Mitigating Spurious Bias for Learning Robust Machine Learning Models

---

A

Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Guangtao Zheng

May 2025

APPROVAL SHEET

This

Dissertation

is submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy

Author: Guangtao Zheng

This Dissertation has been read and approved by the examing committee:

Advisor: Aidong Zhang

Advisor:

Committee Member: Yangfeng Ji

Committee Member: Sheng Li

Committee Member: Jundong Li

Committee Member: Ferdinando Fioretto

Committee Member:

Committee Member:

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, School of Engineering and Applied Science

May 2025

Abstract

Modern machine learning (ML) models have shown strong empirical performance across a wide range of domains. However, they tend to use spurious correlations between target labels and non-essential spurious attributes for predictions, leading to right predictions for the wrong reasons. For example, an image classifier may identify objects based on frequently co-occurring backgrounds rather than the defining features of the objects. This phenomenon, known as spurious bias, can significantly degrade model performance under distribution shifts, where the learned spurious correlations no longer hold, limiting the model's reliability and generalizability in real-world scenarios. This dissertation focuses on spurious bias mitigation for learning ML models that can generalize reliably and robustly in new environments with unknown or known distribution shifts. We propose novel methods tailored for out-of-distribution generalization and generalization under subpopulation shifts, addressing unknown and known distribution shifts, respectively. For out-of-distribution generalization, where target data distributions are unknown during training, we propose to synthesize spurious attributes, such as novel image styles, to explore new data distributions. We design learning algorithms that integrate data exploration into the learning of robust and generalizable features, and demonstrate their effectiveness in challenging settings such as few-shot learning and single domain generalization. Under subpopulation shifts, where proportions of certain data groups are known to vary between training and testing but group annotations are not generally accessible, models may inadvertently rely on spurious attributes in certain data groups for predictions. To address this, we propose multimodal-assisted methods to detect and mitigate spurious bias using pre-trained vision-language models. We further propose fully self-guided methods that leverage internal states of a model for automatic spurious bias detection and mitigation. By directly addressing spurious bias, this dissertation advances the development of robust and trustworthy ML models that make right predictions for the right reasons, improving their reliability across diverse environments.

Acknowledgements

I would like to express my deepest gratitude to the many people who have supported me throughout my journey toward a Ph.D. in Computer Science. First and foremost, I am especially grateful to my advisor, Professor Aidong Zhang, for her invaluable guidance and unwavering support. I feel truly fortunate to have had the opportunity to work under her mentorship. She fosters a lab environment where students are encouraged to pursue cutting-edge research with intellectual freedom. Her keen attention to detail, high-level thinking, and candid feedback have been instrumental in shaping my research. Above all, she genuinely cares about her students' professional development, and her mentorship has been a cornerstone of my growth during my doctoral studies.

I am also grateful to my dissertation committee members—Professor Yangfeng Ji, Professor Jundong Li, Professor Shen Li, and Professor Ferdinando Fioretto—for their insightful feedback and constructive suggestions, which have significantly improved the quality of my dissertation.

I am thankful to my collaborators, lab mates, and friends for their support and companionship along the way. I deeply appreciate Professor Nathan C. Sheffield, with whom I have collaborated since the beginning of my Ph.D. on interdisciplinary research at the intersection of genomics and machine learning. I have learned so much from his expertise and perspective. I also thank Erfaneh Gharavi, Nathan J. LeRoy, Claude Hu, and Julia Rymuza for many insightful discussions. Special thanks go to Mengdi Huai for her generous mentoring during my early years as a Ph.D. student, and to Wenqian Ye for his collaboration and support for my research projects. I am also thankful to my lab mates—Guangxu Xun, Kishlay Jha, Qiuling Suo, Jiayi Chen, Jianhui Sun, Hyun Jae Cho, Guangzhi Xiong, Sanchit Sinha, Lei Gong, Sikun Guo, and Amir Hassan Shariatmadari for making the lab a supportive place. I am also grateful to my friends Yilong Yang, Song Wang, and Peng Wang, and many others whose names are not listed here.

Last but certainly not least, I am profoundly thankful for the unconditional love and support of my family. I am forever indebted to my parents, L. Zheng and X. Zheng, who have supported every decision I've made. I also want to express my deepest appreciation to my wife, Hanjie Chen. From our early days as master's students to the long and demanding journey of pursuing our Ph.D.s, she has been my unwavering source of strength and support. Her endless love and encouragement have sustained me through countless moments of self-doubt and the many challenges that accompany academic life. She believed in me when I struggled to believe in myself, celebrated every small achievement, and reminded me of my goals whenever I felt lost. I am truly grateful for walking this path with her by my side.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Spurious Correlations

Modern machine learning (ML) models have shown strong empirical performance in many application areas. However, this may be achieved by exploiting spurious correlations [2, 3, 4, 5] in the input data. Spurious correlations are brittle associations between spurious attributes of input samples and the corresponding target labels. Figure 1.1 illustrates a spurious correlation between the target Y of an input X and the spurious attribute A determined by a hidden environment variable E. For example, in an environment E where images show a cow on a grassland, the target Y=`cow` and the attribute A=`grassland` formulate a spurious correlation, and the correlation will break in a new environment E' where all images show the correlation between Y=`cow` and A'=`beach` [2, 6]. The attributes `grassland` and `beach` are spurious as they are non-essential to the label `cow` and are not truly predictive of cows in all possible images.

Spurious correlations are prevalent in real-world learning scenarios and tend to be learned by ML models as their decision shortcuts. Common examples include identifying a cow in an image by simply detecting the grassland background in the image rather than the cow itself [2, 6], or detecting pneumonia by only identifying hospital-specific metal tokens in chest X-ray scans [7]. The tendency of using spurious correlations in predictions, known as **spurious bias**, can result in high overall performance

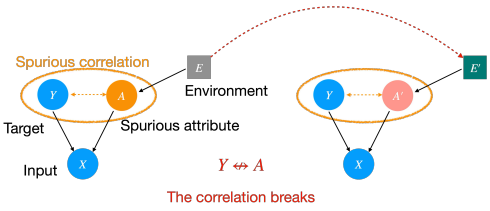


Figure 1.1: Illustration of a spurious correlation between the target Y and the spurious attribute A.

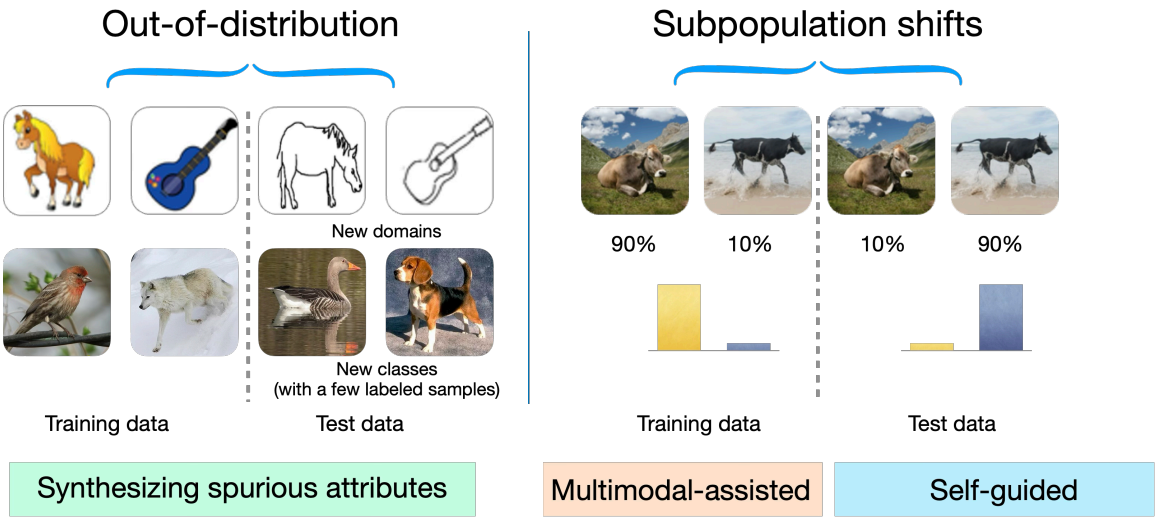


Figure 1.2: Overview of our contributions. We propose to mitigate spurious bias to improve models' generalization under distribution shifts. We consider out-of-distribution generalization (left) and generalization under subpopulation shifts (right), covering unknown and known distribution shifts, respectively. We propose to explore new data distributions by synthesizing spurious attributes to mitigate spurious bias for models generalizing to new domains and new classes with a few labeled samples. To mitigate spurious bias for generalizing under subpopulation shifts, we propose multimodal-assisted and self-guided methods to automatically detect and mitigate spurious biases in models.

when the test data is similar to the training data. However, the bias can also lead to poor generalization under distribution shifts where the spurious correlations do not hold, such as the chest X-ray scans with new metal tokens. Consequently, the non-robustness caused by spurious bias can pose significant risks, especially in critical domains.

Mitigating spurious bias is crucial for obtaining robust and generalizable models under distribution shifts, ensuring that the models are reliable and trustworthy. In this dissertation, we design a suite of spurious bias mitigation methods to improve model generalization under a wide range of distribution shifts. In particular, we consider **out-of-distribution generalization** and **generalization under subpopulation shifts**, covering unknown and known distribution shifts, respectively.

1.2 Out-of-Distribution Generalization

For out-of-distribution generalization, we consider two common settings: (1) generalizing to novel domains [8, 9, 10, 11] and (2) generalizing to novel classes with a few labeled samples [12, 13, 14, 15, 16].

Generalizing to Novel Domains. In this setting, the goal is to train a model on a source domain that can directly generalize (without any fine-tuning) to novel domains that differ in styles but share the same targets as the source domain, such as the images of horse and guitar in the left part of Figure 1.2. Spurious bias is often developed from the model's strong reliance on domain-specific features and can be revealed by a significant performance drop of the model on samples without those features, i.e., samples from novel domains. However, since samples from novel domains are inaccessible during training, it is challenging to reveal spurious bias and then mitigate it accordingly. A common and practical solution is to produce potential domain shifts through data augmentation and mitigate the model's reliance on domain-specific features [17, 18, 19, 20, 21]. However, existing approaches often produce limited domain shifts or the produced domain shifts do not benefit model generalization, resulting in suboptimal mitigation of spurious bias.

Generalizing to Novel Classes. In this setting of out-of-distribution generalization, the goal is to quickly adapt a base model to recognize novel classes with a few labeled samples from these classes (e.g., one sample per class) [22, 23, 12]. Spurious bias often arises from the complex interaction between the prior knowledge in the pre-trained base model and the learning from a few samples of novel classes during the adaptation process. For example, the prior knowledge in the base model may bias the adaptation process so that the adapted model recognizes novel objects in images by solely identifying backgrounds in the images, thereby hindering the generalization to novel classes. Existing approaches [22, 24, 25, 23, 12, 26, 15] mainly focus on training a base model that works well with a specifically designed adaptation method so that a few labeled samples can be effectively used to learn novel classes. However, since only a few labeled samples of novel classes are available for adapting the base model during testing, it is challenging to identify and then mitigate the spurious biases developed during the adaptation process. Therefore, generalization to novel classes may be limited. In such a case, data augmentation [27] remains an effective and practical approach to produce new data distributions and mitigate the reliance on spurious correlations. However, the existing approach [27] adopts a manual design of data augmentation strategies and does not consider data augmentation and spurious bias mitigation in tandem, limiting the generalization to out-of-distribution data.

1.3 Generalization under Subpopulation Shifts

Subpopulation shifts [28, 29] are known a priori compared with distribution shifts induced by novel classes or domains. They refer to changes in the proportion of certain subpopulations or groups of data between training and testing. For instance, as shown in the right part of Figure 1.2, images of cows on grass fields constitute 90% of the training data but only 10% of the test data, while images of cows on beaches become the majority, comprising 90% of the test data. Models trained in this setting tend to exploit the spurious correlations in the majority data for predictions and achieve high overall performance on the data with a distribution similar to that of the training data. However, these models tend to perform poorly overall on the test data with subpopulation shifts, or equivalently, on certain subpopulations of the test data.

To improve model generalization under subpopulation shifts, mitigating spurious bias is crucial. Unlike the out-of-distribution generalization, this setting generally assumes that the training data contains all spurious correlations that may occur in the test data, which allows detection of spurious biases in models and design of targeted mitigation strategies during training. However, annotations of spurious correlations are typically required [3, 30]. Each annotation, also known as a group label, is a tuple of a class label and a spurious attribute. In practice, acquiring group labels is a significant barrier as it requires costly and labor-intensive human annotations. Recent approaches relax the requirement on group labels of the training data by using proxy signals such as prediction losses [31, 32], misclassification [33], or inferred group labels [34, 35], or by fine-tuning on a small portion of validation set with ground truth group labels [4]. Nevertheless, all existing approaches are not completely annotation-free as they still require a validation set with such annotations to select robust models during training.

1.4 Dissertation Contributions and Organization

As depicted in Figure 1.2, we propose three categories of methods: synthesizing spurious attributes, multimodal-assisted, and self-guided methods, addressing unique challenges of spurious bias mitigation under unknown and known distribution shifts. The contributions and organization of the dissertation are summarized as follows:

- **Synthesizing Spurious Attributes (Chapter 3).** We propose to generate out-of-distribution data by synthesizing spurious attributes for improving model generalization to novel domains [36] in Chapter 3.1 and to novel classes [37] in Chapter 3.2. We synthesize

spurious attributes via data augmentations optimized for a given model, producing diverse spurious attributes and effectively reducing the model's reliance on specific ones, in particular low-level spurious attributes such as certain pixels or orientations of images. In Chapter 3.3, we propose to synthesize spurious attributes via a dictionary of learnable latent features which represent high-level spurious attributes such as certain background objects, for generalizing to novel classes [38].

- **Multimodal-Assisted Methods (Chapter 4).** Vision-language models (VLMs) [39, 40, 41] have demonstrated a strong multimodal understanding capability in detecting high-level attributes of images, such as gender or background objects. In Chapter 4.1, we demonstrate that the detected attributes from a VLM can be used to create challenging subpopulation shifts in various classification tasks [42]. With the assistance of VLMs, we propose to automatically detect and mitigate spurious biases in trained models via fine-grained classification on both class labels and detected spurious attributes [5] in Chapter 4.2 and via meta-learning [43] in Chapter 4.3. With pre-trained VLMs, our proposed methods can effectively build models robust to spurious bias and improve their generalization under subpopulation shifts without costly spurious correlation annotations.

- **Self-Guided Methods (Chapter 5).** We propose self-guided methods for spurious bias mitigation by probing the latent space of a model. In Chapter 5.1, we propose to probe the latent representations in a model to detect prediction shortcuts and use them to regularize the model for enhanced robustness to subpopulation shifts [44]. In Chapter 5.2, we propose to identify latent dimensions affected by spurious bias and mitigate their contributions to final predictions [45]. In Chapter 5.3, we further extend our efforts to mitigate spurious bias in multimodal models by leveraging the similarities between vision and text representations to select prompts for mitigating spurious bias in zero-shot classification. Exploiting models' latent representations is free from requiring spurious correlation annotations and can be applied to different data modalities.

Our research advances the development of robust and trustworthy ML models that make right predictions for the right reasons, improving their reliability across diverse environments.

Chapter 2

Related Work

Spurious correlations describe superficial associations between spurious, non-essential attributes and targets in data [46], and they can be used by machine learning models as shortcuts [47, 6, 48, 49]. This shortcut learning phenomenon in machine learning models results in spurious bias — the tendency to use spurious correlations in data for predictions. Spurious bias may surface in various learning scenarios and affect how models generalize to different test data distributions. In this chapter, we discuss related works on training models that can generalize to novel domains (Chapter 2.1), to novel concepts with a few labeled samples (Chapter 2.2), and to subpopulation shifts (Chapter 2.3) from the perspective of spurious bias mitigation.

2.1 Generalizing to Novel Domains

Generalizing to novel domains requires models to learn robust and domain-invariant features. Machine learning models tend to spuriously associate their predictions with domain-specific attributes from source domains, such as associating the predictions on digit images from MNIST [50] with digit size or with the black-and-white image style. Domain generalization methods [8, 9, 10, 11] aim to address this problem. They do not require samples from target domains during training; however, they typically use training samples from multiple source domains instead of one to facilitate learning domain-invariant features. In this dissertation, we consider a more practical scenario, i.e., single domain generalization (SDG), where there is only one single source domain for training and no access to target distributions.

Existing methods on SDG aim to perturb or generate samples with spurious and out-of-domain attributes to improve generalization to novel domains. These methods can be broadly classified into

the following three categories. First, methods that use standard data augmentation [17, 18, 19, 20, 21] can be used to augment the source domain data for out-of-distribution generalization, but they are not very effective in generating samples with large distribution shifts. Second, adversarial data augmentation methods typically augment the source domain data by generating samples either in the pixel space [51, 52] or via perturbing latent feature statistics [53, 54], but these methods also struggle to produce samples with diverse spurious attributes. Third, generative modeling methods [55, 56, 57] use generative models to produce diverse training samples. However, since generative models are also learned from the source domain, the styles of the generated samples are still related to those in the source domain.

To produce samples with large domain shifts for generalizing to novel domains, semantics transformations [58] are proposed to manipulate certain kinds of semantics of an image, such as hue and saturation [59] or color and texture [60]. These transformations are used to produce "unrestricted" perturbations [60] in adversarial samples, which are traditionally generated by finding imperceptible perturbations under a norm ball constraint [60]. Semantics transformations have also been used to improve few-shot generalization [37] via meta-learning [22, 25, 26]. However, these methods cannot be directly adopted in our problem setting since they focus on performing adversarial attacks, while our goal is to improve a model's SDG performance.

Our proposed method, termed adversarial semantics transformations (AdvST) [36], mitigates a model's over reliance on spurious and domain-specific attributes by synthesizing diverse spurious attributes using semantics transformations (Chapter 3.1). AdvST is motivated from the success of standard data augmentation methods, such as rotation, scale, and color jittering, in training robust models. It repurposes these augmentation methods as semantics transformations with learnable parameters, generating samples with large domain shifts from the source domain. Compared with a similar method, Adversarial AutoAugment [61], which adversarially learns augmentation policies to improve *in-domain* generalization performance, AdvST uses semantics transformations to manipulate the semantics of an image, such as the hue or rotation degree, that is *independent* of the source domain, allowing us to inject *external* styles to the generated samples. Moreover, AdvST directly generates worst-case samples to improve *out-of-domain* generalization performance. A parallel work [62] uses a pre-defined set of linguistic transformations, such as negation and paraphrasing, to augment text data for improved vision-language inference performance. However, these transformations do not have learnable parameters and cannot be fine-tuned into different ones.

2.2 Generalizing to Novel Classes

Generalizing to novel classes or concepts with a few labeled examples is a key aspect of human-like intelligence, as humans can quickly learn new concepts with minimal supervision. However, modern machine learning models typically require large amounts of training data to achieve high performance. With a few labeled samples for learning novel classes, models often overfit to spurious correlations in those samples and struggle to generalize to out-of-distribution data of the novel classes [22, 63, 42].

Few-shot classification [23, 12, 64, 65, 66, 67] serves as a practical and representative task for studying this learning scenario and has received great attention recently. Existing methods tackle few-shot classification by designing data augmentation methods or data-efficient learning algorithms.

Existing approaches that design learning algorithms can be broadly categorized into meta-learning and transfer learning. The transfer learning approach [64, 66] first trains a robust embedding model on a base dataset and then fine-tunes it with a few labeled samples from novel classes. In contrast, meta-learning, which is the dominant approach in few-shot classification, leverages a learning-to-learn paradigm, where models are trained across multiple tasks to learn to extract robust and non-spurious features so that they can generalize to unseen tasks using limited data. A task is designed to have a support set and a query set. The support set is used in the inner loop of meta-learning for adapting a model's parameters to novel classes. The query set is used in the outer loop of meta-learning to update the model for better adaptation to the novel classes. Many meta-learning algorithms can be further divided into *optimization-based* and *metric-based* methods. Optimization-based methods [22, 24, 25] use gradient descent to update part or all model parameters in the inner loop to learn a good model initialization that can be fast adapted to a new task within a few gradient descent steps. In contrast, metric-based methods [23, 12, 26, 15] learn a shared embedding network for different tasks with specialized metric functions for classification, such as an SVM classifier [14], a prototype-based classifier with Euclidean distance [12], or a classifier with Earth Mover's distance [68].

Some recent works [27, 69, 70] propose to increase the diversity of tasks in meta learning via data augmentations to mitigate reliance on spurious correlations. By default, many meta-learning algorithms adopt simple data augmentation operations on images in meta-training. A recent work [27] analyzed how support, query, task, and shot augmentations affect the performance of various meta-learning algorithms. Then, they proposed a set of manually designed augmentation policies for meta-learning. However, these policies are manually designed and not directly optimized for meta-

learning. Instead of focusing on the input space, some recent works [69, 70] propose feature mixing and task interpolation to increase the diversity of tasks and mitigate overfitting for gradient-based meta-learning.

We propose the following two methods that improve model generalization to out-of-distribution data from novel classes:

- Learn to learn task transformations (L2TT) [37] (Chapter 3.2): a data augmentation method that leverages differentiable image operations to construct a learnable task transformation layer that can be directly integrated into existing meta-learning frameworks. Compared with automatic data augmentation [71, 72, 73] which finds augmentation policies, differentiable data augmentation [74, 75, 76] is a promising paradigm for data augmentation without resorting to expert knowledge and greatly reduces the cost of searching for optimal data augmentation policies via differentiable image operations. Different from existing differentiable data augmentation methods which often rely on an adversarial loss [75] or an extra reward signal [71] to learn augmentation policies, our method does not have this limitation thanks to the bi-level learning structure of meta-learning. Moreover, instead of simply creating diverse data samples, our proposed L2TT layer optimally transforms training tasks for different meta-learning settings to reduce reliance on spurious correlations in a few labeled samples and improve generalization to new concepts with limited data.

- Knowledge-guided semantics adjustment (KGSA) [38] (Chapter 3.3): a metric-based meta-learning method that leverages a model's latent and semantically meaningful representations to synthesize and mitigate spurious features. Different from existing metric-based methods [23, 12, 26, 15], our method meta-learns a dictionary of spurious features which are adaptively combined and then mitigated from input samples of novel classes. In comparison with the work [77] which obtain classifier weights from external sources, such as class attributes, our work obtains classifier weights only from the training data and adjusts them to mitigate reliance on non-essential features. Additionally, unlike the work in [78] where a shallow network is used to linearly combine memory features and class centroids for a long-tailed recognition problem, our work stores a set of spurious and class-agnostic semantic features that are meta-learned from the training data.

2.3 Generalizing under Subpopulation Shifts

Subpopulation shifts are changes in the proportion of certain subpopulations or groups of data between training and testing [28, 29]. A model with spurious bias tends to have degraded performance on data with shifts in spurious correlations where the learned spurious correlations no longer exist or dominate in data. For example, in the training data, the majority of cow images have grass fields and images of a cow at a beach are the minority; but in the test data, images of a cow at a beach are the majority while images of a cow on a grass field become the minority.

It is critical to mitigate spurious bias for improving a model's robustness to subpopulation shifts. Oracle methods exploit ground truth group labels during training to mitigate the reliance on specific spurious correlations. Here, group labels, specified by class labels and spurious attributes, indicate the presence of spurious correlations in subsets of the training data. Some existing methods [79, 80, 81] use group labels to balance data distributions during training, to formulate a distributionally robust optimization objective [3], or to progressively expand group-balanced training data [30]. Although these methods achieve remarkable success in spurious bias mitigation, the reliance on group labels becomes a barrier in practice, as obtaining such labels often requires domain knowledge and labor-intensive annotation efforts.

To alleviate the dependency on group labels during training, existing approaches propose inferring group labels through various means, such as identifying misclassified samples [31], clustering hidden representations [35], employing invariant learning techniques [82], or training group label estimators using a few samples with group labels [34]. Nevertheless, group labels in the validation data are still needed to specify which biases to address and to select models robust to these biases. Recent last-layer retraining methods [4, 83] leverage group-balanced validation data to fine-tune the last layer of a model.

Without group labels, directly detecting spurious bias typically requires domain knowledge [84, 85] and human annotations [86, 87]. For example, previous works exploit domain knowledge to discover that object backgrounds [49] and image texture [48] could be spuriously correlated with target classes and severely bias the predictions of deep learning models. Recent works [88, 89] use model explanation methods to detect spurious features. Neurons in the penultimate layer of a robust model are also exploited for spurious feature detection with limited human supervision [90, 91].

To automatically detect and effectively mitigate spurious bias in models without group labels or human annotations, we propose **multimodal-assisted** and **self-guided** methods.

For **multimodal-assisted** methods, we propose to exploit the prior knowledge in pre-trained vision-language models (VLMs) and their multimodal understanding capabilities to automatically detect spurious attributes in interpretable text format. We show that these detected attributes can be used to construct various challenging classification tasks with subpopulation shifts [42] (Chapter 4.1). A recent work [92] proposes to use a pre-defined concept bank as an auxiliary knowledge base for spurious attribute detection. In contrast, we propose a fully automatic spurious attribute detection method leveraging pre-trained VLMs. We further propose self-debiasing algorithms that mitigate the reliance on detected spurious attributes via fine-grained classification on both classes and the detected attributes [5] (Chapter 4.2) and via meta-learning [43] (Chapter 4.3). Our proposed mitigation algorithms can effectively use the extracted attributes for spurious bias mitigation without requiring group labels.

For **self-guided** methods, we propose to leverage the latent representations in a model for extracting signals related to spurious bias and mitigating it accordingly. Specifically, we propose ShortcutProbe [44] which leverages a probe set of data to derive shortcut vectors in a model's latent space that represent spurious bias and uses these vectors as regularization to mitigate spurious bias (Chapter 5.1). We further propose NeuronTune [45], a fine-grained mitigation strategy by detecting neurons that are affected by spurious bias via the distributions of their activation values (Chapter 5.2). These approaches are modality-agnostic and do not require group labels. Finally, we extend our effort to multimodal models and propose to use the similarities between latent representations for vision and text inputs to guide the selection of prompts to mitigate spurious bias in multimodal models in zero-shot classification (Chapter 5.3). Debiasing in the zero-shot setting targets at the multimodal spurious bias developed in the pre-training phase, where text representations of targets tend to be misaligned with spurious features that frequently co-occur with target objects in images [93]. Methods for mitigating multimodal spurious bias without downstream data typically exploit easily obtained text data. Chuang et al. [94] generates text prompts with known spurious attributes to debias the weights of a zero-shot classifier constructed from a pre-trained contrastive language-image pre-training (CLIP) model. Adila et al. [95] leverage large language models (LLMs) to obtain spurious and core attributes of class labels and use these attributes to enhance image representations from a pre-trained CLIP model. While our work also focuses on the zero-shot setting, we propose a self-adaptive spurious bias mitigation method that explores prompts provided with pre-trained CLIP models, without requiring external assistance, such as relying on LLMs or specific knowledge about downstream tasks.

Chapter 3

Synthesizing Spurious Attributes for Out-of-Distribution Generalization

In out-of-distribution generalization, target data distributions are unknown during model training. Machine learning models with spurious bias may fail to generalize to out-of-distribution target data since the learned spurious attributes in the training data may not exist in the test data. Our strategy is to synthesize spurious attributes via data augmentations to explore new data distributions, encouraging models to learn robust and generalizable features. In this chapter, we consider few-shot learning and single domain generalization as the two challenging settings of out-of-distribution generalization. In few-shot learning, models may use the spurious correlations captured from the large set of training data for learning novel classes from a few labeled samples, resulting in inferior few-shot generalization performance. In single domain generalization, models may use the spurious correlations captured in the source domain for predictions, resulting in degraded performance on target domains where the learned spurious correlations no longer exist. In this chapter, we first propose an adversarial data augmentation framework to adaptively generate data with challenging spurious attributes for single domain generalization in Chapter 3.1. Then, in Chapter 3.2, we propose a meta-learning framework with adaptive data augmentations to synthesize various spurious attributes using training data for improved few-shot generalization. In Chapter 3.3, we propose a knowledge-guided

semantics adjustment meta-learning framework to synthesize and mitigate spurious features in the latent space of few-shot classifiers.

3.1 AdvST: Adversarial Learning with Semantics Transformations

3.1.1 Introduction

Domain generalization [8, 96, 97, 9] aims to learn a model that can generalize well to target (test) domains with unknown distribution shifts using multiple source (training) domains. These source domains contain various spurious attributes, allowing the model to learn robust decision rules against spurious attributes. However, having diverse domains for training is a strong assumption due to various practical considerations, such as data collection budgets or privacy issues. A realistic alternative is *single domain generalization* (SDG) [52, 97], which only requires data from a single source domain for model training. SDG is challenging for deep image classifiers. Although they have achieved impressive performance on benchmarks, they strongly hinge on the implicit assumption that training and test data follow the same distribution. Their performance can drop significantly when they use spurious correlations for predictions and there are shifts between training and test data distributions caused by, for example, changes in object appearance or data collection methods.

Data augmentation is an effective approach to SDG. It augments the source domain data with various spurious attributes to expand the coverage on the unseen target domain during model training. Methods of data augmentation include using adversarial learning [51, 52, 55] or using generative models [55, 56, 57] to generate diverse data samples. The utility of standard data augmentations, such as scale, or CutOut [17], has not been fully exploited in SDG. In practice, these augmentation methods have been widely used in model training for *in-distribution* generalization. However, their applications in SDG are limited. In most cases, they serve as a part of the data preprocessing procedure in other SDG methods [51, 52, 55]. Although it is intuitive that applying multiple standard data augmentations to the source domain data can generate diverse samples and hence improve a model's SDG performance, we lack a principled approach to fully realize the benefit brought from multiple standard data augmentations.

Therefore, in this section, we revisit standard data augmentations for SDG and develop methods that make them a strong competitor in SDG. We consider the composition of several standard data augmentation as a semantics transformation which can manipulate certain kinds of semantics of

a sample, such as the brightness and hue of an image. Normally, standard data augmentations have pre-specified and fixed parameters. Here, we make these parameters learnable in a semantics transformation so that we can tune these parameters to produce semantically significant variations and bring new spurious attributes, such as different styles, that are different from the source domain data. With semantics transformations, we can transform data in the source domain to a fictitious one which has large domain shifts from the source and possibly covers data in target domains, yielding favorable SDG performance.

To learn semantics transformations for SDG, we propose AdvST, an adversarial learning framework that trains a robust model and generates challenging data samples iteratively with mini-max optimization. In the maximization phase, we learn the parameters of semantics transformations so that the samples transformed by semantics transformations maximize the prediction loss of the model. To avoid learning a trivial solution where the information in the source domain samples is completely lost after semantics transformations, we additionally regularize the distance between the source domain samples and the transformed ones in the deep feature space of the model to keep the core features of the source domain data. In the minimization phase, we train the model with the new samples generated by semantics transformations.

We theoretically show that the learning objective of AdvST connects to that of distributionally robust optimization (DRO) [98, 99]. DRO trains a robust model using the worst-case distribution that leads to the worst model performance on an uncertainty set—a set of neighboring distributions with a predefined value of distributional shifts from the training data. Increasing the coverage of the uncertainty set on the target domain data can improve the model's SDG performance. AdvST can be considered as a special form of DRO whose uncertainty set consists of semantics-induced data distributions which are generated by applying semantics transformations to samples from the source distribution. We demonstrate that AdvST can produce samples in the uncertainty set that expand the coverage on the target domain data.

Our method, despite being a simple method utilizing standard data augmentations, is surprisingly competitive in SDG. AdvST consistently outperforms existing state-of-the-art methods in terms of the average SDG performance on three benchmark datasets.

3.1.2 Methodology

Semantics Transformation

We define a semantics transformation as a composition of several standard data augmentation functions that manipulate certain kinds of semantics of a sample. For example, we can perturb both the hue and brightness of an image x with $\tau(x;\omega) = o_h(o_b(x;\omega_b);\omega_h)$, where o_h is the function that changes the hue of x, o_b changes the brightness of x, and $\omega = \omega_b \cup \omega_h$ denotes the set of parameters for τ. We construct a set of M semantics transformations $\mathcal{T} = \{\tau_i(\cdot;\omega_i), i = 1, \ldots, M\}$ by randomly composing $L(1 \leq L \leq L_{\max})$ unique standard data augmentation functions (Table 3.1).

Intuitively, a semantics transformation with a large L can produce diverse samples with various spurious attributes. However, depending on the target domain data, the semantics transformations that produce more diverse samples are not necessarily better than those producing less diverse ones. Since we have no knowledge about target domains in SDG, we first uniformly choose the length for semantics transformations and then uniformly choose a semantics transformation with the selected length. Thus, we derive the distribution over M semantics transformations as $G(\tau^L) = \frac{1}{M_L L_{\max}}$, where τ^L denotes a semantics transformation with L standard augmentations, $L_{\max}$ is the maximum number of standard augmentations in τ^L, and M_L is the total number of τ^L and satisfies $M = \sum_{L=1}^{L_{\max}} M_L$.

Learning Objective of AdvST

SDG aims to train a model that is robust to unseen domain shifts with the training samples from a single source domain. The robustness of the trained model to unseen domain shifts depends on how much the training data covers target domains. Therefore, with semantics transformations, we aim to generate new data samples that have large domain shifts from the source domain, increasing the chance of covering data samples from unseen target domains.

A key property of the samples from target domains is that they often yield a high average prediction loss because of their large domain shifts from the source. This motivates AdvST, an adversarial learning framework that learns semantics transformations to generate challenging samples with significant semantics variations for model training.

Given a model f_θ with parameters θ, a set of source domain samples $\mathcal{D}_S = \{(x_n, y_n)\}_{n=1}^N$ with N pairs of training sample x_n and its label y_n, and a distribution G over a set of M semantics

Algorithm 1 Adversarial learning with semantics transformations (AdvST)

Input: Source dataset $\mathcal{D}_S$, extended training set $\mathcal{D}$ with K domains, distribution over M semantics transformations G, initial model weights θ_0, number of training epochs E, batch size B, number of batches per epoch N_B, and number of updates in the maximization procedure $T_{\max}$

Output: learned weights θ

$\theta \leftarrow \theta_0$, $\mathcal{D}$.add($\mathcal{D}_S$)
for $e = 1, \ldots, E$ **do**
  //Minimization procedure
  for $b = 1, \cdots, N_B$ **do**
    Get a batch of B samples $\mathcal{B}$ from $\mathcal{D}$
    Update θ with Equation (3.4)
  end for
  //Maximization procedure
  Initialize an empty $\mathcal{D}_e$
  for $(x_n, y_n) \in \mathcal{D}_S$ **do**
    Sample τ from G and initialize its parameters ω_n^0
    for $t = 1, \cdots, T_{max}$ **do**
      Generate a sample $x_n^t = \tau(x_n; \omega_n^{t-1})$
      Update ω_n^t with Equation (3.3)
    end for
    Append $(\tau(x_n, \omega_n^{T_{\max}}), y_n)$ to $\mathcal{D}_e$
  end for
  $\mathcal{D}$.add($\mathcal{D}_e$)
end for
return θ

transformations $\{\tau_i(\cdot;\omega_i)\}_{i=1}^M$, we express the learning objective of AdvST as:

$$\theta^* = \min_{\theta\in\Theta} \max_{\psi\in\Psi} \mathop{\mathbb{E}}_{\tau\sim G} \mathop{\mathbb{E}}_{\xi\sim\mathcal{D}_S} \big[\ell(\theta;\xi') - \lambda d_\theta(\xi',\xi)\big], \tag{3.1}$$

where $\xi = (x, y)$ denotes a tuple of a sample x and its label y, $\xi' = (\tau(x;\omega), y)$ is the tuple of the same label y and a new sample obtained by applying the semantics transformation τ to x, $\ell(\theta;\xi)$ is the prediction loss for $\xi = (x, y)$, Θ denotes the set of all possible values of θ, $\psi = \cup_{i=1}^M \omega_i$ denotes the union of the parameters of M semantics transformations, Ψ is the set of all possible values of ψ, λ is a nonnegative regularization parameter, and d_θ is the squared Euclidean distance function between ξ and ξ' in the deep feature space of the model f_θ, i.e., $d_\theta(\xi, \xi') = \|v - v'\|_2^2$ with the embeddings v and v' of x and x', respectively.

The objective in (3.1) aims to train a robust model with the challenging samples generated from the samples in the source domain while maintaining the core features of the original data. The novel part of (3.1) is that instead of generating images in the pixel space, we adversarially learn the parameters of semantics transformations, exploiting the domain knowledge in standard data augmentations to generate diverse images.

Learning Algorithm

We adopt an iterative optimization algorithm [51, 52] to solve (3.1). Specifically, the algorithm consists of a minimization and a maximization optimization procedures.

Maximization Procedure. We generate worst-case samples via optimized semantics transformations. Specifically, we first sample a semantics transformation τ from G. Then, we sample an example x_n from $\mathcal{D}_S$. We solve the inner maximization problem in (3.1) by applying $T_{\max}$ steps of stochastic gradient ascent to the parameters of the sampled semantics transformation τ. To facilitate generating diverse samples, we add a maximum entropy regularizer [52] during the optimization. In the tth ($1 \leq t \leq T_{\max}$) iteration, we have the following steps:

$$x_n^t = \tau(x_n; \omega_n^{t-1}) \tag{3.2}$$

$$\omega_n^t = \omega_n^{t-1} + \beta \nabla_{\omega_n^{t-1}} \Big(\ell(\theta; x_n^t, y_n) - \lambda d_\theta((x_n^t, y_n), (x_n, y_n)) + \epsilon l_{ent}(\theta; x_n^t, y_n) \Big), \tag{3.3}$$

where ω_n^t denotes the learnable parameters of τ for the n-th data sample at iteration t, $l_{ent}(\theta; x_n^t, y_n)$ is an entropy regularization term to further promote learning diverse samples, ϵ is a nonnegative regularization parameter, and β denotes the learning rate in this procedure. We repeat the above steps until all samples in $\mathcal{D}_S$ have been processed. The synthetic data points $\{(\tau(x_n; \omega_n^{T_{\max}}), y_n)\}_{n=1}^N$ are treated as a new domain of data. We add these generated samples to the extended training set denoted as $\mathcal{D}$, which is initialized as $\mathcal{D}_S$.

Minimization Procedure. We use samples generated from the maximization step to train a robust model θ against unseen distribution shifts. To avoid model forgetting, at each iteration, we sample a batch of B samples $\mathcal{B}$ from the extended training set $\mathcal{D}$ to also use previously generated samples. We add a regularizer $\ell_{reg}(\theta; \mathcal{B})$ consisting of standard supervised contrastive [100] and entropy loss [52] terms to facilitate learning robust representations. At each iteration, we update the model parameters θ using mini-batch stochastic gradient descent as follows

$$\theta \leftarrow \theta - \alpha \nabla_\theta \Big(\frac{1}{B} \sum_{(x,y) \in \mathcal{B}} \ell(\theta; x, y) + \ell_{reg}(\theta; \mathcal{B}) \Big), \tag{3.4}$$

where "$\leftarrow$" denotes value assignment, and α denotes the learning rate.

The complete algorithm is shown in Algorithm 1. We further analyze the space and time complexities of the algorithm with practical considerations in the following.

Space Complexity. In the iterative optimization, we keep adding the generated samples to the extended training set $\mathcal{D}$. The size of $\mathcal{D}$ increases with the iteration number, which is not scalable

when the initial training set $\mathcal{D}_S$ or the iteration number is large. *Therefore, we implement $\mathcal{D}$ as a domain pool that only stores the generated samples from the most recent K runs of the maximization procedures.* In practice, depending on the size of $\mathcal{D}_S$, we set K in the range of 2 to 5 to ensure that we have sufficient samples for training without incurring the scalability issue.

Time Complexity. The time complexity of each iteration of the optimization is $N_B C_\mu + T_{\max} C_G N_B$, where C_μ denotes the complexity of updating the model, C_G denotes the complexity of updating the parameters of semantics transformations, and N_B denotes the number of training batches. Generally, we have $C_G \approx C_\mu$ because C_G and C_μ both include back-propagating the gradients throughout the whole model, and the number of parameters in semantics transformations is negligible compared to the number of model parameters. Therefore, the time complexity is $O(ET_{\max}N_B)$, where E is the total iterations (epochs). In practice, to reduce the impact of $T_{\max}$, we could perform the maximization on different batches in parallel or do early stopping when the difference in loss between consecutive maximization steps is lower than a given threshold.

3.1.3 Theoretical Analysis: Connection to DRO

DRO Formulation

The learning objective of SDG can be expressed via DRO [99] since it does not rely on the notion of a known target distribution. Specifically, DRO chooses a set of probability distributions $\mathcal{U}$ called uncertainty set, and then finds a decision θ from Θ that provides the best hedge against $\mathcal{U}$ by solving the following mini-max problem:

$$\min_{\theta \in \Theta} \max_{Q \in \mathcal{U}} \mathbb{E}_{\xi \sim Q}[\ell(\theta; \xi)], \tag{3.5}$$

where $\ell(\theta; \xi)$ is the prediction loss with the data-label pair $\xi = (x, y)$, Θ denotes the set of all possible model parameters, and $\mathcal{U}$ contains distributions that are at most δ-distance away from the source distribution P. The uncertainty set, $\mathcal{U} = \{Q | D(P, Q) < \delta\}$, depends on a distance metric $D(\cdot, \cdot)$ and a predefined threshold $\delta > 0$. The objective in (3.5) finds an optimized model under the worst-case distribution Q^* found in $\mathcal{U}$ that maximizes the prediction loss.

Semantics-Induced Distribution

Given a set of M semantics transformations, a semantics-induced distribution $Q_\psi(\xi')$ is defined as follows

$$Q_\psi(\xi') = \sum_{\tau_i} G(\tau_i) \int_\xi p(\xi'|\tau_i, \xi, \omega_i) dP, \tag{3.6}$$

where $\xi' = (x', y')$, $\xi = (x, y)$ is a sample from the source distribution P, $\psi = \cup_{i=1}^{M} \omega_i$ denotes the parameters of M semantics transformations, and $p(\xi'|\tau_i, \xi, \omega_i)$ is the probability of obtaining ξ' from ξ and the ith semantics transformation τ_i with parameters ω_i. We require that transformed samples are still assigned with their original labels. Therefore, we have $p(\xi'|\tau_i, \xi, \omega_i) = 0$ if $y' \neq y$. Moreover, if τ_i is a deterministic transformation, then $p(\xi'|\tau_i, \xi, \omega_i) = 1$ when $\tau_i(x; \omega_i) = x'$ and $y' = y$ and $p(\xi'|\tau_i, \xi, \omega_i) = 0$ otherwise. If τ_i is a stochastic transformation, then $p(\xi'|\tau_i, \xi, \omega_i)$ follows the distribution of $\tau_i(x; \omega_i)$. A sample ξ' from Q_ψ can be obtained by first sampling ξ from P with $y = y'$ and τ_i from G, and then obtaining $x' = \tau_i(x; \omega_i)$. Given G and P, Q_ψ fully depends on ψ. We denote the set of all semantics-induced distributions as $\mathcal{Q}_\Psi = \{Q_\psi | \psi \in \Psi\}$, where Ψ is the set of all possible parameters ψ.

Uncertainty Set of AdvST

The uncertainty set of AdvST consists of semantics-induced distributions Q_ψ around the source distribution P to simulate unseen target distributions. These distributions should not deviate too much from the source to avoid hedging against noisy distributions that are not learnable. Hence, we need a proper distance metric $D(\cdot, \cdot)$ to control the distribution shifts. Since semantics transformations create new data samples, we use Wasserstein distances (Definition 3.1) as the metric D to allow a data distribution Q_ψ to have a different support from that of P.

Definition 3.1. (*Wasserstein distances* [101, 102, 103]) Let Ξ be a measurable space. Given a transportation cost function $c : \Xi \times \Xi \to [0, \infty)$, which is nonnegative, lower semi-continuous, and satisfies $c(\xi, \xi) = 0$, for probability measures Q and P on Ξ, the Wasserstein distance between Q and P is

$$W_c(Q, P) = \inf_{J \in \prod(Q,P)} \mathbb{E}_{(\xi,\xi') \sim J}[c(\xi, \xi')], \tag{3.7}$$

where $\prod(Q, P)$ denotes all joint distributions with marginal distributions being P and Q.

We define the transportation cost function c in the deep feature space [52, 51] to include distributions whose samples have large style variations since these samples may still be close to the samples from the source distribution in the deep feature space. To exclude noisy distributions whose data samples change their original labels in the source domain after transformations, we design the cost of moving a source distribution sample to such a sample as infinity. Specifically, the cost function of moving $\xi = (x, y) \sim P$ to $\xi' = (x', y') \sim Q_\psi$ given the model θ is defined as follows

$$c_\theta((x, y), (x', y')) := ||v - v'||_2^2 + \infty \cdot 1\{y \neq y'\}, \tag{3.8}$$

where v and v' are the model-dependent embeddings for x and x', respectively. Therefore, the uncertainty set that we consider in AdvST is

$$\mathcal{U}_\Psi = \{Q | Q \in \mathcal{Q}_\Psi, W_c(Q, P) < \delta\}, \tag{3.9}$$

where δ ($\delta > 0$) denotes the predefined distance threshold between the source P and the semantics-induced distributions $\mathcal{Q}_\Psi$.

DRO Learning Objective for AdvST

Directly solving Equation (3.5) with $\mathcal{U} = \mathcal{U}_\Psi$ is intractable since it requires searching over the infinite dimension space of distribution functions. We consider the following Lagrangian relaxation with the penalty parameter λ:

$$\min_{\theta \in \Theta} \max_{Q \in \mathcal{Q}_\Psi} \{\mathbb{E}_{(x,y)\sim Q}[\ell(\theta; x, y)] - \lambda W_c(Q, P)\}. \tag{3.10}$$

However, Equation (3.10) is still hard to compute. For the inner maximization term of Equation (3.10), Proposition 3.1 provides a tractable form which only requires the source distribution P and the distribution over semantics transformations G.

Proposition 3.1. Let $\ell : \Theta \times \mathcal{X} \times \mathcal{Y} \to [0, \infty)$ denote the loss function which is upper semi-continuous and integrable. The transportation cost function $c : \Xi \times \Xi \to [0, \infty)$ with $\Xi = \mathcal{X} \times \mathcal{Y}$ is a lower semi-continuous function satisfying $c(\xi, \xi) = 0$ for $\xi \in \Xi$. Let G denote the distribution over

M semantics transformations $\{\tau_i | i = 1, \ldots, M\}$. Then, for any given P and $\lambda \geq 0$, it holds that

$$\begin{aligned} &\sup_{Q \in \mathcal{Q}_\Psi} \{\mathbb{E}_Q[\ell(\theta; x, y)] - \lambda W_c(Q, P)\} \\ &= \mathbb{E}_{\tau_i \sim G} \mathbb{E}_P \big[\sup_{\xi \in \Xi_i} (\ell(\theta; \xi) - \lambda c_\theta(\xi, (x, y)))\big]. \end{aligned} \tag{3.11}$$

where $\mathcal{Q}_\Psi$ is a set of distributions induced by M semantics transformations parameterized by ψ, $\Xi_i = \{(x', y) | x' = \tau_i(x; \omega_i), \xi \in \Xi_0, \omega_i \subset \psi\}$, and $\Xi_0 \subseteq \Xi$ is the support of P.

The proof of Proposition 3.1 (see Appendix A.1) includes taking the dual reformulation of Equation (3.10) and considering a semantics-induced distribution Q as a mixture of M distributions. We observe that the objective in (3.1) actually minimizes the empirical version of (3.11) with P and c_θ being replaced by $\mathcal{D}_S$ and d_θ, respectively.

3.1.4 Experiment

Experimental Settings

Datasets. We use the following three benchmark datasets in the experiments and arrange them in increasing order of difficulty. (1) **Digits** is used for digit classification and contains five datasets: MNIST [50], MNIST-M [104], SVHN [105], SYN [104], and USPS [106]. Each dataset has the same 10 digits ranging from 0 to 9. We use MNIST as the source domain and the other four as the test domains. (2) **PACS** [96] is a collection of four domains, namely, Art, Cartoon, Photo and Sketch. The four domains share seven common object categories and differ in the styles of their images. We use one domain as the source domain and the other three as the unseen target domains. (3) **DomainNet** [107] is a large-scale dataset which has 345 object classes and contains six domains, namely Real, Infograph, Clipart, Painting, Quickdraw, and Sketch. We use Real as the source domain and the remaining five as the test domains. This is the most challenging dataset in our experiments due to the large number of classes and the high variability of domains.

AdvST implementations. We used 12 standard augmentations commonly used in image transformations (Table 3.1), such as Rotate and Translate, to construct semantics transformations. Most augmentation functions have specific learnable parameters controlling the magnitude of the transformations. We designed a semantics transformation as a composition of at most $L_{\max} = 3$ standard augmentations since more augmentations bring marginal gains. We used the differentiable library [108] to implement these transformations. We denote our method as **AdvST** when $\epsilon = 0$ in Equation (3.3) and **AdvST-ME** when $\epsilon > 0$.

Standard transformations	Description	Number of Parameters
HSV	Perturb in the HSV color space	3
Contrast	Perturb the contrast of an image	1
Invert	Invert pixel values at a given threshold	1
Sharpness	Perturb the sharpness of an image	1
Shear	Shear an image in horizontal and vertial directions	2
Translate	Move an image in horizontal and vertial directions	2
Rotate	Rotate an image	1
Scale	Change the size of an image	1
Solarize	Reverse the tone of an image	1
Equalize	Improve global contrast of an image via equalization	None
Posterize	Reduce the number of bits for each color channel	None
Cutout	Produce occlusions in an image	None

Table 3.1: Standard data augmentations used in experiments.

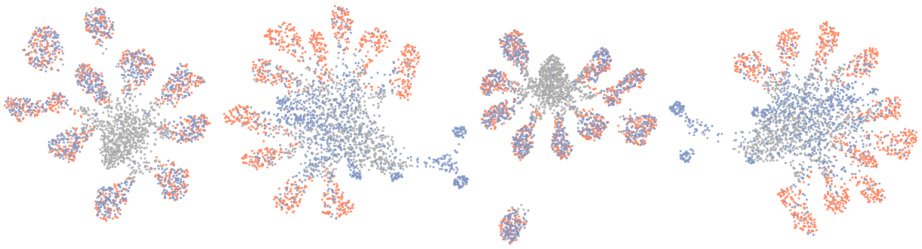
Config.	Semantics	Contrastive	Entropy	Avg.
1				59.3±1.5
2	✓			77.0±0.4
3	✓	✓		77.8±0.2
4	✓		✓	78.8±0.2
5	✓	✓	✓	80.0±0.4

Table 3.2: Ablation study on the Digits dataset. We report average classification accuracy over the four target domains.

Ablation Studies

We conducted ablation studies on AdvST-ME using the Digits dataset. We evaluated how semantics transformations (Semantics), the contrastive regularizer (Contrastive), and the entropy regularizer (Entropy) affect the average SDG performance. We observe from Table 3.2 that semantics transformations can significantly boost the average classification accuracy by 17.7% (Configurations 1 and 2). The contrastive and entropy regularizers can further boost the performance of Configuration 1 by 0.8% (Configuration 3) and 1.8% (Configuration 4), respectively. Our method (Configuration 5) achieves the highest average classification accuracy with all three components.

We further compared the coverage of generated samples on target domain data between our methods, AdvST and AdvST-ME, and their pixel-level counterparts, ADA [51] and ME-ADA [52], which directly generate images in the pixel space. We visualized how the samples generated by ADA, ME-ADA, and our methods distribute in the embedding space in Figure 3.1. We color the samples from the source domain MNIST orange and the samples from the four target domains gray.



(a) ADA (b) AdvST (c) ME-ADA (d) AdvST-ME

Figure 3.1: Visualization of how samples from the source domain, target domains, and synthetic domains distribute in the embedding space. We compare AdvST and AdvST-ME with their non-semantics counterparts ADA and ME-ADA.

We give details for obtaining the figure in Appendix A.1. From Figure 3.1(a) and (c), we observe that most of the synthetic samples distribute very close to the source domain data and have little coverage on the target domains. In contrast, the synthetic samples in Figure 3.1(b) and (d) deviate from the source domain and have broad coverage on the target domains.

We provide analyses on the sensitivity of λ and the effect of different semantics transformations in Appendix A.1.

Comparison on Digits

Baselines. We included ADA, ME-ADA, and the following methods for comparison: ERM, which trains a model only using the standard cross-entropy loss; CCSA [109], which aligns samples from different domains to improve generalization; d-SNE [110], which minimizes the maximum distance between sample pairs of the same class and maximizes the minimum distance among sample pairs of different categories; JiGen [111], which is a multi-task learning method that combines the target recognition task and the Jigsaw classification task; M-ADA [55], which uses generative models and meta-learning [22, 112, 113] to improve ADA; AutoAug [114] and RandAug [20], which augment data based on the searched augmentation policies; RSDA [115], which randomly searches image transformations to train a robust model; and PDEN [57] and L2D [56], which use generative models for data augmentation.

Results. We observe from Table 3.3 that our methods, AdvST and AdvST-ME, significantly improve the performance of the pixel-level adversarial data augmentations, ADA and ME-ADA, across the four target domains and achieve a maximum gain of 25.5% in average classification accuracy. Regarding per-domain performance, our methods achieve the best performance on all the target domains except the MNIST-M domain. It is common to observe that a method does not

Method	SVHN	MNIST-M	SYN	USPS	Avg.
ERM	27.8	52.7	39.7	76.9	49.3
CCSA	25.9	49.3	37.3	83.7	49.1
d-SNE	26.2	51.0	37.8	93.2	52.1
JiGen	33.8	57.8	43.8	77.2	53.1
ADA	35.5	60.4	45.3	77.3	54.6
ME-ADA	42.6	63.3	50.4	81.0	59.3
M-ADA	42.6	67.9	49.0	78.5	59.5
AutoAug	45.2	60.5	64.5	80.6	62.7
RandAug	54.8	74.0	59.6	77.3	66.4
RSDA	47.7	81.5	62.0	83.1	68.5
L2D	62.9	**87.3**	63.7	84.0	74.5
PDEN	62.2	82.2	69.4	85.3	74.8
AdvST	**67.5±0.7**	79.8±0.7	**78.1±0.9**	94.8±0.4	**80.1±0.5**
AdvST-ME	66.7±1.0	80.0±0.5	77.9±0.7	**95.4±0.4**	80.0±0.4

Table 3.3: Classification accuracy (%) results on the four target domains SVHN, MNIST-M, SYN, and USPS, with MNIST as the source domain. Best results are in bold font.

Target	MixUp	CutOut	ADA	ME-ADA	AugMix	RandAug	ACVC	L2D	AdvST	AdvST-ME
Art	52.8	59.8	58.0	60.7	63.9	67.8	67.8	67.6	**69.2±1.4**	67.0±1.1
Cartoon	17.0	21.6	25.3	28.5	27.7	28.9	30.3	42.6	**55.3±2.0**	53.2±1.1
Sketch	23.2	28.8	30.1	29.6	30.9	37.0	46.4	47.1	**67.7±1.5**	67.2±2.2
Avg.	31.0	36.7	37.8	39.6	40.8	44.6	48.2	52.5	**64.1±0.4**	62.5±0.8

Table 3.4: Classification accuracy (%) comparison on the PACS dataset. Best results are in bold font.

perform the best on all the target domains. For example, PDEN performs better than L2D on SYN but worse than L2D on MNIST-M. We reason that the knowledge that helps a model generalize in one domain does not necessarily work for the other. To demonstrate this, we trained models on one of the five domains and evaluated their generalization performance on each of the remaining domains. From the accuracy heatmap in Figure 3.2, we see that the learned knowledge for MNIST-M cannot transfer well to SYN and vice versa, which explains the performance tradeoff between

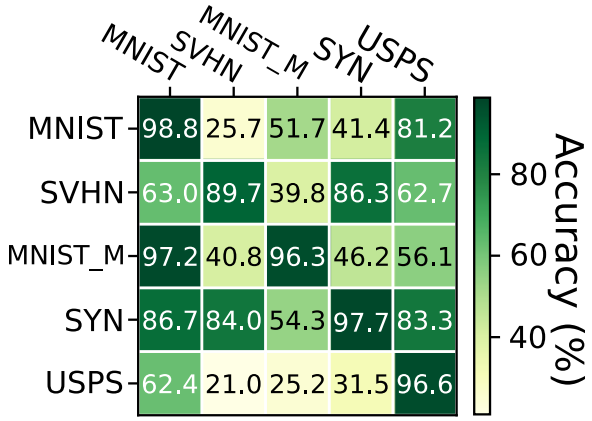


Figure 3.2: Accuracy heatmap for models trained individually on the five domains from the Digit dataset using ERM.

Target	MixUp	CutOut	CutMix	ADA	ME-ADA	RandAug	AugMix	ACVC	AdvST	AdvST-ME
Painting	38.6	38.3	38.3	38.2	39.0	41.3	40.8	41.3	42.3±0.1	**42.4±0.2**
Infograph	13.9	13.7	13.5	13.8	14.0	13.6	13.9	12.9	14.8±0.2	**14.9±0.1**
Clipart	38.0	38.4	38.7	40.2	41.0	41.1	41.7	**42.8**	41.5±0.4	41.7±0.2
Sketch	26.0	26.2	26.9	24.8	25.3	30.4	29.8	30.9	30.8±0.3	**31.0±0.2**
Quickdraw	3.7	3.7	3.6	4.3	4.3	5.3	6.3	**6.6**	5.9±0.2	6.1±0.2
Avg.	24.0	24.1	24.2	24.3	24.7	26.3	26.5	26.9	27.1±0.2	**27.2±0.1**

Table 3.5: Classification accuracy (%) comparison on the DomainNet dataset. Best results are in bold font.

MNIST-M and SYN when comparing AdvST with AdvST-ME or AdvST with L2D. Nevertheless, our methods achieve the best average classification accuracy over the four target domains among all the methods.

Comparison on PACS

Baselines. We compared our methods AdvST and AdvST-ME with ADA, ME-ADA, MixUp [116], CutOut [17], CutMix [117], RandAug [20], AugMix [18], and L2D [56]. We also included ACVC [118], which applies attention consistency to learning from augmented samples.

Results. We used Photo as the source domain and evaluated models on the Art, Cartoon, and Sketch domains. Generalizing raw images to artificial images is the most challenging SDG setting in the PACS dataset since the domain shift between the source and target domains is substantial. Results in Table 3.4 show that our methods significantly improve the performance of pixel-level adversarial data augmentations, ADA and ME-ADA, in all three domains. Moreover, our method AdvST performs the best on the three target domains and achieves the best average classification accuracy over the three domains. AdvST-ME performs the second best in this setting, indicating that maximizing output entropy to further encourage generating diverse samples does not help the generalization from a natural domain to an artificial one.

Comparison on DomainNet

Baselines. We compared our methods AdvST and AdvST-ME with ADA, ME-ADA, MixUp [116], CutOut [17], CutMix [117], RandAug [20], and AugMix [18].

Results. Table 3.5 shows our results in the most challenging SDG setting, DomainNet, which has 345 classes and significant domain shifts from the source domain, such as Real to Infograph and Real to Quickdraw. Under this challenging setting, our methods outperforms pixel-level adversarial data augmentations, ADA and ME-ADA, and complex data augmentations, such as RandAug and AugMix.

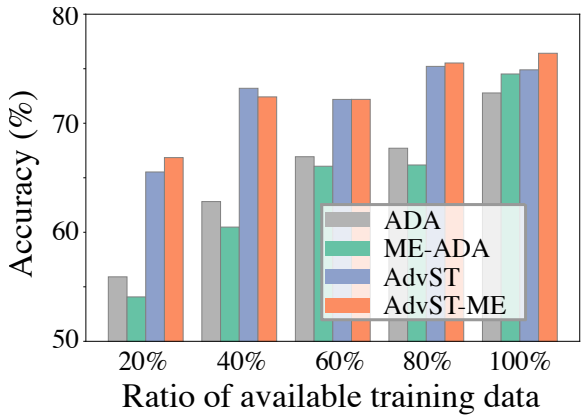


Figure 3.3: Average classification accuracy under different ratios of available training data.

Learning with Limited Source Data

We further demonstrated the utility of our methods by evaluating the average classification accuracy of our methods on target domains with limited training data. We used the Art dataset from PACS as the source domain and the remaining three datasets in PACS as the target domains. We used partial training data of the Art domain and reported the average classification accuracy over the three target domains in Figure 3.3. We observe that under different ratios of available training data, our methods, AdvST and AdvST-ME, consistently outperform ADA and ME-ADA, respectively. The gains are significant when the ratio is small, demonstrating the effectiveness of our method when there is a lack of available training data.

3.1.5 Conclusion

We revisited data augmentation for SDG and focused on leveraging the domain knowledge in standard data augmentations. We conceptualized a composition of several standard data augmentations as a semantics transformation with learnable parameters and proposed AdvST, an adversarial learning framework that aims to train a robust model with samples with diverse spurious attributes generated by semantics transformations. We theoretically showed that AdvST optimizes a DRO objective with semantics-induced distributions. Although built on standard data augmentations, AdvST is surprisingly competitive. It achieves the best average domain generalization performance on three benchmark datasets and is effective with limited source data.

3.2 Learning to Learn Task Transformations for Improved Few-Shot Classification

3.2.1 Introduction

Learning new concepts with a small amount of data is recognized as a hallmark of human intelligence [22]. In contrast, modern deep neural networks typically are trained with a large amount of labeled data. Meta-learning, which learns a meta-model that can quickly generalize to new concepts with a few labeled examples and adaptation steps, has recently attracted tremendous interest [12, 13, 14, 15, 16]. A widely used test bed for meta-learning algorithms is few-shot image classification where classifications are performed on new image categories after learning a few labeled training examples for each category.

In few-shot image classification, existing meta-learning algorithms [12, 13, 14, 15, 16] often adopt data augmentation methods in their implementations for performance improvement. These data augmentation methods produce samples with diverse spurious attributes to facilitate the learning of robust features. However, these methods are often manually designed as a sequence of fixed image transformation functions, ignoring the training dynamics of meta-learning. As the training progresses, the meta-model could gradually memorize the seen tasks. Despite that a fixed augmentation strategy is applied, the augmented tasks along with the spurious attributes could be memorized by the meta-model at a certain training stage, and thus the meta-model may lack the ability to generalize to new tasks. We need to provide harder tasks with images containing more diverse spurious attributes than existing ones. However, this is not possible with fixed augmentation strategies.

Moreover, existing data augmentation methods are often designed to be agnostic to various meta-learning settings specified by meta-model architectures and meta-learning algorithms. Hence, the difference between various meta-learning settings is ignored, resulting in tasks that are suboptimal for certain meta-learning settings. For example, if the same augmented tasks are provided to a deep and shallow meta-models, the deep one may simply remember the provided tasks, leading to severe overfitting. Similarly, as will be shown later, each algorithm also has its own level of task difficulty at which the algorithm is most effective in training a meta-model that generalizes well to unseen tasks.

To address the above challenges, we aim to construct tasks with task difficulty levels optimized for a certain meta-learning setting and at each training stage. Direct optimization of the task construction is infeasible in practice since we need to search all possible combinations of examples in a

training set to obtain optimal tasks. To circumvent this, instead of constructing tasks from scratch, we propose to learn to transform given training tasks to get transformed ones with optimized task difficulty. The task difficulty depends on the spurious attributes synthesized by the learned transformations. From the information theoretic perspective [119], the information shared between the meta-model input and the corresponding output is reduced when the input goes through additional transformations. By transforming an input task, we control the amount of information flowing from the input to the output. With less information provided to the meta-model, it becomes more challenging to learn new concepts from the input task. Therefore, learning to transform tasks provides a feasible way to provide tasks with optimized task difficulties during meta-training.

Inspired by the above idea, we propose to add a task transformation layer between a training task and a meta-model. The layer transforms a training task by applying learnable transformation functions to all the examples in the task. We design a task transformation function as a sequence of differentiable image operations with learnable transformation magnitudes. This design has two benefits: 1) the image operations, such as changing brightness and rotating an image, are label-preserving, and can avoid the change of labels and unwanted biases in the transformed task caused by an arbitrary task transformation function which may distort the semantics of images in the task; 2) differentiable image operations allow us to back propagate through a specific meta-learning setting, enabling efficient optimization of the task transformation functions. To add the flexibility in how a training task can be transformed, the task transformation layer is designed as a stochastic function which follows a learnable distribution containing a set of transformation functions with learnable probabilities. During meta-training, the layer is jointly optimized with the meta-model, allowing the transformed tasks to co-adapt with the meta-model.

We summarize our contributions as follows:

- We propose a new meta-learning framework with a task transformation layer that mediates the discrepancy between training tasks and meta-learning settings specified by meta-model architecture and meta-learning algorithms, and controls task difficulty in accommodation to training dynamics.

- We design the task transformation layer as a differentiable and stochastic function for efficient optimization. As a benefit of such design choice, we get a new metric indicating the overall task difficulty required for training on a specific dataset in a certain meta-learning setting.

- We show that our method can consistently improve the few-shot generalization performance of various meta-models trained with different meta-learning algorithms, meta-model architectures on two benchmark datasets.

3.2.2 Preliminaries

The goal of meta-learning is to learn a meta-model that can quickly generalize to unseen but related tasks with only a handful of labeled examples per task. The meta-model is meta-trained on a sequence of training tasks under a meta-learning algorithm. A meta-learning algorithm $\mathcal{E}$ can be specified by the inner loop learning algorithm $\mathcal{A}$ and the outer loop learning algorithm $\mathcal{B}$, i.e., $\mathcal{E} = \{\mathcal{A}, \mathcal{B}\}$. A typical meta-learning setting can be specified with the meta-learning algorithm $\mathcal{E}$ and the meta-model f_θ parameterized by θ. The meta-learning framework under this setting is formulated as:

$$f_{\theta^*} = \mathcal{B}\Big(\mathbb{E}_{\mathcal{T}\sim p(\mathcal{T})}\mathcal{L}(\mathcal{T})\Big) \tag{3.12}$$

$$\text{s.t.} \quad \mathcal{L}(\mathcal{T}) = \frac{1}{n_Q}\sum_{(x,y)\in\mathcal{Q}} \ell(f_{\hat{\theta}}(x), y), \tag{3.13}$$

$$f_{\hat{\theta}} = \mathcal{A}(f_\theta, \mathcal{S}), \tag{3.14}$$

where $\mathcal{T} = \{\mathcal{S}, \mathcal{Q}\}$ is a training task and consists of a support set $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{n_S}$ and a query set $\mathcal{Q} = \{(x_j, y_j)\}_{j=1}^{n_Q}$ containing n_S and n_Q sample-label pairs, respectively; $\mathcal{A}$ is the inner loop algorithm, which fine-tunes the meta-model f_θ with the training data in $\mathcal{S}$ from task $\mathcal{T}$, and outputs the adapted model $f_{\hat{\theta}} = \mathcal{A}(f_\theta, \mathcal{S})$; and $\mathcal{B}$ is the outer loop algorithm that outputs an optimized meta-model f_{θ^*} considering all the training tasks from the task distribution $p(\mathcal{T})$. Typically, $p(\mathcal{T})$ describes the distribution of training tasks that are constructed randomly from a training dataset. To construct an N-way K-shot task, we first randomly sample N image categories from the training dataset, and then randomly sample K images for each of the N categories for the support set $\mathcal{S}$. For the query set $\mathcal{Q}$, we sample K_q images for each category so that $n_Q = N \cdot K_q$. The task loss $\mathcal{L}(\mathcal{T})$ is calculated on the query set $\mathcal{Q}$ with the adapted model $f_{\hat{\theta}}$ and the cross-entropy loss function $\ell(\cdot, \cdot)$.

The above formulation subsumes a wide range of meta-learning algorithms. For gradient-based algorithms, both $\mathcal{A}$ and $\mathcal{B}$ are certain kinds of optimizers. For example, in MAML [22], $\mathcal{A}$ and $\mathcal{B}$ are designed as a stochastic gradient descent optimizer and an Adam optimizer [120], respectively. For metric-based algorithms, $\mathcal{B}$ is commonly designed as an optimizer, such as SGD with momentum [121], and $\mathcal{A}$ is designed as a classification method with a certain metric, such as ridge regression in

R2D2 [13], and an SVM classifier in MetaOptNet [14]. Hence, our analysis based on the general meta-learning framework can be applied to many meta-learning algorithms that follow this framework.

Problem Definition. From Equation (3.12) to Equation (3.14), we observe that under a given meta-learning setting specified by the meta-learning algorithm $\mathcal{E}$ and the meta-model f_θ with a certain architecture, the only factor that affects the performance of the optimized meta-model f_{θ^*} is the task distribution $p(\mathcal{T})$. In practice, tasks are randomly sampled from the training set for different meta-learning settings, and $p(\mathcal{T})$ represents the random task distribution. However, this ignores the difference between various meta-learning settings as they require different levels of task difficulties. Moreover, the training dynamics is also not considered when the meta-model parameter θ is continuously updated during training with a fixed $p(\mathcal{T})$. To achieve improved few-shot performance under a specific meta-learning setting, it is critical to construct tasks with optimized task difficulties that fit specific meta-learning algorithms and meta-models during training.

3.2.3 Methodology

To address the mismatch between $p(\mathcal{T})$, $\mathcal{E}$, and f_θ, an ideal approach is to learn to construct tasks with optimized task difficulties for the considered meta-learning setting. However, this approach is challenging in practice since we need to search combinatorially in the whole training dataset which is computationally prohibitive. To circumvent this, we propose to dynamically control task difficulty by introducing a learnable task transformation layer which can be jointly optimized with the meta-model during training.

Task Transformation Layer

We add a task transformation layer between an input task and a meta-model to control the task difficulty. The layer perturbs the task with learnable transformation functions to optimize the task difficulty at a certain training stage in a certain meta-learning setting. Based on the data processing theorem from information theory [119], additional transformations to the input reduce the information shared between the meta-model input and the corresponding output. Therefore, we optimize a task difficulty by controlling the amount of information flowing from the input to the output via transforming an input task. With less information provided to the meta-model, it becomes more challenging to learn new concepts from the input task.

To add the flexibility in how a task can be transformed by the task transformation layer, we design the layer as a stochastic function which samples a task transformation function τ for each task from

the distribution $p_\tau(\tau;\omega)$ parameterized by ω. Intuitively, using multiple task transformations at a time can create many diverse distributions to increase the chance of producing $\tilde{p}(\mathcal{T}')$ optimal for a given meta-learning setting. The goal of the layer is to transform a population of training tasks such that they follow a task distribution $\tilde{p}(\mathcal{T}')$ optimized at a certain training stage in a specific meta-learning setting. In practice, since $\tilde{p}(\mathcal{T}') = \int_{\tau,\mathcal{T}} p(\mathcal{T}'|\mathcal{T},\tau)p_\tau(\tau;\omega)p(\mathcal{T})$, the task transformation layer first samples a τ from $p_\tau(\tau;\omega)$ and then transforms a training task $\mathcal{T}$ from $p(\mathcal{T})$ via $p(\mathcal{T}'|\mathcal{T},\tau)$. We design $p(\mathcal{T}'|\mathcal{T},\tau)$ to be a distribution over $\mathcal{T}'$ calculated as follows,

$$\mathcal{T}' = \tau(\mathcal{T}) = \{x'|x' = \tau(x), x \in \mathcal{T}\}, \tag{3.15}$$

where x' and x have the same dimension, and the sampled task transformation τ is either a deterministic mapping or a stochastic one. In other words, $\mathcal{T}'$ is obtained by transforming all the samples in task $\mathcal{T}$ with the same function τ. Although τ is applied sample-wise, we still recognize it as a task transformation function since it is learned from a population of training tasks. In practice, our design enjoys fast convergence and small memory consumption. In addition to the above method, we could transform all the samples in a task with a single transformation, transform each sample in a task with an individual transformation, or other methods in between. We leave more sophisticated designs of transforming a task as our future work.

Task Transformation Functions

The task transformation function τ used by the task transformation layer needs to satisfy certain constraints. With an arbitrary τ, the semantics of images in a task may be distorted, leading to change of labels and unwanted biases in the transformed task. We also require τ sampled from $p_\tau(\tau;\omega)$ to be differentiable so that ω can be directly optimized. Inspired from differentiable data augmentation [74], we design a task transformation function as a sequence of differentiable and label-preserving image operations, such as changing brightness and rotating an image, with learnable transformation magnitudes.

Specifically, given L as the length of a task transformation τ, we have $\tau(\cdot) = O_L \circ \cdots \circ O_1(\cdot)$, where $\circ$ denotes function composition, and $O_1, \ldots, O_L$ are differentiable image operations. We denote each image operation as $O(\cdot) = g(\cdot; m)$, where $g \in \mathcal{G}$ is an image operation in the set of candidate image operations $\mathcal{G}$, $m \in \mathcal{M}$ is a learnable transformation magnitude for g, and $\mathcal{M}$ is the set of all possible magnitudes. In general, each image operation has its own range of transformation magnitude. For example, a rotation operation has the magnitude in degrees ranging from $-180°$

to 180°, while a contrast operation has the magnitude in pixel intensity ranging from 0 to 1. We normalize these magnitudes in different ranges to the same interval to stabilize the learning of task transformations. We set $\mathcal{M} = [0, 1]$ such that given $m \in \mathcal{M}$, for a function g with the transformation range $[m^g_{\min}, m^g_{\max}]$, the actual magnitude is $m \cdot (m^g_{\max} - m^g_{\min}) + m^g_{\min}$. In other words, when $m = 0$, $g(x; m)$ gives the original input x; when $m = 1$, $g(x; m)$ gives the most transformed image. For example, $O(\cdot)$ could be a rotate-90° operation with g being the rotate function and $m = 0.75$ ($m \in [0, 1]$ corresponds to a degree in $[-180°, 180°]$).

Sampling Strategy

The task transformation layer samples a task transformation function τ from the distribution $p_\tau(\tau; \omega)$ which can be factored as the product of L conditional probability distributions, i.e.,

$$p_\tau(\tau; \omega) = \prod_{l=1}^{L} p(O_l | O_1, \ldots, O_{l-1}), \tag{3.16}$$

where L is a hyperparameter denoting the length of τ, and $p(O_l|O_1, \ldots, O_{l-1})$ is a conditional distribution supported on $|\mathcal{G}|$ operations with learnable magnitudes, where $|\cdot|$ denotes the size of a set. The previous operations $O_1, \ldots, O_{l-1}$ are sampled from the support of $p(O_k|O_1, \ldots, O_{k-1})$ with $1 \leq k < l$. A task transformation function with length L is constructed by sampling an operation O_l from $p(O_l|O_1, \ldots, O_{l-1})$ with l staring from 1 to L. In summary, we design $p_\tau(\tau; \omega)$ to be a learnable distribution over the $|\mathcal{G}|^L$ task transformation functions with ω including all learnable magnitudes and probabilities of image operations.

It is straightforward to directly sample O_l from $p(O_l|O_1, \ldots, O_{l-1})$, but this sampling process is not differentiable with respect to the parameters in $p(O_l|O_1, \ldots, O_{l-1})$ which is a categorical distribution by design. To address this, we apply a differentiable relaxation on $p(O_l = O|O_1, \ldots, O_{l-1})$ via Gumbel-Softmax reparameterization [122]. Concretely, in the forward pass, we select $O_l = \arg\max_{O \in \mathcal{O}^L_l}(p_O + r_O)$, where $p_O = p(O_l = O|O_1, \ldots, O_{l-1})$, $\mathcal{O}^L_l$ is the support of $p(O_l|O_1, \ldots, O_{l-1})$ and by design $|\mathcal{O}^L_l| = |\mathcal{G}|$, $r_O = -\log(-\log(u))$, and $u \sim \text{Uniform}(0, 1)$. In the backward pass, we have all the operations involved as $O_l(\cdot) = \sum_{O \in \mathcal{O}^L_l} s_O O(\cdot)$, where s_O is calculated as

$$s_O = \frac{\exp((p_O + r_O)/\epsilon)}{\sum_{O \in \mathcal{O}^L_l} \exp((p_O + r_O)/\epsilon)}, \forall O \in \mathcal{O}^L_l, \tag{3.17}$$

where ϵ is the temperature of the softmax function, and it controls the sampling uncertainty: a larger ϵ will generate a task distribution with more randomly sampled task transformation functions. To sample diverse task transformation functions during training, we set ϵ to a large number, e.g. $\epsilon = 20$.

Discussion. To sample the l-th operation for a task transformation function, we need $O(|\mathcal{G}|^l)$ parameters to specify $p(O_l|O_1, \ldots, O_{l-1})$, and a total of $O(|\mathcal{G}|^L)$ parameters to determine the whole distribution $p_\tau(\tau; \omega)$. Although the additional parameters needed in our method is exponential with respect to L, the value of L is usually very small in practice. We find that using $L \leq 5$ is sufficient to achieve good performance. Moreover, $|\mathcal{G}|$ is usually in the order of tens or less. Hence, the number of additional learnable parameters induced by our method is negligible when it is compared with that of a meta-model.

Learning Objective

Our proposed new meta-learning framework, termed *learning to learn task transformations (L2TT)*, optimizes a training task distribution by transforming randomly constructed tasks with a set of learnable task transformation functions. Given $p_\tau(\tau; \omega)$ and the distribution of randomly constructed tasks $p(\mathcal{T})$, the learning objective of our proposed method L2TT is:

$$f_{\theta^*}, \omega^* = \mathcal{B}\Big(\mathbb{E}_{\mathcal{T}' \sim \tilde{p}(\mathcal{T}')} \mathcal{L}(\mathcal{T}')\Big) \tag{3.18}$$

$$\text{s.t.} \quad \tilde{p}(\mathcal{T}') = \int_\tau \int_{\mathcal{T}} p(\mathcal{T}'|\mathcal{T}, \tau) p(\mathcal{T}) p_\tau(\tau; \omega), \tag{3.19}$$

$$\mathcal{L}(\mathcal{T}') = \frac{1}{n_Q} \sum_{(x,y) \in \mathcal{Q}'} \ell(f_{\hat{\theta}}(x), y), \tag{3.20}$$

$$f_{\hat{\theta}} = \mathcal{A}(f_\theta, \mathcal{S}'), \tag{3.21}$$

where $p(\mathcal{T}'|\mathcal{T}, \tau)$ denotes the distribution of the transformed task $\mathcal{T}'$ given $\mathcal{T}$ and τ. By providing the transformed support set $\mathcal{S}'$ and the query set $\mathcal{Q}'$ in $\mathcal{T}'$ to the inner and outer loop learning procedures, respectively, we naturally embed task transformations in the learning to learn framework and can jointly optimize them with the meta-model. Equation (3.19) shows the new task distribution $\tilde{p}(\mathcal{T}')$ depends on $p_\tau(\tau; \omega)$. By optimizing ω, we equivalently optimize the task distribution such that it is well tuned with the underlying model architecture and the meta-learning algorithm.

One of the learning outcomes is the optimized task transformation function distribution $p(\tau; \omega^*)$. If we average all the learned magnitudes with the corresponding probabilities over all the transformation functions, we obtain a new metric called the average task transformation magnitude (AvgM-

TT). This metric indicates the overall task difficulty required by training a given meta-model with the provided meta-learning algorithm and the training dataset. As shown in the experiments, AvgM-TT provides a quantitative way of comparing between different meta-learning algorithms, meta-model architectures, and training datasets.

3.2.4 Experiments

We conduct experiments to answer the following research questions (RQs). **RQ1**: Is our L2TT meta-learning framework effective for different meta-learning algorithms and meta-model architectures? **RQ2**: How our method compares with other methods that can also change images in a task? **RQ3**: How the learned task transformations differ for various combinations of meta-learning algorithms and meta-model architectures? We also show the results of different designs of task transformations in ablation study.

Experimental Setup

Datasets. **CIFAR-FS** [13] is a lightweight yet challenging few-shot image classification benchmark, and it allows fast prototyping. The dataset consists of all 100 classes from CIFAR-100 [123], and the classes are split into 64, 16, and 20 for meta-training, meta-validation, and meta-testing respectively. There are 600 images of size 32×32 in each class. **miniImageNet** [124] is a challenging few-shot classification benchmark without demanding computational resources. It contains 100 classes with each having 600 images. Each image is down sampled to have the size of 84×84. We follow the dataset split from [124] and divide the dataset into three non-overlapping sets of classes, forming meta-training, meta-validation, and meta-testing sets. The class numbers in the three sets are 64, 16, and 20, respectively.

Baseline Methods. For **RQ1** and **RQ3**, we select four meta-learning algorithms, including three metric-based meta-learning algorithms: R2D2 [13], MetaOptNet [14], and ProtoNet [125], and one gradient-based meta-learning algorithm MAML [22]. We select ResNet12 and CNN64 (4 64-filter convolutional layers) as the meta-model architectures. We follow the implementations in [27] and give the implementation details in Appendix A.2. For **RQ2**, the most related methods for comparison are data augmentation methods. We only consider augmentation in the input pixel space for fair comparison since the image operations that we use all work in this space. Specifically, we include SimpleAug, AutoAugment [71], and MetaDA [27] in the experiments. SimpleAug uses the following transformation `RandomCrop`→ `ColorJitter`→`RandomHorizontalFlip` to transform each sample in a

Architecture	Meta-learning algorithm	Meta-learning framework	CIFAR-FS		miniImageNet	
			1-shot	5-shot	1-shot	5-shot
ResNet-12	R2D2	Standard	72.53±0.11	84.16±0.08	60.59±0.10	75.90±0.08
		L2TT	**75.96±0.11**	**86.72±0.08**	**63.56±0.11**	**78.25±0.08**
ResNet-12	ProtoNet	Standard	70.42±0.12	83.25±0.08	58.28±0.10	75.82±0.08
		L2TT	**73.63±0.11**	**85.76±0.08**	**60.82±0.11**	**78.16±0.08**
ResNet-12	MetaOptNet	Standard	71.56±0.12	84.03±0.08	60.51±0.10	76.34±0.08
		L2TT	**74.34±0.11**	**86.19±0.08**	**62.50±0.10**	**78.17±0.08**
CNN64	ProtoNet	Standard	61.10±0.12	79.28±0.09	47.43±0.10	70.55±0.08
		L2TT	**63.73±0.11**	**81.06±0.09**	**49.12±0.10**	**71.57±0.08**
CNN64	MAML	Standard	55.81±0.11	75.50±0.09	42.38±0.10	64.64±0.09
		L2TT	**58.50±0.11**	**76.16±0.09**	**47.70±0.10**	**64.75±0.09**

Table 3.6: Performance comparison between our proposed L2TT and the Standard meta-learning frameworks under different meta-learning algorithms and meta-model architectures on the CIFAR-FS and miniImageNet datasets.

task, and it is the default augmentation method in many meta-learning algorithms [12, 13, 14, 15, 16]. AutoAugment contains sets of augmentation policies optimized for specific datasets, e.g., CIFAR-100 and ImageNet. Each policy is a chain of image operations. MetaDA [27] manually designed a set of augmentation policies which augment a task by transforming the samples in the support set, in the query set, or in the whole task. Since MetaDA adopts CutMix [126] to change the query samples (called QC), the training objective changes from predicting the labels to additionally predicting the areas of the two image patches in a mixed image created by Cutmix. For fair comparison, we apply QC to SimpleAug, AutoAugment, and our method and get SimpleAug-QC, AutoAugment-QC, L2TT-QC, respectively.

Results

For **RQ1**, we evaluate our proposed L2TT framework on the CIFAR-FS and miniImageNet datasets in five meta-learning settings specified by meta-model architectures and meta-learning algorithms. The Standard meta-learning framework from Equation (3.12) to Equation (3.14) is the default meta-learning framework for the five meta-learning settings, and we use SimpleAug as the data augmentation method. The 5-way few-shot classification accuracies with 95% confidence intervals are reported in Table 3.6, and the best results are highlighted in bold fonts. Compared with the Standard framework, our proposed L2TT framework achieves consistent performance improvement on the two datasets across all the five meta-learning settings with different meta-model architecture and meta-learning algorithms. This universal improvement verifies the effectiveness of our method in different meta-learning settings. Moreover, it also implies that the mismatch between training tasks, meta-learning algorithms, and meta-model architectures, is prevalent in many meta-learning settings. By learning to learn task transformations, we can optimize the training task distribution

Architecture	Meta-learning algorithm	Data augmentation method	CIFAR-FS		miniImageNet	
			1-shot	5-shot	1-shot	5-shot
ResNet-12	R2D2	SimpleAug-QC	76.01±0.11	86.21±0.08	64.24±0.11	78.42±0.08
		AutoAugment-QC	76.36±0.11	86.86±0.08	64.93±0.11	78.82±0.08
		MetaDA	76.00±0.11	87.25±0.08	63.78±0.10	78.96±0.08
		L2TT-QC	**77.40±0.11**	**87.76±0.08**	**65.82±0.10**	**80.36±0.08**
ResNet-12	ProtoNet	SimpleAug-QC	75.04±0.11	86.29±0.08	60.69±0.11	76.43±0.09
		AutoAugment-QC	75.51±0.11	86.57±0.08	61.94±0.11	77.41±0.09
		MetaDA	75.00±0.11	87.06±0.08	60.63±0.11	78.13±0.08
		L2TT-QC	**76.52±0.11**	**87.12±0.08**	**63.07±0.11**	**78.79±0.08**
ResNet-12	MetaOptNet	SimpleAug-QC	73.20±0.12	86.29±0.08	64.71±0.11	79.03±0.08
		AutoAugment-QC	73.37± 0.11	86.45±0.08	65.19±0.10	79.47±0.08
		MetaDA	73.97±0.11	86.98±0.08	64.00±0.10	79.43±0.08
		L2TT-QC	**76.65±0.11**	**87.84±0.08**	**65.60±0.10**	**80.99±0.08**
CNN64	ProtoNet	SimpleAug-QC	63.26±0.12	80.63±0.08	49.95±0.10	71.39±0.08
		AutoAugment-QC	62.74±0.12	79.87±0.09	49.90±0.10	69.53±0.09
		MetaDA	63.07±0.12	80.86±0.08	49.67± 0.10	71.39±0.08
		L2TT-QC	**63.75±0.11**	**81.41±0.08**	**50.67±0.10**	**71.76±0.08**
CNN64	MAML	SimpleAug-QC	58.80±0.12	76.67±0.10	48.57±0.10	66.03±0.09
		AutoAugment-QC	55.82±0.12	72.16±0.10	47.08±0.10	63.59±0.09
		MetaDA	60.20±0.12	77.36±0.09	47.51±0.10	66.79±0.09
		L2TT-QC	**61.20±0.12**	**78.12±0.09**	**48.91±0.10**	**66.90±0.08**

Table 3.7: Performance comparison between various data augmentation methods for different meta-learning algorithms and meta-model architectures on the CIFAR-FS and miniImageNet datasets.

for a give meta-model architecture and a meta-learning algorithm to provide further performance improvement. The performance gains achieved by our method are larger when we use deeper meta-model architectures, e.g., the gains achieved with ResNet-12 are larger than those with CNN64.

In our method, a task transformation function is designed to have several image operations. This design choice relates our method to the widely used data augmentation methods and gives rise to **RQ2**. Although the goal of our method is different from a typical data augmentation method which aims to generate diverse samples, our method of learning to learn task transformations generates diverse samples like a typical data augmentation method does. We compare our method with three data augmentation methods: SimpleAug-QC, AutoAugment-QC, and MetaDA. For fair comparison, we also include QC in our method and denote the new method as L2TT-QC. Table 3.7 shows the 5-way few-shot classification accuracies with 95% confidence intervals of the five meta-models trained with these methods. We highlight the best results in bold fonts in Table 3.7. From the data augmentation perspective, we observe that L2TT-QC performs the best in all the five meta-learning settings and on the two datasets.

Moreover, among the methods compared in Table 3.7, MetaDA is a competitive method. It is specifically designed for meta-learning and has 9 augmentation policies manually designed with the expert knowledge about meta-learning, such as considering the support-query structure of a task in designing the policies. However, MetaDA does not consider the difference in various meta-learning settings. In contrast, our method learns task transformations which can be automatically optimized

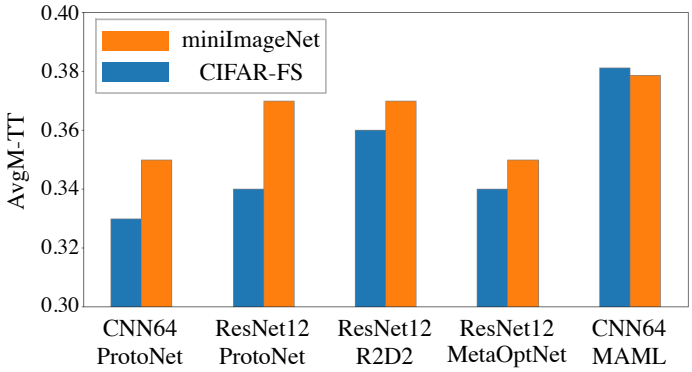


Figure 3.4: AvgM-TTs for different meta-learning algorithms and meta-model architectures on the miniImageNet and CIFAR-FS datasets.

for a specific meta-learning setting. Additionally, MetaDA shows a tendency to overfit to the 5-shot setting since it outperforms AutoAugment-QC in almost all the 5-shot cases but is inferior to AutoAugment-QC in more than half of the 1-shot cases, especially in those with the miniImageNet dataset. Since all the models are meta-trained in the same 5-way 5-shot setting, the results show that MetaDA cannot generalize well to a low-shot setting where the number of labeled samples in the support set of a task is small. Our method does not have the above limitation.

For **RQ3**, we calculate AvgM-TT for the five meta-learning settings and show the results in Figure 3.4. By design, a higher magnitude indicates a more aggressive transformation on the images in a task. Among the first four metric-based settings in Figure 3.4, we observe that the AvgM-TTs for the miniImageNet dataset are higher than those for the CIFAR-FS dataset. This indicates that we generally need "harder" tasks for the larger and complex miniImageNet dataset than those for the smaller and simple CIFAR-FS dataset. However, for the gradient-based setting CNN64-MAML, the two AvgM-TTs on the two datasets are very similar, indicating that MAML is not very sensitive to training data with varying complexities. The two values are also larger than those in the four metric-based settings. This indicates that MAML is easier to suffer from overfitting than the three metric-based algorithms and needs "harder" tasks in training. We also observe that deeper meta-model architectures require "harder" tasks, as shown by the higher AvgM-TTs of ResNet12-ProtoNet than those of CNN64-ProtoNet on respective datasets. Overall, AvgM-TT indicates the overall complexity involving training datasets, model architectures, and meta-learning algorithms.

Ablation Studies

Task transformations are constructed by sampling from a distribution of task transformations. Different design choices affect the overall performance. We study two important hyperparameters L

L	Shot	ϵ		
		10	20	∞
1	1-shot	73.08±0.12	73.73±0.12	73.65±0.12
	5-shot	84.03±0.08	84.39±0.08	84.25±0.08
2	1-shot	75.30±0.11	75.92±0.11	72.65±0.11
	5-shot	86.05±0.08	86.30±0.08	84.87±0.08
3	1-shot	74.75±0.11	**75.96±0.11**	75.03±0.11
	5-shot	85.74±0.08	**86.72±0.08**	85.97±0.08
4	1-shot	75.11±0.11	72.04±0.11	73.44±0.11
	5-shot	86.16+-0.08	84.60±0.08	85.36±0.08

Table 3.8: Analysis on different design choices for task transformation functions. We study various length-L task transformation functions sampled with different levels of sampling uncertainty controlled by τ. We use R2D2 with ResNet-12 on the CIFAR-FS dataset.

and ϵ of the distribution. The function length L controls the number of image operations in a task transformation. The temperature ϵ controls the uncertainty during the sampling of a task transformation. A larger ϵ will produce more random sampling results. We use $\epsilon = \infty$ to denote the uniform sampling of image operations. In other words, all possible task transformations have equal chance of being selected during training. We use ResNet-12 and R2D2 in our L2TT framework with different Ls and ϵs. The results are show in Table 3.8. We observe improved performance for each ϵ when L increases from 1 to 3. A larger L increases the representation power of a task transformation and offers more flexibility in how a training task distribution can be transformed. However, as shown by the results when $L = 4$, a too large L hurts the performance. We also note that uniformly sampling image operations will result in suboptimal performance, as shown by the results when $\epsilon = \infty$. This inferior performance with $\epsilon = \infty$ indicates that task transformations are not of equal importance, and this also justifies our probabilistic modeling of task transformations.

3.2.5 Conclusion

We introduced a task transformation layer to address the mismatch between training tasks and a given meta-learning setting specified by the meta-model architecture and the meta-learning algorithm during meta-training. The added layer adjusts the difficulty of an input task by transforming the task to control the information flowing from the meta-model input to its output. We implemented the task transformation layer as a stochastic function with differentiable image operations as its building blocks. Experimental results demonstrated the effectiveness of our method in improving the generalization performance of various meta-learning algorithms on different model architectures and datasets.

3.3 Knowledge-Guided Semantics Adjustment for Improved Few-Shot Classification

3.3.1 Introduction

The ability of learning from few examples is attractive for deep learning models since it can greatly alleviate the need of the labor-intensive process of collecting many labeled examples for training. However, because of their large hypothesis space [65], widely used off-the-shelf deep learning models have difficulty in generalization when trained on few labeled examples. To systematically analyze and address this challenge, novel few-shot learning settings have been developed. Few-shot classification [127, 23, 12, 128, 64] is one example of such pursuit.

In few-shot classification, tasks are constructed such that each task consists of a support set and a query set. For each task, a model learns from the support set, which only has very few labeled examples per class, and predicts examples in the query set. In practice, the labels provided in each task, such as Class 1 and Class 2, carry no semantic meanings. Without any prior knowledge, it is challenging for a deep neural network to correctly infer the actual class from few images especially when they have multiple objects. Without substantial labeled data, the network may capture spurious features, e.g., any of the objects or a combination of them, from an image as the class object that distinguishes the image from other images in different classes. This is illustrated in Figure 3.5, where we fine-tune and evaluate a well-trained meta-learning model proposed in [26] on a challenging 2-way 1-shot testing task. The task is constructed with two classes representing two breeds of dogs, but the provided labels do not contain such information. Each class has one labeled image in the support set. We use class activation map (CAM) [1] as the visualization tool to show important areas contributing to predictions made by the model. For the support set images, we observe that in Class 2, the helmet, although being unrelated to the true class, is considered as important. As a result, the model tends to treat the task as distinguishing dogs with or without helmets, and thus the query image from Class 2 is misclassified as Class 1 since there is no helmet in the image.

It should be noted that even for humans, without any prior knowledge or context about the task, there is still a possibility that the query image is considered as Class 1. However, with the prior knowledge that for many tasks, dogs are more probable to be an object of interest than other objects in an image, we tend to treat this task as to classify two breeds of dogs, and we will focus on the dogs instead of the helmet or other spurious features when making predictions since the helmet is not of interest according to our prior knowledge. To mimic such human-level behavior in

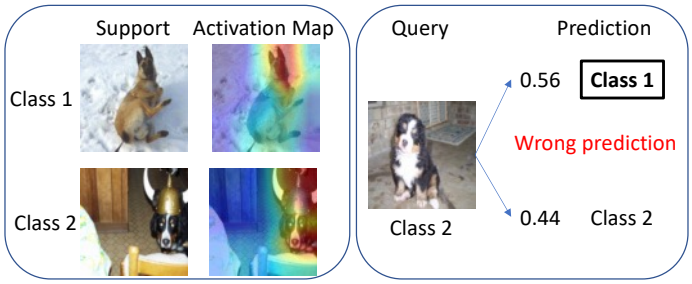


Figure 3.5: Illustration on how class-unrelated objects in an image affects model prediction. Important areas contributing to the correct predictions made by the model are highlighted by CAM [1] with warmer colors representing higher importance. The helmet is considered important by the model for correctly recognizing Class 2 but it is unrelated to dogs.

the existence of prior knowledge to improve the performance of a few-shot classifier, we propose to learn semantically meaningful features along with their relative importance as the prior knowledge in few-shot classification. These features represent common patterns found in many images from many few-shot tasks, and we use them by first detecting the strengths of these features in each image and adjusting the strengths to suppress unimportant and spurious ones which are not of interest as compared to others.

More specifically, we propose a knowledge-guided semantics adjustment module (KGSA), which contains a set of learnable semantic features and a learnable importance kernel. Each of the semantic features represents a common pattern found in many images, such as the shape of dogs or some spurious features. The KGSA detects these semantic features in each image embedding and uses the importance kernel to determine which features are to be suppressed. Moreover, since an image embedding is obtained from a nonlinear feature extractor, it contains complex correlations between the objects in the input image, hindering the ability of KGSA in detecting and suppressing class-unrelated patterns in the image. To circumvent this, we propose a simple and effective sampling-based image representation decomposition (IRD) module to mitigate the coupling between objects in image embeddings. With the incorporation of IRD, spurious and class-unrelated patterns can be effectively detected and suppressed by KGSA in an image embedding, and class-related patterns can be captured by the model to infer the true class of the image.

In summary, the main contributions of this section are:

- We propose to learn the prior knowledge on important objects for object recognition in few-shot classification by proposing a KGSA module which consists of a set of semantic features and an importance kernel. The KGSA can detect interested object patterns using the semantic

features and suppress spurious and class-unrelated ones synthesized from the importance kernel to help a model infer the class of a given image.

- We propose a simple and effective IRD module to improve the effectiveness of the KGSA module by mitigating the coupling between objects in an image embedding.
- We meta-train the KGSA on many randomly sampled few-shot classification tasks for improved few-shot classification performance. We analyze and validate the effectiveness of our method on two benchmark datasets [124, 128]. Our method achieves the best few-shot classification performance among all the relevant state-of-the-art approaches and shows superiority in dealing with extremely low-shot tasks.

3.3.2 Problem Definition

In few-shot classification, a few-shot task $\mathcal{T}$ contains a support set $\mathcal{S} = \{(\mathbf{x}_i, z_i)\}_{i=1}^{n_S}$ and a query set $\mathcal{Q} = \{(\mathbf{x}_j, z_j)\}_{j=1}^{n_Q}$, where $\mathbf{x}_i$ and $\mathbf{x}_j$ are the examples, z_i and z_j are the corresponding labels, n_S denotes the size of the support set, and n_Q denotes the size of the query set. Typically, $\mathcal{S}$ has N-way K-shot examples with $n_S = N \cdot K$, i.e., there are N classes with K examples per class in the set. In $\mathcal{Q}$, there are $n_Q = N \cdot K_q$ examples with the same N classes as in $\mathcal{S}$ and K_q examples per class. A model in few-shot classification has an embedding network $f_{\boldsymbol{\theta}}(\cdot)$ parameterized by $\boldsymbol{\theta}$ and a classifier $\ell_{\boldsymbol{\phi}}(\cdot)$ parameterized by $\boldsymbol{\phi}$. The embedding network maps an example $\mathbf{x}$ to its embedding $\mathbf{e} = f_{\boldsymbol{\theta}}(\mathbf{x}) \in \mathbb{R}^{D \times 1}$ with D denoting the embedding dimension. The classifier takes $\mathbf{e}$ as its input and predicts its class membership. The objective in few-shot classification is to learn a model that can quickly adapt to an unseen task using its support set data, i.e., $\boldsymbol{\theta}^*, \boldsymbol{\phi}^* = A(\boldsymbol{\theta}, \boldsymbol{\phi}, \mathcal{S})$, where $\boldsymbol{\theta}^*, \boldsymbol{\phi}^*$ are task-specific weights, A represents a learning algorithm which uses $\mathcal{S}$, $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ as inputs. Unlike the transductive setting, where $\boldsymbol{\theta}^*, \boldsymbol{\phi}^* = A(\boldsymbol{\theta}, \boldsymbol{\phi}, \mathcal{S}, \mathcal{Q})$, we focus on the inductive learning setting since $\boldsymbol{\theta}^*$ and $\boldsymbol{\phi}^*$ only depend on $\mathcal{S}$. Moreover, following the common setting in meta-learning [22, 15, 68], we assume the task distributions in meta-training and meta-testing are similar, i.e., tasks are all sampled from the same dataset so that knowledge transfer is possible.

Challenges. Without enough data in $\mathcal{S}$ and the proper context (since the provided labels carry no semantic meanings), $\boldsymbol{\theta}^*$ and $\boldsymbol{\phi}^*$ tend to fit to $\mathcal{S}$ with spurious patterns. As illustrated in Figure 3.5, Class 2 may be inferred as any dog with a helmet, but it actually represents a specific breed of dogs. Hence, the fitted model will generalize poorly on the query set $\mathcal{Q}$ since the wrong class is inferred.

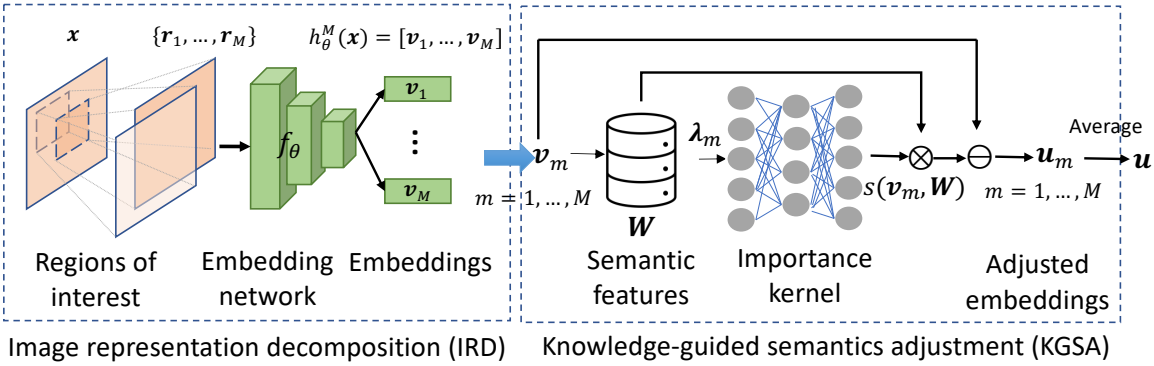


Figure 3.6: The overview of our proposed method. Our method includes IRD and KGSA as the two key components.

3.3.3 Methodology

Method Overview

Figure 3.6 illustrates the framework of the proposed method which consists of two components. KGSA detects the strengths of certain patterns represented by the semantic features and adjusts the strengths of these features to get the final embedding of an image such that class-unrelated patterns in the image are suppressed. IRD is introduced to improve the effectiveness of KGSA by decomposing an image embedding into a set of embeddings. Each embedding represents an area of interest of the image. In this way, we decouple complex correlations between objects induced by the nonlinearity of the embedding network, facilitating KGSA in detecting object patterns and adjusting their contributions to the image embedding. The set of embeddings from IRD are processed and averaged by KGSA to get the final embedding for the input. We explain each component in detail in the following sections.

Image Representation Decomposition (IRD)

Due to the nonlinearity of the embedding network, multiple objects in an image are coupled in the image embedding, making it difficult to detect and suppress class-unrelated object patterns. To address this, we introduce an IRD module to decompose an image embedding into a set of patch embeddings. Ideally, each image patch represents a region of interest, containing an object to be recognized by the KGSA. However, in practice, it is hard to find regions of interest from an image such that each region contains a meaningful object. We could use region proposal networks [129] to find such regions, but training them in the few-shot setting is a separate and challenging problem.

Therefore, we design a simple random sampling approach in the IRD module and leave more complex implementations of the IRD module as our future work.

In our method, we randomly sample M regions, $\{\mathbf{r}_i | i = 1, \ldots, M\}$, from a given image $\mathbf{x}$. Then the new embedding for the image $\mathbf{x}$ is $h^M_{\boldsymbol{\theta}}(\mathbf{x}) = [f_{\boldsymbol{\theta}}(\mathbf{r}_1), \ldots, f_{\boldsymbol{\theta}}(\mathbf{r}_M)]$, where $h^M_{\boldsymbol{\theta}}(\mathbf{x}) \in \mathbb{R}^{D \times M}$ is the embedding matrix for the input $\mathbf{x}$. Note that each $\mathbf{r}_i$ is resized such that its embedding dimension is D. In the trivial case when $M = 1$, the IRD will output the original image embedding $f_{\boldsymbol{\theta}}(\mathbf{x})$.

One potential limitation of the random sampling strategy is that we may miss or cut out some objects in an image. However, this is not critical in both training and testing phases. In the training phase, the imperfectly sampled image patches that contain only part of objects will act as a regularization, avoiding remembering specific objects. In the testing phase, we can sample more image patches than those in the training to ensure sufficient coverage of all meaningful objects in the image without incurring large computational overhead. Although the sampling method is often used for data augmentation, in our method, it serves as a simple and effective method to decouple complex relations between objects.

Knowledge-Guided Semantics Adjustment (KGSA)

KGSA consists of a set of learnable semantic features $\mathbf{W}$ and a learnable importance kernel $\kappa_{\boldsymbol{\psi}}(\cdot)$ with parameters $\boldsymbol{\psi}$. We denote KGSA as $g_{\boldsymbol{\omega}}(\cdot)$ with $\boldsymbol{\omega} = \{\mathbf{W}, \boldsymbol{\psi}\}$. Then, for each $\mathbf{v}_m = h^M_{\boldsymbol{\theta}}(\mathbf{x})[m]$ from the m-th element of the IRD output $h^M_{\boldsymbol{\theta}}(\mathbf{x})$, KGSA outputs adjusted embedding $\mathbf{u}_m$ for each $\mathbf{v}_m$ by subtracting from $\mathbf{v}_m$ the spurious feature synthesized from the semantic features $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_L] \in \mathbb{R}^{D \times L}$ with certain suppression weights, i.e.,

$$\mathbf{u}_m = g_{\boldsymbol{\omega}}(\mathbf{v}_m) = \mathbf{v}_m - \mathbf{W} s(\mathbf{v}_m, \mathbf{W}), m = 1, \ldots, M, \tag{3.22}$$

where $s(\mathbf{v}_m, \mathbf{W}) \in \mathbb{R}^{L \times 1}$ denotes the suppression weight vector dependent on $\mathbf{v}_m$ and $\mathbf{W}$. Finally, all the M adjusted vectors $\{\mathbf{u}_m | m = 1, \ldots, M\}$ are averaged to get the adjusted embedding $\mathbf{u}$ for the input $\mathbf{x}$, i.e.,

$$\mathbf{u} = 1/M \cdot \sum_{m=1}^{M} \mathbf{u}_m. \tag{3.23}$$

In our design of KGSA, calculation of the weights follows the *detection-and-adjustment* procedure due to its flexibility in imposing constraints on the semantic features and its low computational complexity.

Detection

KGSA first detects the strengths of semantic features $\mathbf{W}$ in the embedding $\mathbf{v}_m$ by calculating the strength vector $\boldsymbol{\lambda}_m = [\lambda_{1,m}, \ldots, \lambda_{L,m}]^T$, a sparse decomposition of $\mathbf{v}_m$ in terms of $\mathbf{W}$. Specifically, $\boldsymbol{\lambda}_m$ is obtained via solving the following problem:

$$\boldsymbol{\lambda}_m = \arg\min_{\boldsymbol{\lambda}} \|\mathbf{W}\boldsymbol{\lambda} - \mathbf{v}_m\|_2^2 + \gamma\|\boldsymbol{\lambda}\|_2^2, \tag{3.24}$$

where γ is a non-negative hyperparameter controlling the magnitude of $\boldsymbol{\lambda}_m$, i.e., $\|\boldsymbol{\lambda}_m\|_2$. The solution $\boldsymbol{\lambda}_m$ quantifies the similarities of these semantic features to the embedding $\mathbf{v}_m$. The advantage of introducing Equation (3.24) is that we can impose an explicit sparse constraint on $\boldsymbol{\lambda}_m$ and consequently enforce these semantic features to be as general as possible to cover many similar objects. Moreover, we can get the closed form solution for $\boldsymbol{\lambda}_m$ as $\boldsymbol{\lambda}_m = (\mathbf{W}^T\mathbf{W} + \gamma\mathbf{I})^{-1}\mathbf{W}^T\mathbf{v}_m$, where $\mathbf{I} \in \mathbb{R}^{L\times L}$ is an identity matrix. Note that γ not only controls the magnitude of $\boldsymbol{\lambda}$, hence the representation capability of $\mathbf{W}$, but also makes sure the inverse operation is not ill-conditioned. In practice, we find that our method is not sensitive to γ in terms of few-shot classification performance, so we set $\gamma = 1$.

Adjustment

Then, KGSA adjusts $\boldsymbol{\lambda}_m$ using the importance kernel $\kappa_{\boldsymbol{\psi}}(\cdot)$ since KGSA could detect two semantic features that have large λs and needs to determine which one should be suppressed. For example, if the two semantic features $\mathbf{w}_1$ and $\mathbf{w}_2$ correspond to head covering and the shape of dogs, respectively, then we expect to suppress $\mathbf{w}_1$ while preserving $\mathbf{w}_2$ for images in Figure 3.5, since we need to focus on dogs according to the prior knowledge. The importance kernel is proposed to learn the relative importance of all the semantic features such that only class-unrelated ones are suppressed. We implement the importance kernel as an L-input L-output neural network. As a result, the output may contain negative numbers. Subtracting features with negative weights will strengthen them in $\mathbf{v}_m$, adding class-unrelated noise in $\mathbf{v}_m$ since the semantic features are designed to be class-agnostic. Hence, we calculate the suppression weight vector as

$$s(\mathbf{v}_m, \mathbf{W}) = \max(\kappa_{\boldsymbol{\psi}}(\boldsymbol{\lambda}_m), \mathbf{0}), \tag{3.25}$$

where $\max(\cdot,\cdot)$ is an element-wise maximum operator, and $\mathbf{0}$ is a length-L all-zero column vector.

Learning Objective

In essence, KGSA is designed to highlight class-related objects in a dataset by suppressing unrelated ones so as to improve the few-shot classification performance of a model on the dataset. Specifically, we use the adjusted embedding vectors of the support set data to build a prototype-based few-shot classifier which adopts the nearest-neighbor classification rule used in many few-shot classification literature [12, 130, 15, 26]. In an N-way K-shot task, each of the N class prototypes is calculated as follows,

$$\boldsymbol{\mu}_c = \frac{1}{|\mathcal{F}_c|} \sum_{\mathbf{u} \in \mathcal{F}_c} \mathbf{u}, c = 1, \ldots, N, \tag{3.26}$$

where $|\cdot|$ denotes the size of a set, $\mathcal{F}_c = \{\mathbf{u} | \mathbf{u} = 1/M \sum_{m=1}^{M} g_{\boldsymbol{\omega}}(h_{\boldsymbol{\theta}}^M(\mathbf{x}_s)[m]), \mathbf{x}_s \in \mathcal{S}_c\}$, and $\mathcal{S}_c$ is the set of support examples from class c. The N class prototypes will instantiate a prototype-based classifier $p(\cdot|\mathbf{x}_q, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N)$ whose prediction for a query example $\mathbf{x}_q \in \mathcal{Q}$ is $\hat{c} = \arg\max_{c=1,\ldots,N} p(y = c|\mathbf{x}_q, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N)$, where $p(y = c|\mathbf{x}_q, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N) = \exp(\tau d(\mathbf{u}_q, \boldsymbol{\mu}_c)) / \sum_{c'} \exp(\tau d(\mathbf{u}_q, \boldsymbol{\mu}_{c'}))$, $d(\cdot, \cdot)$ denotes a similarity metric function, $\boldsymbol{u}_q$ is the embedding for the query example $\mathbf{x}_q$ obtained from the KGSA, and τ controls the softness of the output probability distribution. In the following, we use cosine similarity as the metric function thanks to the good generalization performance it brings to few-shot classification [26], and we choose τ via hyperparameter selection.

Overall Objective. To train KGSA with IRD, we minimize the risk function $\mathcal{L} = \mathbb{E}_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}}$ with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\omega}$ on tasks sampled from $p(\mathcal{T})$ as follows,

$$\boldsymbol{\theta}^*, \boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\theta}, \boldsymbol{\omega}} \mathbf{E}_{\mathcal{T} \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}}, \tag{3.27}$$

where the task loss $\mathcal{L}_{\mathcal{T}}$ is defined as $\mathcal{L}_{\mathcal{T}} = -\frac{1}{|\mathcal{Q}|} \sum_{(\mathbf{x}, z) \in \mathcal{Q}} \log p(y = z|\mathbf{x}, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N)$, $\mathcal{T} = \{\mathcal{S}, \mathcal{Q}\}$, z is the label for $\mathbf{x}$, and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N$ are obtained from the support set $\mathcal{S}$ following Equation (3.26). The true risk $\mathcal{L}$ is hard to obtain since $p(\mathcal{T})$ is unknown; hence, we calculate the empirical risk of B tasks as $\mathcal{L}_E = \frac{1}{B} \sum_{b=1}^{B} \mathcal{L}_{\mathcal{T}_b}$ with $\mathcal{T}_b$ sampled from the training data. Then, we use gradient descent with a learning rate β to minimize $\mathcal{L}_E$ each time after sampling B tasks as $(\boldsymbol{\theta}, \boldsymbol{\omega}) = (\boldsymbol{\theta}, \boldsymbol{\omega}) - \beta \nabla_{\boldsymbol{\theta}, \boldsymbol{\omega}} \mathcal{L}_E$. The above iteration will terminate when the validation accuracy stops increasing. Due to our specific implementation, no learnable parameters are required for the IRD module.

Discussion. In our method, two important hyperparameters affect the effectiveness of KGSA: the number of sampled regions M in IRD and the number of semantic features L in KGSA. With a larger M, IRD can provide more accurate object information to KGSA. However, a larger M

Method	Backbone	miniImageNet		tieredImageNet	
		1-shot	5-shot	1-shot	5-shot
TADAM [130]	ResNet-12	58.50 ± 0.30	76.70 ± 0.30	-	-
ProtoNet [12]	ResNet-12	60.37 ± 0.83	78.02 ± 0.57	65.65 ± 0.92	83.40 ± 0.65
Shot-Free [131]	ResNet-12	59.04 ± 0.43	77.64 ± 0.39	66.87 ± 0.43	82.64 ± 0.43
LEO [25]	WRN-28-10	61.76 ± 0.08	77.59 ± 0.10	66.33 ± 0.05	81.44 ± 0.09
MetaOptNet [14]	ResNet-12	62.64 ± 0.61	78.63 ± 0.46	65.99 ± 0.72	81.56 ± 0.53
CAN [132]	ResNet-12	63.85 ± 0.48	79.44 ± 0.34	69.89 ± 0.51	84.23 ± 0.37
MTL [133]	ResNet-12	61.20 ± 1.80	75.50 ± 0.80	65.62 ± 1.80	80.61 ± 0.9
Meta-Baseline [26]	ResNet-12	63.17 ± 0.23	79.26 ± 0.17	68.62 ± 0.27	83.29 ± 0.18
Neg-Cosine [134]	ResNet-12	63.85 ± 0.81	81.57 ± 0.56	-	-
DSN-MR [135]	ResNet-12	64.60 ± 0.72	79.51 ± 0.50	67.39 ± 0.82	82.85 ± 0.56
FEAT [15]	ResNet-12	66.78 ± 0.20	82.05 ± 0.14	70.80 ± 0.23	84.79 ± 0.16
DeepEMD [68]	ResNet-12	65.91 ± 0.82	82.41 ± 0.56	71.16 ± 0.87	86.03 ± 0.58
MELR [136]	ResNet-12	67.40 ± 0.43	83.40 ± 0.28	72.14 ± 0.51	87.01 ± 0.35
CPLAE [137]	ResNet-12	67.46 ± 0.44	83.22 ± 0.29	72.23 ± 0.50	87.35 ± 0.34
Our method	ResNet-12	$\mathbf{70.34 \pm 0.45}$	$\mathbf{85.27 \pm 0.28}$	$\mathbf{74.30 \pm 0.50}$	$\mathbf{87.58 \pm 0.33}$

Table 3.9: Performance comparison to prior works on the miniImageNet and tieredImageNet datasets. Results with citations are reported from the literature with "-" representing "not reported". The best performance is in the bold font.

will take more time and memory to run the optimization in Equation (3.27). The effectiveness of KGSA on many unseen tasks also depends on L. For a small and simple dataset which contains a small number of classes, the KGSA with a large L will overfit to the dataset, capturing features corresponding to many specific objects. On the other hand, for a large dataset with many classes, we expect L to be larger. In practice, we will use a held-out validation set to select the best values of L.

3.3.4 Experiments

Experimental Setup

Dataset. We use miniImageNet [124] and tieredImageNet [128] in the experiments. For miniImageNet, we follow the dataset split from [124], resulting in 64, 16, and 20 classes in the meta-train, meta-validation, and meta-test sets. The tieredImageNet dataset has 608 classes which are divided into 351, 97, and 160 classes for meta-training, meta-validation, and meta-testing, respectively. All images are resized to 84×84.

Evaluation. All the experiments are conducted in the inductive setting. The statistics in the batch normalization layers are fixed to avoid knowledge sharing between tasks. We randomly sample 2000 N-way K-shot tasks from the test split of a dataset and report the mean accuracy and the standard deviation with a 95% confidence interval over those tasks.

Model	miniImageNet	
	1-shot	5-shot
Base	65.78 ± 0.46	82.07 ± 0.31
Base+KGSA	67.05 ± 0.45	82.64 ± 0.31
Base+IRD	67.34 ± 0.46	82.02 ± 0.31
Base+KGSA+IRD	68.67 ± 0.45	83.67 ± 0.30
Model	tieredImageNet	
	1-shot	5-shot
Base	71.76 ± 0.52	85.41 ± 0.35
Base+KGSA	71.88 ± 0.52	86.00 ± 0.35
Base+IRD	72.19 ± 0.51	85.60 ± 0.35
Base+KGSA+IRD	72.76 ± 0.52	86.74 ± 0.35

Table 3.10: Ablation study on KGSA and IRD ($M = 5$ in both training and testing). Results on 5-way tasks are reported.

Results on Benchmark Datasets

Table 3.9 shows the performance comparison on 5-way few-shot tasks. For fair comparison, we only consider methods using inductive learning and supervised learning since our method does not use additional unlabeled data. We observe from Table 3.9 that our method achieves the best performance among all the listed baseline methods in both 1- and 5-shot settings with significant performance gains achieved in 1-shot settings. Moreover, from the perspective of additionally added parameters, the overhead of our method is negligible compared to the embedding network, since according to our experiment setting, the number of added parameters equals $L \times D + 2L \times H$, which is the combination of numbers from the semantic features and the importance kernel in KGSA.

Ablation Studies

Contributions of IRD and KGSA. According to the discussion in the Learning Objective section, we set M to 5 in both training and testing to speed up evaluation and to best showcase the advantage of KGSA when low quality image embeddings are provided by the IRD with small M. We denote the model [26] whose embedding network is pre-trained and then fine-tuned with a few-shot classifier as the "Base" model, and we follow the training procedures in [15]. We add a KGSA module to "Base" to get "Base+KGSA", and an IRD module to "Base" to get "Base+IRD". Finally, we add both the KGSA and the IRD modules to "Base" and obtain "Base+KGSA+IRD". We observe from Table 3.10 that with KGSA alone, the performance of "Base+KGSA" is on par with some recent works [15, 68]. Adding IRD can improve the performance of "Base" and "Base+KGSA", with a bigger improvement from the latter. For example, on the miniImageNet dataset, the performance gains for adding IRD

Adjusting Strength Vectors	Shot	L				
		1	5	10	50	100
Importance kernel	1-shot	66.42	66.85	67.05	66.97	66.62
	5-shot	82.29	82.64	82.55	82.54	82.34
Linear transform	1-shot	66.30	66.32	66.29	66.13	65.83
	5-shot	82.35	82.52	82.48	81.19	81.15

Table 3.11: Classification accuracy of the "Base+KGSA" model configured with different Ls and different methods for adjusting strength vectors.

to "Base" and "Base+KGSA" in the 1-shot case are 1.56% and 1.62%, respectively; for the 5-shot case, IRD can improve the accuracy of "Base+KGSA" by 1.03%, while being ineffective for "Base".

Semantic Features and Importance Kernel in KGSA. We study how the number of semantic features L affects the effectiveness of KGSA and analyze the necessity of importance kernel in KGSA. Table 3.11 shows the classification accuracies of the "Base+KGSA" model on the miniImageNet dataset with $L \in \{1, 5, 10, 50, 100\}$ and two different methods of adjusting the strength vector defined in Equation (3.24). The baseline is "Linear transform" which scales each strength value with a learnable parameter, while "Importance kernel" represents feed forward neural networks proposed in our method. We observe that models adopting importance kernels perform better than those adopting linear transforms as L increases, and we see a significant performance drop on the latter when $L \in \{50, 100\}$. This is because the importance kernel is context dependent: it compares strength values of different semantic features in an image embedding and adjusts these values jointly instead of transforming them separately. Hence, an importance kernel is necessary in KGSA. We also observe that in different settings, the optimal L is different; a larger shot setting generally requires a smaller L. Moreover, models with $L = 1$ have competitive performance. However, it is not the result of introducing additional parameters since adding an additional layer to the embedding network will get inferior performance with classification accuracies on miniImageNet being 63.14 ± 0.46 and 79.57 ± 0.33 for 1-shot and 5-shot, respectively. Instead, when $L = 1$, we observe that the learned semantic feature represents many noisy background objects, and models can benefit by suppressing it in image embeddings.

Number of Sampling Times in IRD. We compare the performance of "Base+IRD" with that of "Base+KGSA+IRD" when IRD is configured with different Ms. Since a higher M in training requires more GPU memory, we train the models with $M = 5$ and evaluate them with $M \in \{5, 10, 15, 25, 50\}$. Table 3.12 shows classification accuracies on the miniImageNet dataset. We observe consistent performance improvement on both shot numbers as M increases, and the performance increment saturates when M is large. Moreover, we show performance gains of our method,

M	Base+IRD		Base+KGSA+IRD	
	1-shot	5-shot	1-shot	5-shot
5	67.34	82.02	68.67 (+1.33)	83.67 (+1.65)
10	67.90	82.57	69.31 (+1.41)	84.43 (+1.86)
15	68.16	82.86	69.65 (+1.49)	84.59 (+1.73)
25	68.41	83.02	69.75 (+1.34)	84.86 (+1.84)
50	68.63	83.07	**70.34** (+1.71)	**85.27** (+2.20)

Table 3.12: Classification accuracy comparison between "Base+IRD" and "Base+KGSA+IRD" with IRD configured with different Ms on the miniImageNet dataset.

"Base+KGSA+IRD", over the "Base+IRD" model in parentheses following the accuracy numbers. We observe that when M increases, the KGSA module benefits more from the IRD module than the "Base" model does, indicating the necessity of IRD in providing high-quality image embeddings for KGSA.

3.3.5 Conclusion

We introduced KGSA which consists of a set of learnable semantic features representing spurious attributes shared by many similar objects and a learnable importance kernel encoding the relative importance of the features. The KGSA follows the detection-and-adjustment procedure to suppress spurious and class-unrelated attributes in image embeddings. To improve the effectiveness of KGSA, we proposed an IRD module to decouple the potential correlations between objects in the image embedding due to the nonlinearity of the embedding network. We meta-trained the KGSA with IRD on many few-shot tasks. Our method outperforms the state-of-the-art methods in two few-shot benchmark datasets and shows superiority in dealing with extremely low-shot tasks.

Chapter 4

Multimodal-Assisted Spurious Bias Mitigation under Subpopulation Shifts

Subpopulation shifts refer to changes in the distribution of specific data groups between the training and test sets. For example, in the training set, waterbirds appearing against water backgrounds may constitute the majority, while those with land backgrounds are the minority. In contrast, the test set may reverse this distribution, with waterbirds on land backgrounds becoming the majority and those on water backgrounds the minority. Models with spurious bias heavily rely on spurious attributes in certain data groups for predictions, resulting in poor generalization in other data groups where the learned spurious attributes do not exist. Although changes in the proportions of data groups are generally known, group annotations are typically unknown or hard to acquire, making it challenging to learn robust models against subpopulation shifts. In this chapter, we propose to use pre-trained vision-language models (VLMs) to automatically detect spurious attributes for spurious bias detection and mitigation. In Chapter 4.1, we demonstrate that a benchmark system with pre-trained VLMs can effectively generate challenging classification tasks with subpopulation shifts for learning novel concepts in a few-shot classification setting. In Chapters 4.2 and 4.3, we design two mitigation methods focusing on spurious-attribute-aware classification and spurious-attribute-agnostic representation learning, respectively, to mitigate spurious bias detected via VLMs.

4.1 Benchmarking Spurious Bias Using Vision-Language Models

4.1.1 Introduction

In the traditional learning setting [138, 139, 140], deep learning models tend to rely on spurious correlations as their prediction shortcuts or exhibit *spurious bias*, where spurious correlations [48, 141, 47, 6, 139, 142, 143] are the brittle associations between classes and spurious attributes — attributes of inputs non-essential to the classes. For example, models that predict classes using the associated backgrounds [49] or image textures [48] may experience significant performance drops when the associated backgrounds or textures change to different ones. To reveal spurious bias, it is important to create evaluation tasks with subpopulation shifts. In this section, we aim to generate subpopulation shifts for detecting spurious bias and comprehensively evaluate our proposed method. To this end, we focus on the few-shot classification [12, 13, 14, 15, 16] (FSC) setting as this setting requires generating multiple classification tasks for evaluation which aligns well with our goal.

In FSC, few-shot classifiers can transfer the knowledge learned from base classes to recognize novel classes with a few labeled samples. However, they face potential risks when deployed in the real world, such as data distribution shifts [144, 145] and adversarial examples [146, 147]. In the low-data regime, spurious bias becomes more evident. For example, in Figure 4.1, the correlation between the class `bird` and the attribute `tree branch` in the support (training) image may form a shortcut path from `tree branch` to predicting the image as `bird` and hinder the learning of the desired one that uses class-related attributes, such as `head`, `tail`, and `wing`. The shortcut will fail to generalize in the query (test) image where no `tree branch` can be found. In general, few-shot image classifiers are susceptible to spurious bias.

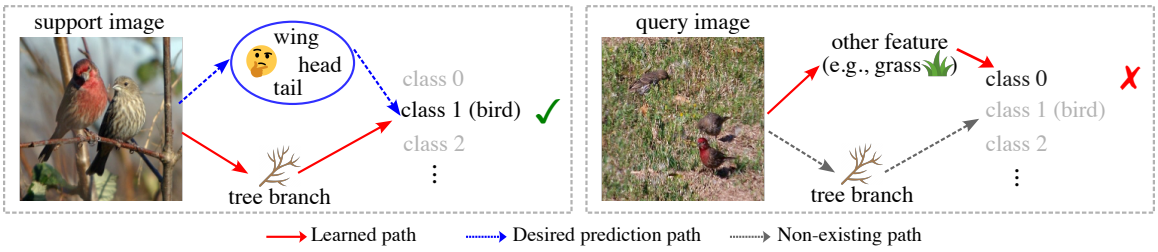


Figure 4.1: Exploiting the spurious correlation between the class `bird` and the spurious attribute `tree branch` to predict `bird` leads to an incorrect prediction on the test image showing birds on a grass field. For clarity, we only show the case for one class.

However, there lacks a dedicated benchmarking framework that evaluates the robustness of few-shot classifiers to spurious bias. The standard benchmarking procedure in FSC trains a few-shot classifier on base classes from a training set with ample samples and evaluates the classifier on FSC test tasks constructed from a test set with novel classes. The problem with this procedure is the lack of explicit control over the spurious correlations in the constructed FSC tasks. Each FSC test task contains randomly sampled support and query samples. Thus, spurious correlations in the majority of the test set samples can be demonstrated in these tasks, providing unfair advantages for few-shot classifiers with high reliance on the spurious correlations.

In this section, we propose a systematic and rigorous benchmark framework, termed Few-Shot Tasks with Attribute Biases (FewSTAB), to fairly compare the robustness of various few-shot classifiers to spurious bias. Our framework explicitly controls spurious correlations in the support and query samples when constructing an FSC test task to reveal the robustness pitfalls caused by spurious bias. To achieve this, we propose attribute-based sample selection strategies that select support and query samples with biased attributes. These attributes together with their associated classes formulate spurious correlations such that if the support samples induce spurious bias in a few-shot classifier, i.e., the classifier learns the spurious correlations in the support samples as its prediction shortcuts, then the query samples can effectively degrade the classifier's performance, exposing its non-robustness to spurious bias.

Our framework exploits the spurious attributes in test data for formulating spurious correlations in FSC test tasks. Some existing datasets [139, 148, 149] provide spurious attribute annotations. However, they only have a few classes and cannot provide enough classes for training and testing. Many benchmark datasets for FSC do not have annotations on spurious attributes, and obtaining these annotations typically involves labor-intensive human-guided labeling [86, 87]. To address this, we further propose to use a pre-trained vision-language model (VLM) to automatically identify distinct attributes in images in the high-level text format. Our attribute-based sampling methods can use the identified attributes to simulate various spurious correlations. Thus, we can reuse any existing FSC datasets for benchmarking few-shot classifiers' robustness to spurious bias, eliminating the need for the manual curation of new datasets.

The main contributions of our work are summarized as follows:

- We propose a systematic and rigorous benchmark framework, termed Few-Shot Tasks with Attribute Biases (FewSTAB), that specifically targets spurious bias in few-shot classifiers,

demonstrates their varied degrees of robustness to spurious bias, and benchmarks spurious bias in varied degrees.

- We propose novel attribute-based sample selection strategies using a pre-trained VLM for constructing few-shot evaluation tasks, allowing us to reuse any existing few-shot benchmark datasets without manually curating new ones for the evaluation.
- FewSTAB provides a new dimension of evaluation on the robustness to spurious bias along with a new design guideline for building robust few-shot classifiers. We demonstrate the effectiveness of FewSTAB by applying it to models trained on three benchmark datasets with ten FSC methods.

4.1.2 Related Work

In Chapter 2.2, we have reviewed related works on generalizing to novel classes with a few labeled samples. In this section, we provide detailed discussions on few-shot classification and the associated robustness issues, as well as methods for discovering spurious attributes and revealing spurious biases.

Few-Shot Classification. Few-shot classification [23, 12, 64, 65, 66, 67] has received vast attention recently. Few-shot classifiers can be trained with meta-learning or transfer learning on base classes to learn the knowledge that can be transferred to recognize novel classes with a few labeled samples. The transfer learning approaches [64, 66] first learn a good embedding model and then fine-tune the model on samples from novel classes. The meta-learning approaches can be further divided into optimization-based and metric-based methods. The optimization-based methods [22, 150, 24, 151, 152] aim to learn a good initialized model such that the model can adapt to novel classes efficiently with a few gradient update steps on a few labeled samples. The metric-based methods [23, 125, 25, 14, 130, 13, 15, 68] aim to learn a generalizable representation space with a well-defined metric, such as Euclidean distance [12], to learn novel classes with a few labeled samples. Recently, large vision-language models [41, 153, 154] are used for few-shot classification. However, they have completely different training and inference pipelines from the models that we consider in this section.

Robustness in Few-Shot Classification. There are several notions of robustness for few-shot classifiers. The common one requires a few-shot classifier to perform well on the in-distribution samples of novel classes in randomly sampled FSC test tasks. The robustness to adversarial perturbations further requires a few-shot classifier to perform well on samples with imperceptible perturbations [146, 147]. Moreover, the cross-domain generalization [155, 67, 156] aims to test how robust a few-shot classifier is on samples from novel classes with domain shifts, which are typically reflected by the

changes in both image styles and classes. In contrast, we focus on a new notion of robustness: the robustness to spurious bias. There is a lack of rigorous evaluation methods on the topic. We provide a new evaluation method that specifically targets spurious bias and can systematically demonstrate few-shot classifiers' varied degrees of vulnerability to spurious bias, which has not been addressed in the existing literature.

Benchmarks for Spurious Bias. There are some existing datasets [139, 148, 149] that are designed to benchmark spurious bias in image classifiers. However, these datasets are only applicable to the traditional learning setting [138, 139, 140] since the classes in them are not sufficient for the training and testing of few-shot classifiers. Existing benchmarks in few-shot classification are not tailored for benchmarking spurious bias in few-shot classifiers. A recent work [157] creates a large-scale few-shot classification benchmark dataset with spurious-correlation shifts. In contrast, we propose a benchmark framework that can reuse existing few-shot classification datasets and provide a new dimension of evaluation.

Discovering Spurious Attributes. A spurious attribute is non-essential to a class and only exists in some samples. Early works on discovering spurious attributes [86, 87] require a predefined list of spurious attributes and expensive human-guided labeling of visual attributes. Recent works [158, 159, 160, 91] greatly reduce the need for manual annotations by using the neurons of robust models to detect visual attributes. However, they still need humans to annotate the detected visual attributes. We automate this process by using a pre-trained VLM to obtain distinct attributes as words. Instead of discovering spurious correlations, we simulate them via attribute-based sampling for benchmarking.

4.1.3 Preliminary

Few-Shot Classification Tasks. A typical FSC task $\mathcal{T}$ has a support set $\mathcal{S}$ for training and a query set $\mathcal{Q}$ for testing. In this task, there are C classes ($c = 1, \ldots, C$) with $N_{\mathcal{S}}$ (a small number) training samples and $N_{\mathcal{Q}}$ test samples per class in $\mathcal{S}$ and $\mathcal{Q}$, respectively. The task is called a C-way $N_{\mathcal{S}}$-shot task.

Few-Shot Classifiers. A few-shot classifier f_θ with parameters θ aims to classify the samples in $\mathcal{Q}$ after learning from $\mathcal{S}$ with a learning algorithm $\mathcal{O}$ in a few-shot task $\mathcal{T}$. Here, $\mathcal{O}$ could be any learning algorithms, such as the optimization method [22] or a prototype-based classifier learning method [12, 26]. To acquire a good few-shot learning capability, f_θ is typically meta-trained or pre-

Symbol	Meaning
$\mathcal{T}$	An FSC task
$\mathcal{S}$	Support (training) set in $\mathcal{T}$
$\mathcal{Q}$	Query (test) set in $\mathcal{T}$
c	A class in $\mathcal{T}$
C	Number of classes per task
$N_{\mathcal{S}}$	Number of samples (shots) per class in $\mathcal{S}$
$N_{\mathcal{Q}}$	Number of samples per class in $\mathcal{Q}$
$\mathcal{O}$	A few-shot adaptation algorithm
ψ	An attribute detector
Ω	An automatic word selector
$\mathcal{D}_{train}$	The base training set
$\mathcal{D}_{val}$	The validation set for selecting a few-shot classifier
$\mathcal{D}_{test}$	The test set for evaluating a few-shot classifier
$\mathcal{C}_{train}$	Classes in $\mathcal{D}_{train}$
$\mathcal{C}_{test}$	Classes in $\mathcal{D}_{test}$
$\mathcal{D}_c$	A set of all samples belonging to class c
a	A text-format attribute
$\mathcal{A}$	A set of text-format attributes

Table 4.1: Meanings of major symbols used in the section.

trained [161] on a base training set $\mathcal{D}_{train} = \{(x_n, y_n)|y_n \in \mathcal{C}_{train}, n = 1, \ldots, N_{train}\}$ with N_{train} sample(x)-label(y) pairs, where $\mathcal{C}_{train}$ is a set of base classes.

Performance Metrics. The performance of a few-shot classifier is typically measured by its average classification accuracy over $N_{\mathcal{T}}$ C-way $N_{\mathcal{S}}$-shot tasks randomly sampled from $\mathcal{D}_{test} = \{(x_n, y_n)|y_n \in \mathcal{C}_{test}, n = 1, \ldots, N_{test}\}$ where N_{test} sample-label pairs from novel classes $\mathcal{C}_{test}$ do not appear in $\mathcal{D}_{train}$, i.e., $\mathcal{C}_{train} \cap \mathcal{C}_{test} = \emptyset$. We denote this metric as *standard accuracy* $\mathrm{Acc}(f_\theta)$, i.e.,

$$\mathrm{Acc}(f_\theta) = \frac{1}{N_{\mathcal{T}}} \sum_{t=1}^{N_{\mathcal{T}}} \sum_{c=1}^{C} M_c(\mathcal{T}_t; f_\theta, \mathcal{O}), \tag{4.1}$$

where $M_c(\mathcal{T}_t; f_\theta, \mathcal{O})$ denotes the classification accuracy of f_θ on the query samples from the class c in $\mathcal{T}_t$ after f_θ is trained on $\mathcal{S}$ with $\mathcal{O}$. The metric $\mathrm{Acc}(f_\theta)$ in Equation (4.1) only shows the average learning capability of f_θ over C randomly selected novel classes. To better characterize the robustness of f_θ to spurious bias, we define the *class-wise worst classification accuracy* over tasks as

$$\mathrm{wAcc}(f_\theta) = \frac{1}{N_{\mathcal{T}}} \sum_{t=1}^{N_{\mathcal{T}}} \min_{c=1,\ldots,C} M_c(\mathcal{T}_t; f_\theta, \mathcal{O}). \tag{4.2}$$

A larger $\mathrm{wAcc}(f_\theta)$ indicates that f_θ is more robust to spurious bias.

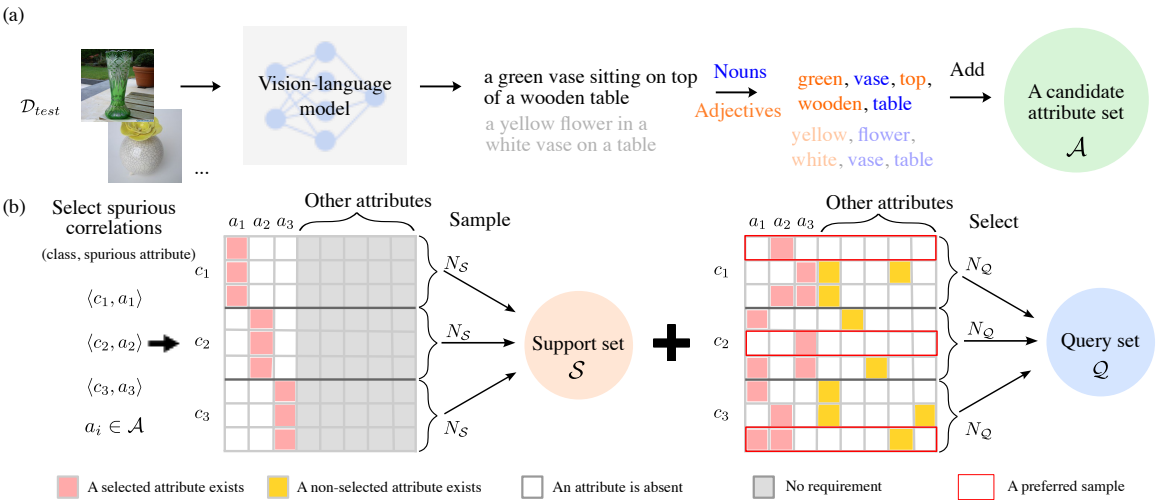


Figure 4.2: FewSTAB overview. (a) Extract distinct attributes using a pre-trained VLM. (b) Generate an FSC task for the evaluation of spurious bias in few-shot classifiers.

Spurious Correlations. A spurious correlation is the association between a class and an attribute of inputs that is *non-essential* to the class, and it *only* holds in some samples. We formally define it as follows.

Definition 4.1. Let $\mathcal{D}_c$ denote a set of sample-label pairs having the label c, and let $\psi : \mathcal{X} \to \mathcal{B}_{\mathcal{A}}$ be an attribute detector, where $\mathcal{X}$ is the set of all possible inputs, $\mathcal{B}_{\mathcal{A}}$ denotes all possible subsets of $\mathcal{A}$, and $\mathcal{A}$ is the set of all possible attributes. The class c and an attribute $a \in \mathcal{A}$ form a spurious correlation, denoted as $\langle c, a \rangle$, if and only if the following conditions hold:

1. There exists $(x, c) \in \mathcal{D}_c$ that satisfies $a \in \psi(x)$, and
2. There exists $(x', c) \in \mathcal{D}_c$ that satisfies $a \notin \psi(x')$.

We define a as the **spurious attribute** in $\langle c, a \rangle$.

Definition 4.1 specifies that all the spurious correlations are based on $\mathcal{D}_c$. In the remainder of the section, we define $\mathcal{D}_c = \{(x, c) | \forall (x, c) \in \mathcal{D}_{test}\}$ with $c \in \mathcal{C}_{test}$ as we focus on *evaluating* the robustness to spurious bias.

We list major symbols in the section alongside their meanings in Table 4.1.

4.1.4 Methodology

Attribute-Based Sample Selection

We first propose two attribute-based sample selection methods to reveal spurious bias in a few-shot classifier. Consider a training set $\mathcal{S}$ in a few-shot test task $\mathcal{T}$, which has C classes with each class

$c \in \mathcal{C}_{test}$ associating with a unique spurious attribute $a \in \mathcal{A}$. We aim to discover samples that can exhibit a classifier's spurious bias on $\langle c, a \rangle$ induced from $\mathcal{S}$. Motivated by existing findings [139, 143, 162] that classifiers with high reliance on $\langle c, a \rangle$ tend to perform poorly on samples without it, we propose an attribute-based sample selection strategy below.

Intra-Class Attribute-Based Sample Selection. Given $\mathcal{D}_c$ and the training set $\mathcal{S}$ having the spurious correlation $\langle c, a \rangle$, we generate a set $\mathcal{I}_{\langle c,a \rangle}$ of sample-label pairs which have class c but do not contain attribute a, i.e.,

$$\mathcal{I}_{\langle c,a \rangle} = \{(x, c) | \forall (x, c) \in \mathcal{D}_c, a \notin \psi(x)\}. \tag{4.3}$$

The above proposed method demonstrates a few-shot classifier's robustness to individual spurious correlation $\langle c, a \rangle$ and does not consider a multi-class classification setting where spurious attributes from some other class c' exist in samples of the class c. In this case, these attributes may mislead the classifier to predict those samples as the class c' and severely degrade the performance on the class c. For example, consider using the spurious correlations $\langle$`vase`, `blue`$\rangle$ and $\langle$`bowl`, `green`$\rangle$ for predicting `vase` and `bowl`, respectively. An image showing a vase in green is more effective in revealing the robustness to $\langle$`vase`, `blue`$\rangle$ as it is more likely to be misclassified as `bowl` than other images. Motivated by this, we propose the *inter-class attribute-based sample selection* below.

Inter-Class Attribute-Based Sample Selection. Given $\mathcal{D}_c$ and the training set $\mathcal{S}$ with the spurious correlations $\langle c, a \rangle$ and $\langle c', a' \rangle$, where $c' \neq c$ and $a' \neq a$, we generate a set $\mathcal{I}_{\langle c,a \rangle}^{\langle c',a' \rangle}$ of sample-label pairs which have class c, do not contain attribute a, but contain attribute a' from another class c':

$$\mathcal{I}_{\langle c,a \rangle}^{\langle c',a' \rangle} = \mathcal{I}_{\langle c,a \rangle} \cap \{(x, c) | \forall (x, c) \in \mathcal{D}_c, a'_{\langle c' \rangle} \in \psi(x)\}, \tag{4.4}$$

where $a'_{\langle c' \rangle}$ denotes a' in $\langle c', a' \rangle$, and $\mathcal{I}_{\langle c,a \rangle}$ is defined in Equation 4.3.

Considering that there are C classes in the training set $\mathcal{S}$ with each class associating with a unique spurious attribute a, to effectively demonstrate the reliance on the spurious correlation $\langle c, a \rangle$ with the inter-class attribute-based sample selection, we consider all the spurious correlations in $\mathcal{S}$. Specifically, we apply the above selection strategy to all the $C - 1$ spurious correlations in $\mathcal{S}$ other than $\langle c, a \rangle$ and obtain $\mathcal{I}_{\langle c,a \rangle}^{C}$ as the union of the $C - 1$ sets as follows:

$$\mathcal{I}_{\langle c,a \rangle}^{C} = \bigcup_{\langle c',a' \rangle \in \mathcal{C}_{\setminus c}} \mathcal{I}_{\langle c,a \rangle}^{\langle c',a' \rangle}, \tag{4.5}$$

where $\mathcal{C}_{\setminus c}$ denotes all the spurious correlations in $\mathcal{S}$ other than $\langle c, a \rangle$.

The inter-class attribute-based sample selection is built upon the intra-class attribute-based sample selection. In the remainder of the section, we use the inter-class method as our default sample selection strategy, which is more effective empirically (Chapter 4.1.5, Ablation Studies). In certain cases, however, where there are not enough desired samples during task construction, we resort to the intra-class sample selection strategy (Chapter 4.1.5, Implementation details).

In the following, we introduce **FewSTAB**, a benchmark framework that uses the proposed selection strategies to construct FSC tasks containing samples with biased attributes for benchmarking spurious bias in few-shot classifiers.

FewSTAB (Part 1): Text-Based Attribute Detection

Our attribute-based sample selection methods require knowing the attributes in images, which typically involves labor-intensive human labeling. To make our method scalable and applicable to few-shot classifiers trained on different datasets, we adopt a pre-trained VLM to automatically identify distinct attributes in images in text format, which includes the following two steps.

Step 1: Generating Text Descriptions. We use a pre-trained VLM [40, 39] ϕ to automatically generate text descriptions for images in $\mathcal{D}_{test}$. The VLM is a model in the general domain and can produce text descriptions for various objects and patterns. For example, for the current input image in the `vase` class in Figure 4.2(a), besides the class object `vase`, the VLM also detects the vase's color `green`, and another object `table` with its material `wooden`.

Step 2: Extracting Informative Words. From the generated text descriptions, we extract nouns and adjectives as the detected attributes via *an automatic procedure* Ω. The two kinds of words are informative as a noun describes an object, and an adjective describes a property of an object. All the detected attributes form the candidate attribute set $\mathcal{A}$. We realize the attribute detector ψ defined in Definition 4.1 as $\psi(x) = \Omega(\phi(x))$.

Remark 1: A VLM in general can extract many distinct attributes from the images. On some images, the VLM may detect non-relevant attributes, such as detecting a duck from a bird image. A more capable VLM could warrant a better attribute detection accuracy and benefit individual measurements on few-shot classifiers. Although being a VLM-dependent benchmark framework, FewSTAB can produce consistent and robust relative measurements among all the compared FSC methods, regardless the choice of VLMs (Chapter 4.1.5, Ablation Studies).

Remark 2: The candidate set $\mathcal{A}$ constructed with all the extracted words may contain attributes that represent the classes in $\mathcal{D}_{test}$. However, during our attribute-based sample selection, these

attributes will not be used since they always correlate with classes and therefore do not satisfy the definition of spurious attributes in Definition 4.1. We provide details of ϕ and Ω in Chapter 4.1.5.

FewSTAB (Part 2): FSC Task Construction

Constructing a C-way $N_{\mathcal{S}}$-shot FSC task $\mathcal{T}$ for benchmarking spurious bias in few-shot classifiers involves constructing a support (training) set $\mathcal{S}$ and a query (test) $\mathcal{Q}$ with biased attributes.

Constructing the Support Set. The support set contains the spurious correlations that we aim to demonstrate to a few-shot classifier. As a fair and rigorous benchmark system, FewSTAB makes no assumptions on the few-shot classifiers being tested and randomly samples C classes from $\mathcal{C}_{test}$. For each sampled class, it randomly selects a spurious correlation $\langle c, a\rangle$ in $\mathcal{D}_{test}$ with $a \in \mathcal{A}$. To effectively demonstrate the spurious correlation $\langle c, a\rangle$ to a few-shot classifier, we select samples of the class c such that (1) they all have the spurious attribute a and (2) do not have spurious attributes from the other $C-1$ spurious correlations. We construct $\mathcal{S}_c$ with $N_{\mathcal{S}}$ samples for the class c that satisfy the above two conditions. Thus, the spurious attribute a becomes predictive of the class c in $\mathcal{S}_c$. We take the union of all C such sets to get $\mathcal{S} = \cup_{c=1}^{C}\mathcal{S}_c$. Figure 4.2(b) demonstrates the case when $C = 3$. Note that we have no requirements for other non-selected attributes in $\mathcal{A}$ to ensure that we have enough samples for $\mathcal{S}_c$.

Constructing the Query Set. To evaluate the robustness to the spurious correlations formulated in $\mathcal{S}$, we first construct a candidate set $\mathcal{I}^{C}_{\langle c,a\rangle}$ in Equation (4.5) for each spurious correlation $\langle c, a\rangle$ in $\mathcal{S}$. Since we have no requirements on the non-selected attributes that are *not* used to formulate spurious correlations in $\mathcal{S}$, a few-shot classifier may predict query samples via some of these attributes, e.g., the yellow blocks in Figure 4.2(b), bypassing the test on the formulated spurious correlations in $\mathcal{S}$. To address this, we propose query sample selection below.

Query sample selection: We select query samples from $\mathcal{I}^{C}_{\langle c,a\rangle}$ that are *least likely* to have non-selected spurious attributes, such as the ones enclosed with red boxes in Figure 4.2(b). To achieve this, we first calculate the fraction of sample-label pairs in $\mathcal{I}^{C}_{\langle c,a\rangle}$ that have the attribute a as

$$p_a = |\{x | a \in \psi(x), \forall (x, c) \in \mathcal{I}^{C}_{\langle c,a\rangle}\}|/|\mathcal{I}^{C}_{\langle c,a\rangle}|, \tag{4.6}$$

where $|\cdot|$ denotes the size of a set, $a \in \tilde{\mathcal{A}}$, and $\tilde{\mathcal{A}}$ contains all non-selected attributes. A larger p_a indicates that the attribute a occurs more frequently in data and is more likely to be used in formulating prediction shortcuts. We then calculate the likelihood score for each $(x, c) \in \mathcal{I}^{C}_{\langle c,a\rangle}$ as $s(x) = \sum_{a\in\psi(x),a\in\tilde{\mathcal{A}}} p_a$, i.e., the summation of all p_a of non-selected attributes in x. The likelihood

VLM	Unique attributes			Avg. attributes per class		
	miniImageNet	tieredImageNet	CUB-200	miniImageNet	tieredImageNet	CUB-200
ViT-GPT2	1111	2532	159	190.40	230.94	25.78
BLIP	2032	6710	247	254.40	310.40	29.74

Table 4.2: Statistics of detected attributes in $\mathcal{D}_{test}$ by two VLMs.

score will be zero if there are no non-selected attributes in x. A large $s(x)$ indicates that the image x can be predicted via many non-selected attributes. Therefore, we select $N_{\mathcal{Q}}$ samples from $\mathcal{I}^C_{\langle c,a\rangle}$ that have the lowest likelihood scores to construct $\mathcal{Q}_c$. Then, we have $\mathcal{Q} = \cup_{c=1}^{C} \mathcal{Q}_c$, which contains samples for evaluating the robustness of a few-shot classifier to the spurious correlations in $\mathcal{S}$.

Complexity Analysis. The text-based attribute detection only needs to use VLMs once to extract attributes for each test set of a dataset. For the task construction, in a nutshell, we analyze the attributes of samples from each of the C classes and do the sampling. Thus, the computational complexity is $O(N_{\mathcal{T}} C N_c N_{\mathcal{A}})$, where N_c is the maximum number of samples per class in test data, $N_{\mathcal{A}}$ is the number of extracted attributes. We only need to run the process *once* and use the generated tasks to benchmark various models.

4.1.5 Experiments

Experimental Setup

Datasets. We used two general datasets, miniImageNet [124] and tieredImageNet [128], and one fine-grained dataset, CUB-200 [163]. Each dataset consists of $\mathcal{D}_{train}$, $\mathcal{D}_{val}$, and $\mathcal{D}_{test}$ for training, validation, and test, respectively (see Appendix A.3). All images were resized to 84×84.

FSC Methods. We trained FSC models with ten algorithms covering three major categories. For gradient-based meta-learning algorithms, we chose ANIL [164], LEO [25], and BOIL [165]. For metric-based meta-learning algorithms, we chose ProtoNet [12], DN4 [166], R2D2 [13], CAN [132], and RENet [167]. For transfer learning algorithms, we chose Baseline++ [64] and RFS [66]. See Appendix A.3 for more details. Any backbones can be used as the feature extractor. For fair comparisons between different methods, we used the ResNet-12 backbone adopted in [130].

Text-Based Attribute Detection. We used a pre-trained VLM named ViT-GPT2 [40] to generate text descriptions for images in $\mathcal{D}_{test}$. After that, we used Spacy (`https://spacy.io/`) to extract nouns and adjectives from these descriptions automatically. We also used another pre-trained VLM, BLIP [39], to test whether FewSTAB can produce consistent results. The statistics of the detected attributes are shown in Table 4.2.

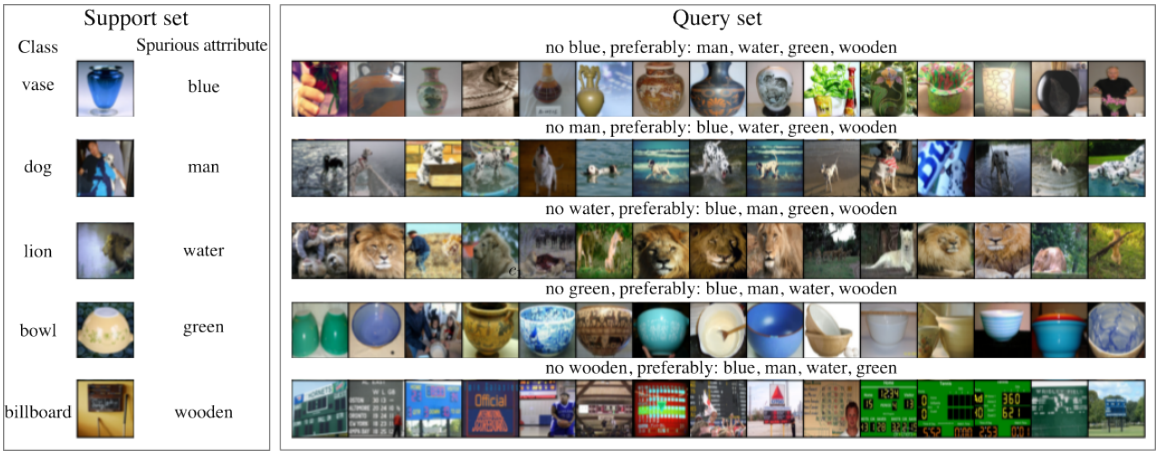


Figure 4.3: A 5-way 1-shot task constructed by our inter-class attribute-based sample selection using samples from the miniImageNet dataset. Note that due to the limited capacity of a VLM, the attributes may not well align with human understandings.

Implementation Details. We trained FSC models with the implementation in [161]. Each model was trained on $\mathcal{D}_{train}$ of a dataset with one of the ten FSC methods. For each meta-learning based method, we trained two models using randomly sampled 5-way 1-shot and 5-way 5-shot tasks, respectively. All the tasks have 15 samples per class in the query set. We saved the model that achieves the best validation accuracy on $\mathcal{D}_{val}$ for evaluation. For FewSTAB, if we do not have enough desired samples to construct a support set, we redo the construction from the beginning. If there are not enough desired samples to construct a query set, we first try to use the intra-class attribute-based sample selection; if the desired samples are still not enough, we redo the construction from the beginning. We created 3000 tasks for model evaluation. All experiments were conducted on the NVIDIA RTX 8000 GPUs.

Visualization of a Constructed Task

We show a 5-way 1-shot task constructed by FewSTAB in Figure 4.3. Each class in the support set correlates with a unique spurious attribute. The query samples of a class do not have the spurious attribute correlated with the class and some of them have spurious attributes associated with other classes in the support set. For example, the query samples of the class `lion` do not have the spurious attribute `water`, and some of them have spurious attributes from other classes in the support set, such as `man` and `green`. FewSTAB introduces biased attributes in the task so that query samples can be easily misclassified as other classes by a few-shot classifier that relies on the spurious correlations in the support set.

Shot	Method	miniImageNet		tieredImageNet		CUB-200	
		wAcc-A (↑)	wAcc-R (↑)	wAcc-A (↑)	wAcc-R (↑)	wAcc-A (↑)	wAcc-R (↑)
1	ANIL	10.38±0.30	14.36±0.33	11.21±0.30	15.63±0.36	13.78±0.40	16.94±0.43
	LEO	14.26±0.46	21.35±0.54	16.00±0.55	29.63±0.71	28.29±0.80	40.22±0.87
	BOIL	12.48±0.23	14.93±0.24	12.27±0.21	14.13±0.23	19.15±0.29	22.50±0.29
	ProtoNet	14.03±0.49	21.96±0.58	14.50±0.50	27.13±0.69	34.62±0.85	46.61±0.89
	DN4	12.37±0.46	19.28±0.56	11.99±0.47	23.62±0.65	35.22±0.86	47.26±0.88
	R2D2	18.05±0.53	26.50±0.60	16.41±0.54	30.41±0.71	36.70±0.90	48.82±0.88
	CAN	17.37±0.53	25.96±0.60	18.84±0.60	36.29±0.78	22.74±0.72	31.95±0.78
	RENet	19.10±0.57	28.85±0.65	18.83±0.61	35.70±0.78	32.43±0.81	43.98±0.81
	Baseline++	15.30±0.48	23.18±0.56	17.51±0.54	31.62±0.71	9.17±0.47	14.59±0.58
	RFS	18.00±0.53	27.12±0.61	18.35±0.60	35.24±0.77	32.45±0.80	44.49±0.82
5	ANIL	14.83±0.40	25.37±0.52	13.72±0.39	30.60±0.51	31.63±0.55	45.47±0.53
	LEO	26.31±0.59	41.33±0.59	29.49±0.72	57.22±0.72	46.62±0.82	59.76±0.77
	BOIL	13.09±0.22	15.21±0.23	14.90±0.22	18.55±0.24	19.17±0.28	21.33±0.27
	ProtoNet	32.07±0.58	51.95±0.52	30.95±0.70	62.53±0.62	60.06±0.74	75.68±0.50
	DN4	27.60±0.58	42.68±0.62	16.07±0.62	40.63±0.81	59.25±0.77	73.58±0.56
	R2D2	35.37±0.59	50.84±0.55	31.12±0.72	61.08±0.65	58.66±0.82	75.20±0.54
	CAN	36.44±0.65	54.23±0.55	31.17±0.76	64.19±0.62	41.31±0.74	61.61±0.58
	RENet	36.19±0.63	56.52±0.58	30.27±0.76	63.49±0.64	52.93±0.82	71.82±0.56
	Baseline++	29.52±0.57	44.94±0.57	30.01±0.72	59.06±0.67	16.86±0.52	29.84±0.69
	RFS	36.85±0.64	55.66±0.55	31.15±0.76	62.71±0.67	54.98±0.81	74.33±0.53

Table 4.3: Comparison between wAcc-R and wAcc-A with 95% confidence interval on the miniImageNet, tieredImageNet, and CUB datasets. Numbers in the Shot column indicate that the models are both trained (if applicable) and tested on 5-way 1- or 5-shot tasks. Darker colors indicate higher values.

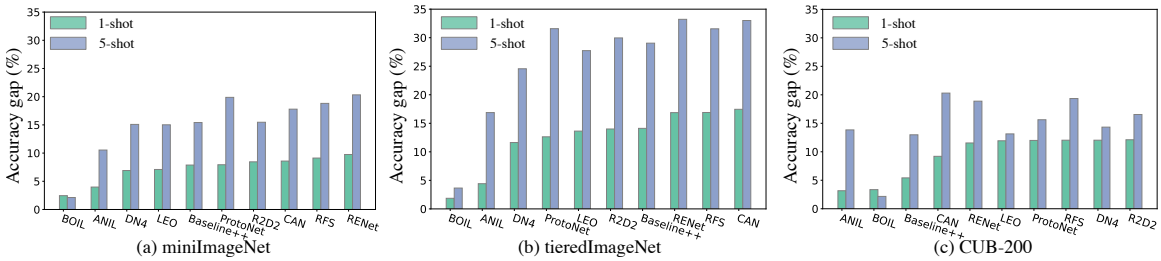


Figure 4.4: Accuracy gaps (wAcc-R minus wAcc-A) on the 5-way 1-shot and 5-way 5-shot tasks from the (a) miniImageNet, (b) tieredImageNet, and (c) CUB-200 datasets.

Effectiveness of FewSTAB

FewSTAB Can Effectively Reveal Spurious Bias in Few-shot Classifiers. We show in Table 4.3 the wAcc (Equation (4.2)) on 5-way 1/5-shot test tasks that are randomly sampled (wAcc-R) and are constructed with FewSTAB (wAcc-A), respectively. FewSTAB generates FSC test tasks only based on the class-attribute correlations in data. In each test setting, the FSC methods in Table 4.3 are evaluated with the same FSC tasks. We observe that wAcc-A is consistently lower than wAcc-R on the three datasets and on two test-shot numbers, showing that FewSTAB is more effective than the standard evaluation procedure (random task construction) in exhibiting the spurious bias in

various few-shot classifiers. We additionally show that FewSTAB also works on the most recent FSC methods and can reflect the improvement made to mitigate spurious bias (see Appendix A.3).

FewSTAB Reveals New Robustness Patterns among FSC Methods. In Table 4.4, we calculate the Spearman's rank correlation coefficients [168] between the values of wAcc-R and wAcc-A from Table 4.3. The coefficients are bounded from 0 to 1, with larger values indicating that the ranks of FSC methods based on wAcc-R are more similar to those based on wAcc-A. In the 1-shot setting, it is not effective to control the spurious correlations since we only have one sample per class in the support set. Hence, the coefficients are large, and the ranks based on wAcc-A are similar to those based on wAcc-R. In the 5-shot cases, we have more samples to demonstrate the spurious correlations. The coefficients become smaller, i.e., the ranks based on wAcc-A show different trends from those based on wAcc-R. In this case, FewSTAB reveals new information on FSC methods' varied degrees of robustness to spurious bias.

Dataset	1-shot	5-shot
miniImageNet	0.96	0.95
tieredImageNet	0.96	0.90
CUB-200	1.00	0.94

Table 4.4: Spearman's rank correlations between wAcc-A and wAcc-R in Table 4.3.

FewSTAB Can Benchmark Spurious Bias in Varied Degrees. As shown in Figure 4.4, the accuracy gap, defined as wAcc-R minus wAcc-A, in general, becomes larger when we switch from 5-way 1-shot to 5-way 5-shot tasks. Compared with the random task construction, FewSTAB creates more challenging tasks in the 5-shot case for demonstrating spurious bias in few-shot classifiers. In other words, with a higher shot value in the constructed test tasks, FewSTAB aims to benchmark spurious bias in a higher degree.

A New Dimension of Evaluation and a New Design Guideline

FewSTAB creates a new dimension of evaluation on the robustness to spurious bias. We demonstrate this with a scatter plot (Figure 4.5) of Acc (Equation (4.1)) and wAcc-A of the ten few-shot classifiers. FewSTAB offers new information regarding different few-shot classifiers' robustness to spurious bias as we observe that Acc does not well correlate with wAcc-A. A high wAcc-A indicates that the classifier is robust to spurious bias, while a high Acc indicates that the classifier can correctly predict most

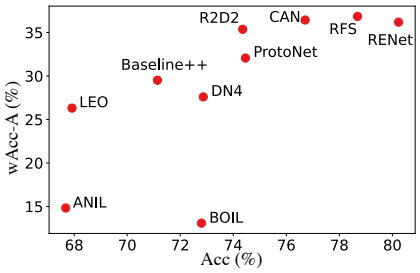


Figure 4.5: Acc versus wAcc-A of the ten FSC methods tested on 5-way 5-shot tasks from miniImageNet.

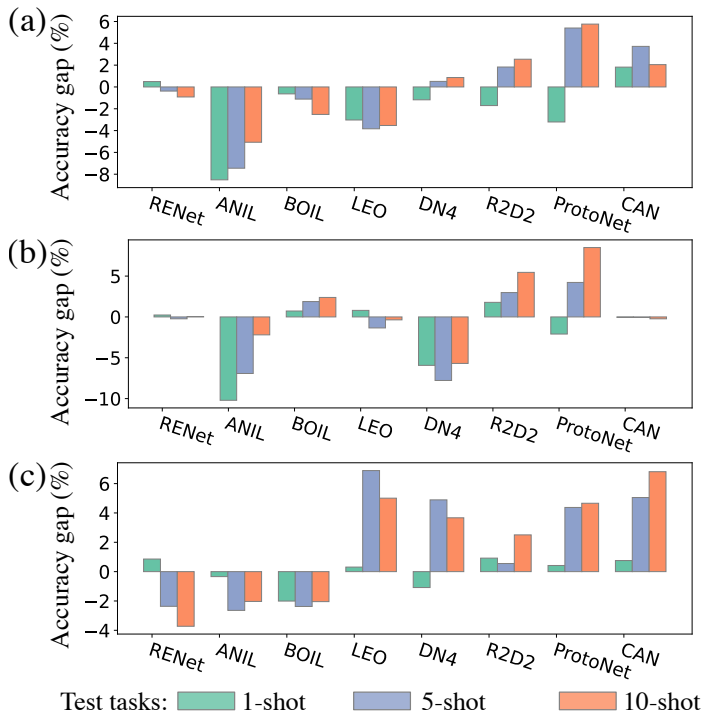


Figure 4.6: Accuracy gaps of few-shot classifiers tested on 1-shot, 5-shot, and 10-shot tasks constructed from (a) miniImageNet, (b) tieredImageNet, and (c) CUB-200 datasets.

of the samples. With the scatter plot, we can view tradeoffs between the two metrics on existing few-shot classifiers. A desirable few-shot classifier should appear in the top-right corner of the plot.

FewSTAB Enables Designs for Varied Degrees of Robustness

As demonstrated in Chapter 4.1.5, FewSTAB can benchmark spurious bias in varied degrees, which in turn enables practitioners to design robust few-shot classifiers targeted for different degrees of robustness to spurious bias. The reason for differentiating designs for varied degrees of robustness is that the same design choice may not work under different robustness requirements. For example, increasing shot number in training tasks is a common strategy for improving the few-shot generalization of meta-learning based methods. We trained few-shot classifiers with 5-way 5-shot and 5-way 1-shot training tasks randomly sampled from $\mathcal{D}_{train}$, respectively. We then calculated the *accuracy gap defined as the wAcc-A of a model trained on 5-shot tasks minus the wAcc-A of the same model trained on 1-shot tasks*. A positive and large accuracy gap indicates that this strategy is effective in improving the model's robustness to spurious bias. In Figure 4.6, on each of the three datasets, we give results of the eight meta-learning based FSC methods on the 5-way 1-, 5-, and 10-shot FewSTAB tasks which are used to demonstrate the strategy's robustness to increased degrees of

Attribute-based sample selection		Query sample selection	Avg. drop (%)
Intra-class	Inter-class		
✓			5.13
✓	✓		13.30
✓	✓	✓	15.05

Table 4.5: Comparison between different techniques used by FewSTAB for constructing FSC tasks.

Dataset	1-shot	5-shot
miniImageNet	0.98	1.00
tieredImageNet	1.00	0.99
CUB-200	1.00	0.98

Table 4.6: Spearman's rank correlation coefficients between wAcc-A obtained using ViT-GPT2 and BLIP.

spurious bias. This strategy does not work consistently under different test shots. For example, in Figure 4.6(a) this strategy with CAN only works the best on the 5-way 5-shot FewSTAB tasks.

Ablation Studies

Techniques Used in FewSTAB. We analyze how different sample selection methods affect the effectiveness of FewSTAB in Table 4.5. With only intra-class attribute-based sample selection, we randomly select query samples from Equation (4.3). For inter-class attribute-based sample selection and intra-class attribute-based sample selection (automatically included by Equation (4.4)), we randomly select query samples from Equation (4.5). FewSTAB uses all the techniques in Table 4.5. We define accuracy drop as wAcc-R minus wAcc-A, and we use the drop averaged over the ten FSC methods tested on 5-way 5-shot tasks from the miniImageNet dataset as our metric. A larger average drop indicates that the corresponding sample selection method is more effective in reflecting the spurious bias in few-shot classifiers. We observe that all proposed techniques are effective and the inter-class attribute-based sample selection is the most effective method.

Choice of VLMs. Although our main results are based on the pre-trained ViT-GPT2 model [40], we show in Table 4.6 that when switching to a different VLM, i.e., BLIP [39], the relative ranks of different few-shot classifiers based on wAcc-A still hold with high correlations. In other words, FewSTAB is robust to different choices of VLMs.

Detection Accuracy of VLMs. A VLM may miss some attributes due to its limited capacity, resulting in a small detection accuracy. However, the detection accuracy of a VLM has little impact on our framework. To demonstrate this, we adopt a cross-validation strategy, i.e., we use the outputs from one VLM as the ground truth to evaluate those from another VLM, since assessing the detection accuracy of a VLM typically requires labor-intensive human labeling. On the CUB-200 dataset, we observe that the detection accuracy of ViT-GPT2 based on the BLIP's outputs is 70.12%, while the detection accuracy of BLIP based on the ViT-GPT2's outputs is 59.28%. Although the two VLMs

differ significantly in the detected attributes, our framework shows almost consistent rankings of the evaluated FSC methods (Table 4.6).
Additional results are presented in Appendix A.3.

4.1.6 Conclusion

In this section, we propose attribute-based sample selection strategies with a pre-trained VLM to select samples with subpopulation shifts for revealing spurious bias in models. These strategies have been proven effective in FSC, where we designed a systematic and rigorous benchmark framework called FewSTAB for evaluating the robustness of few-shot classifiers to spurious bias. FewSTAB adopts these sample selection strategies to construct FSC test tasks with subpopulation shifts so that the reliance on spurious correlations can be effectively revealed. FewSTAB can automatically benchmark spurious bias in few-shot classifiers on any existing test data thanks to its use of a pre-trained VLM for automated attribute detection. With FewSTAB, we provided a new dimension of evaluation on the robustness of few-shot image classifiers to spurious bias and a new design guideline for building robust few-shot classifiers. FewSTAB can reveal and enable designs for varied degrees of robustness to spurious bias. In the following sections, we will exploit these strategies for automatic spurious bias mitigation.

4.2 Learning Robust Classifiers with Self-Guided Spurious Correlation Mitigation

4.2.1 Introduction

Deep neural classifiers have shown strong empirical performance in many application areas. However, some of the high performance may be credited to their strong reliance on spurious correlations [3, 33, 34], which are brittle associations between non-essential spurious attributes of inputs and the corresponding targets in many real-world datasets. For example, a deep neural classifier trained with empirical risk minimization (ERM) can achieve a high accuracy of predicting cow by just detecting the `grassland` attribute of images, given that cow correlates with `grassland` in most images. However, the correlation is spurious as the `grassland` attribute is not essential for the class cow, and the classifier exhibits severe performance degradation on images showing a cow at a beach [6, 2].

Mitigating the reliance on spurious correlations is crucial for obtaining robust models. Existing methods typically assume that spurious correlations are known (1) fully in both the training and validation data for model training and selection [169, 30] or (2) only in the validation data for model selection [31, 82, 4, 170]. However, obtaining annotations of spurious correlations typically requires expert knowledge and human supervision, which is a significant barrier in practice.

In this section, we tackle the setting where annotations of spurious correlations are not available. To train a classifier robust to spurious correlations without knowing them, we propose a novel self-guided spurious correlation mitigation framework that automatically detects and analyzes the classifier's reliance on spurious correlations and relabels training data tailored for spurious correlation mitigation.

Our framework exploits the classifier's reliance on individual attributes contained in multiple training samples. To this end, we first propose to automatically detect all possible attributes from a target dataset based on a pre-trained vision-language model (VLM). The VLM learns the mapping between real-world images and their text descriptions. Thus, it is convenient to extract informative words from the descriptions as the attributes which summarize similar input features that could be exploited by a classifier for predictions.

The detected attributes and class labels formulate all possible correlations that the classifier might exploit for predictions. The classifier may exploit some of the correlations for predictions and be invariant to others. To measure the classifier's reliance on these class-attribute correlations or their degrees of spuriousness, we propose a spuriousness metric to quantify how likely the classifier relies on the correlations in a set of data. Given a class-attribute correlation, a large value of the metric shows the classifier's strong reliance on the correlation and a significant impact of the correlation on the classifier's performance.

With the spuriousness values for all the possible correlations, the prediction behaviors of the classifier on samples naturally emerge. The spuriousness values of all the class-attribute correlations relevant to a sample collectively characterize the classifier's prediction behavior on the sample, i.e., how likely those correlations are exploited by the classifier for predicting the class label of the sample. Therefore, we can discover diverse prediction patterns of the classifier. For example, some of the prediction behaviors are frequently used on many samples while some are not.

To mitigate the classifier's reliance on spurious correlations, we demonstrate to the classifier that multiple prediction behaviors for samples in the same class are different but should lead to the same class label. In this way, the classifier is not only aware of multiple attributes leading to the same class, but also encouraged to discover more robust features for predictions. To achieve this, we relabel the

training data with fine-grained labels so that the classifier is trained to distinguish samples from the same class with different prediction behaviors. Moreover, considering the imbalanced sample distributions over different prediction behaviors, we adopt a balanced sampling approach to train the adapted classifier for improved robustness against spurious correlations.

We consolidate the detection and mitigation methods in an iterative learning procedure since it is possible that mitigating certain spurious prediction behaviors may increase the reliance on others. Our method, termed as *Learning beyond Classes (LBC)*, has the following **contributions**:

- We completely remove the spurious correlation annotation barrier for learning a robust classifier by proposing an automatic detection method that exploits the prior knowledge in a pre-trained vision-language model.
- We mitigate a classifier's reliance on spurious correlations by diversifying its outputs to recognize different prediction behaviors and balancing its training data.
- Our method debiases a biased classifier with a new self-guided procedure of iteratively identifying and mitigating the classifier's spurious prediction behaviors. We demonstrate that LBC achieves the best performance on five real-world datasets where spurious correlations are unknown or unavailable.

4.2.2 Problem Formulation

We focus on an ERM-trained classifier f_θ with parameters θ learned from a training dataset $\mathcal{D}_{tr} = \{(x_n, y_n)\}_{n=1}^{N}$ of N pairs of image $x_n \in \mathcal{X}$ and label $y_n \in \mathcal{C}$, i.e.,

$$\theta = \arg\min_{\theta'} \mathbb{E}_{(x,y)\in\mathcal{D}_{\text{tr}}} \ell(f_{\theta'}(x), y), \tag{4.7}$$

where $\mathcal{X}$ is the set of all input samples, $\mathcal{C}$ is the set of classes, and $\ell(\cdot,\cdot)$ is the cross-entropy loss function. Before we introduce the problem regarding the classifier, we describe the following concepts we use in this section.

Group Labels. A group label $\langle c, a\rangle$ is a fine-grained label that uniquely annotates a group of samples which are in the class c and have the attribute a.

Spurious Attributes. A spurious attribute a describes non-predictive conceptual features of inputs. For example, the spurious attribute $a =$ `water` may describe different water backgrounds in images.

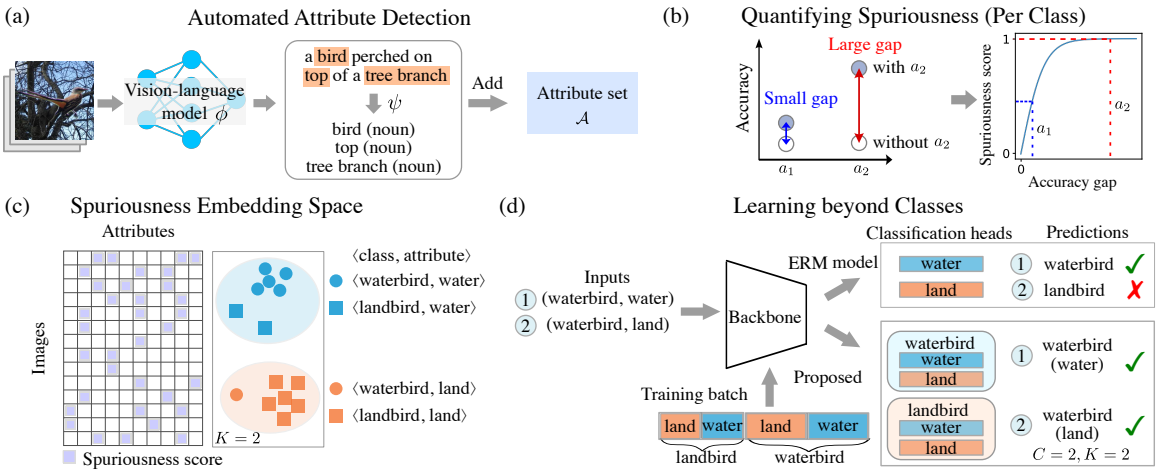


Figure 4.7: Method overview. (a) Detecting attributes with a pre-trained VLM. (b) Quantifying the spuriousness of correlations between classes and detected attributes. (c) Clustering in the spuriousness embedding space for relabeling the training data. (d) Diversifying the outputs of the classifier and training the classifier with balanced training data.

Spurious Correlations. A spurious correlation is the association between a spurious attribute a and a class c that exists *only* in some samples of class c. We use $\langle c, a \rangle$ to denote both a group label and a spurious correlation if a is spurious.

A real-world dataset can be partitioned into several groups with different group labels. Typically, a classifier f_θ can perform well on in-distribution data. However, if the group labels are imbalanced over the dataset, the classifier will excessively learn from the spurious correlations in the majority groups and potentially ignore those in the minority groups. This phenomenon exposes a robustness issue of the classifier, especially when it is deployed in an environment where data distributions are more shifted towards the minority groups.

Existing problem formulations assume the availability of group labels in both the training and validation data for model training and selection, respectively, or in the validation data only, where the annotations of the group labels are often expensive to obtain. Here, we consider the novel setting where **no group labels are available**, which completely removes the barrier of human annotation in existing methods.

4.2.3 Methodology

Automated Spurious Correlation Detection

Without knowing what spurious correlations to be mitigated, we propose an automated and scalable method to discover all potential spurious correlations in a target dataset. Our method exploits the

prior knowledge in a pre-trained vision language model (VLM) and consists of automatic attribute detection and quantifying the spuriousness of correlations between detected attributes and classes.

Automatic Attribute Detection

The detection procedure has the following two steps.

1. Generating Text Descriptions. We use a pre-trained VLM ϕ to automatically generate text descriptions of images without human supervisions. Since the model is general-purposed and is not specifically fine-tuned on the target dataset, it can discover general objects and patterns. For example, in Figure 4.7(a), besides the class object `bird`, the model also detects `tree branch`.

2. Extracting Informative Words. We detect attributes by identifying nouns and adjectives from text descriptions as these types of words are informative in representing objects and patterns in images. We use an automatic procedure ψ (Chapter 4.2.4) to extract these informative words from the text descriptions obtained in the first step. For example, we extract `bird`, `top`, and `tree branch` from the description as shown in Figure 4.7(a). We add the detected attributes to a set $\mathcal{A}$ as the set of all attributes detected from the images.

Remark: It is possible that the pre-trained VLM may generate inaccurate text descriptions for some images due to its inductive bias learned during pre-training. For example, it may describe a lemon on a tree as a yellow bird. However, in this case, we observe that `bird` is still informative as it refers to any object with a tree background with similar attributes. Therefore, although some extracted words may not be self-explanatory in representing certain kinds of features, they can still be representative of many similar samples.

Quantifying Spuriousness

Not all detected attributes in $\mathcal{A}$ may form a spurious correlation with a class since some correlations may not have corresponding images, or some attributes are not spurious and represent class objects. More importantly, not all correlations are exploited by a classifier. To quantify the likelihood of a class-attribute correlation being spurious and used by a classifier, i.e., *spuriousness* of the correlation, we propose a novel metric, termed *spuriousness score*, that unifies the above cases.

Our spuriousness score for $\langle c, a\rangle$ is motivated by the observation that the classifier f_θ will generalize poorly on samples of class c without the attribute a if f_θ excessively relies on a for predicting the class c. Intuitively, the score will be higher if f_θ has a larger performance discrepancy on images with and without a and vice versa. We formally define our spuriousness score for $\langle c, a\rangle$ as follows.

Definition 1 (Spuriousness Score): Given a class $c \in \mathcal{C}$, an attribute $a \in \mathcal{A}$, and a classifier f_θ trained on $\mathcal{D}_{tr}$, the spuriousness of $\langle c, a \rangle$ is calculated as follows,

$$\gamma(a, c; f_\theta, \mathcal{D}_{tr}) = \tanh\Big(\text{Abs}(\log \frac{M(\mathcal{D}_{tr}^{\langle c,a \rangle}; f_\theta)}{M(\mathcal{D}_{tr}^{\langle c,\hat{a} \rangle}; f_\theta)})\Big), \tag{4.8}$$

with $\gamma(a, c; f_\theta, \mathcal{D}_{tr}) = 0$ when $\mathcal{D}_{tr}^{\langle c,\hat{a} \rangle} = \emptyset$ or $\mathcal{D}_{tr}^{\langle c,a \rangle} = \emptyset$, where $\mathcal{D}_{tr}^{\langle c,a \rangle} \subset \mathcal{D}_{tr}$ denotes the subset of all training samples from class c **with** the attribute a, $\mathcal{D}_{tr}^{\langle c,\hat{a} \rangle} \subset \mathcal{D}_{tr}$ denotes the subset of all training samples from class c **without** the attribute a, $M(\cdot; f_\theta)$ denotes the classification accuracy of f_θ on a given set of samples, and $\text{Abs}(\cdot)$ denotes taking the absolute value. Moreover, the division can produce larger values than the simple difference between the two accuracies, making different correlations more distinctive. Using $\log(\cdot)$ avoids encountering extreme values from the division, and $\tanh(\text{Abs}(\cdot))$ bounds the score in the range from 0 to 1. Figure 4.7(b) gives an example.

Remark: When $\mathcal{D}_{tr}^{\langle c,\hat{a} \rangle} = \emptyset$, a is always associated with c in $\mathcal{D}_{tr}$, e.g., a is a class object. When $\mathcal{D}_{tr}^{\langle c,a \rangle} = \emptyset$, a and c are not associated, i.e., no samples correspond to the association between a and c. In both cases, the spuriousness score is zero as they do not fit in our description of spurious correlations. We exclude such attributes from $\mathcal{A}$. Moreover, the $\text{Abs}(\cdot)$ operator implies that when the fraction in Equation (4.8) is lower than 1, i.e., a leads to incorrect predictions other than c and results in a small nominator, $\langle c, a \rangle$ still has a high spuriousness score as it also represents a robustness pitfall.

With the spuriousness score, we can measure how likely the detected attribute a and the class c form a spurious correlation that is exploited by the classifier f_θ. The lower the spuriousness score, the less f_θ relies on attribute a in its prediction, thus the more robust the classifier is.

Spuriousness-Guided Training Data Relabeling

Applying our spuriousness score (Equation (4.8)) to all the possible correlations between the classes and the detected attributes, we can holistically characterize the spuriousness of images by showing how likely correlations relevant to an image are exploited by the classifier in predicting the image's class label. To analyze the spuriousness of all the training images and pinpoint the robustness pitfalls of the classifier, we propose *spuriousness embedding*, which is defined as follows.

Definition 2 (Spuriousness Embedding): Given an image x with label y, a classifier f_θ, the detected attribute set $\mathcal{A}$ with N_A attributes, a VLM ϕ, an attribute extraction procedure ψ, and the training set $\mathcal{D}_{tr}$, we design the spuriousness embedding for (x, y), denoted as $SE(x, y)$, as a

N_A-dimensional vector, whose i_a-th element is defined as follows,

$$SE(x, y)[i_a] = \gamma(a, y; f_\theta, \mathcal{D}_{tr}) \cdot \mathbb{1}_{a \in \psi(\phi(x))}, a \in \mathcal{A}, \tag{4.9}$$

where a denotes an attribute of x, i_a denotes the dimension index of $SE(x, y)$ that corresponds to a, and $\mathbb{1}_{\{\cdot\}}$ is the indicator function that equals one if the condition in the subscript is true and equals zero otherwise.

With spuriousness embeddings, we embed all images in $\mathcal{D}_{tr}$ in the spuriousness embedding space. Each point in the space represents both an image and a vector characterizing an individual spurious prediction behavior of f_θ in using relevant class-attribute correlations to predict the image.

In the ideal scenario, a robust classifier produces all zero vectors in the space; practically, we expect to observe a dispersed distribution of points as the classifier does not excessively rely on specific spurious prediction behaviors. In contrast, a biased classifier tends to produce an uneven distribution of points from the same class. We demonstrate in Figure 4.7(c) with two clusters of similar prediction behaviors frequently used in predictions, i.e., using the water or land attributes. The partition based on the spuriousness of the images separates same-class samples unevenly, exposing potential prediction failure modes of the classifier, such as predicting a waterbird image with a land background as a landbird.

Therefore, we cluster images into K clusters in the spuriousness embedding space across different classes to capture potential failure modes in classification. Here, K is a hyperparameter and is controlled by our design choice. Formally, we represent the above process as follows,

$$p_K(x, y) = CLU(SE(x, y); \mathcal{D}_{tr}, K), \forall (x, y) \in \mathcal{D}_{tr}, \tag{4.10}$$

where $p_K(x, y)$ is the cluster label for (x, y), and CLU denotes a clustering algorithm. In the following, we use KMeans as CLU. Now, $p_K(x, y)$ denotes a *super-attribute* that covers similar detected attributes that may be used to predict the class label y. For example, the blue cluster in Figure 4.7(c) represents the `water` super-attribute which may cover relevant attributes such as `pond` and `river`.

We use the cluster labels $p_K(x, y)$ to guide the debiasing of the classifier by relabeling training data with fine-grained labels formulated with $p_K(x, y)$ and y, which we unify as one symbol $g_K(x, y) = p_K(x, y) + (y - 1) \cdot K$.

Learning beyond Classes

The spuriousness-guided training data relabeling provides fine-grained training labels defined by the cluster labels and classes. To effectively use the relabeled training data for robust classifier learning, we propose two novel strategies along with training and inference procedures in the following.

Diversifying Outputs of the Classifier

The constructed training labels instruct the classifier that multiple prediction behaviors for the same class should lead to a correct and consistent prediction outcome. To achieve this, we diversify the outputs of the classifier from predicting class labels to distinguishing between different prediction behaviors for the same class. In this way, the classifier is not only aware of other attributes leading to the same class, but also encouraged to discover more robust features for predictions.

Specifically, we separate f_θ into a backbone e_{θ_1} and a C-way classification head q_{θ_2}, i.e, $f_\theta = q_{\theta_2} \circ e_{\theta_1}$, where C is the number of classes, and the parameters $\theta = \theta_1 \cup \theta_2$. Then, we replace q_{θ_2} with h_{θ_3} which is a $(K \cdot C)$-way classification head, resulting in a transformed model $\tilde{f}_{\tilde{\theta}} = h_{\theta_3} \circ e_{\theta_1}$ with $\tilde{\theta} = \theta_3 \cup \theta_1$. Each output of $\tilde{f}_{\tilde{\theta}}$ corresponds to a combination of class c and cluster label k. Figure 4.7(d) gives an example with $K = 2$ and $C = 2$. In this example, instead of predicting two classes, i.e., waterbird and landbird, we replace the classification head of the ERM model with a new classification head on the correlations: (waterbird-`water`), (waterbird-`land`), (landbird-`water`), and (landbird-`land`).

Balancing Training Data

As discussed in Chapter 4.2.3, different prediction behaviors may correspond to uneven numbers of samples, which may bias the predictions of the adapted classifier. To address this, we consider within-class and cross-class balancing strategies.

Within-Class Balancing. We sample K equal-sized training sets $\mathcal{B}_c^k \subseteq \mathcal{G}_c^k$ with images from class c and K different clusters, where $\mathcal{G}_c^k$ is defined as

$$\mathcal{G}_c^k = \{(x, y) | y = c, p_K(x, y) = k, \forall (x, y) \in \mathcal{D}_{tr}\}. \tag{4.11}$$

In this way, we assign equal importance to predicting the K clusters within the class c.

Cross-Class Balancing. Considering that predictions for different classes may show varied reliance on spurious correlations, we additionally balance the size of $\mathcal{B}_c = \cup_{k=1}^{K} \mathcal{B}_c^k$. Specifically, for each class c, we calculate the variance of cluster sizes within class c, i.e., $\sigma_c = Var(\{|\mathcal{G}_c^k| \ | k = 1, \ldots, K\})$, where $|\cdot|$ denotes the size of a set, and $Var(\cdot)$ denotes calculating the variance of a set of numbers. The variance measures the degree of imbalanced prediction behaviors for class c. We control the

size of $\mathcal{B}_c$ based on σ_c such that we sample more training data for a class that exhibits a larger degree of imbalanced prediction behaviors. Concretely, given the batch size B, we set $|\mathcal{B}_c| = B \cdot \rho_c$, $|\mathcal{B}_c^k| = B \cdot \rho_c / K$, where $\rho_c = \log(\sigma_c) / \sum_{c=1}^{C} \log \sigma_c$, and we use $\log(\cdot)$ to avoid encountering extreme values.

Training and Inference

The overall learning objective is as follows,

$$\tilde{\theta}^* = \arg\min_{\tilde{\theta}} \mathbb{E}_{\mathcal{B} \sim \mathcal{D}_{tr}} \mathbb{E}_{(x,y) \in \mathcal{B}} \ell(\tilde{f}_{\tilde{\theta}}(x), g_K(x, y)), \tag{4.12}$$

where $\mathcal{B} = \cup_{c=1}^{C} \mathcal{B}_c$.

It is possible that after training on Equation (4.12), the model develops reliance on other spurious correlations. Therefore, we iteratively update the spuriousness scores based on the updated model and perform the mitigation procedure again. We call this method *learning beyond classes (LBC)*. The whole procedure is listed in Algorithm 1 in Appendix A.4.1.

Model Selection. Given a validation set $\mathcal{D}_{val}$ without group labels, we develop a selection metric called *pseudo unbiased validation accuracy* $Acc_{unbiased}^{pseudo}$ to select the best model during training. Specifically, we group the validation data based on the existence of the detected attribute a, i.e.,

$$\mathcal{D}_{val}^{a} = \{(x, y) | (x, y) \in \mathcal{D}_{val}, a \in \psi(\phi(x))\}. \tag{4.13}$$

Then, we calculate $Acc_{unbiased}^{pseudo}$ as the average over the accuracy on $\mathcal{D}_{val}^{a}$ as follows,

$$Acc_{unbiased}^{pseudo} = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} M(\mathcal{D}_{val}^{a}; \tilde{f}_{\tilde{\theta}}). \tag{4.14}$$

During inference, the predicted label is calculated as $\hat{c} = \lceil (\arg\max_c \tilde{f}_{\tilde{\theta}}(x)) / K \rceil$, where $\lceil c \rceil$ denotes taking the smallest integer greater than or equal to c.

4.2.4 Experiments

Datasets

Waterbirds. Waterbirds [3] is a dataset for recognizing waterbirds and landbirds. It is generated synthetically by combining images of the two birds from the CUB dataset [163] and the backgrounds, water, and land, from the Places dataset [174], producing (waterbird, water), (waterbird, land), (landbird, land), and (landbird, water) groups.

Method	Backbone	Group information		CelebA			Waterbirds		
		Train	Validation	Worst(↑)	Average(↑)	Gap(↓)	Worst(↑)	Average(↑)	Gap(↓)
GroupDRO [3]	ResNet-50	✓	✓	88.9	92.9	4.0	91.4	93.5	2.1
LfF [33]	ResNet-50	✗	✓	78.0	85.1	7.1	78.0	91.2	13.2
CVaR DRO [171]	ResNet-50	✗	✓	64.4	82.5	18.1	75.9	96.0	20.1
JTT [31]	ResNet-50	✗	✓	81.1	88.0	6.9	86.7	93.3	6.6
DFR [4]	ResNet-50	✗	✓	$69.4_{\pm 1.4}$	$93.3_{\pm 0.1}$	$23.9_{\pm 1.3}$	$80.2_{\pm 2.3}$	$92.1_{\pm 0.6}$	$11.9_{\pm 2.7}$
LBC (Ours)	ResNet-50	✗	✓	$\mathbf{87.4}_{\pm 1.8}$	$92.4_{\pm 0.3}$	$\mathbf{5.0}_{\pm 2.1}$	$\mathbf{88.1}_{\pm 1.4}$	$94.1_{\pm 0.3}$	$\mathbf{6.0}_{\pm 1.7}$
ERM	ResNet-50	✗	✗	45.7	95.5	49.8	66.4	90.2	23.8
LfF [33]	ResNet-50	✗	✗	24.4	85.1	60.7	44.1	91.2	47.1
CVaR DRO [171]	ResNet-50	✗	✗	36.1	82.5	46.4	62.0	95.2	33.2
JTT [31]	ResNet-50	✗	✗	40.6	88.0	47.4	62.5	93.3	30.8
DivDis [172]	ResNet-50	✗	✗	55.0	90.8	35.8	81.0	90.7	9.7
MaskTune [173]	ResNet-50	✗	✗	$78.0_{\pm 1.2}$	$91.3_{\pm 0.1}$	$13.3_{\pm 1.3}$	$86.4_{\pm 1.9}$	$93.0_{\pm 0.7}$	$6.6_{\pm 2.6}$
DFR [4]	ResNet-50	✗	✗	46.0	95.8	49.8	77.4	92.1	14.7
LBC (Ours)	ResNet-50	✗	✗	$\mathbf{81.2}_{\pm 1.5}$	$92.2_{\pm 0.3}$	$\mathbf{11.0}_{\pm 1.8}$	$\mathbf{87.3}_{\pm 1.8}$	$93.2_{\pm 0.9}$	$\mathbf{5.9}_{\pm 2.7}$

Table 4.7: Worst-group and average accuracy (%) comparison with state-of-the-art methods on the CelebA and Waterbirds datasets. The ResNet-50 backbones are pretrained on ImageNet. GroupDRO reveals the theoretically best performance given all the group information in worst-group results and performance gaps. The best worst-group results and performance gaps are in **boldface**.

CelebA. CelebA [149] is a large-scale image dataset of celebrity faces. The task is to identify hair color, non-blond or blond, with gender as the spurious attribute. There are four groups in the CelebA dataset: (blond, male), (blond, female), (non-blond, male), (non-blond, female).

ImageNet-9. ImageNet-9 [49] comprises images with different background and foreground signals, which can be used to assess how much models rely on image backgrounds. This dataset is a subset of ImageNet [175] containing nine super-classes. This dataset helps learn the robustness of vision models to their dependence on the backgrounds of images.

ImageNet-A. ImageNet-A [176], is a dataset of real-world images, adversarially curated to test the limits of classifiers such as ResNet-50. While these images are from standard ImageNet classes [175], their complexity increases the challenge, often causing misclassifications in multiple models. We use this dataset to test the robustness of a classifier after training it on ImageNet-9.

NICO. The NICO dataset [148] is designed for non-independent and identically distributed and out-of-distribution image classification, simulating real-world scenarios where testing distributions differ from training ones. It labels images with both main concepts and contexts (e.g., 'dog on grass'), enabling studies on transfer learning, domain adaptation, and generalization. NICO contains two superclasses: Animal and Vehicle, with 19 classes, 188 contexts, and nearly 25,000 images in total.

Experimental Setup

Spurious Attribute Detection. We generate text descriptions for images using a pre-trained vision-language model [40], which has an encoder-decoder structure where the encoder is a vision

Class	Attributes
Landbird	pool , boat , building , pond , surfboard , sandy , beach , water , body, frisbee
Waterbird	stream , forest , building , pile, front, middle, animal, photo, tree , branch

Table 4.8: Top-10 detected attributes selected based on their spuriousness scores for each class in the Waterbirds dataset. We highlight several attributes that are relevant to water backgrounds in blue and those that are relevant to land backgrounds in orange.

transformer [177] and the decoder is the GPT-2 [178] language model. We set the maximum length of the sequence to be generated as 16 and the number of beams for beam search to 4. After generating text descriptions for test images, we use Spacy (`https://spacy.io/`) to extract nouns and adjectives from the descriptions automatically. We additionally filter out words with frequencies less than 10 to remove potential annotation noise. In our experiments, we only need to do this procedure once for each dataset.

Training Settings. We use ResNet-50 and ResNet-18 as the backbone networks. For each dataset, we first train an ERM model, which is first initialized with ImageNet pre-trained weights, for 100 epochs. We set the learning rate to 0.001 which decays following a cosine annealing scheduler and use an SDG optimizer with 0.9 momentum and 10^{-4} weight decay. Then, we use the ERM-trained models as the initial models for our LBC training. For all the datasets, we fix the learning rate to 0.0001 and the batch size to 128. We sample 20 batches per epoch and train for 50 epochs. The cluster size K is set to 3. We report our results averaged over 5 runs. We provide training details in the Appendix A.4.3. All experiments are conducted on NVIDIA RTX A6000 GPUs.

Evaluation Metrics. We adopt different evaluation metrics on different datasets. For a dataset with group labels defined by classes and biased attributes, we partition the test data into groups. *Average accuracy* measures the overall performance of a classifier on the test data; however, it may be dominated by the majority group of samples with certain biases that the classifier may heavily rely on for predictions. Therefore, we mainly focus on the *worst-group accuracy* which is a widely accepted robustness measure that gives the lower-bound performance of a classifier on various dataset biases. To measure the tradeoff between the average and worst accuracy, we additionally calculate the *gap* between the two metrics. For datasets without group labels, we report different kinds of average accuracies on specifically designed test sets, which we will explain in the respective sections.

Method	Spurious attribute label	ImageNet-9		ImageNet-A
		Validation(↑)	Unbiased(↑)	Test(↑)
StylisedIN [179]	✓	$88.4_{\pm 0.5}$	$86.6_{\pm 0.6}$	$24.6_{\pm 1.4}$
LearnedMixin [84]	✓	$64.1_{\pm 4.0}$	$62.7_{\pm 3.1}$	$15.0_{\pm 1.6}$
RUBi [180]	✓	$90.5_{\pm 0.3}$	$88.6_{\pm 0.4}$	$27.7_{\pm 2.1}$
ERM	✗	$90.8_{\pm 0.6}$	$88.8_{\pm 0.6}$	$24.9_{\pm 1.1}$
ReBias [181]	✗	$91.9_{\pm 1.7}$	$90.5_{\pm 1.7}$	$29.6_{\pm 1.6}$
LfF [33]	✗	86.0	85.0	24.6
CaaM [182]	✗	95.7	95.2	32.8
SSL+ERM [183]	✗	$94.18_{\pm 0.07}$	$93.18_{\pm 0.04}$	$34.21_{\pm 0.49}$
LWBC [183]	✗	$94.03_{\pm 0.23}$	$93.04_{\pm 0.32}$	$35.97_{\pm 0.49}$
LBC (Ours)	✗	$\mathbf{96.97}_{\pm 0.17}$	$\mathbf{96.03}_{\pm 0.12}$	$\mathbf{40.63}_{\pm 1.79}$

Table 4.9: Validation, Unbiased, and Test metrics (%) evaluated on the ImageNet-9 and ImageNet-A datasets. All methods use ResNet-18 as the backbone. The best results are in **boldface**.

Method	Spurious attribute label	NICO	
		Validation(↑)	Test(↑)
RUBi [180]	✓	43.86	44.37
IRM [138]	✓	40.62	41.46
ERM	✗	43.77	42.61
CBAM [184]	✗	42.15	42.46
ReBias [181]	✗	44.92	45.23
LfF [33]	✗	41.83	40.18
CaaM [182]	✗	46.38	46.62
SSL+ERM [183]	✗	$55.63_{\pm 0.54}$	$52.24_{\pm 0.27}$
LWBC [183]	✗	$56.05_{\pm 0.45}$	$52.84_{\pm 0.31}$
LBC (Ours)	✗	$\mathbf{68.26}_{\pm 2.15}$	$\mathbf{65.34}_{\pm 2.54}$

Table 4.10: Validation and Test metrics (%) evaluated on the NICO dataset. All methods use ResNet-18 as the backbone pretrained on ImageNet. The best results are in **boldface**.

Effectiveness of Spuriousness Score

We calculated the spuriousness scores for the correlations between all the detected attributes and classes in the Waterbirds dataset and selected the top-10 attributes with the highest spuriousness scores for each class. Table 4.8 shows that these attributes are mostly relevant to water and land backgrounds, which are spurious by design. Interestingly, our spuriousness score can find attributes in one class that are heavily exploited to predict some other class. For example, `pool` from `landbird` is detected in images of `landbird` with a pool, but it tends to bias the predictions toward `waterbird` since it is relevant to water backgrounds. As discussed in Chapter 4.2.3, this arises from the Abs operator in our definition of spuriousness score in Equation (4.8): when the nominator term is much smaller than the denominator term, e.g., when a=`pool` and c=`landbird`, the spuriousness score is still high. Examples from other datasets are shown in Appendix A.4.4.

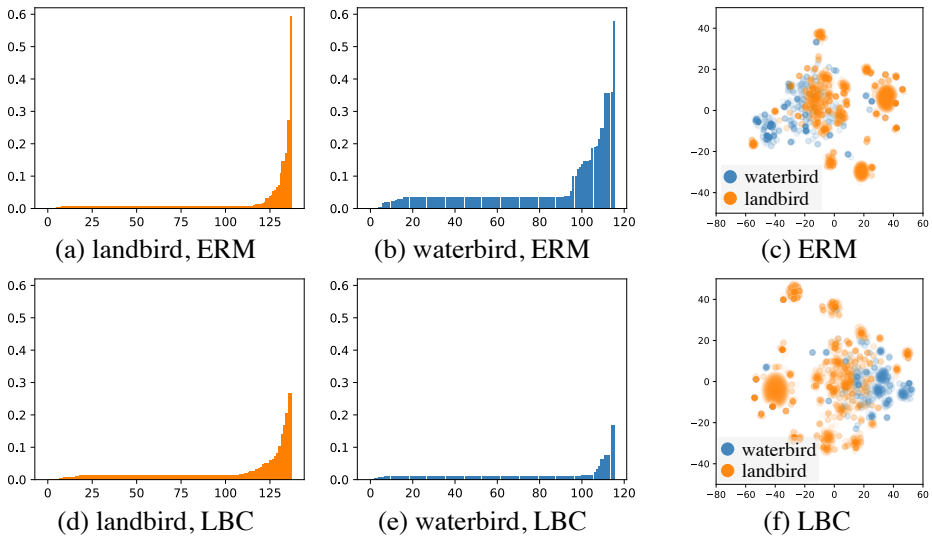


Figure 4.8: (a) and (b): Spuriousness scores for the attributes detected from landbird and waterbird based on an ERM model. (d) and (e): Spuriousness scores based on our LBC model. (c) and (f): Spurious embeddings of the images in the Waterbirds dataset based on the ERM and LBC model, respectively.

LBC Reduces Reliance on Spurious Correlations

We have shown that our spuriousness score can effectively reflect a model's reliance on spurious correlations. To show the efficacy of our LBC method in learning a robust model to spurious correlations, we calculated the spuriousness scores for the class-attribute correlations based on an ERM and our LBC-trained models. Comparing the results between Figure 4.8(a) and Figure 4.8(d), as well as between Figure 4.8(b) and Figure 4.8(e), we observe that LBC significantly reduces the spuriousness scores of the correlations between the detected attributes and the two classes. Moreover, the spuriousness embeddings, which represent images with different prediction behaviors, become more dispersed in Figure 4.8(f) than in Figure 4.8(c). This indicates that LBC successfully mitigates the ERM classifier's reliance on certain prediction behaviors and diversifies the prediction behaviors for different classes.

Comparison with Existing Methods

Datasets with Group Labels. We first compare our method with baselines in Table 4.7 on the CelebA and Waterbirds datasets, which provide group labels of all the data samples. In the first setting where only the group labels of the validation data are used, our method achieves the best worst-group accuracies and the best gaps between the average and the worst-group accuracies on the two datasets, striking a favorable balance between the robustness of the classifier and its

overall performance. Our worst-group accuracies are also close to the upper bounds established by GroupDRO, while our average accuracies are competitive or better than those of GroupDRO. In our main setting where no group labels are available, our method outperforms the baselines with the best worst-group accuracies and worst-average gaps. More significantly, our method is the most robust in terms of the least drops in worst-group accuracy when switching from the first setting to the second one. This shows the effectiveness of our designed model selection metric in selecting robust models.

Datasets with Texture Biases. Experiments on ImageNet-9 and ImageNet-A test how much a classifier relies on the spurious texture bias. In Table 4.9, texture group labels [181, 183] are used as the spurious attribute labels. The "Validation" denotes the average accuracy on the validation set, "Unbiased" denotes the average accuracy over several texture groups, and "Test" denotes the average accuracy on the ImageNet-A dataset which contains misclassified samples by an ImageNet-trained model. Our method outperforms other methods on the three metrics, showing that our method is effective in mitigating a classifier's reliance on texture biases and correcting its failure modes in classification.

Dataset with Object-Context Correlations. The NICO dataset is created to evaluate a classifier's reliance on object-context correlations. In Table 4.10, "Validation" denotes the average accuracy on the validation data which contains the same object-context correlations as in the training data, and "Test" denotes the average accuracy on the test data which contain not only existing object-context correlations but also unseen ones. Our method effectively mitigates the reliance on object-context correlations and achieves the best on the two metrics.

Ablation Studies

We first analyzed the effectiveness of the four proposed components: (1) predicting prediction behaviors (PPB), (2) within-class balancing (WCB), (3) cross-class balancing, and (4) spuriousness embeddings (SE). We remove one component and observe the worst-group accuracies achieved by the remaining ones. In Figure 4.9(a), \PPB denotes that we keep the original classifier to predict classes, \WCB denotes that we randomly sample from the same class, \CCB denotes that we equally sample images with different identified prediction behaviors, and \SE denotes that we use binary attribute embeddings for images. We observe that all four components positively contribute to our method since removing any one of them results in reduced accuracy. Among the four components, balanced data sampling (WCB and CCB), especially WCB, is most critical to our method. Figure 4.9(b) shows that a large K exceeding 10 has suboptimal worst-group accuracies and that when

$K = 2$, it limits the discovery of diverse prediction behaviors. Typically, $K = 3$ works for most of the cases. More results are shown in Appendix A.4.

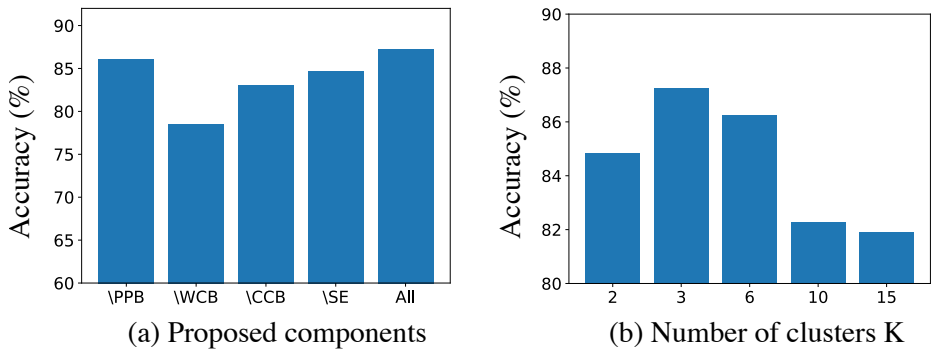


Figure 4.9: Worst-group accuracy comparison of (a) leave-one-out study on the four proposed components and (b) analysis on the number of clusters K on the Waterbirds dataset.

4.2.5 Conclusion

We completely removed the barrier of expert knowledge and human annotations for spurious correlation mitigation by proposing a self-guided framework. Our framework incorporates an automated approach empowered by a VLM to detect attributes in images and quantifies their spuriousness with class labels. We formulated a spuriousness embedding space based on the spuriousness scores to identify distinct prediction behaviors of a classifier. We trained the classifier to recognize the identified prediction behaviors with balanced training data. Experiments showed that our framework improves the robustness of a classifier against spurious correlations without knowing them in the data.

4.3 Spuriousness-Aware Meta-Learning for Learning Robust Classifiers

4.3.1 Introduction

Spurious correlations are prevalent in real-world datasets. They are brittle associations between certain input attributes and the corresponding target variables. For example, the class `cow` is correlated with `grassland` when most training images show a cow on a grassland, but the correlation breaks when a cow is at a beach [2, 6]. The grassland feature is spurious as it does not always correlate with the label `cow` and is not truly predictive for all cow images. Deep image classifiers often use spurious correlations as their prediction shortcuts [2], such as inferring an image as representing a

`cow` by focusing on the grassland background of the image. Although this shortcut learning strategy can achieve high overall performance when the majority of samples have spurious correlations, it generalizes poorly on samples where spurious correlations do not hold. Thus, mitigating the reliance on spurious correlations is crucial for obtaining robust image classifiers.

Existing approaches require annotations of spurious correlations or group labels, which separate data into multiple groups with each containing samples of the same class and sharing the same attribute. For example, a group label (cow, grass field) represents all cow images with grass fields as the background. The group labels are used to formulate new optimization objectives [3] or used for model selection and/or model fine-tuning [34, 33, 4, 170]. However, knowing the group labels in data requires expert knowledge and costly human annotations, which cannot scale to large datasets. Completely removing the requirement for group labels while learning robust classifiers is also a challenging task since we have no knowledge about what spurious correlations we need to mitigate.

In this paper, we propose a novel learning framework to train an image classifier to be robust to spurious correlations without the need of group labels. We design our framework to iteratively detect and mitigate the spurious correlations that the classifier heavily relies on for predictions. To achieve this, we first propose an automatic spurious attribute detection method empowered by a pre-trained vision-language model (VLM). The VLM enables us to detect text-format attributes which represent many similar pixel-level features and are interpretable to humans. These attributes together with class labels can formulate various class-attribute correlations which we may find to be spurious in data, and these correlations can cover many potential scenarios where an image classifier fails to generalize because of its reliance on one or multiple of these spurious correlations. Therefore, to train a robust classifier against spurious correlations in general without the guidance of group labels, we focus on mitigating the classifier's reliance on the detected correlations.

However, it is not efficient to mitigate all of them with equal importance, since among the detected correlations, some are trivial for the classifier as the classifier is robust to them, while some may pose a great risk to the robustness of the classifier. Thus, we propose a novel spuriousness metric to quantify the *spuriousness* of the correlation between a detected attribute and a class label, which measures a classifier's reliance on these class-attribute correlations for predictions, with a larger value indicating a greater reliance on the correlation. With the spuriousness metric, we can identify harmful spurious correlations.

To train a robust classifier, we propose a SPUriousness-aware MEta-learning (termed SPUME) strategy. Unlike the classical settings where only a few spurious correlations are known and needed to be mitigated, our setting has numerous correlations established by the detected attributes and

class labels, especially when the dataset that we use has rich features. Using meta-learning, we can distribute the detected spurious correlations with high spuriousness values into multiple meta-learning tasks by carefully curating the data in those tasks. We exploit the support (training) and query (test) sets in a meta-learning task so that samples in the support and query sets have different spurious correlations. Such a task *simulates* a challenging learning scenario where the classifier will perform poorly on the query set when it has a high reliance on the spurious correlations in the support set. By meta-training the classifier on these spuriousness-aware meta-learning tasks, our classifier can learn to be invariant to the spurious correlations.

Our **contributions** are as follows:

- We propose an automatic method to detect spurious correlations in data, which exploits the prior knowledge contained in a pre-trained VLM and extracts spurious attributes in interpretable text format.
- We tackle the problem of mitigating the reliance on spurious correlations with a novel meta-learning strategy.
- We propose a novel spuriousness metric to guide the construction of meta-learning tasks with the detected spurious attributes.
- We demonstrate that a classifier with high average accuracy does not necessarily have high worst-group accuracy which is commonly used for measuring the robustness to spurious correlations. Our method, termed as *SPUrious-aware MEta-learning (SPUME)*, can train classifiers robust to spurious correlations on five benchmark datasets without knowing the spurious correlations a priori.

4.3.2 Problem Formulation

Consider a training dataset $\mathcal{D}_{\text{tr}} = \{(x_n, y_n)\}_{n=1}^{N}$ with $x_n \in \mathcal{X}$, $y_n \in \mathcal{Y}$, where $\mathcal{X}$ denotes the input space containing all possible inputs, $\mathcal{Y}$ denotes the set of K classes. In real-world scenarios, a sample x_n in $\mathcal{D}_{\text{tr}}$ typically has *spurious attributes* and these attributes have *spurious correlations* with the label y_n. We describe the two important concepts below.

• **Spurious Attributes:** A spurious attribute $a \in \mathcal{A}$ describes some common patterns in the input space $\mathcal{X}$ and spuriously correlates with some label $y \in \mathcal{Y}$, where $\mathcal{A}$ denotes all possible spurious attributes. In other words, a can be in samples of multiple classes or only in some samples of a class, and therefore is not essential to any of the classes. For example, the "land background" attribute

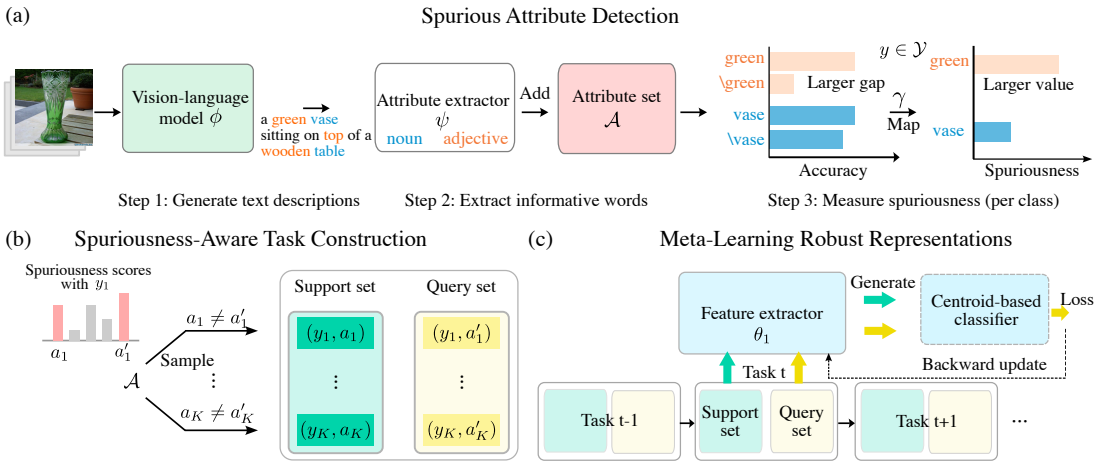


Figure 4.10: Overview of SPUME. (a) Detect attributes from training data and measure their spuriousness in three steps. "\green" denotes without the attribute "green". (b) Construct spuriousness-aware meta-learning tasks guided by the spuriousness scores of the detected attributes. (c) Meta-train a robust feature extractor using the constructed tasks.

can exist in images of waterbird and landbird classes [3], and "land background" is non-essential to either of the classes.

• **Spurious Correlations:** A spurious correlation, denoted as $\langle y, a \rangle$, describes the brittle association between the spurious attribute a and the label y. The spurious correlation $\langle y, a \rangle$ does not always hold in the sense that a can be associated with multiple y's or y can correlate with other attributes in some samples. Knowing all the spurious correlations in $\mathcal{D}_{\text{tr}}$, we can divide $\mathcal{D}_{\text{tr}}$ into multiple data groups $\mathcal{D}_{\text{tr}}^g$, $g \in \mathcal{G}$, where $g = (y, a)$ denotes the group label for samples with the label y and having the spurious attribute a, and $\mathcal{G} = \mathcal{Y} \times \mathcal{A}$ denotes the set of all group labels.

Given a deep neural classifier f_θ with parameters θ, we train it with empirical risk minimization (ERM) on the training set $\mathcal{D}_{\text{tr}}$ and obtain the optimized classifier f_{θ^*} as follows:

$$\theta^* = \arg\min_\theta \mathbb{E}_{(x,y)\in\mathcal{D}_{\text{tr}}} \ell(f_\theta(x), y), \tag{4.15}$$

where $\ell(\cdot, \cdot)$ is the cross-entropy loss function.

The problem occurs when data groups $\{\mathcal{D}_{\text{tr}}^g | g \in \mathcal{G}, \mathcal{D}_{\text{tr}}^g \subset \mathcal{D}_{\text{tr}}\}$ in $\mathcal{D}_{\text{tr}}$ are imbalanced in sizes or the inductive bias of the classifier f_θ favors particular data groups. For example, a majority group $\mathcal{D}_{\text{tr}}^g$ with the group label $g = (y, a)$ in $\mathcal{D}_{\text{tr}}$, which has significantly more samples than other groups, may bias the optimization in Equation (4.15) towards favoring the data in $\mathcal{D}_{\text{tr}}^g$ having the spurious

correlation $\langle y, a \rangle$, i.e.,

$$\theta^* \approx \arg\min_{\theta} \mathbb{E}_{(x,y)\in\mathcal{D}_{\text{tr}}^{g}} \ell(f_\theta(x), y), \tag{4.16}$$

with $|\mathcal{D}_{\text{tr}}^{g}| \gg |\mathcal{D}_{\text{tr}}^{g'}|$, where $g, g' \in \mathcal{G}$ and $g \neq g'$, and $|\cdot|$ denotes the size of a set. As a result, the classifier f_{θ^*}, instead of utilizing the core features in samples to predict y, may superficially learn the mapping from a to y, which is non-robust when the correlation between a and y breaks. More specifically, since a is a spurious attribute, there may exist $\langle y', a \rangle$ in samples from class y' with $y \neq y'$. Then, it is very likely that f_{θ^*} will wrongly predict these samples as y instead of y'. For example, when f_{θ^*} learns to use water backgrounds (a) to predict waterbirds (y), it fails to recognize landbirds (y') with water backgrounds. Similarly, when the inductive bias in f_{θ^*} favors certain spurious correlations, the classifier will encounter the same generalization problem.

Spurious correlations pose a great challenge to the robustness of machine learning models. To address this, typically, all or partial group labels of the training data is required for various purposes, such as formulating the group robustness objective [3], reweighting the training data, or selecting models [31]. However, acquiring group labels for a dataset typically involves human-guided annotations, which is costly and not scalable, especially when the dataset is large. In the following, without the need of group labels, we propose a novel spuriousness-aware meta-learning framework to train a classifier to be robust to spurious correlations.

4.3.3 Spuriousness-Aware Meta-Learning

We give the overview of our framework in Figure 4.10, where we first detect spurious attributes with a pre-trained VLM (Figure 4.10(a)). To effectively use the detected spurious attributes for spurious correlation mitigation, we propose a novel meta-learning strategy and provide details on how to construct spuriousness-aware meta-training tasks (Figure 4.10(b)) and meta-learn robust representations (Figure 4.10(c)).

Automatic Spurious Attribute Detection

To automatically detect spurious attributes in a target dataset without human-guided annotations, we propose to exploit the prior knowledge in a pre-trained VLM. Our method detects spurious attributes in *text format* and consists of the following three steps.

Step 1: Generate Text Descriptions. We generate a text description for each image using a pre-trained VLM ϕ, which is capable of generating text descriptions of images at scale. Moreover, since the model is trained on massive data and is not specifically fine-tuned on the target dataset, it can discover general objects and patterns. For example, in Figure 4.10(a), besides the class object `vase`, the VLM also detects the vase's color `green` and a background object `table` with its material `wooden`.

Step 2: Extract Informative Words as Attributes. We extract informative words from the text descriptions of images as attributes. We select nouns, which describe objects, and adjectives, which describe certain properties of objects, as the informative words. For example, we extract `green`, `vase`, `top`, `wooden`, and `table` from the description in Figure 4.10(a). We instantiate the attribute extractor ψ with an automatic procedure (Section 4.3.4) to extract these informative words from the text descriptions obtained in the first step. Then, these extracted words are added to the attribute set $\mathcal{A}$ as the *possible* spurious attributes.

Remark. VLMs can detect general objects and patterns. However, due to the inductive bias learned during pre-training, VLMs may generate text descriptions for some images that are not aligned with human understandings, such as describing a red-and-green background as a "Christmas tree". Although "Christmas tree" is not self-explanatory in this case, it is still a valid and useful attribute, representing samples having similar red-and-green backgrounds. This also highlights the benefit of using VLMs: they can detect patterns that are not easily perceived by humans. A limitation of such a VLM-based detection approach is that VLMs may struggle on describing images from domain-specific tasks where, for example, slight changes in orientation of objects or variations in geographies are important for robust predictions. Nevertheless, our proposed spurious attribute detection approach is not restricted to a specific VLM, and it can be improved if more capable VLMs are available.

Step 3: Measure Spuriousness. To know whether a detected attribute $a \in \mathcal{A}$ is spurious, we need to consider it in the correlation with a class label y, since among all the correlations between the attributes in $\mathcal{A}$ and class labels, some of them may be vacuous — they do not exist in the training data (e.g., a only exists in images of the class y' with $y' \neq y$), and some of them are not spurious (e.g., the attribute a is detected exclusively in all the images of the class y). Moreover, we are interested in identifying spurious correlations that are likely to be exploited by a classifier for predictions as these correlations directly affect the robustness of the classifier.

To unify the above cases, we propose a metric to quantify the likelihood of the correlation $\langle y, a\rangle$ being spurious *and* used by a classifier, i.e., *spuriousness* of the correlation. The metric γ considers y, a, the training data $\mathcal{D}_{\text{tr}}$, and the classifier f_θ, and maps them to a finite value, which we call *spuriousness score*. We defines γ as follows.

Definition 4.2 (Spuriousness Metric)**.** Given a class label $y \in \mathcal{Y}$, an attribute $a \in \mathcal{A}$, and a classifier f_θ trained on $\mathcal{D}$ with $\theta \in \Theta$, the spuriousness metric for $\langle y, a\rangle$ is a mapping $\gamma : \mathcal{Y}\times\mathcal{A}\times\mathcal{D}\times\Theta \to [\alpha, \beta]$, where $\mathcal{D}$ denotes a set of sample-label pairs, Θ denotes the set of all possible θ, and $[\alpha, \beta]$ denotes the output value range of γ, with α being the lowest and β being the highest. When the data group size $|\mathcal{D}^{(y,a)}| = 0$ or $|\mathcal{D}^{(y,\hat{a})}| = 0$, where $\hat{a}$ denotes all attributes in $\mathcal{A}$ other than a, the mapping γ outputs α.

Given the training set $\mathcal{D}_{tr}$, $|\mathcal{D}_{tr}^{(y,a)}| = 0$ and $|\mathcal{D}_{tr}^{(y,\hat{a})}| = 0$ correspond to that $\langle y, a\rangle$ does not exist in $\mathcal{D}_{tr}$ and that $\langle y, a\rangle$ exists exclusively in samples of class y, respectively. For both cases, the spuriousness of $\langle y, a\rangle$ should be the smallest.

Then, we specifically design γ based on the performance of the classifier f_θ. The motivation is that the classifier f_θ will generalize poorly on samples of the class y without the attribute a if f_θ excessively relies on a for predicting the label y. Therefore, as demonstrated in Figure 4.10(a), the spuriousness will be higher if f_θ has a larger performance discrepancy on images with and without a and be lower when the performance discrepancy is smaller. We formally define our spuriousness metric for $\langle y, a\rangle$ as follows,

$$\gamma(y, a; \mathcal{D}_{tr}, f_\theta) = \tanh\Big(\text{abs}\big(\log \frac{J(\mathcal{D}_{tr}^{(y,a)}; f_\theta)}{J(\mathcal{D}_{tr}^{(y,\hat{a})}; f_\theta)}\big)\Big), \tag{4.17}$$

with $\gamma(y, a; \mathcal{D}_{tr}, f_\theta) = 0$ when $\mathcal{D}_{tr}^{(y,\hat{a})} = \emptyset$ or $\mathcal{D}_{tr}^{(y,a)} = \emptyset$, where $\mathcal{D}_{tr}^{(y,a)} \subset \mathcal{D}_{tr}$ denotes the subset of all training data from the class c *with* the attribute a, $\mathcal{D}_{tr}^{(y,\hat{a})} \subset \mathcal{D}_{tr}$ denotes the subset of all training data from the class c *without* the attribute a, $J(\cdot; f_\theta)$ denotes the classification accuracy of f_θ on a given set of samples, and abs$(\cdot)$ denotes taking the absolute value. The division in Equation (4.17) aims to produce larger values than the simple difference between the two accuracies, making different correlations more distinctive. Moreover, using $\log(\cdot)$ avoids encountering extreme values from the division, and tanh(abs$(\cdot)$) bounds the score in the range from 0 to 1. Other designs of γ are possible, and we have shown in our experiments that our method proposed in the following is robust to different choices of spuriousness metrics.

Discussion. With the detected attributes and our spuriousness metric, we can identify spurious correlations that are likely to be used for predictions by a classifier and thus pose a potential risk to the robustness of the classifier. To improve the robustness to spurious correlations, we need to mitigate the classifier's reliance on those spurious correlations. Since there are multiple spurious correlations, mitigating all of them at once is a challenging task. To address this, we formulate the problem in a novel *meta-learning* [22, 23, 12, 26] setting, where we construct meta-learning tasks with each task containing some potentially harmful spurious correlations. Now, our goal is to learn a good classifier that performs well across all these tasks with various spurious correlations.

In the following, we first introduce how to construct meta-learning tasks with the identified spurious correlations. Then, we give the details of using the constructed tasks for meta-learning.

Spuriousness-Aware Task Construction

To mitigate spurious correlations via meta-learning, we first create meta-learning tasks which will be used in meta-training. A meta-learning task typically consists of a support (training) set $\mathcal{S}$ providing training samples for learning novel concepts and a query (test) set $\mathcal{Q}$ containing test samples for the evaluation of the learning outcome. We use the two sets to *simulate* spurious correlations in meta-learning tasks so that these spurious correlations can be effectively mitigated via meta-learning.

As illustrated in Figure 4.10(b), for *each class* y_k with $k = 1, \ldots, K$, we first sample two attributes a_k and a'_k from $\mathcal{A}$ based on their spuriousness scores, where $a_k \neq a'_k$. Specifically, we normalize the scores as probabilities, and an attribute with a higher spuriousness score will be more likely to be selected than another attribute with a lower spuriousness score. In this way, we target the spurious correlations that pose a high risk to the robustness of the classifier.

Then, the two sampled attributes formulate two spurious correlations with y_k, i.e., $\langle y_k, a_k \rangle$ and $\langle y_k, a'_k \rangle$, based on which, we get two data groups, $\mathcal{D}_{\text{tr}}^{(y_k, a_k)}$ and $\mathcal{D}_{\text{tr}}^{(y_k, a'_k)}$, from the training set $\mathcal{D}_{\text{tr}}$. These two groups of data together represent a shift in the correlation between the two spurious attributes and the class label. If the classifier learns to rely on the spurious correlation in one group of data for predictions, then it will fail on the other group of data with a different spurious correlation. Thus, crafting such a shift facilitates learning a robust classifier.

Next, for efficient training, we randomly sample N_S data points per class from the two data groups to construct the *non-overlapping* support set $\mathcal{S}_k$ and the query set $\mathcal{Q}_k$, i.e.,

$$\mathcal{S}_k = \bigcup_{i=1}^{N_S} \left\{ (x_i, y_k) | (x_i, y_k) \in \tilde{\mathcal{D}}_{\text{tr}}^{(y_k, a_k)} \right\}, \tag{4.18}$$

Algorithm 2 SPUME

Input: A training dataset $\mathcal{D}_{tr}$, a feature extractor h_{θ_1}, a pre-trained VLM ϕ, an attribute extractor ψ, a spuriousness metric γ, the number of tasks per epoch N_T, the number of classes K, and the number of training epochs E.
Output: Learned weights θ_1^*

```
Build the attribute set A = ∪_{(x,y)∈D_tr} ψ(φ(x))
for e = 1, ..., E do
    Generate class centroids using Equation (4.20) with S = D_tr
    Generate spuriousness scores using Equation (4.17)
    Set T(D_tr, A, γ, θ_1) as an empty set
    for t = 1, ..., N_T do
        Sample K pairs of attributes from A for each class
        Construct a spuriousness-aware meta-learning task T using Equation (4.18) and (4.19)
        Add T to T(D_tr, A, γ, θ_1)
    end for
    Set θ_1 = θ_1^* using Equation (4.23)
end for
return θ_1^*
```

and

$$\mathcal{Q}_k = \bigcup_{i=1}^{N_S} \left\{ (x_i, y_k) | (x_i, y_k) \in \tilde{\mathcal{D}}_{\text{tr}}^{(y_k, a_k')} \right\}, \tag{4.19}$$

where $\tilde{\mathcal{D}}_{\text{tr}}^{(y_k,a_k)} = \mathcal{D}_{\text{tr}}^{(y_k,a_k)} - \mathcal{D}_{\text{tr}}^{(y_k,a_k')}$ and $\tilde{\mathcal{D}}_{\text{tr}}^{(y_k,a_k')} = \mathcal{D}_{\text{tr}}^{(y_k,a_k')} - \mathcal{D}_{\text{tr}}^{(y_k,a_k)}$ are sets of elements unique to $\mathcal{D}_{\text{tr}}^{(y_k,a_k)}$ and $\mathcal{D}_{\text{tr}}^{(y_k,a_k')}$, respectively. Taking the above set difference ensures that the two spurious correlations won't appear in the same set since some samples may have both the attributes a_k and a_k'.

After constructing the two sets for *each class*, we obtain the constructed task $\mathcal{T} = \{\mathcal{S}, \mathcal{Q}\}$ with $\mathcal{S} = \cup_{k=1}^{K} \mathcal{S}_k$ and $\mathcal{Q} = \cup_{k=1}^{K} \mathcal{Q}_k$. If K is large, we can randomly select a subset of K classes to construct $\mathcal{T}$. The constructed task $\mathcal{T}$ demonstrates to the classifier that the spurious correlations in $\mathcal{T}$ are highly risky for it, and that the classifier should be invariant to them in order to perform well on this task. Importantly, the construction of meta-learning tasks also ensures that biases in VLMs won't be passed down to the classifier as the construction process effectively decorrelates biased attributes from VLMs with prediction targets.

Meta-Learning Robust Representations

To train a robust classifier using the constructed tasks, we modify f_θ so that it fits in with the meta-learning paradigm. Specifically, we discard the last linear classification layer of f_θ and keep

its feature extractor $h_{\theta_1} : \mathcal{X} \to \mathbb{R}^D$, where $\theta_1 \subset \theta$ and D is the number of dimensions in the feature extractor's outputs. Thus, learning a robust classifier is equivalent to learning robust representations.

As illustrated in Figure 4.10(c), for the t'th task, we use the representations of the samples in the support set $\mathcal{S}$ provided by h_{θ_1} to generate (learn) a centroid-based classifier with K class-centroids $\mathcal{W} = \{\mathbf{w}_1, \ldots, \mathbf{w}_K\}$ calculated as follows

$$\mathbf{w}_k = \frac{1}{N_S} \sum_{n=1}^{N_S} h_{\theta_1}(x_n), (x_n, y_k) \in \mathcal{S}. \tag{4.20}$$

Next, we evaluate whether the classifier depends on the spurious correlations in $\mathcal{S}$ by testing it on the query set $\mathcal{Q}$ where the spurious correlations in $\mathcal{S}$ do not hold. The output probability of the classifier on y_k is calculated as follows

$$p(y_k|x_n, \theta_1, \mathcal{S}) = \frac{\exp(\tau d(\mathbf{w}_k, h_{\theta_1}(x_n)))}{\sum_{k'=1}^{K} \exp(\tau d(\mathbf{w}_{k'}, h_{\theta_1}(x_n)))}, \tag{4.21}$$

where $d(\cdot, \cdot)$ denotes the cosine similarity between two embedding vectors, and τ denotes a scaling hyperparameter. Then, the task loss $\ell_{\mathcal{T}}$ on $\mathcal{T} = \{\mathcal{S}, \mathcal{Q}\}$ is as follows

$$\ell_{\mathcal{T}}(\theta_1) = \mathop{\mathbb{E}}_{(x_n, y_n) \in \mathcal{Q}} - \log p(y_n|x_n, \theta_1, \mathcal{S}). \tag{4.22}$$

A high loss indicates that the classifier, and in turn the feature extractor h_{θ_1}, rely on the spurious correlations in the support set and cannot generalize well on the query set.

Learning Objective. We minimize the loss in (4.22) over tasks constructed with various spurious correlations to find a feature extractor $h_{\theta_1^*}$ that is robust to multiple spurious correlations, i.e.,

$$\theta_1^* = \arg\min_{\theta_1} \mathbb{E}_{\mathcal{T} \in T(\mathcal{D}_{\text{tr}}, \mathcal{A}, \gamma, \theta_1)} \ell_{\mathcal{T}}(\theta_1), \tag{4.23}$$

where $T(\mathcal{D}_{\text{tr}}, \mathcal{A}, \gamma, \theta_1)$ denotes all possible meta-learning tasks constructed from $\mathcal{D}_{\text{tr}}$ based on the detected attributes $\mathcal{A}$, the spuriousness metric γ, and the feature extractor θ_1.

To solve (4.23), we adopt an iterative optimization procedure. We first fix θ_1 and construct a set of meta-training tasks based on $\mathcal{A}$, θ_1, and γ. Then, we update θ_1 using the constructed tasks. The above steps are iterated until some stop criterion is met. We name our method as *SPUriousness-aware MEta-Learning (SPUME)* and give the training details in Algorithm 2.

Complexity Analysis. VLMs do not incur training cost because they are only used for data preparation. Extracting attributes (Line 1, Algorithm 2) is a onetime offline process, and empirically,

its time cost scales linearly with the dataset size. Spuriousness measurement (Line 4, Algorithm 2) is performed periodically during training, and its time complexity grows linearly with the amount of data it uses. The total training cost is $O(E(C_m + C_s))$, where E is the number of training epochs, C_m and C_s are the time cost of meta-learning a classifier and obtaining spuriousness scores per epoch, respectively, with $C_m \gg C_s$, since the latter only requires forward passes through the classifier. Moreover, using a metric-based meta-learning technique (Equation (4.20)) leads to C_m being comparable to training a standard classifier. Therefore, our method does not incur significant training cost compared with the ERM training.

Model Selection. We divide the validation data $\mathcal{D}_{\text{val}}$ into groups based on the detected attributes $\mathcal{A}$ and calculate the average accuracy over these groups as follows,

$$Acc_{pu} = \frac{1}{|\mathcal{A}| \cdot |\mathcal{Y}|} \sum_{a \in \mathcal{A}} \sum_{y \in \mathcal{Y}} J\big(\mathcal{D}_{\text{val}}^{(y,a)}; h_{\theta_1}\big). \tag{4.24}$$

We call this metric *pseudo-unbiased accuracy*, which fairly measures the performance of the classifier on various groups inferred with the detected attributes in $\mathcal{A}$.

Inference. We first create a centroid-based classifier using Equation (4.20) with all the data in $\mathcal{D}_{\text{tr}}$. Then, given a test sample x, the prediction is $\hat{y} = \arg\max_{y \in \mathcal{Y}} p(y|x, \theta_1, \mathcal{D}_{\text{tr}})$.

4.3.4 Experiments

Datasets

We tested our method on five image classification datasets with various types of spurious correlations, which are introduced below. Detailed dataset statistics are give in Table A.4.1 in Appendix A.5.

Waterbirds [3] contains waterbird and landbird classes. It is a synthetic dataset generated by combining images of the two kinds of birds from the CUB dataset [163] with water and land backgrounds from the Places dataset [174], producing (landbird, land), (landbird, water), (waterbird, land), and (waterbird, water) groups.

CelebA [149] is a large-scale image dataset of celebrity faces. It contains images showing two hair colors, non-blond and blond, which are spuriously correlated with gender. There are four groups in the CelebA dataset: (non-blond, female), (non-blond, male), (blond, female), and (blond, male).

ImageNet-9 [185] is a subset of ImageNet [175] and contains nine super-classes. It is known to have correlations between object classes and image textures. We followed the setting in [183] and [181] to prepare training and validation data.

Dataset	Number of detected attributes		Average number of attributes per image	
	BLIP	ViT-GPT2	BLIP	ViT-GPT2
Waterbirds	160	144	3.301	4.314
CelebA	683	345	3.913	4.291
NICO	239	199	3.104	3.995
ImageNet-9	540	442	3.276	4.311

Table 4.11: Statistics of the attributes detected from the Waterbirds, CelebA, NICO, and ImageNet-9 datasets.

ImageNet-A [176] is a dataset of real-world images, adversarially curated to test the limits of classifiers such as ResNet-50. While these images are from standard ImageNet classes [175], they are often misclassified in multiple models. We used this dataset to test the robustness of a classifier after training it on ImageNet-9.

NICO [148] is designed for out-of-distribution image classification, simulating real-world scenarios where testing distributions differ from training ones. It labels images with both main concepts (e.g., cat) and contexts (e.g., at home). We used the Animal super-class in NICO and followed the setting in [186, 187] for data preparation.

Experimental Setup

Spurious Attribute Detection. We used two pre-trained VLMs, ViT-GPT2 [40] and BLIP [39] to generate text descriptions for images. ViT-GPT2 has an encoder-decoder structure with a vision transformer [177] as the encoder and the language model GPT-2 [178] as the decoder. BLIP has a multimodal mixture of encoder-decoder architecture. After generating text descriptions, we used Spacy (`https://spacy.io/`) to extract nouns and adjectives from the descriptions automatically. We additionally filtered out words with frequencies less than 10 to remove potential annotation noise and to ensure that we have enough samples to construct a meta-learning task with selected spurious attributes. We give the statistics of the detected spurious attributes in the four datasets (ImageNet-A is not included as it is only used for testing) in Table 4.11. BLIP detects more attributes than ViT-GPT2 overall but less attributes for each image. Based on the two VLMs, our method has two variations, namely **SPUME-BLIP** and **SPUME-ViT-GPT2**. In the following experiments, we report the results of both methods.

Training Settings. We set $N_S = 10$ for sampling each class of images for both the support and query sets of a task. Following existing settings [3, 183, 187], we used ResNet-50 as the feature

extractor for the experiments on the Waterbirds and CelebA datasets, and used ResNet-18 on the ImageNet-9 and NICO datasets. All models were initialized with weights pre-trained on ImageNet. We used a stochastic gradient descent (SDG) optimizer with a momentum of 0.9 and a weight decay of 10^{-4} during meta-training. We trained a model for 100 epochs and used the cosine annealing scheduler to control the decay of learning rate. Without any group labels, our method used the pseudo-unbiased accuracy on the validation set defined in Equation (4.24) for model selection, while other methods used the average validation accuracy. We repeated each experiment three times and calculated the averaged results with standard deviations. We provide additional training details in Appendix A.5. All experiments were conducted on NVIDIA A100 GPUs.

Baselines. We compare our methods with state-of-the-art methods on mitigating spurious correlations and provide descriptions of the baseline methods in Appendix A.5. For fair comparison, the same feature extractor was used for methods compared on each dataset. Group labels were not used for model training and selection for all the compared methods. Note that we did not include VLMs as baselines, as they were exclusively used for extracting attributes from training data in our method. Moreover, directly using VLMs requires a completely different design, e.g., designing proper input prompts for classification.

Evaluation Metrics. To evaluate the robustness to spurious correlations on the Waterbirds and CelebA datasets, which provide group labels, we adopted the widely accepted robustness metric, **worst-group accuracy**, that gives the lower-bound performance of a classifier on the test set with various dataset biases. We also calculated the **accuracy gap** between the standard average accuracy and the worst-group accuracy as a measure of a classifier's reliance on spurious correlations. A high worst-group accuracy with a low accuracy gap indicates that the classifier is robust to spurious correlations and can fairly predict samples from different groups. We adopted **average accuracy** for the evaluations on the NICO, ImageNet-9, and ImageNet-A datasets as the these datasets are specifically constructed to evaluate the robustness to distributional shifts.

Visualization of a Spuriousness-Aware Task

We show a spuriousness-aware meta-learning task constructed from the Waterbirds dataset with $N_S = 5$ in Figure 4.11. For images in the same class, their backgrounds differ significantly in the support and query sets. Specifically, the landbird images selected based on the attribute "horse" in the support set have land backgrounds, while the same-class images selected based on the attribute

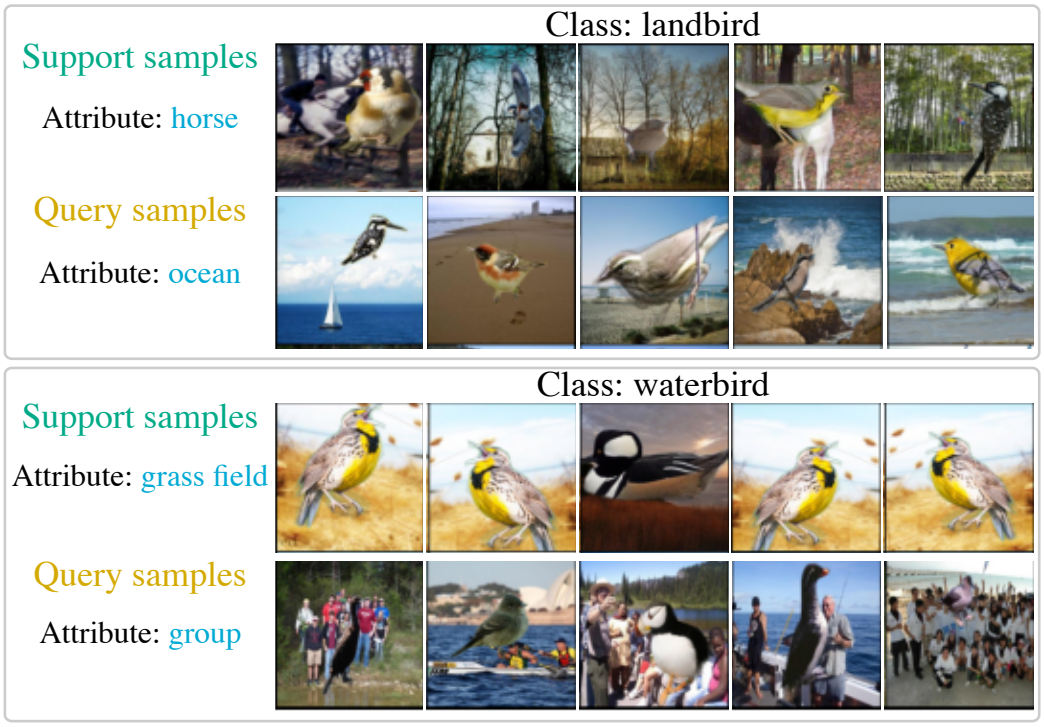


Figure 4.11: A meta-learning task with $N_S = 5$ constructed from the Waterbirds dataset. Images in the support set differ significantly from images in the query set in terms of their backgrounds.

"ocean" in the query set mainly have water backgrounds. Similarly, the query images of waterbird selected based on the attribute "group" have backgrounds filled with a group of people, while the corresponding support images selected based on the attribute "grass field" have grass backgrounds without irrelevant objects.

The constructed task creates a challenging learning scenario for classifiers that rely on spurious correlations for predictions. For example, a classifier that learns to use the land backgrounds to predict landbird from the support set will fail to predict landbird images with water backgrounds in the query set. Optimizing a classifier's performance on these spuriousness-aware tasks facilitates the classifier to learn to be invariant to spurious correlations.

SPUME Mitigates Reliance on Spurious Correlations

We calculated the spuriousness scores for all the detected class-attribute correlations before and after applying SPUME-BLIP to a classifier with the ResNet-50 backbone initialized with ImageNet pre-trained weights. We sorted the scores in the "before" scenarios and kept the order in the corresponding "after" scenarios. From Figures 4.12(a), 4.12(c), 4.12(e), and 4.12(g), we observe that the initial classifiers exhibit high reliance on the detected class-attribute correlations which have high

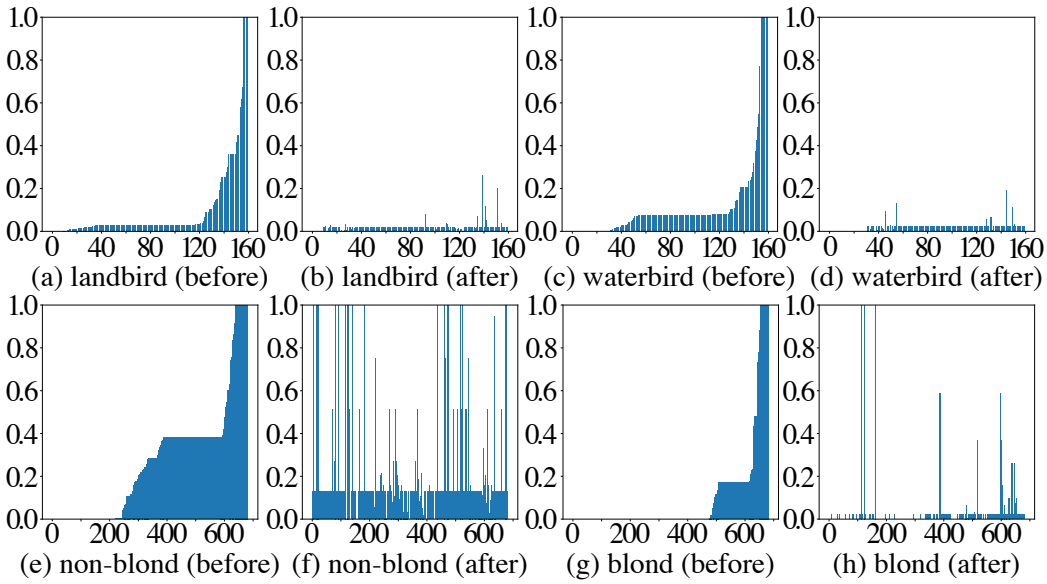


Figure 4.12: Spuriousness scores for all the class-attribute correlations before and after applying SPUME-BLIP to a classifier. The horizontal axes represent the indexes of detected attributes or class-attribute correlations, and the vertical axes represent the spuriousness scores. (a)-(d) Spuriousness scores on the Waterbirds dataset with landbird and waterbird classes. (e)-(h) Spuriousness scores on the CelebA dataset with non-blond and blond classes.

spuriousness scores. After applying SPUME-BLIP to the classifiers on the Waterbirds dataset, we observe from Figures 4.12(b) and 4.12(d) that the reliance on most of class-attribute correlations are mitigated and these correlations all have low spuriousness scores. On the CelebA dataset, which has more class-attribute correlations than the Waterbirds dataset, it becomes more challenging to mitigate the reliance on all these correlations. As observed from Figures 4.12(f) and 4.12(h), some correlations, which have low spuriousness scores initially, become highly spurious. Nevertheless, SPUME-BLIP can still mitigate the reliance on most of the class-attribute correlations having high spuriousness scores. Moreover, since spuriousness scores are not directly incorporated into our optimization objective in (4.23), the decrease in spuriousness scores demonstrates the effectiveness of our spuriousness-aware meta-learning strategy in mitigating the reliance on spurious correlations.

Quantitative Evaluation

We compared our methods with prior methods on mitigating spurious correlations on the five datasets. On each of the datasets, we show the reported results of these methods when they are available and give the details of these methods in Appendix A.5.

Method	Worst-group acc. (↑)	Acc. gap (↓)
ERM	66.4	23.8
LfF [33]	44.1	47.1
CVaR DRO [171]	62.0	33.2
JTT [31]	62.5	30.8
DFR [4]	77.4	14.7
DivDis [172]	81.0	9.7
SPUME-ViT-GPT2	**85.9±0.2**	6.9±0.8
SPUME-BLIP	85.7±0.2	**6.1±0.4**

Table 4.12: Comparison of worst-group accuracy (%) and accuracy gap (%) on the Waterbirds dataset. All methods do not have access to ground-truth group labels.

Method	Worst-group acc. (↑)	Acc. gap (↓)
ERM	45.7	49.8
LfF [33]	24.4	60.7
CVaR DRO [171]	36.1	46.4
JTT [31]	40.6	47.4
DFR [4]	46.0	49.8
DivDis [172]	55.0	35.8
MaskTune [173]	78.0	13.3
SPUME-ViT-GPT2	84.4±1.2	5.9±0.7
SPUME-BLIP	**86.0±1.0**	**4.1±1.0**

Table 4.13: Comparison of worst-group accuracy (%) and accuracy gap (%) on the CelebA dataset. All methods do not have access to ground-truth group labels.

For experiments on the Waterbirds and CelebA datasets, we aimed to simulate a more realistic learning scenario and thus did not provide group labels during model training, even though the two datasets provide group labels. During testing, we used the group labels to formulate the *worst-group accuracy* and calculated the *accuracy gap* as the standard average accuracy minus the worst-group accuracy. The two metrics measure a classifier's robustness to *specific* spurious correlations specified by the group labels, and our goal is to train the classifier to be robust to these spurious correlations without knowing them.

Our methods, SPUME-ViT-GPT2 and SPUME-BLIP achieve the best worst-group accuracy and the best accuracy gap on the Waterbirds and CelebA datasets (Tables 4.12 and 4.13), suggesting that our trained classifiers have strong and balanced prediction capability across different data groups. Note that the spurious attribute detection process proposed in Section 4.3.3 could introduce biases present in VLMs into the detected spurious attributes. More specifically, biases in different VLMs result in different sets of attributes. Consequently, SPUME simulates different sets of spurious correlations during meta-training. However, this wouldn't be a significant concern. Since our spurious attribute detection process can detect many distinctive attributes with well-established VLMs,

Method	Accuracy (↑)
ERM	75.9
REx [188]	74.3
Group DRO [3]	77.6
JiGen [111]	85.0
Mixup [189]	80.3
CNBB [148]	78.2
DecAug [186]	85.2
SIFER [190]	86.2±0.9
SPUME-ViT-GPT2	88.2±1.1
SPUME-BLIP	**89.2±0.4**

Table 4.14: Comparison of average accuracy (%) on the NICO dataset. Most of the methods (DecAug, DRO, etc) use group information for training, while we do not use it.

SPUME can mitigate many potential spurious correlations. Thus, biases in VLMs won't significantly affect the effectiveness of our framework. We demonstrate this by showing that SPUME with two well-established VLMs are effective and have comparable performance across different datasets (Tables 4.12 and 4.13). Moreover, SPUME-BLIP performs much better than SPUME-ViT-GPT2 on the CelebA dataset where BLIP detects approximately twice as many attributes as ViT-GPT2 does (Table 4.11), suggesting that detecting more attributes benefits SPUME in training more robust classifiers.

The NICO dataset provides object-context correlations and aims to evaluate the out-of-distribution generalization capability of a classifier by testing it in new contexts. We did not use the provided correlations during training and calculated the standard average accuracy on the test set with new object-context correlations. SPUME-ViT-GPT2 and SPUME-BLIP outperform previous methods with higher average accuracies (Table 4.14).

For the experiments on the ImageNet-9 which does not provide information on spurious correlations, we trained and tested our methods on the ImageNet-9 dataset. We also tested our methods on the ImageNet-A dataset which contains images representing various failure prediction modes in an ImageNet pre-trained classifier. The accuracy gap is calculated as the average validation accuracy on the ImageNet-9 dataset minus the average accuracy on the ImageNet-A dataset. Our methods achieve the best on ImageNet-A while well balancing between different prediction modes with the lowest accuracy gaps (Table 4.15).

Method	ImageNet-9 (↑)	ImageNet-A (↑)	Acc. gap (↓)
ERM	90.8±0.6	24.9±1.1	65.9
ReBias [181]	91.9±1.7	29.6±1.6	62.3
LfF [33]	86.0	24.6	61.4
CaaM [182]	95.7	32.8	62.9
SSL+ERM [183]	94.2±0.1	34.2±0.5	60
LWBC[183]	94.0±0.2	36.0±0.5	58
SIFER [187]	**97.8±0.1**	40.0±0.8	57.8
SPUME-ViT-GPT2	95.3±0.5	**44.3±0.8**	**51.0±1.1**
SPUME-BLIP	95.5±0.2	42.5±0.8	53.0±0.7

Table 4.15: Comparison of average accuracy (%) and accuracy gap (%) on the ImageNet-9 and ImageNet-A datasets.

Method	Worst-group acc (↑)	Acc. gap (↓)
ERM	66.4	23.8
ERM-Cosine	75.5	17.5
SPUME-Random	78.7±0.9	10.5±0.8
SPUME-BLIP	85.7±0.2	**6.1±0.4**
SPUME-ViT-GPT2	**85.9±0.3**	6.9±0.8

Table 4.16: Worst-group accuracy and accuracy gap comparisons between meta-learning based methods with spuriousness-aware (SPUME-BLIP and SPUME-ViT-GPT2) and random (SPUME-Random) task constructions, and ERM-trained models on the Waterbirds dataset.

Ablation Study

Spuriousness-Aware Task Construction. To evaluate the effectiveness of using VLMs to guide the construction of meta-learning tasks, we compared SPUME with SPUME-Random which uses randomly constructed tasks during training. We also included the classical ERM model and the ERM-Cosine model that uses cosine distance for predictions to compare with the meta-learning based approaches. We observe from Table 4.16 that switching to the cosine-distance-based classifier increases the robustness to spurious correlations. Moreover, SPUME-Random outperforms ERM by 12.3% in the worst-group accuracy and improves the accuracy gap by 13.3%, demonstrating that meta-learning is a promising approach to improve the robustness to spurious correlations. Additionally, using spuriousness-aware meta-learning tasks constructed with the VLMs (BLIP and ViT-GPT2) can further improve robustness to spurious correlations. Specifically, SPUME-BLIP achieves 7.0% and 4.4% increments over SPUME-Random in the worst-group accuracy and accuracy gap, respectively, and SPUME-ViT-GPT2 achieves 7.2% and 3.6% increments in the two metrics.

Different Designs of the Spuriousness Metric. We have given our design of spuriousness metric in Equation (4.17). Here, we explore other possible design choices shown in Table 4.17, where

Metric	Worst-group acc. (↑)	Acc. gap (↓)
tanh(abs(log(η)))	**85.7±0.2**	**6.1±0.4**
abs(δ)	85.5±0.2	6.3±0.3
Constant	85.1±0.2	6.7±0.3
tanh(log(η))	84.8±0.2	7.3±0.4
δ	84.5±0.5	7.4±0.9

Table 4.17: Analysis on different designs of spuriousness metrics. We tested SPUME-BLIP on the Waterbirds dataset.

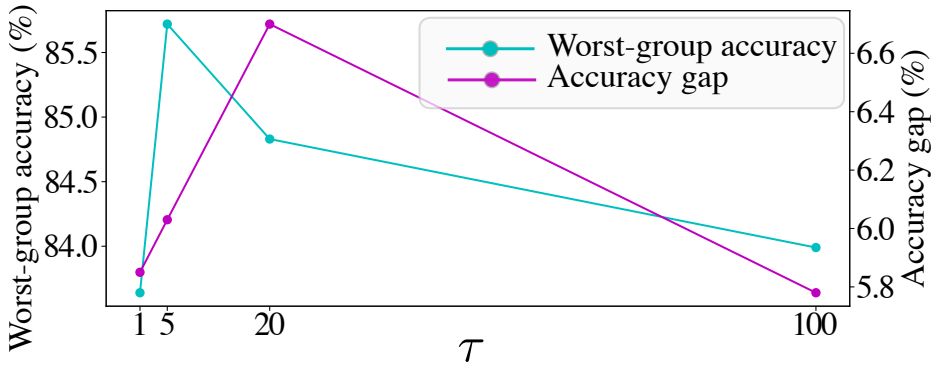


Figure 4.13: Worst-group accuracy and accuracy gap comparisons between SPUME-BLIP with different τ's on Waterbirds.

$\delta = J(\mathcal{D}_{tr}^{(y,a)}; f_\theta) - J(\mathcal{D}_{tr}^{(y,\hat{a})}; f_\theta)$, $\eta = J(\mathcal{D}_{tr}^{(y,a)}; f_\theta)/J(\mathcal{D}_{tr}^{(y,\hat{a})}; f_\theta)$, $J(\cdot;\cdot)$ is the accuracy measure used in Equation (4.17), and "Constant" represents that we assign the same score for all the detected attributes. Our method SPUME-BLIP works well with different spuriousness metrics and still outperforms the baselines we compared in Table 4.12. Moreover, our method works well with non-negative spuriousness metrics as SPUME with tanh(abs(log(η))) or abs(δ) performs better than with the other two metrics.

Scaling Parameter of the Centroid-Based Classifier. We analyzed how the scaling parameter τ of the centroid-based classifier (Equation (4.21)) affects the performance of SPUME. Figure 4.13 shows the worst-group accuracies and accuracy gaps of SPUME-BLIP with different τ's on the Waterbirds dataset. A very large or small τ, e.g., $\tau = 100$ or $\tau = 1$, harms to robustness of the trained classifiers. In practice, we set τ to be in the range from 5 to 50.

Effects of Using VLMs. Although SPUME uses VLMs for data preprocessing, the robustness does not directly come from the outputs of VLMs. To show this, we added an additional layer after the backbone to predict detected attributes for each image, acting as a regularization. We then fine-tuned the whole model on the Waterbirds and CelebA datasets, respectively. The worst-

group accuracies on the two datasets are 71.7% and 47.2%, respectively, which are close to ERM trained models. Therefore, the attributes themselves do not provide useful regularization on the robustness of the classifier. Moreover, directly using VLMs for predictions requires a completely different inference pipeline and is not as effective as our proposed SPUME. Details are provided in Appendix A.5.

4.3.5 Conclusion

We proposed a novel framework to train a classifier to be robust against spurious correlations in settings where spurious correlations are not known or specified. We first adopted a pre-trained VLM to automatically extract text-format attributes from a target dataset. Then, we quantified the spuriousness of the correlations between detected attributes and class labels using a spuriousness metric. To effectively mitigate multiple detected spurious correlations, we adopted a meta-learning strategy which iteratively meta-trains a classifier on multiple meta-learning tasks constructed to represent various class-attribute correlations with high spuriousness values. Our framework, SPUME, mitigates many highly spurious correlations in training samples and performs the best under different robustness measures on five benchmark datasets. In the future, we aim to explore more capable VLMs and combine other approaches, e.g., customized data augmentations, for mitigating a model's reliance on a wider range of spurious correlations.

Chapter 5

Self-Guided Spurious Bias Mitigation under Subpopulation Shifts

In the previous chapter, we propose multimodal-assisted methods to address spurious bias without requiring group annotations. However, those approaches require a relatively long data preprocessing step to extract spurious attributes from training data, depend on specific choices of vision-language models (VLMs), and are limited to the vision modality. In this chapter, we propose fully self-guided methods that probe the latent space of a trained model, allowing direct spurious bias mitigation with any data modalities. In Chapter 5.1, we propose to detect prediction shortcuts in the latent space of a model via a probe set and regularize the retraining of the model via the detected prediction shortcuts. In Chapter 5.2, we propose to suppress the contributions from neurons that are identified as primarily encoding spurious features for spurious bias mitigation. In Chapter 5.3, we extend the idea of these latent-space methods to improve the robustness of zero-shot classification of a VLM by selecting prompts that minimize the correlations between spurious features and prediction targets using the similarities between image and text embeddings.

5.1 ShortcutProbe: Probing Prediction Shortcuts for Learning Robust Models

5.1.1 Introduction

Deep learning models have shown remarkable performance across domains, but this success is often achieved by exploiting spurious correlations [139, 31, 29, 83, 46, 42] between spurious attributes or shortcut features [2] and targets. For example, models have been found to use correlations between textures and image classes [48] for object recognition instead of focusing on defining features of objects. This issue becomes even more problematic in high-stakes domains like healthcare. For instance, models predicting pneumonia were shown to rely on correlations between metal tokens in chest X-ray scans from different hospitals and the disease's detection outcomes [7], rather than the pathological features of pneumonia itself. The tendency of using spurious correlations is referred to as *spurious bias*. Models with spurious bias often fail to generalize on data groups lacking the learned spurious correlations, leading to significant performance degradation and non-robust behaviors across different data groups. This robustness issue can have severe social consequences, especially in critical applications.

Mitigating spurious bias is crucial for robust generalization across data groups with varying spurious correlations. Existing methods on mitigating spurious bias [139, 4] rely on group labels. Group labels represent spurious correlations with class labels and spurious attributes. For example, `(waterbirds, water)` [139] represents a spurious correlation between waterbirds and water backgrounds in the images of waterbirds with water backgrounds. Using group labels specifies explicitly the spurious correlations that a model should avoid. However, obtaining group labels requires expert knowledge and labor-intensive annotation efforts. Moreover, group labels fail to capture subtle spurious biases, such as using certain pixels in images for predictions.

In this section, we propose a post hoc approach that can automatically detect and mitigate potential spurious biases in a model rather than relying on group labels. Our key innovation is reframing the task of identifying and mitigating spurious biases as detecting and leveraging *prediction shortcuts* in the model's latent space. Prediction shortcuts are latent features derived from input embeddings and predominantly contribute to producing the same prediction outcome across different classes. In essence, prediction shortcuts represent non-defining features of certain classes that the model heavily uses for predictions. By operating in the model's latent space, our approach

leverages the expressiveness of latent embeddings, enabling direct identification of spurious biases across diverse input formats without requiring group labels.

We present our post hoc approach as a novel framework called *ShortcutProbe*, which first identifies prediction shortcuts in a given model and leverages them to guide model retraining for spurious bias mitigation. ShortcutProbe utilizes a probe set without group labels, typically containing a diverse mix of features, to uncover potential prediction shortcuts. These shortcuts are identified as latent features extracted from sample embeddings belonging to different classes but producing the same prediction outcome. By optimizing these features to maximize the model's confidence in their corresponding predictions, ShortcutProbe effectively encodes spurious attributes in non-generalizable prediction shortcuts that the model overly relies on for predictions.

With the identified prediction shortcuts, ShortcutProbe mitigates spurious biases by retraining the model to be invariant to these shortcuts, as they are irrelevant to true prediction targets. This invariance is achieved by applying regularization during retraining, which ensures that the identified prediction shortcuts no longer contribute to the model's predictions of the true targets.

We theoretically demonstrate that when the spurious attributes in the training data are new to the model as reflected by the high prediction loss, the tendency of using spurious attributes for predictions is high after training on the data. Our method aims to revert the process of learning spurious attributes by retraining the model so that the learned spurious attributes induce high prediction losses, effectively unlearning the spurious attributes and reducing the influence of spurious correlations.

In summary, our contributions in this section are as follows:

- We introduce ShortcutProbe, a novel post hoc framework for mitigating spurious bias without requiring group labels. ShortcutProbe identifies prediction shortcuts and leverages them as a form of regularization for training robust models.
- We provide a theoretical analysis revealing that our spurious bias mitigation approach effectively unlearns spurious attributes.
- Through extensive experiments, we show that our method successfully trains models robust to spurious biases without prior knowledge about these biases.

5.1.2 Preliminary

A *spurious correlation* is the correlation between a *spurious attribute* present in the training samples and a prediction target. For example, the class `waterbird` and the attribute `water background`

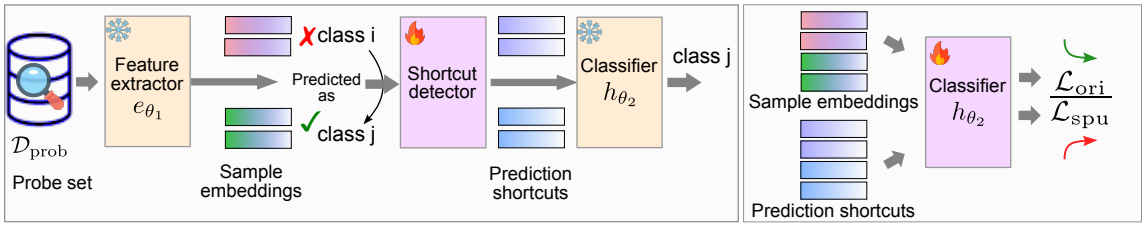


Figure 5.1: Illustration of ShortcutProbe. (a) The framework uses a set of probe data $\mathcal{D}_{\text{prob}}$ to identify prediction shortcuts by learning a shortcut detector to extract similar features from samples of different classes i and j that are all predicted as the same class j. Feature extractor e_{θ_1} and classifier h_{θ_2} are frozen during this stage. (b) ShortcutProbe then retrains the classifier with the probe data (the loss of the probe data $\mathcal{L}_{\text{ori}}$) while using the identified prediction shortcuts as regularization (the loss of the prediction shortcuts $\mathcal{L}_{\text{spu}}$).

might form a spurious correlation in the images of `waterbird`, where some of them have water backgrounds, e.g., pond or river, and some do not. Spurious attributes are not truly predictive of the targets. A group label, e.g., (`waterbird`, `water background`), consists of a prediction target and a spurious attribute.

Consider a model $f_\theta : \mathcal{X} \to \mathbb{R}^{|\mathcal{Y}|}$ with parameter θ trained on a training dataset $\mathcal{D}_{\text{tr}} = \{(x_i, y_i)\}_{i=1}^{N}$ with N sample-label pairs, where $x_i \in \mathcal{X}$ denotes a sample in the input set $\mathcal{X}$, $y_i \in \mathcal{Y}$ denotes a label in the label set $\mathcal{Y}$, and $|\cdot|$ denotes the size of a set. The model $f_\theta = e_{\theta_1} \circ h_{\theta_2}$ can be considered as a feature extractor $e_{\theta_1} : \mathcal{X} \to \mathbb{R}^D$ followed by a classifier $h_{\theta_2} : \mathbb{R}^D \to \mathbb{R}^{|\mathcal{Y}|}$, where $\theta = \theta_1 \cup \theta_2$, h_{θ_2} is the last layer of the model, and D denotes the number of dimensions.

Due to the existence of spurious attributes in $\mathcal{D}_{\text{tr}}$, the model can exploit them for predictions, such as recognizing waterbirds by detecting the existence of water backgrounds [3]. This presents a challenge: It is hard to determine whether a high-performing model is truly robust or simply "right for the wrong reasons", i.e., relying on spurious attributes. Although models with spurious biases typically exhibit degraded performance when the learned spurious attributes are absent from input data, e.g., a waterbird on a land background, it remains challenging to identify the specific spurious attributes encoded by the model without group labels, which hinders the development of effective spurious bias mitigation strategies.

5.1.3 Methodology

Method Overview

We introduce *ShortcutProbe*, a post hoc framework that automatically detects and mitigates prediction shortcuts without requiring group labels to specify which spurious biases to address. The

framework comprises two key steps: (1) **Prediction shortcut detection**, where a probe set is used to identify prediction shortcuts, and (2) **Spurious bias mitigation**, where the identified shortcuts are used in retraining the model to mitigate spurious biases in the model.

We provide an overview of our framework in Figure 5.1. The process begins with a set of probe data $\mathcal{D}_{\text{prob}}$, which typically contains samples with various spurious attributes that reflect a model's non-robustness to spurious correlations. These samples are mapped to the latent embedding space of the model f_θ through its feature extractor e_{θ_1}, allowing us to model *any* prediction shortcuts the model might use for predictions. This strategy bypasses the need to explicitly define spurious correlations through group labels, a task that is especially challenging for subtle features, such as specific pixels in images. More concretely, as shown in Figure 5.1(a), we first train a shortcut detector that extracts potential prediction shortcuts from samples of different classes but having the same prediction. The identified prediction shortcuts encode spurious attributes shared across classes and capture the model's non-robustness across different data groups. Next, in Figure 5.1(b), the model is retrained with $\mathcal{D}_{\text{prob}}$ to mitigate spurious biases by unlearning the identified prediction shortcuts.

Prediction Shortcut Detection

Given a model f_θ and a probe set $\mathcal{D}_{\text{prob}}$, we aim to detect prediction shortcuts by learning a shortcut detector $g_\psi : \mathbb{R}^D \to \mathbb{R}^D$ to identify prediction shortcuts from input embeddings. Intuitively, prediction shortcuts can be identified from samples of different classes but having the same prediction outcome, an indication that similar features exist in these samples but are irrelevant to classes. In other words, prediction shortcuts can be shared among samples from different classes, necessitating a shared representational space to encode diverse prediction shortcuts. We formalize this intuition in the following definition.

Definition 5.1 (Prediction shortcuts)**.** Given input sample-label pairs (x, y) and (x', y'), where $y \neq y'$, a trained model $f_\theta = e_{\theta_1} \circ h_{\theta_2}$, sample embeddings $\mathbf{v} = e_{\theta_1}(x)$ and $\mathbf{v}' = e_{\theta_1}(x')$ for x and x', respectively, a vector space $\mathcal{V} \subset \mathbb{R}^D$ spanned by K base column vectors in $\mathbf{A} \in \mathbb{R}^{D \times K}$, prediction shortcuts $\mathbf{s}_x \in \mathcal{V}$ and $\mathbf{s}_{x'} \in \mathcal{V}$ for the two samples satisfy the following conditions:

- $\text{Pred}\big(h_{\theta_2}(e_{\theta_1}(x))\big) = y$, $\text{Pred}\big(h_{\theta_2}(\mathbf{s}_x)\big) = y$, and
- $\text{Pred}\big(h_{\theta_2}(e_{\theta_1}(x'))\big) = y$, $\text{Pred}\big(h_{\theta_2}(\mathbf{s}_{x'})\big) = y$,

where $\text{Pred}(f_\theta(x)) = \arg\max_{\mathcal{Y}} f_\theta(x)$, $\mathbf{s}_x = \mathbf{P}_\mathbf{A}\mathbf{v}$, $\mathbf{s}_{x'} = \mathbf{P}_\mathbf{A}\mathbf{v}'$, and

$$\mathbf{P}_\mathbf{A} = \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T \tag{5.1}$$

is the projection matrix such that the prediction shortcut $\mathbf{s}_x$ is the best estimate of $\mathbf{v}$ in the vector space $\mathcal{V}$ in the sense that the distance $\|\mathbf{s}_x - \mathbf{v}\|_2^2$ is minimized.

In Definition 5.1, we define a prediction shortcut as a projection of a sample embedding. It exists in the vector space $\mathcal{V}$ shared by samples of different classes with K degrees of freedom. Here, K governs the complexity of the vector space representing prediction shortcuts. A smaller K results in a less expressive vector space that may fail to adequately capture prediction shortcuts. Conversely, a larger K provides greater flexibility in representing prediction shortcuts but may encode irrelevant information. The optimal value of K depends on the complexity of the probe data; values that are too small or too large can impede learning and lead to suboptimal performance. We treat K as a tunable hyperparameter.

By representing prediction shortcuts as vectors, we can in theory capture any spurious bias, even the intricate ones. For instance, a vector $\mathbf{s}$ might correspond to features of water backgrounds that are used to predict waterbirds in any image with water backgrounds, revealing a spurious bias in the model. Alternatively, $\mathbf{s}$ could represent a specific feature corresponding to certain pixels in input images, capturing the prediction shortcut based on low-level pixel values—an aspect that is challenging to articulate through group labels.

Learning the Shortcut Detector. Based on the definition of the prediction shortcut, we design the shortcut detector g_ψ as a function that implements the projection operation defined in Equation (5.1) with the learnable parameter $\psi = \mathbf{A} \in \mathbb{R}^{D\times K}$, that is for a sample embedding $\mathbf{v} \in \mathbb{R}^D$, $g_\psi(\mathbf{v}) = \mathbf{P}_\mathbf{A}\mathbf{v}$. Learning g_ψ essentially learns a shared vector space spanned by $\mathbf{A}$ that could cover prediction shortcuts in samples in the probe set.

To effectively learn g_ψ, for each class y, we first collect samples from the probe set $\mathcal{D}_{\text{prob}}$ to formulate two non-overlapping sets $\mathcal{D}^y_{\text{cor}}$ and $\mathcal{D}^y_{\text{pre}}$, i.e.,

$$\mathcal{D}^y_{\text{cor}} = \{(x, y)|(x, y) \in \mathcal{D}_{\text{prob}}, \text{Pred}(f_\theta(x)) = y\}, \tag{5.2}$$

and

$$\mathcal{D}^y_{\text{pre}} = \{(x', y')|(x', y') \in \mathcal{D}_{\text{prob}}, \text{Pred}(f_\theta(x')) = y \neq y'\}, \tag{5.3}$$

where $\mathcal{D}^y_{\text{cor}}$ and $\mathcal{D}^y_{\text{pre}}$ contain samples that are correctly and incorrectly predicted as y. The two sets together demonstrate a possibility that certain features shared across classes are incorrectly

associated with prediction targets, allowing us to extract these features as potential prediction shortcuts.

Next, we propose the following objective to identify prediction shortcuts:

$$\mathcal{L}_{\text{det}} = \mathop{\mathbb{E}}_{y\in\mathcal{Y}} \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}^y_{\text{cor}}\cup\mathcal{D}^y_{\text{pre}}} \ell\big(h_{\theta_2}(g_\psi(\mathbf{v})), y\big), \tag{5.4}$$

where $\ell : \mathbb{R}^{|\mathcal{Y}|} \times \mathcal{Y} \to \mathbb{R}$ is the loss function, $\psi = \mathbf{A}$, and $\mathbf{v} = e_{\theta_1}(x)$. To ensure that prediction shortcuts are relevant to the given samples, we regularize the semantic similarity between $g_\psi(\mathbf{v})$ and $\mathbf{v}$ as follows,

$$\mathcal{L}_{\text{reg}} = \mathop{\mathbb{E}}_{y\in\mathcal{Y}} \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}^y_{\text{cor}}\cup\mathcal{D}^y_{\text{pre}}} \|g_\psi(\mathbf{v}) - \mathbf{v}\|_2^2. \tag{5.5}$$

The overall learning objective for ψ is

$$\psi^* = \arg\min_{\psi} \mathcal{L}_{\text{det}} + \eta\mathcal{L}_{\text{reg}}, \tag{5.6}$$

where $\eta > 0$ represents the regularization strength. The objective in Equation (5.6) aims to encode the properties of prediction shortcuts in Definition 5.1 into the shortcut detector g_ψ while maintaining the relevance of the prediction shortcuts to input samples. Training g_ψ is lightweight as there are only DK learnable parameters. With g_ψ, we can identify multiple prediction shortcuts from samples in $\mathcal{D}_{\text{prob}}$.

Spurious Bias Mitigation

Mitigating spurious biases in a model requires that the spurious attributes captured during training are no longer predictive of the targets. Although identifying spurious attributes can be challenging, the shortcut detector introduced in the previous section identifies prediction shortcuts as potential spurious attributes utilized by the model, providing valuable guidance for addressing spurious biases.

In the following, we formulate an optimization objective that incorporates the above constraint to learn a robust model using the probe set $\mathcal{D}_{\text{prob}}$. First, a general requirement is that the trained model should produce correct and consistent predictions on training samples. To this end, for each class y, we sample from $\mathcal{D}^y_{\text{cor}}$ and $\mathcal{D}^y_{\text{mis}}$, where $\mathcal{D}^y_{\text{cor}}$ is the set of correctly predicted sample-label pairs defined in Equation (5.2), and $\mathcal{D}^y_{\text{mis}}$ is defined as follows,

$$\mathcal{D}^y_{\text{mis}} = \{(x', y) | (x', y) \in \mathcal{D}_{\text{prob}}, \text{Pred}(f_\theta(x')) \neq y\}, \tag{5.7}$$

representing misclassified sample-label pairs from the class y. Then, the training objective $\mathcal{L}_{\text{ori}}$ is as follows,

$$\mathcal{L}_{\text{ori}}(\mathcal{D}_{\text{prob}};\theta) = \mathop{\mathbb{E}}_{y\in\mathcal{Y}} \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}^y_{\text{cor}}\cup\mathcal{D}^y_{\text{mis}}} \ell(f_\theta(x),y), \tag{5.8}$$

which mitigates potential spurious biases by ensuring consistent predictions in $\mathcal{D}^y_{\text{mis}}$ and $\mathcal{D}^y_{\text{cor}}$.

Moreover, to ensure that the training targets at mitigating the spurious biases in the model, we further formulate a regularization term using the prediction shortcuts identified by our shortcut detector to guide the training process. Specifically, as the identified prediction shortcuts are not predictive of the targets, we aim to maximize the loss on the prediction shortcuts defined as follows,

$$\mathcal{L}_{\text{spu}}(\mathcal{D}_{\text{prob}};\theta) = \mathop{\mathbb{E}}_{y\in\mathcal{Y}} \mathop{\mathbb{E}}_{(x,y)\in\mathcal{D}^y_{\text{cor}}\cup\mathcal{D}^y_{\text{mis}}} \ell(h_{\theta_2}(g_\psi(\mathbf{v})),y), \tag{5.9}$$

where $\mathbf{v} = e_{\theta_1}(x)$.

We incorporate the above two loss terms into the **overall training objective** as follows,

$$\theta_2^* = \arg\min_{\theta_2} \lambda\mathcal{L}_{\text{ori}}/\mathcal{L}_{\text{spu}}, \tag{5.10}$$

where $\lambda > 0$ is the regularization strength. Here, we retrain only the final classification layer of the model while keeping the feature extractor frozen. This approach significantly reduces computational complexity and allows us to reuse the previously learned sample embeddings. Details of the training algorithm are provided in Appendix A.6.

Choice of the Probe Set

The probe set plays a crucial role in both detecting prediction shortcuts and mitigating spurious biases. To achieve the full potential of ShortcutProbe, we use a held-out dataset—unseen by the model—to construct the probe set $\mathcal{D}_{\text{prob}}$. This choice is essential because the model may have memorized the training samples, making it difficult to identify prediction shortcuts based on discrepancies in predictions for samples of the same class. Moreover, we select samples with high predication confidence (measured by output entropy) from the held-out dataset to construct $\mathcal{D}_{\text{prob}}$ so that prediction shortcuts can be easily detected from these samples. Details of constructing $\mathcal{D}_{\text{prob}}$ and results on different choices of $\mathcal{D}_{\text{prob}}$ are provided in Section 5.1.4.

Theoretical Analysis

We theoretically demonstrate that minimizing the proposed objective $\mathcal{L}_{\text{ori}}/\mathcal{L}_{\text{reg}}$ effectively unlearns the spurious correlations between spurious attributes and their associated targets captured in the model. Without loss of generality, we analyze this in the context of a general linear regression setting. Consider an input sample $x \in \mathcal{X}$, a prediction target $y \in \mathcal{Y}$, and a spurious-only sample $\tilde{x}$ that lacks any defining features in x related to y. Let $x_1, \ldots, x_N$ denote N training samples, $\varphi : \mathcal{X} \to \mathbb{R}^D$ be a generic feature map, and $J_{\mathbf{w}}(x) = \varphi(x)^T\mathbf{w}$ represent a generalized linear regression model with parameters $\mathbf{w} \in \mathbb{R}^D$.

We denote the correlation between the model output for a spurious-only sample $\tilde{x}$ and a prediction target y as $\rho(J_{\mathbf{w}}(\tilde{x}), y)$. The following lemma [191] gives an upper bound on the correlation.

Lemma 5.1. The correlation between the model output for the spurious-only sample $\tilde{x}$ and the prediction target y is upper bounded as follows:

$$\rho(J_{\mathbf{w}}(\tilde{x}), y) \leq \gamma_{\varphi}\sigma_{\mathcal{Y}}\sqrt{\mathcal{R}_{\mathcal{X}}}, \tag{5.11}$$

where $\mathcal{R}_{\mathcal{X}}$ is the generalization error, $\sigma_{\mathcal{Y}}$ is the standard deviation of prediction targets, and

$$\gamma_{\varphi} = \mathbb{E}_{\tilde{x},x}\Big[\frac{\varphi(\tilde{x})^T\mathbf{O}\varphi(x)}{\|\mathbf{O}\varphi(x)\|_2^2}\Big], \tag{5.12}$$

where $\mathbf{O} = \mathbf{I} - \mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V}$ is the orthogonal projection matrix, and $\mathbf{V} = [\varphi(x_1), \ldots, \varphi(x_N)]^T \in \mathbb{R}^{N\times D}$ is the feature matrix.

Given that $\mathcal{R}_{\mathcal{X}}$ and $\sigma_{\mathcal{Y}}$ are independent of $\tilde{x}$, the spurious-only sample $\tilde{x}$ affects the correlation upper bound via the feature alignment term γ_{φ} between the spurious attribute of $\tilde{x}$ and the original feature of x. We further interpret this term in the following proposition.

Proposition 5.1. The feature alignment term γ_{φ} is the ratio between the expected prediction error on the spurious sample $\tilde{x}$ and the expected prediction error on the original sample x:

$$\gamma_{\varphi} = \mathbb{E}_{\tilde{x},x}\Big[\frac{\varphi(\tilde{x})^T\mathbf{O}\varphi(x)}{\|\mathbf{O}\varphi(x)\|_2^2}\Big] = \frac{\mathbb{E}_{\tilde{x}}[\|\mathbf{O}\varphi(\tilde{x})\|_2]}{\mathbb{E}_{x}[\|\mathbf{O}\varphi(x)\|_2]}. \tag{5.13}$$

The term $\|\mathbf{O}\varphi(x)\|_2$ in Equation (5.13) denotes the error term for the sample x, while $\|\mathbf{O}\varphi(\tilde{x})\|_2$ denotes the error term for the spurious sample $\tilde{x}$. Since x and $\tilde{x}$ are independent, the feature alignment term γ_{φ} is the ratio between the loss on spurious-only samples and the loss on the original samples. We provide a detailed proof in Appendix A.6.

Method	Waterbirds			CelebA			CheXpert		
	WGA (↑)	Average (↑)	Gap (↓)	WGA (↑)	Average (↑)	Gap (↓)	WGA (↑)	Average (↑)	Gap (↓)
ERM [193]	$80.3_{\pm 3.1}$	$93.3_{\pm 0.4}$	13.0	$45.6_{\pm 2.9}$	$95.2_{\pm 0.2}$	49.6	$22.0_{\pm 1.6}$	$90.8_{\pm 0.1}$	68.8
JTT [31]	$86.7_{\pm 1.0}$	$93.3_{\pm 0.2}$	6.6	$40.6_{\pm 1.2}$	$88.6_{\pm 0.2}$	48.0	$60.4_{\pm 4.9}$	$75.2_{\pm 0.8}$	14.8
DFR [4]	$90.3_{\pm 2.1}$	$95.0_{\pm 1.3}$	4.7	$72.2_{\pm 2.0}$	$\mathbf{92.9}_{\pm 0.1}$	20.7	$72.7_{\pm 1.5}$	$78.7_{\pm 0.4}$	6.0
AFR [32]	$88.7_{\pm 4.2}$	$95.0_{\pm 1.0}$	6.3	$77.8_{\pm 1.5}$	$91.0_{\pm 0.4}$	13.2	$72.4_{\pm 2.0}$	$76.8_{\pm 1.1}$	4.4
ShortcutProbe (Ours)	$\mathbf{90.8}_{\pm 0.6}$	$\mathbf{95.0}_{\pm 0.3}$	**4.2**	$\mathbf{83.4}_{\pm 0.9}$	$91.4_{\pm 0.1}$	**8.0**	$\mathbf{75.0}_{\pm 0.7}$	$\mathbf{79.0}_{\pm 0.2}$	**4.0**

Table 5.1: Comparison of worst-group accuracy (WGA) and average accuracy (%) with baseline methods on the Waterbirds, CelebA, and CheXpert datasets. The best results are highlighted in **boldface**. All bias mitigation methods use the same half of the validation set.

The prediction shortcuts obtained by our shortcut detector approximate the spurious-only features. Thus, the loss $\mathcal{L}_{\text{spu}}$ approximates the nominator of the regularization term in Equation (5.13). Moreover, $\mathcal{L}_{\text{ori}}$ approximates the denominator in Equation (5.13). Note that Lemma 1 gives the upper bound of the spurious correlation *before* learning it. The objective in Equation (5.4) trains the shortcut detector to learn the correlation by minimizing γ_ϕ. In the mitigation step, the objective in Equation (5.10) unlearns the correlation by maximizing γ_ϕ.

5.1.4 Experiments

Datasets

Image Datasets. **Waterbirds** [3] contains waterbird and landbird classes selected from the CUB dataset [163]. The two bird classes are mixed with water and land backgrounds from the Places dataset [174]. **CelebA** [149] is a large-scale image dataset of celebrity faces. Images showing two hair colors, non-blond and blond, are spuriously correlated with gender. **CheXpert** [192] is a chest X-ray dataset containing six spurious attributes from the combination of race (White, Black, Other) and gender (Male, Female). Two diagnose results, i.e., "No Finding" (positive) and "Finding" (negative) are the labels. **ImageNet-9** [185] is a subset of ImageNet [175] and contains nine super-classes. It is known to have correlations between object classes and image textures. We prepared the training and validation data as in [183] and [181]. **ImageNet-A** [176] is a dataset of real-world images, adversarially curated to test the limits of classifiers such as ResNet-50. While these images are from standard ImageNet classes [175], they are often misclassified in multiple models. We used this dataset to test the robustness of a classifier after training it on ImageNet-9. **NICO** [148] is designed for out-of-distribution image classification, simulating real-world scenarios where testing distributions differ from training ones. It labels images with both main concepts (e.g., cat) and contexts (e.g., at home). We used the Animal super-class in NICO and followed the setting in [186, 187] for data preparation.

Method	MultiNLI			CivilComments		
	WGA (↑)	Average (↑)	Gap (↓)	WGA (↑)	Average (↑)	Gap (↓)
ERM [193]	$67.0_{\pm 0.4}$	$82.2_{\pm 0.2}$	15.2	$58.5_{\pm 1.3}$	$92.2_{\pm 0.1}$	33.7
JTT [31]	$71.6_{\pm 0.8}$	$80.7_{\pm 0.4}$	9.1	$68.3_{\pm 0.9}$	$\mathbf{89.0}_{\pm 0.3}$	20.7
DFR [4]	$72.6_{\pm 1.7}$	$81.8_{\pm 0.4}$	9.2	$76.6_{\pm 0.8}$	$85.8_{\pm 0.5}$	9.2
AFR [32]	$66.6_{\pm 0.3}$	$82.2_{\pm 0.2}$	15.6	$74.6_{\pm 5.1}$	$84.7_{\pm 2.5}$	10.1
ShortcutProbe (Ours)	$\mathbf{74.3}_{\pm 0.7}$	$\mathbf{82.6}_{\pm 0.3}$	**8.3**	$\mathbf{79.9}_{\pm 0.6}$	$88.5_{\pm 0.2}$	**8.6**

Table 5.2: Comparison of worst-group accuracy (WGA) and average accuracy (%) with baseline methods on the MultiNLI and Civilcomments datasets. Best results are highlighted in **boldface**. All bias mitigation methods use the same half of the validation set.

Text Datasets. **MultiNLI** [194] is a text classification dataset with 3 classes: neutral, contradiction, and entailment, representing the natural language inference relationship between a premise and a hypothesis. The spurious feature is the presence of negation, which is highly correlated with the contradiction label. **CivilComments** [195] is a binary text classification dataset aimed at predicting whether a comment contains toxic language. Spurious features involve references to eight demographic identities: male, female, LGBTQ, Christian, Muslim, other religions, Black, and White.

Experimental Setup

Constructing the Probe Set. From the chosen data source, such as the training or validation set, we sorted the samples within each class by their prediction losses and divided them into two equal halves: a high-loss set and a low-loss set. This approach approximates the incorrectly and correctly predicted samples, respectively, while ensuring that the incorrectly predicted set is non-empty, even when all samples are correctly classified. Within each set, we then selected the top $r\%$ of samples with the highest prediction confidence (i.e., those with the lowest output entropy).

Training Details. We first trained a base model initialized with pretrained weights using empirical risk minimization (ERM) on the training dataset. Then, we retrained the model on half of the validation set using various bias mitigation methods. For our method, we first constructed the probe set using the same half of the validation set and used the probe set for shortcut detection and mitigation. The remaining half of the validation set was used for model selection and hyperparameter tuning. For experiments on the Waterbirds, CelebA, and CheXpert datasets, we used ResNet-50 as the backbone network, and we used ResNet-18 on the ImageNet-9/A and NICO datasets to ensure a fair comparison with baseline methods. For text datasets, we used a pretrained BERT model [196]. We trained models using each method three times with different random seeds and reported

the average results as well as their standard deviations. Detailed training settings are provided in Appendix A.6[1].

Evaluation Metrics. Without group labels, we used the worst-class accuracy [29] for model selection, which is defined as the worst per-class accuracy on an evaluation set. For performance evaluation on the Waterbirds, CelebA, CheXpert, MultiNLI, and Civilcomments datasets, we adopted the widely accepted metric, *worst-group accuracy*, which is the lowest accuracy among multiple groups of the test set with each group containing a certain spurious correlation. We also calculated the *accuracy gap* defined as the standard average accuracy minus the worst-group accuracy to measure the degree of a classifier's spurious biases. A high worst-group accuracy with a low accuracy gap indicates that the classifier is robust to spurious biases and can fairly predict samples from different groups. We adopted *average accuracy* for the evaluations on the NICO, ImageNet-9, and ImageNet-A datasets as these datasets are specifically constructed to evaluate the robustness against distributional shifts.

We give the full details of our experimental setup in Appendix A.6. We report our results averaged over 3 runs. All experiments are conducted on NVIDIA RTX 8000 GPUs.

Analysis of Probe Set

Our method relies on a probe set for detecting prediction shortcuts and mitigating spurious bias. A good probe set can be used to effectively reveal and mitigate spurious biases in a model, such as those curated with group labels [3]. Here, we show that our method, ShortcutProbe, performs effectively without group labels when a probe set is carefully selected from *readily available sources*, such as the training data and held-out validation data.

To demonstrate, we constructed a probe set as described in Experimental Setup, using the training set or half of the validation set (with the other half reserved for model selection) as the data source. For each data source, we varied the proportion $r\%$ from 20% to 100%. This adjustment created different probe sets, ranging from those containing only samples with highly confident predictions (small r) to those including all samples from the selected data source ($r = 100$).

Fig. 5.2(a) shows the performance of ShortcutProbe measured by worst-group accuracy (WGA) under different probe sets, while Fig. 5.2(b) presents the sizes of these probe sets. Compared to ERM models, we observe that using the training data for retraining results in minimal improvement

[1]Code is available at `https://github.com/gtzheng/ShortcutProbe`.

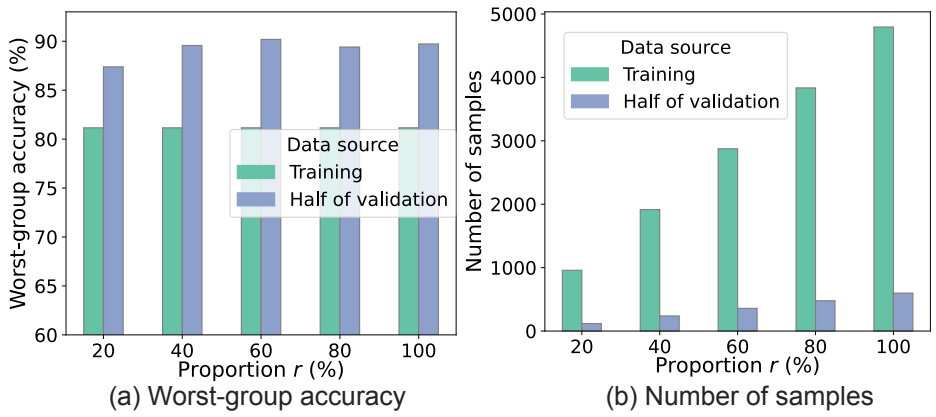


Figure 5.2: Analyses on different probe sets constructed from the Waterbirds dataset. (a) Worst-group accuracy comparison between models trained with training data and half of the validation data. (b) Numbers of samples in respective probe sets.

on the Waterbirds dataset. This is because most of the training data can be correctly predicted by the model, resulting in a probe set that is not informative for learning prediction shortcuts.

In contrast, by leveraging a relatively small amount of the held-out data compared to the size of the training data, our method demonstrates a significant improvement in robustness to spurious biases. Additionally, our approach benefits most from samples with high prediction confidence, i.e., when r is small. However, as shown in Fig. 5.2(a), setting r too small results in an insufficient number of training samples, leading to suboptimal WGA performance. In the following, unless otherwise specified, we use half of the validation set to construct the probe set and treat r as a tunable hyperparameter.

Main Results

We focus on a challenging and practical setting where group labels are unavailable in both the training and validation data. This scenario requires detecting and mitigating spurious biases using only the data and models available in a standard ERM training setup. As baselines, we selected state-of-the-art last-layer retraining methods DFR [4] and AFR [32], as well as JTT [31], which retrains the entire model. For DFR, which typically requires group labels, we used class labels instead. All baseline methods listed in Tables 5.1 and 5.2 utilized half of the validation data for training. Similarly, our method employed the same half of the validation data to construct a probe set, derived as a subset of this portion. The remaining half of the validation data was reserved for model selection.

As shown in Tables 5.1 and 5.2, our method achieves the highest WGA and the smallest accuracy gap between average accuracy and WGA, indicating its ability to strike a strong balance across

Method	Accuracy (↑)
ERM	75.9
REx [188]	74.3
Group DRO [3]	77.6
JiGen [111]	85.0
Mixup [189]	80.3
CNBB [148]	78.2
DecAug [186]	85.2
SIFER [190]	86.2
ShortcutProbe (Ours)	**90.5**$_{\pm 0.6}$

Table 5.3: Comparison of average accuracy (%) on the NICO dataset.

different data groups. Methods that achieve high average accuracy, such as DFR on the CelebA dataset, prioritize learning spurious features in the probe set. Although maintaining good predictivity on average, DFR still suffer from the prediction shortcuts, as shown by its low WGA. Our method remains effective on larger backbone networks beyond ResNet-50, such as ResNet-152 and ViT (see Appendix A.6). Unlike other baseline methods, our method uses only a portion of available data for training, highlighting the effectiveness of the probe set in detecting and mitigating spurious biases. Notably, when we applied the same probe set with baseline methods, this led to degraded performance in both WGA and accuracy gap, underscoring the unique advantages of our approach.

We further tested the out-of-distribution generalization of our method on the NICO dataset. Images of each class in the test set are associated with an unseen context. Our method achieves the best classification accuracy without using group labels (Table 5.3), demonstrating its effectiveness in mitigating the reliance on contexts. We present additional results on the ImageNet-9 and ImageNet-A datasets in Appendix A.6 to demonstrate our method's effectiveness in combating distributional shifts and the capability of achieving good tradeoff between in-distribution and out-of-distribution performance.

Ablation Studies

Prediction Shortcuts. We evaluated the effectiveness of prediction shortcuts in Figure 5.3(a). We began by using a randomly initialized shortcut detector to optimize the objective in Equation(5.10). Additionally, we tested the performance without the spurious bias regularization term $\mathcal{L}_{\text{spu}}$, represented as $\lambda = 0$ in Figure 5.3(a), as well as for λ values of 1, 10, and 50. Our results indicate that the model performs the worst when prediction shortcuts are not used as regularization. Interestingly, our method still demonstrates effectiveness even when the prediction shortcuts

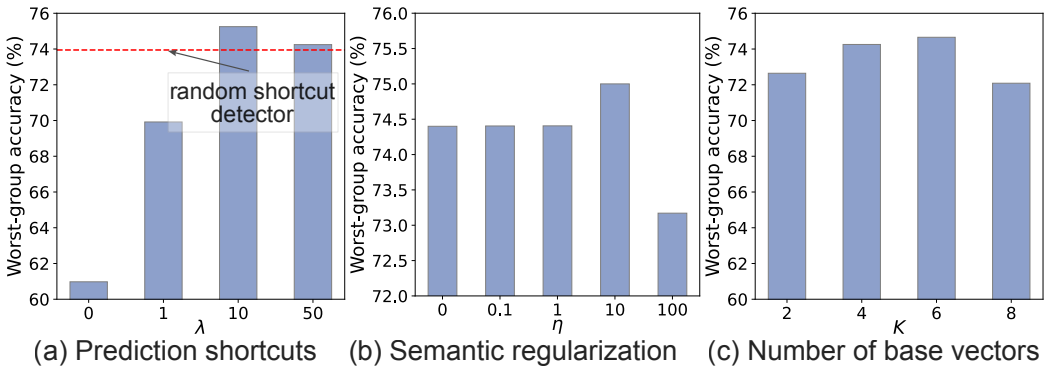


Figure 5.3: Analyses on how (a) prediction shortcuts as well as their regularization strength λ, (b) semantic regularization strength η, and (c) number of base vectors K affect a model's robustness to spurious biases. We report the worst-group accuracy on the CheXpert dataset.

are random. As λ increases, the regularization strength decreases. The model achieves optimal performance when an appropriately balanced λ is selected.

Semantic Regularization. We analyzed the impact of the semantic regularization strength η defined in Equation (5.5). Figure 5.3(b) shows that incorporating this regularization with a moderate value of η enhances the model's robustness.

Number of Base Vectors. The number of base vectors, K, determines the representational capacity of the shortcut detector. A value of K that is too small will limit the detector's ability to identify spurious attributes effectively, while an excessively large K may lead to overfitting on the probe data. Notably, as shown in Figure 5.3(c), the optimal value of K is 6, which coincides with the true number of spurious attributes in the CheXpert dataset.

5.1.5 Conclusion

In this work, we proposed a novel post hoc framework to mitigate spurious biases without requiring group labels. Our approach first learns a shortcut detector in the latent space of a given model via a diverse probe set. To mitigate spurious biases, we retrained the model to be invariant to detected prediction shortcuts using a novel regularized training objective. We theoretically demonstrated that this objective effectively unlearns the spurious correlations captured during training. Experiments confirmed that our method successfully mitigates spurious biases and enhances model robustness to distribution shifts. Future work may explore constructing a more diverse probe set to further enhance the detection and mitigation of spurious biases.

5.2 NeuronTune: Towards Self-Guided Spurious Bias Mitigation

5.2.1 Introduction

Deep neural networks trained using empirical risk minimization (ERM) often develop spurious bias: a tendency to rely on spurious correlations for predictions. A spurious correlation refers to a non-causal relationship between a class and an attribute that is not essential for defining the class, commonly referred to as a spurious attribute [46]. For example, the class of waterbird and the background of the water can form spurious correlations in the predictions of waterbird [3], as the background of the water is a spurious attribute. In contrast, core attributes, such as bird feathers, causally determine a class. A model with spurious bias may achieve a high prediction accuracy [6, 48, 2, 49, 42] even without core attributes, such as identifying an object only by its frequently co-occurring background [2]. However, the model may perform poorly on the data lacking the learned spurious correlations, which poses a great challenge to robust model generalization.

Existing methods [3, 4, 197] that mitigate spurious bias are mostly at the sample level, using a curated set of samples with annotations of spurious correlations called *group labels* to retrain a biased model. A group label (class, spurious attribute) annotates a sample with a spurious attribute and its class label, representing a spurious correlation. However, group labels are difficult to acquire and often require costly human-guided annotations. To circumvent this, group label estimation [34] and various sample reweighting mechanisms [33, 31, 183, 32, 83] are adopted using the idea that spurious bias can be identified through the misclassification of bias-conflicting samples.

Despite significant progress in spurious bias mitigation, existing sample-level methods that rely on group labels or sample reweighting offer limited and indirect control over how spurious bias is addressed. On the one hand, group labels are data annotations that are *external* to a model and may not accurately reflect the specific spurious bias developed in the model. On the other hand, sample reweighting does not directly target the internal mechanisms that give rise to spurious bias. This highlights the need for a **self-guided approach** that **directly intervenes in a model's decision process**, providing more targeted and model-relevant signals for mitigating spurious bias than sample-level approaches.

To this end, we focus on developing self-guided methods that directly analyze the internal prediction mechanism of a model to identify components of the model that are affected by spurious bias and then mitigate their influence to final predictions. We take a step towards this goal by

proposing a novel method termed **NeuronTune**, which systematically reduces spurious bias in deep neural networks. NeuronTune first probes in the latent embedding space of a trained model to identify dimensions (neurons) of sample embeddings *affected* by spurious bias, termed *biased dimensions*—those where spurious attributes predominantly contribute to prediction errors [198, 90]. Those dimensions can be identified when high activation magnitudes are strongly associated with incorrect predictions, indicating that features represented by those dimensions are not truly predictive of target classes. Importantly, rather than attempting to explicitly distinguish dimensions representing spurious and core attributes, an inherently challenging task given the complex entanglement of features in deep networks, NeuronTune instead identifies *biased dimensions* and suppresses the contributions of the these dimensions to final predictions. This intervention encourages the model to discover robust decision rules and mitigates spurious bias in the model.

Compared with the existing sample-level methods for spurious bias mitigation, NeuronTune provides direct intervention at the neuron level, allowing for more precise and targeted control over the mitigation of spurious bias during model tuning. Unlike approaches that rely on sample-level annotations such as group labels, NeuronTune enables the model to self-debias without external supervision. This makes it applicable in standard ERM training settings, where no additional annotations beyond class labels are available. As a result, NeuronTune serves as a practical and effective post hoc tool for mitigating spurious bias.

We theoretically demonstrate that neuron activations coupled with their final prediction outcomes provide self-identifying information on whether the neurons are affected by spurious bias. Our theoretical findings further suggest a practical metric for identifying biased dimensions and proves that NeuronTune can bring a model closer to the unbiased one. Experiments on vision and text datasets with different model architectures confirm the effectiveness of our method.

5.2.2 Preliminaries

We consider a standard classification problem. The training set $\mathcal{D}_{\text{train}} = \{(\mathbf{x}, y)|\mathbf{x} \in \mathcal{X}, y \in \mathcal{Y}\}$ typically contains data groups $\mathcal{D}_g^{\text{tr}}$ with $\mathcal{D}_{\text{train}} = \cup_{g\in\mathcal{G}}\mathcal{D}_g^{\text{tr}}$, where $\mathbf{x}$ denotes a sample in the input space $\mathcal{X}$, y is the corresponding label in the finite label space $\mathcal{Y}$, $g := (y, a)$ denotes the group label defined by the combination of a class label y and a spurious attribute $a \in \mathcal{A}$, where $\mathcal{A}$ denotes all spurious attributes in $\mathcal{D}_{\text{train}}$, and $\mathcal{G}$ denotes all possible group labels. Sample-label pairs in the group $\mathcal{D}_g^{\text{tr}}$ have the same class label y and the same spurious attribute a.

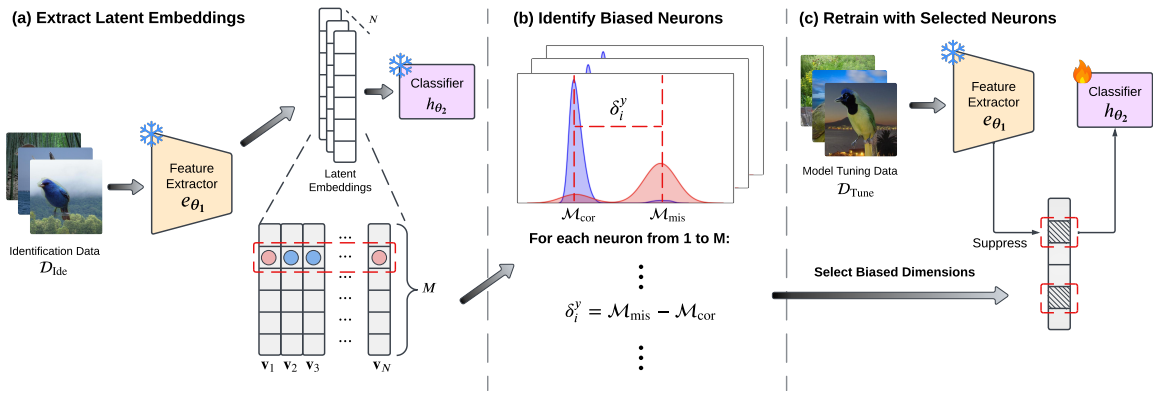


Figure 5.4: Practical implementation of NeuronTune. (a) Extract latent embeddings $\mathbf{v}_1, \ldots, \mathbf{v}_N$ and prediction outcomes (blue for correct and red for incorrect predictions) from an ERM-trained model using the identification data $\mathcal{D}_{\text{Ide}}$. (b) Identify biased neurons (dimensions) utilizing the statistics $\mathcal{M}_{\text{mis}}$ and $\mathcal{M}_{\text{cor}}$ derived from neuron activations for correct (blue) and incorrect (red) predictions from Equation (5.18). (c) Retrain the last prediction layer on $\mathcal{D}_{\text{Tune}}$ while keeping the feature extractor frozen and suppressing identified biased dimensions.

Our Scenario. We consider *unsupervised spurious bias mitigation*, where no group labels are available, resembling a standard ERM training. A commonly used performance metric is the worst-group accuracy (WGA), which is the accuracy on the worst performing data group in the test set $\mathcal{D}_{\text{test}}$, i.e., WGA $= \min_{g \in \mathcal{G}} \text{Acc}(f, \mathcal{D}_g^{\text{te}})$, where $\mathcal{D}_g^{\text{te}}$ denotes a group of data in $\mathcal{D}_{\text{test}}$ with $\mathcal{D}_{\text{test}} = \cup_{g \in \mathcal{G}} \mathcal{D}_g^{\text{te}}$, and f denotes a trained model. Typically, data in $\mathcal{D}_{\text{train}}$ is imbalanced across groups, and the model f tends to favor certain data groups, resulting in a low WGA. Improving WGA without knowing group labels during training is challenging.

We propose *NeuronTune*, a self-guided method for mitigating spurious bias without requiring group labels. NeuronTune identifies neurons (dimensions) affected by spurious bias in a model's latent space and tunes the model while suppressing the identified neurons. In Section 5.2.3, we present an analytical framework that outlines the design principles and theoretical properties of NeuronTune. Section 5.2.4 introduces a practical implementation for mitigating spurious bias in real-world settings.

5.2.3 NeuronTune: An Analytical Framework

At the core of NeuronTune is the identification of neurons that are affected by spurious bias. We establish an analytical framework to (1) elucidate the principle of neuron selection, (2) derive a selection metric that follows the principle of neuron selection, and (3) reveal the mechanism of NeuronTune in mitigating spurious bias.

Data and Prediction Models

Data Model. We design a data generation process that facilitates learning spurious correlations. Following the setting in [138, 199], we model a sample-label pair $(\mathbf{x}, y)$ in $\mathcal{D}_{\text{train}}$ as:

$$\mathbf{x} = \mathbf{x}_{\text{core}} \oplus \mathbf{x}_{\text{spu}} \in \mathbb{R}^{D\times 1},\ y = \boldsymbol{\beta}^T \mathbf{x}_{\text{core}} + \varepsilon_{\text{core}}, \tag{5.14}$$

where $\mathbf{x}_{\text{core}} \in \mathbb{R}^{D_1\times 1}$ is the core component, $\oplus$ denotes the vector concatenation operator, and the spurious component $\mathbf{x}_{\text{spu}} \in \mathbb{R}^{D_2\times 1}$ with $D_1 + D_2 = D$ is associated with the label y with the following relation:

$$\mathbf{x}_{\text{spu}} = (2a-1)\boldsymbol{\gamma} y + \boldsymbol{\varepsilon}_{\text{spu}}, a \sim \text{Bern}(p), \tag{5.15}$$

where $(2a-1) \in \{-1, +1\}$, $a \sim \text{Bern}(p)$ is a Bernoulli random variable, and p is close to 1, indicating that $\mathbf{x}_{\text{spu}}$ is mostly predictive of y but not always. In (5.14) and (5.15), $\boldsymbol{\beta} \in \mathbb{R}^{D_1\times 1}$ and $\boldsymbol{\gamma} \in \mathbb{R}^{D_2\times 1}$ are coefficients with unit L^2 norm, and $\varepsilon_{\text{core}} \in \mathbb{R}$ and $\boldsymbol{\varepsilon}_{\text{spu}} \in \mathbb{R}^{D_2\times 1}$ represent the variations in the core and spurious components, respectively. We set $\varepsilon_{\text{core}}$ and each element in $\boldsymbol{\varepsilon}_{\text{spu}}$ as zero-mean Gaussian random variables with the variances η^2_{core} and η^2_{spu}, respectively. We set $\eta^2_{\text{core}} \gg \eta^2_{\text{spu}}$ to facilitate learning spurious correlations [3].

Prediction Model. We adopt a linear regression model with two linear layers [199] defined as $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$, where $\mathbf{W} \in \mathbb{R}^{M\times D}$ denotes the embedding matrix simulating a feature extractor, $\mathbf{b} \in \mathbb{R}^{M\times 1}$ denotes the last layer, and M is the number of embedding dimensions. The model $f(\mathbf{x})$ can be further expressed as follows,

$$\begin{aligned} f(\mathbf{x}) &= \sum_{i=1}^{M} b_i(\mathbf{x}^T_{\text{core}}\mathbf{w}_{\text{core},i} + \mathbf{x}^T_{\text{spu}}\mathbf{w}_{\text{spu},i}) \\ &= \mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} + \mathbf{x}^T_{\text{spu}}\mathbf{u}_{\text{spu}}, \end{aligned} \tag{5.16}$$

where $\mathbf{w}_{\text{core},i} \in \mathbb{R}^{D_1\times 1}$, $\mathbf{w}_{\text{spu},i} \in \mathbb{R}^{D_2\times 1}$, $\mathbf{w}_i^T = [\mathbf{w}^T_{\text{core},i}, \mathbf{w}^T_{\text{spu},i}] \in \mathbb{R}^{1\times D}$ is the i-th row of $\mathbf{W}$, $\mathbf{u}_{\text{core}} = \sum_{i=1}^{M} b_i\mathbf{w}_{\text{core},i}$, and $\mathbf{u}_{\text{spu}} = \sum_{i=1}^{M} b_i\mathbf{w}_{\text{spu},i}$. The training objective is $\ell_{\text{tr}}(\mathbf{W}, \mathbf{b}) = \frac{1}{2}\mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{\text{train}}}\|f(\mathbf{x}) - y\|_2^2$.

Remark: To better understand our data and prediction models, consider that a in Eq. (5.15) controls subpopulations in data, e.g., when $a = 1$, it may represent a group of waterbirds on water, and when $a = 0$, it may represent a group of waterbirds on land. The probability p controls the severity of imbalance in subpopulations. When p is close to one, the data is severely imbalanced in

subpopulations. After training with ERM, the model minimizes the training loss, i.e., maximizes the average-case accuracy, but obtains a large nonzero weight on the spurious feature (Lemma 1 in Appendix) and is away from the optimal model (Corollary 1 in Appendix). For example, the model may focus on correctly classifying waterbirds on water, at the expense of its ability to recognize waterbirds on land.

Principle of Neuron Selection

NeuronTune aims to identify neurons that reflect spurious bias. Proposition 5.2 specifies the principle of NeuronTune in terms of what neurons are to be identified and suppressed during model tuning.

Proposition 5.2 (**Principle of NeuronTune**). Given the model $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$ trained with the data specified in (5.14) and (5.15), it captures spurious correlations when $\boldsymbol{\gamma}^T\mathbf{w}_{\text{spu},i} < 0, i \in \{1, \ldots, M\}$. The principle of NeuronTune is to suppress neurons containing negative $\gamma^T\mathbf{w}_{\text{spu},i}$.

If $\gamma^T\mathbf{w}_{\text{spu},i} \geq 0$, the model handles the spurious component correctly. Specifically, when $a = 1$, the spurious component $\mathbf{x}_{\text{spu}}$ positively correlates with the core component $\mathbf{x}_{\text{core}}$ and contributes to the output, whereas when $a = 0$, its correlation with $\mathbf{x}_{\text{core}}$ breaks with a negative one and has a negative contribution to the output. The relations reverse when $\gamma^T\mathbf{w}_{\text{spu},i} < 0$, i.e., the model utilizes $\mathbf{x}_{\text{spu}}$ even when the correlation breaks, demonstrating a strong reliance on the spurious component instead of the core component. The proof is in Appendix A.7.2.

Metric for Neuron Selection

Guided by the principle of NeuronTune in Proposition 5.2, the following theorem gives a practical metric to select neurons that are affected by spurious bias.

Theorem 5.1 (**Metric for Neuron Selection**). Given the model $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$, we cast it to a classification model by training it to regress $y \in \{-\mu, \mu\}$ ($\mu > 0$) on $\mathbf{x}$ based on the data model specified in (5.14) and (5.15), where $\mu = \mathbb{E}[\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}]$. The metric δ_i^y defined in the following can identify neurons affected by spurious bias when $\delta_i^y > 0$:

$$\delta_i^y = \text{Med}(\bar{\mathcal{V}}_i^y) - \text{Med}(\hat{\mathcal{V}}_i^y),$$

where $\bar{\mathcal{V}}_i^y$ and $\hat{\mathcal{V}}_i^y$ are the sets of activation values for misclassified and correctly predicted samples with the label y from the i-th neuron, respectively; an activation value is defined as $\mathbf{x}_{\text{core}}^T\mathbf{w}_{\text{core},i} + \mathbf{x}_{\text{spu}}^T\mathbf{w}_{\text{spu},i}$, and $\text{Med}(\cdot)$ returns the median of an input set of values.

We show in Appendix A.7.2 that the theorem establishes the approximation $\delta_i^y \approx -2\mu\gamma^T \mathbf{w}_{\text{spu},i}$, which confirms that neurons selected by the metric defined above follow the principle in Proposition 5.2. Adopting medians in the metric makes the metric robust to outlier values.

Let $\mathcal{M}_{\text{mis}} = \text{Med}(\bar{\mathcal{V}}_i^y)$ and $\mathcal{M}_{\text{cor}} = \text{Med}(\hat{\mathcal{V}}_i^y)$. Intuitively, a high $\mathcal{M}_{\text{mis}}$ indicates that high activations at the i-th dimension contribute to misclassification when predicting the class y. A low $\mathcal{M}_{\text{cor}}$ implies that the i-th dimension has little effect in correctly predicting the class y. Thus, a large difference between $\mathcal{M}_{\text{mis}}$ and $\mathcal{M}_{\text{cor}}$, i.e., a large δ_i^y, indicates that the i-th dimension represents features that are irrelevant to the class y. In other words, with a high likelihood, the dimension is affected by spurious bias. In contrast, a negative δ_i^y highlights the relevance of the i-th dimension for predictions as most correctly predicted samples have high activation values in this dimension, and most incorrectly predicted samples have low activation values.

Remark: Proposition 5.2 and Theorem 5.1 state that when a spurious correlation breaks, neurons that continue to positively contribute to mispredictions will be selected. For example, in the case of waterbird with water and land backgrounds, neurons that cause misclassification on images of waterbird appearing on land will be identified.

Mechanism of NeuronTune

NeuronTune mitigates spurious bias by retraining the last layer while suppressing (zeroing out) the identified neurons. The following theorem shows that this improves model robustness and explains how it achieves this.

Theorem 5.2 (**NeuronTune Mitigates Spurious Bias**)**.** Consider the model $f^*(\mathbf{x}) = \mathbf{x}^T\mathbf{u}^*$ trained on the biased training data with $p \gg 0.5$, where $\mathbf{u}^{*T} = [\mathbf{u}_{\text{core}}^{*T}, \mathbf{u}_{\text{spu}}^{*T}]$. Under the mild assumption that $\boldsymbol{\beta}^T\mathbf{w}_{\text{core},i} \approx \boldsymbol{\gamma}^T\mathbf{w}_{\text{spu},i}, \forall i = 1, \ldots, M$, then applying NeuronTune to $f^*(\mathbf{x})$ produces a model that is closer to the unbiased one.

The assumption $\boldsymbol{\beta}^T\mathbf{w}_{\text{core},i} \approx \boldsymbol{\gamma}^T\mathbf{w}_{\text{spu},i}, \forall i = 1, \ldots, M$ generally holds for a biased model, as the model has learned to associate spurious attributes with core attributes. The proof is in Appendix A.7.2. Denote the NeuronTune solution by $\mathbf{u}_{\text{core}}^\dagger$ and $\mathbf{u}_{\text{spu}}^\dagger$. Our finding reveals that retraining the last layer does not alter the weight on the spurious component, i.e., $\mathbf{u}_{\text{spu}}^\dagger = \mathbf{u}_{\text{spu}}^*$, which is the optimal solution achievable by last-layer retraining methods (see Lemma 3 in Appendix A.7.2). However, it does adjust $\mathbf{u}_{\text{core}}^\dagger$ to be closer to the optimal weight on the core component, $\boldsymbol{\beta}$. Overall, NeuronTune brings the model parameters closer to the optimal, unbiased solution compared to the parameters of the original biased model. Therefore, NeuronTune is guaranteed to outperform

the ERM-trained model. Further discussion on the connection to last-layer retraining methods is provided in Appendix A.7.3.

Remark: Our findings suggest that our approach makes a slight trade-off in average-case accuracy to achieve improved worst-group accuracy. For example, our method may slightly reduce the model's ability to classify waterbird on water due to a *relative* decrease in reliance on the water feature, while significantly enhancing its ability to classify waterbird on land.

5.2.4 NeuronTune: Practical Implementation

For real-world spurious bias mitigation, we consider a well-trained ERM model $f_{\boldsymbol{\theta}}$ where $\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}'} \mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{\text{train}}} \ell(f_{\boldsymbol{\theta}'}(\mathbf{x}), y)$, and ℓ denotes the cross-entropy loss function. The model $f_{\boldsymbol{\theta}} = e_{\boldsymbol{\theta}_1} \circ h_{\boldsymbol{\theta}_2}$ consists of a feature extractor $e_{\boldsymbol{\theta}_1} : \mathcal{X} \to \mathbb{R}^M$ followed by a linear classifier $h_{\boldsymbol{\theta}_2} : \mathbb{R}^M \to \mathbb{R}^{|\mathcal{Y}|}$, where M is the number of dimensions of latent embeddings obtained from $e_{\boldsymbol{\theta}_1}$, $\circ$ denotes the function composition operator, and $\boldsymbol{\theta} = \boldsymbol{\theta}_1 \cup \boldsymbol{\theta}_2$.

NeuronTune aligns best with our theoretical analysis when implemented as a last-layer retraining method where the feature extractor $e_{\boldsymbol{\theta}_1}$ is fixed and the last layer is linear and tunable. Figure 5.4 gives an overview of NeuronTune which mainly includes identifying affected neurons and model tuning with identified neurons.

Identifying Affected Neurons

As shown in Figure 5.4(a), we use a set of identification data $\mathcal{D}_{\text{Ide}}$, which typically contains a set of diverse features not seen by the model, to identify dimensions (neurons) affected by spurious bias in the model's latent space. We first extract latent embeddings and prediction outcomes for samples of class y in $\mathcal{D}_{\text{Ide}}$, i.e.,

$$\mathcal{V}^y = \{(\mathbf{v}, o) | \mathbf{v} = e_{\boldsymbol{\theta}_1}(\mathbf{x}), \forall (\mathbf{x}, y) \in \mathcal{D}_{\text{Ide}}\}, \tag{5.17}$$

where $o = \mathbb{1}\{\arg\max f_{\boldsymbol{\theta}}(\mathbf{x}) == y\}$, $\mathbf{v} \in \mathbb{R}^M$ is an M-dimensional latent embedding of $\mathbf{x}$, and o is the corresponding prediction outcome with $\mathbb{1}$ being an indicator function.

Identification Criterion. As shown in Figure 5.4(b), for each embedding dimension i, we separate $\mathcal{V}^y$ into two sets $\hat{\mathcal{V}}_i^y$ and $\bar{\mathcal{V}}_i^y$, representing values at the i-th embedding dimension from $\mathcal{V}^y$, contributing respectively to correct and incorrect predictions, i.e.,$\hat{\mathcal{V}}_i^y = \{\mathbf{v}[i] | (\mathbf{v}, 1) \in \mathcal{V}^y\}$, and $\bar{\mathcal{V}}_i^y = \{\mathbf{v}[i] | (\mathbf{v}, 0) \in \mathcal{V}^y\}, \forall i = 1, \ldots, M,\ y \in \mathcal{Y}$, where $\mathbf{v}[i]$ denotes the i-th dimension of $\mathbf{v}$. We propose a *spuriousness score* δ_i^y to measure the spuriousness of the i-th dimension when predicting

the class y. Following the insight from Theorem 5.1, we define δ_i^y as follows:

$$\delta_i^y = \mathcal{M}_{\text{mis}} - \mathcal{M}_{\text{cor}}, \tag{5.18}$$

where $\mathcal{M}_{\text{mis}} = \text{Med}(\bar{\mathcal{V}}_i^y)$ and $\mathcal{M}_{\text{cor}} = \text{Med}(\hat{\mathcal{V}}_i^y)$.

Theorem 5.1 assumes that each dimension of input embeddings consists of a linear combination of spurious and core components. While it generally holds that each dimension represents a mixture of spurious and core components, in real-world scenarios, the combination is typically nonlinear. To account for this, we introduce λ as a threshold and identify dimensions using the following criterion:

$$\mathcal{S} = \{i | \delta_i^y > \lambda, \forall i = 1, \ldots, M, y \in \mathcal{Y}\}. \tag{5.19}$$

We set λ to 0 by default, as it works well in practice.

In the following, we refer to a dimension as a **biased dimension** when $\delta_i^y > \lambda$ and **unbiased dimension** otherwise. A biased (unbiased) dimension does not imply that the dimension exclusively represents spurious (core) attributes. In practice, an unbiased dimension exhibits high activation values for target classes, whereas a biased dimension shows high activation values for undesired classes. Visualizations of several identified biased and unbiased dimensions on real-world datasets are provided in Appendix A.7.9.

We include the dimensions identified for all the classes into the set $\mathcal{S}$ since an identified biased dimension for one class cannot serve as a core contributor to predicting some other class in a well-defined classification task. For example, consider that the dimension representing "blue color" is biased for the "rectangle" class while being unbiased for the "blue color" class. This happens when we have a blue rectangle as the input, which makes the classification ambiguous.

Additionally, while our approach may resemble traditional variable selection such as ℓ_1 regularization, it goes further by specifically addressing spurious bias—a factor often ignored in traditional methods. Notably, our method operates in an unsupervised setting without requiring group labels. Further details on its advantages are provided in Appendix A.7.6.

Model Tuning with Identified Neurons

As illustrated in Figure 3.6(c), we tune the last prediction layer while suppressing the signals from the identified biased dimensions. In this way, we explicitly intervene the internal decision process of the model to discover robust decision rules beyond using spurious correlations.

Learning Objective. Concretely, given a model tuning dataset $\mathcal{D}_{\text{Tune}}$, we optimize the following objective,

$$\boldsymbol{\theta}_2^* = \arg\min_{\boldsymbol{\theta}_2} \mathop{\mathbb{E}}_{\mathcal{B}\sim\mathcal{D}_{\text{Tune}}} \mathop{\mathbb{E}}_{(x,y)\in\mathcal{B}} \ell(h_{\boldsymbol{\theta}_2}(\tilde{\mathbf{v}}), y), \tag{5.20}$$

where $\mathcal{B}$ contains *class-balanced* sample-label pairs from $\mathcal{D}_{\text{Tune}}$, addressing that the classifier may favor certain classes during model tuning, and $\tilde{\mathbf{v}}$ is the latent embedding after zeroing-out activations on the biased dimensions in $\mathcal{S}$. Unless otherwise stated, we use $\mathcal{D}_{\text{train}}$ as $\mathcal{D}_{\text{Tune}}$.

Model Selection. Without group labels, it is challenging to select robust models [31, 29]. We address this by designing a novel model selection metric, termed *spuriousness fitness score (SFit)*, which is the sum of magnitudes of spuriousness scores across dimensions and classes, i.e., SFit = $\sum_{m=1}^{M}\sum_{y\in\mathcal{Y}} \text{Abs}(\delta_m^y)$, where $\text{Abs}(\cdot)$ returns the absolute value of a given input. The score holistically summarizes whether biased and unbiased dimensions in the model are distinguishable. A low SFit indicates that the model tends to memorize samples. Empirically, we find that a high SFit effectively selects a robust model.

NeuronTune is highly efficient as it only requires tuning the last layer of the model. We use (5.19) and (5.20) to iteratively perform the biased dimension detection and model tuning while using SFit for model selection.

5.2.5 Experiments

Datasets

We tested NeuronTune on four image datasets and two text datasets, each with different types of spurious attributes. (1) **Waterbirds** [3] is an image dataset for recognizing waterbird and landbird. It is generated synthetically by combining images of the two bird types from the CUB dataset [163] and the backgrounds, water and land, from the Places dataset [174]. (2) **CelebA** [149] is a large-scale image dataset of celebrity faces. The task is to identify hair color, non-blond or blond, with male or female as the spurious attribute. (3) **ImageNet-9** [49] is a subset of ImageNet [175] containing nine super-classes. It comprises images with different background and foreground signals and can be used to assess how much models rely on image backgrounds. (4) **ImageNet-A** [176] is a dataset of real-world images, adversarially curated to test the limits of classifiers such as ResNet-50. We used this dataset to test the robustness of a classifier after training it on ImageNet-9. (5) **MultiNLI** [194] is a text classification dataset with three classes: neutral, contradiction, and entailment, representing the natural language inference relationship between a premise and a hypothesis. The spurious attribute

is the presence of negation. (6) **CivilComments** [195] is a binary text classification dataset aimed at predicting whether an internet comment contains toxic language. The spurious attribute involves references to eight demographic identities. The dataset uses standard splits provided by the WILDS benchmark [28].

Experimental Setup

Training Details. We first trained ERM models on each of the datasets. We used ResNet-50 and ResNet-18 [200] models pretrained on ImageNet for experiments on the Waterbirds and CelebA datasets, and on the ImageNet-9 and ImageNet-A datasets, respectively. For text datasets, we used the BERT model [196] pretrained on Book Corpus and English Wikipedia data. We followed the settings in Izmailov et al. [170] for ERM training, with the best models selected based on the average validation accuracy. For our NeuronTune training, unless otherwise stated, we used the validation data as $\mathcal{D}_{\text{Ide}}$ and the training data as $\mathcal{D}_{\text{Tune}}$. We took the absolute values of neuron activations before the identification process, ensuring that high activation magnitudes reflect strong contributions to predictions. We ran the training under five different random seeds and reported average accuracies along with standard deviations. We provide full training details in Appendix A.7.8. Code is available at `https://github.com/gtzheng/NeuronTune`.

Evaluation Metrics. To evaluate the robustness to spurious bias, we adopt the widely accepted robustness metric, *worst-group accuracy (WGA)*, that gives the lower-bound performance of a classifier on the test set with various dataset biases. We also focus on the *accuracy gap* between the standard average accuracy and the worst-group accuracy as a measure of a classifier's reliance on spurious correlations. A high worst-group accuracy and a low accuracy gap indicate that the classifier is robust to spurious correlations and can fairly predict samples from different groups.

Synthetic Experiment

We considered an input $\mathbf{v} = [v^c, v^s, v^\epsilon] \in \mathbb{R}^3$ that has three dimensions: a core dimension with the core component $v^c \in \mathbb{R}$, a spurious dimension with the spurious component $v^s \in \mathbb{R}$, and a noise dimension with the noise component v^ϵ. We generated training and test sets with sample-label pairs $(\mathbf{v}, y)$, where $y \in \{-1, +1\}$. The core component in $\mathbf{v}$ is a noisy version of the label y in both sets. The spurious component in the training set is a noisy version of the spurious attribute $a = 0$ in 95% (5% for $a = 1$) of samples with $y = -1$ and in 5% (95% for $a = 1$) of samples with $y = +1$. The noise component is an independent zero-mean Gaussian variable. In the test set, for each label, we reduced the 95% group to 10%, effectively reversing the majority and minority group roles. We adopted a

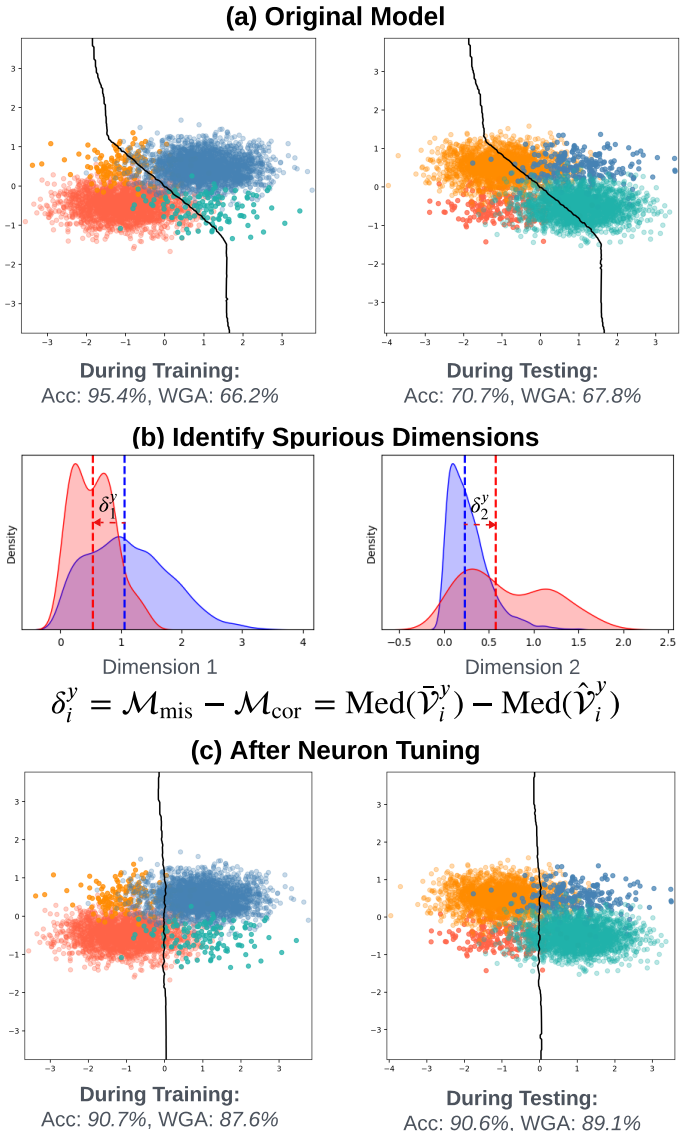


Figure 5.5: Synthetic experiment. (a) Training and test data distributions along with the decision boundaries of the trained model. (b) Value distributions of the correctly (blue) and incorrectly (red) predicted samples at the first (left) and second (right) dimensions of input embeddings, with the second dimension identified as a biased dimension. (c) NeuronTune improves WGA. Data groups $(y = +1, a = 1)$: red dots; $(y = +1, a = 0)$: orange dots; $(y = -1, a = 0)$: blue dots; $(y = -1, a = 1)$: green dots.

logistic regression model $\phi_{\tilde{\mathbf{w}}}(\mathbf{v}) = 1/(1 + \exp\{-(\mathbf{w}^T\mathbf{v} + b)\})$ with $\tilde{\mathbf{w}} = [\mathbf{w}, b]$. The model predicts $+1$ when $\phi_{\tilde{\mathbf{w}}}(\mathbf{v}) > 0.5$ and -1 otherwise. We trained $\phi_{\tilde{\mathbf{w}}}$ on the generated training data and tested it on the corresponding test data. Details of the data generation are provided in Appendix A.7.1.

Figure 5.5 illustrates spurious bias and how NeuronTune mitigates it. First, we observe that the decision boundary of the trained model tends to separate the majority groups of training samples. This leads to a high average accuracy but a small WGA on the training set (Figure 5.5(a), left)

Algorithm	Group annotations		Waterbirds			CelebA		
	Train	Val	WGA (↑)	Acc. (↑)	Acc. Gap (↓)	WGA (↑)	Acc. (↑)	Acc. Gap (↓)
ERM [193]	-	-	72.6	97.3	24.7	47.2	95.6	48.4
JTT [31]	No	Yes	86.7	93.3	6.6	81.1	88.0	6.9
SELF† [83]	No	Yes	**93.0**$_{\pm 0.3}$	94.0$_{\pm 1.7}$	**1.0**	83.9$_{\pm 0.9}$	91.7$_{\pm 0.4}$	7.8
CNC [35]	No	Yes	88.5$_{\pm 0.3}$	90.9$_{\pm 0.1}$	2.4	**88.8**$_{\pm 0.9}$	89.9$_{\pm 0.5}$	**1.1**
BAM [201]	No	Yes	89.2$_{\pm 0.3}$	91.4$_{\pm 0.4}$	2.2	83.5$_{\pm 0.9}$	88.0$_{\pm 0.4}$	4.5
AFR [32]	No	Yes	90.4$_{\pm 1.1}$	94.2$_{\pm 1.2}$	3.8	82.0$_{\pm 0.5}$	91.3$_{\pm 0.3}$	9.3
DFR† [4]	No	Yes	92.4$_{\pm 0.9}$	94.9$_{\pm 0.3}$	2.5	87.0$_{\pm 1.1}$	92.6$_{\pm 0.5}$	5.6
BPA [202]	No	No	71.4	-	-	82.5	-	-
GEORGE [203]	No	No	76.2	95.7	19.5	52.4	94.8	42.4
BAM [201]	No	No	89.1$_{\pm 0.2}$	91.4$_{\pm 0.3}$	2.3	80.1$_{\pm 3.3}$	88.4$_{\pm 2.3}$	8.3
NeuronTune	No	No	92.2$_{\pm 0.3}$	94.4$_{\pm 0.2}$	2.2	83.1$_{\pm 1.1}$	92.0$_{\pm 0.5}$	8.9
NeuronTune†	No	No	**92.5**$_{\pm 0.9}$	94.5$_{\pm 0.3}$	**2.0**	**87.3**$_{\pm 0.4}$	90.3$_{\pm 0.5}$	**3.0**

Table 5.4: Comparison of worst-group accuracy (%), average accuracy (%), and accuracy gap (%) on the image datasets. † denotes using a fraction of validation data for model tuning. The best result in each group of methods is in **boldface**.

and poor performance on the test set (Figure 5.5(a), right). Then, Figure 5.5(b) demonstrates the value distributions of the first (core) and second (spurious) dimensions of the input samples with $y = -1$. NeuronTune identified the second dimension as a biased dimension, which indeed represents spurious attributes. Next, Figure 5.5(c) shows that NeuronTune significantly improves WGA on both the training and test sets by suppressing the contributions from biased dimensions. Finally, independent of how NeuronTune works, there exists a tradeoff between average accuracy and WGA due to complexity of input samples, as demonstrated in the left parts of Figures. 5.5(a) and 5.5(c).

Comparison with Existing Approaches

We evaluated NeuronTune on both image and text datasets to showcase its effectiveness and versatility in handling different data modalities and model architectures. Our primary comparisons were with methods specifically designed for unsupervised spurious bias mitigation, where no group labels are available for bias mitigation. To provide additional context, we also included methods for semi-supervised spurious bias mitigation, which leverage group labels in the validation set to select robust models.

Results in the lower parts of Tables 5.4 and 5.5 were obtained in the unsupervised spurious bias mitigation setting. In this setting, our method achieves the highest worst-group accuracies and smallest accuracy gaps across the datasets, highlighting its effectiveness in enhancing models' robustness to spurious bias and balancing performance across different data groups. Results in the upper parts of Tables 5.4 and 5.5 were from methods in the semi-supervised spurious bias mitigation

Algorithm	Group annotations		MultiNLI			CivilComments		
	Train	Val	WGA (↑)	Acc. (↑)	Acc. Gap (↓)	WGA (↑)	Acc. (↑)	Acc. Gap (↓)
ERM [193]	-	-	67.9	**82.4**	14.5	57.4	**92.6**	35.2
JTT [31]	No	Yes	72.6	78.6	**6.0**	69.3	91.1	21.8
SELF† [83]	No	Yes	$70.7_{\pm 2.5}$	$81.2_{\pm 0.7}$	10.5	$79.1_{\pm 2.1}$	$87.7_{\pm 0.6}$	8.6
CNC [35]	No	Yes	-	-	-	$68.9_{\pm 2.1}$	$81.7_{\pm 0.5}$	12.8
BAM [201]	No	Yes	$71.2_{\pm 1.6}$	$79.6_{\pm 1.1}$	8.4	$79.3_{\pm 2.7}$	$88.3_{\pm 0.8}$	9.0
AFR [32]	No	Yes	$\mathbf{73.4}_{\pm 0.6}$	$81.4_{\pm 0.2}$	8.0	$68.7_{\pm 0.6}$	$89.8_{\pm 0.6}$	21.1
DFR† [4]	No	Yes	$70.8_{\pm 0.8}$	$81.7_{\pm 0.2}$	10.9	$\mathbf{81.8}_{\pm 1.6}$	$87.5_{\pm 0.2}$	**5.7**
BAM [201]	No	No	$70.8_{\pm 1.5}$	$80.3_{\pm 1.0}$	9.5	$79.3_{\pm 2.7}$	$88.3_{\pm 0.8}$	9.0
NeuronTune	No	No	$72.1_{\pm 0.1}$	$81.1_{\pm 0.6}$	9.0	$82.4_{\pm 0.2}$	$89.2_{\pm 0.1}$	6.8
NeuronTune†	No	No	$\mathbf{72.5}_{\pm 0.3}$	$80.3_{\pm 0.6}$	**7.8**	$\mathbf{82.7}_{\pm 0.4}$	$89.4_{\pm 0.2}$	**6.7**

Table 5.5: Comparison of worst-group accuracy (%), average accuracy (%), and accuracy gap (%) on the text datasets. † denotes using a fraction of validation data for model tuning. The best result in each group of methods is in **boldface**.

Method	ImageNet-9	ImageNet-A	Acc. Gap (↓)
ERM [193]	$90.8_{\pm 0.6}$	$24.9_{\pm 1.1}$	65.9
StylisedIN [179]	$88.4_{\pm 0.5}$	$24.6_{\pm 1.4}$	63.8
RUBi [180]	$90.5_{\pm 0.3}$	$27.7_{\pm 2.1}$	62.8
ReBias [181]	$91.9_{\pm 1.7}$	$29.6_{\pm 1.6}$	62.3
LfF [33]	86.0	24.6	61.4
CaaM [182]	**95.7**	32.8	62.9
SSL+ERM [183]	$94.2_{\pm 0.1}$	$34.2_{\pm 0.5}$	60.0
LWBC [183]	$94.0_{\pm 0.2}$	$36.0_{\pm 0.5}$	58.0
NeuronTune	$93.7_{\pm 0.1}$	$\mathbf{37.3}_{\pm 0.5}$	**56.4**

Table 5.6: Average accuracy (%) and accuracy gap (%) comparison on the ImageNet-9 and ImageNet-A datasets. ResNet-18 was used as the backbone. The best results are in **boldface**.

setting. Methods in this setting benefit from group labels for selecting robust models. Despite this advantage, NeuronTune demonstrates strong self-debiasing capabilities, competing favorably with methods such as AFR and DFR that rely on group labels. When a half of the validation set was used in training, NeuronTune achieved better WGAs and accuracy gaps on three out of four datasets than DFR and SELF that exploited the same set of data for training.

Notably, compared with sample-level last-layer retraining methods, such as AFR, NeuronTune manipulates the neurons within a model, providing more targeted control on how spurious bias is mitigated. Hence, NeuronTune in theory can achieve better robustness to spurious bias (Appendix A.7.3). In general, NeuronTune compares favorably with AFR in terms of WGA and accuracy gap, with larger gains achieved when AFR models were selected without group labels (Appendix A.7.4).

We further used the ImageNet-9 [183, 181] and ImageNet-A [176] datasets to evaluate NeuronTune's robustness to distribution shifts, which are challenging to depict in group labels. We first trained an ERM model from scratch using ImageNet-9 and then fine-tuned its last layer with Neu-

$\mathcal{D}_{\text{Ide}}$	$\mathcal{D}_{\text{Tune}}$	NT	Waterbirds	CelebA	MultiNLI	CivilComments
$\mathcal{D}_{\text{train}}$	$\mathcal{D}_{\text{train}}$	Yes	$78.0_{\pm 2.3}$	$58.5_{\pm 1.2}$	$42.0_{\pm 10.5}$	$80.0_{\pm 10.5}$
$\mathcal{D}_{\text{val}}$	$\mathcal{D}_{\text{train}}$	Yes	$92.2_{\pm 0.3}$	$83.1_{\pm 1.1}$	$72.1_{\pm 0.1}$	$82.4_{\pm 0.2}$
$\mathcal{D}_{\text{val}}$	$\mathcal{D}_{\text{train}}$	No	$82.7_{\pm 0.4}$	$53.9_{\pm 0.0}$	$63.4_{\pm 0.7}$	$81.5_{\pm 0.5}$
$\mathcal{D}_{\text{val}}/2$	$\mathcal{D}_{\text{val}}/2$	Yes	$\mathbf{92.5}_{\pm 0.9}$	$\mathbf{87.3}_{\pm 0.4}$	$\mathbf{72.5}_{\pm 0.3}$	$\mathbf{82.7}_{\pm 0.4}$

Table 5.7: Comparison of worst-group accuracy (%) between different choices of $\mathcal{D}_{\text{Ide}}$ and $\mathcal{D}_{\text{Tune}}$ as well as neuron-based tuning (NT) on the four datasets. The best results are in **boldface**.

ronTune. In Table A.6.7, NeuronTune achieves the best accuracy on the challenging ImageNet-A dataset, which is known for its natural adversarial examples. While this improvement comes with a slight trade-off in in-distribution accuracy on ImageNet-9, NeuronTune maintains the smallest accuracy gap between the two datasets, making it a robust method for out-of-distribution generalization.

Finally, in Tables 5.4, 5.5, and A.6.7, we observe a common trade-off between average accuracy and WGA that exists across many spurious bias mitigation methods. For NeuronTune, this trade-off primarily occurs when samples sharing the same spurious attribute but belonging to different classes are difficult to separate in the latent space, as illustrated in Figure 5.5. While improving sample embeddings could help alleviate this issue, it often demands substantial computational resources. In contrast, NeuronTune, as a post hoc method, efficiently mitigates spurious bias by tuning only the last layer with low computational complexity (Appendix A.7.5) while still achieving a favorable balance between WGA and overall performance.

Ablation Studies

In Table 5.7, we compare NeuronTune's performance between different choices of the identification dataset $\mathcal{D}_{\text{Ide}}$ and the model tuning dataset $\mathcal{D}_{\text{Tune}}$. Additionally, we demonstrate the effectiveness of neuron-based tuning on the identified biased dimensions (denoted as NT).

When using $\mathcal{D}_{\text{Ide}} = \mathcal{D}_{\text{train}}$, we observe a relatively low performance across datasets. After switching to a held-out validation data $\mathcal{D}_{\text{val}}$, we observe significant performance improvements. This highlights the advantage of using a new and independent dataset to identify biased dimensions, as models may have already memorized patterns in $\mathcal{D}_{\text{train}}$. By default, NeuronTune adopts $\mathcal{D}_{\text{val}}$ as $\mathcal{D}_{\text{Ide}}$. It is important to note that using $\mathcal{D}_{\text{val}}$ to identify biased dimensions is analogous to using it for model selection. Hence, $\mathcal{D}_{\text{val}}$ is not directly used for updating model weights.

Next, we disabled NT during model tuning (NT=No), which effectively reduces NeuronTune to class-balanced model tuning. We observe consistent performance degradation across the four datasets, which validates the effectiveness of NT across datasets.

Masking value	0	0.2	0.4	0.6	0.8	1.0
WGA (↑)	$87.3_{\pm 0.4}$	$71.5_{\pm 1.5}$	$72.2_{\pm 1.2}$	$72.9_{\pm 1.5}$	$73.1_{\pm 1.5}$	$73.0_{\pm 1.2}$
Acc. (↑)	$90.3_{\pm 0.5}$	$93.8_{\pm 0.2}$	$93.8_{\pm 0.3}$	$93.8_{\pm 0.2}$	$93.8_{\pm 0.2}$	$93.9_{\pm 0.2}$

Table 5.8: Analysis of the impact of partial suppression (masking value > 0) and full suppression (masking value $= 0$) on the performance of NeuronTune†, evaluated on the CelebA dataset.

Moreover, inspired by the success of DFR [4], which uses a half of the validation data for model tuning, we divided $\mathcal{D}_{\text{val}}$ into two equal halves: one half (denoted as $\mathcal{D}_{\text{val}}/2$) was used as $\mathcal{D}_{\text{Ide}}$, while the other half served as $\mathcal{D}_{\text{Tune}}$. Unlike DFR, our method does not rely on group labels in the validation data. This strategy leads to further performance improvements on datasets such as CelebA and MultiNLI, demonstrating the advantage of using separate and independent datasets for bias identification and model tuning. Identifying the optimal choice for $\mathcal{D}_{\text{Ide}}$ and $\mathcal{D}_{\text{Tune}}$ remains an avenue for future research.

Finally, we analyze different strategies for handling the identified biased dimensions, as shown in Table 5.8. Our default approach, described in Section 5.2.4, fully suppresses the activations on the biased dimensions by multiplying the activations with a masking value of zero. To explore the effect of partial suppression, we varied the masking value from 0.2 to 1.0, where 1.0 corresponds to no suppression. As shown in Table 5.8, on the CelebA dataset, only the full suppression strategy (masking value $= 0$) led to an improvement in WGA. This highlights that while partial suppression may reduce the loss in average accuracy, its impact on spurious bias is similar to no suppression at all. With nonzero masking values, models can still adjust their weights using biased activations, resulting in persistent spurious bias.

5.2.6 Conclusion

We proposed a self-guided spurious bias mitigation method that directly intervenes the prediction mechanisms within a model without using group labels. Our method exploits distinct patterns of neuron activations in a model's latent space to identify biased dimensions and suppresses signals from these dimensions while tuning the remaining model. We theoretically validated our neuron identification method and proved that our method can bring a model closer to an unbiased one than its ERM counterpart. Experiments validated our theoretical findings and showed that our method is a lightweight post hoc bias mitigation method that can work across different data modalities and model architectures. Future work may explore different choices of identification and model tuning data to enhance spurious bias mitigation.

5.3 Self-Adaptive Prompt Exploration for Zero-Shot Spurious Bias Mitigation in Vision-Language Models

5.3.1 Introduction

Zero-shot models hold promising potential for making predictions on any set of classes without requiring data collection or training. Pre-trained vision-language models (VLMs) [41, 204, 205, 206, 207] such as contrastive language-image pre-training (CLIP) models [41] have demonstrated a strong zero-shot prediction capability across diverse downstream tasks. They typically consist of a pre-trained image and a text encoder from which vision and text representations are aligned in a shared joint embedding space. Thus, the zero-shot prediction for an image can be simply achieved by finding the description from a set of candidate descriptions whose representation is most similar to the image representation.

However, recent studies [208, 95, 209] have found that pre-trained CLIP models often develop an undesirable tendency to use spurious correlations between spurious, non-essential features and targets across modalities for making predictions in a certain modality. For instance, as shown in Figure 5.6, there is a misalignment between features from vision and language modalities, i.e., the class label "landbird" is misaligned with a land background due to their strong spurious correlation in the pre-training data. Then, a CLIP model may use the image background to infer the object's label ("waterbird") as "landbird", resulting in misclassification. Such a biased prediction behavior, termed as *multimodal spurious bias*, severely limits the zero-shot generalization capability of pre-trained CLIP models on out-of-distribution data where cross-modal spurious correlations captured during pre-training no longer hold in downstream tasks, e.g., the correlation between "landbird" and a land background no longer holds in Figure 5.6.

Mitigating multimodal spurious bias is essential for ensuring robust generalization across various downstream tasks. Existing methods differ significantly on tackling this problem. Some methods [162, 208, 210, 209] adopt fine-tuning approaches, which focus on biases specific to downstream tasks and require a set of fine-tuning data. Although these methods achieve impressive improvements in robustness to multimodal spurious bias over the vanilla zero-shot approach on multiple downstream tasks, they require labeled data and do not tackle the problem in the zero-shot setting. A recent method [95] tackles multimodal spurious bias within the language modality and does not require training data. But it typically requires specifying potential spurious attributes acquired by prompting a large language model (LLM) regarding a downstream task.

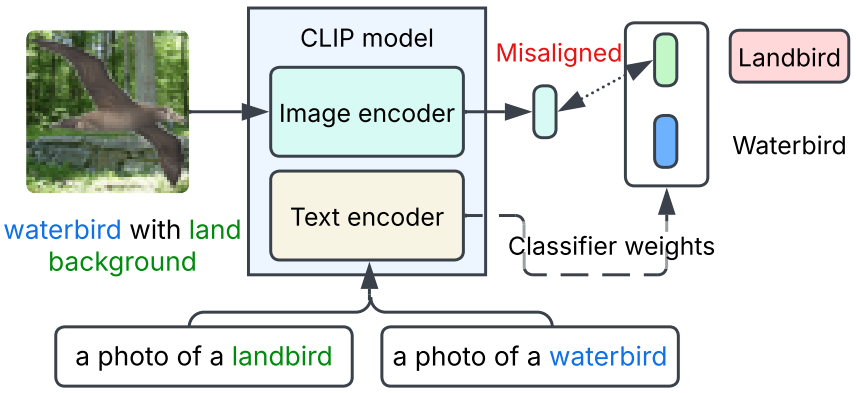


Figure 5.6: Illustration of multimodal spurious bias in a CLIP model. The text representation of "a photo of a landbird" is misaligned with the image representation because of the spurious land background feature, resulting in misclassification.

In this paper, we propose a zero-shot, self-adaptive framework for mitigating multimodal spurious bias, requiring no training data or prior knowledge of the bias. We first formally define multimodal spurious bias, establishing a theoretical foundation for analyzing its impact on zero-shot classification. Our theoretical insights reveal a connection between the strength of multimodal spurious bias and the similarity between input image representations and text representations of class labels, which can be influenced by different prompt templates.

Building on this insight, our approach leverages the prompt templates recommended for pre-trained CLIP models [41], such as "a drawing of a [CLASS]" and "a photo of the [CLASS]", where [CLASS] is a placeholder for class labels. These prompts serve as candidate descriptions for images by substituting actual class labels. A key observation motivating our approach is that different prompts exhibit varying degrees of robustness to multimodal spurious bias, suggesting that prompt selection significantly impacts how multimodal spurious bias affects zero-shot classification performance.

Our proposed framework, termed **S**elf-**A**daptive prompt **E**xploration (SAVE), adaptively selects prompts according to an input image and constructs robust zero-shot classifiers based on selected prompts. SAVE is fine-tuning free and does not rely on prior knowledge about spurious biases such as annotations or spurious attributes obtained through external means like LLMs. Extensive experiments and in-depth analyses on four benchmarks across six models validate the effectiveness of our method in mitigating multimodal spurious bias and improving model generalization.

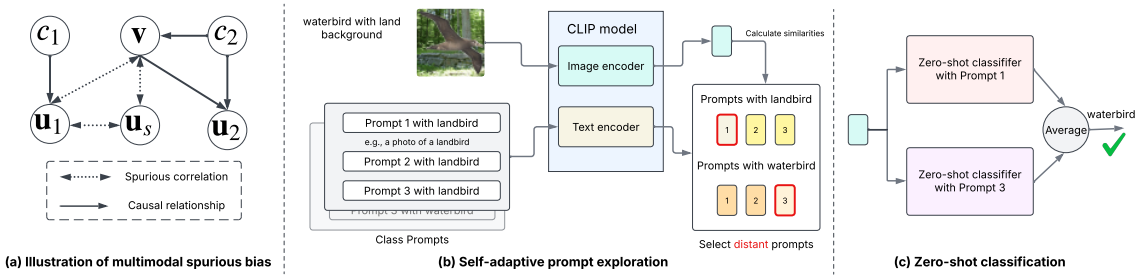


Figure 5.7: Method overview. (a) Illustration of multimodal spurious bias, where c_2 denotes a class label, $\mathbf{v}$ denotes an image representation, $\mathbf{u}_s$ denotes a textual spurious feature, $\mathbf{u}_1$ and $\mathbf{u}_2$ denote text representations for the class c_1 and c_2 respectively. (b) Self-adaptive prompt exploration finds a prompt for each class from a set of candidate prompts that minimizes multimodal spurious bias. (c) Zero-shot classification using an ensemble of zero-shot classifiers constructed with prompts selected from the previous step.

5.3.2 Methodology

We first theoretically analyze the multimodal spurious bias in VLMs. Based on the insights gained in the analysis, we propose a self-adaptive prompt exploration method to mitigate multimodal spurious bias.

Preliminary

A CLIP [41] model is trained to align the representation of an image x from its vision encoder ϕ and the representation of a text description t from its text encoder ψ in a joint embedding space when the text description t matches with the image x. Specifically, let $\mathbf{v} = \phi(x) \in \mathbb{R}^D$ denote the vision representation for the image x and $\mathbf{u} = \psi(t) \in \mathbb{R}^D$ be the text representation for the text description t, where D is the number of embedding dimensions. Then, the CLIP training objective [41] essentially aims to maximize the probability of $\mathbf{v}$ given $\mathbf{u}$ and the probability of $\mathbf{u}$ given $\mathbf{v}$ over all training image-text pairs, i.e.,

$$\phi, \psi = \arg\max_{\phi', \psi'} \mathbb{E}_{p(x,t)} \Big(p(\mathbf{v}|\mathbf{u}) + p(\mathbf{u}|\mathbf{v}) \Big), \tag{5.21}$$

where $p(x, t)$ denotes the joint distribution of matching image-text pairs in the training set. For example, a CLIP model may learn to align the embeddings of an image of landbird and a text description "a photo of a landbird" while pushing embeddings of unrelated images and texts away from each other, such as an image of waterbird and "a photo of a landbird". Ideally, for a matching image-text pair (x, t), we will obtain $p(\mathbf{v}|\mathbf{u}) \approx p(\mathbf{u}|\mathbf{v})$ after training.

Zero-Shot Classification. Given an image x belonging to one of K classes $\{c_k\}_{k=1}^K$, zero-shot classification first constructs K text descriptions by inserting each class name into a predefined text template, such as "a photo of a [CLASS]". Each description is then encoded into a text representation $\mathbf{u}_k$ for each class c_k. Then, the zero-shot prediction $\hat{k}$ is:

$$\hat{k} = \arg\max_k p(\mathbf{u}_k|\mathbf{v}) = \arg\max_k \frac{\mathbf{v}^T\mathbf{u}_k}{\|\mathbf{v}\|_2\|\mathbf{u}_k\|_2}, \tag{5.22}$$

where $\mathbf{v}$ is the vision representation for the input image x, $\|\cdot\|_2$ is the Euclidean norm of a vector, and $p(\mathbf{u}_k|\mathbf{v})$ is defined to be proportional to $\mathbf{v}^T\mathbf{u}_k$.

Multimodal Spurious Bias

In practice, a given text description t may not fully describe the content in x. For example, x could be an image depicting a landbird with a land background, and t could simply be "a photo of a landbird", which only describes the primary object in the image. When a CLIP model learns to align many such image-text pairs where land backgrounds spuriously correlate with the target "landbird", then the model may inadvertently learn to align the representation of "a photo of a landbird" with the representation of land backgrounds, instead of the defining features of landbirds. The misalignment causes a *multimodal spurious bias* in the model, which tends to use land backgrounds in images to infer their descriptions. As illustrated in Figure 5.6, due to the misalignment, an image of waterbird with a land background is incorrectly paired with the description "a photo of a landbird".

To formally define multimodal spurious bias, we introduce $\mathbf{u}'_s \in \mathbb{R}^D$ to represent a latent textual spurious feature, such as the missing "land background" in the description "a photo of a landbird". With $\mathbf{u}'_s$, we can conveniently expand $p(\mathbf{v}|\mathbf{u})$ and $p(\mathbf{u}|\mathbf{v})$ in (5.21) as the marginalization over all possible textual spurious features, i.e.,

$$p(\mathbf{v}|\mathbf{u}) = \int_{\mathbf{u}'_s} p(\mathbf{v}|\mathbf{u}, \mathbf{u}'_s)p(\mathbf{u}'_s|\mathbf{u})d\mathbf{u}'_s, \tag{5.23}$$

and

$$p(\mathbf{u}|\mathbf{v}) = \int_{\mathbf{u}'_s} p(\mathbf{u}|\mathbf{v}, \mathbf{u}'_s)p(\mathbf{u}'_s|\mathbf{v})d\mathbf{u}'_s. \tag{5.24}$$

In the pre-training data, if the majority of images with their text representation $\mathbf{u}$ have a spurious feature represented by $\mathbf{u}_s$, then a CLIP model may learn the strong correlations between the spurious feature $\mathbf{u}_s$ and the image representation $\mathbf{v}$ as well as the text representation $\mathbf{u}$. As a result, the

model will develop a multimodal spurious bias and we will have $p(\mathbf{u}_s|\mathbf{u}) \approx 1$ and $p(\mathbf{u}_s|\mathbf{v}) \approx 1$. We formally define multimodal spurious bias in the following.

Definition 5.2 (**Multimodal Spurious Bias**)**.** Consider a pre-trained CLIP model consisting of a vision encoder ϕ and a text encoder ψ. Given an image-text pair (x, t) and a latent spurious feature $\mathbf{u}_s$, a multimodal spurious bias in the model relevant to $\mathbf{u}_s$ satisfies the following conditions:

$$p(\mathbf{v}|\mathbf{u}) \approx p(\mathbf{v}|\mathbf{u}, \mathbf{u}_s), \tag{5.25}$$

and

$$p(\mathbf{u}|\mathbf{v}) \approx p(\mathbf{u}|\mathbf{v}, \mathbf{u}_s), \tag{5.26}$$

where $\mathbf{v} = \phi(x)$ and $\mathbf{u} = \psi(t)$.

The above conditions indicate that $p(\mathbf{u}_s|\mathbf{u}) \approx 1$ and $p(\mathbf{u}_s|\mathbf{v}) \approx 1$ based on Equation (5.23) and Equation (5.24), and the pre-trained model tends to align $\mathbf{v}$ and $\mathbf{u}$ with $\mathbf{u}_s$. This indicates a misalignment between the vision representation $\mathbf{v}$ and the text representation $\mathbf{u}$. When the pre-trained model is tested on the data with $p(\mathbf{u}_s|\mathbf{u}) \ll 1$ and $p(\mathbf{u}_s|\mathbf{v}) \ll 1$, i.e., the spurious features in the test data no longer have strong correlations with input images and the corresponding text descriptions compared to the training data, such as the waterbird image with a land background in Figure 5.6 where a land background is no longer associated with landbird, then the model may struggle on most of the test data, showing degraded zero-shot classification performance.

Theoretical Insights

We first theoretically analyze how multimodal spurious bias affects zero-shot classification. The insights derived from our analysis will guide the design of our multimodal spurious bias mitigation method in the following section.

Without loss of generality, we consider a zero-shot classification task with two classes, c_1 and c_2. Given a prompt template, we can obtain text representations for the two classes as $\mathbf{u}_1$ and $\mathbf{u}_2$. Consider an image representation $\mathbf{v}$ from class c_2 with an unknown spurious feature described by the text representation $\mathbf{u}_s$. The zero-shot prediction $\hat{k}$ can be obtained as follows,

$$\hat{k} = \arg\max_{k \in \{1,2\}} p(\mathbf{u}_k|\mathbf{v}) \tag{5.27}$$

We assume a multimodal spurious bias between $\mathbf{u}_1$, $\mathbf{v}$, and $\mathbf{u}_s$, as indicated by the dashed arrows in Figure 5.7(a). Then, the zero-shot prediction may be biased towards the class label c_1, instead of the true class label c_2, as supported by the following theorem.

Theorem 5.3. Consider a pre-trained CLIP model from which we obtain two text representations $\mathbf{u}_1$, $\mathbf{u}_2$ for the class c_1 and c_2, respectively, an image representation $\mathbf{v}$ with the class label c_2, and a textual spurious feature $\mathbf{u}_s$ related to $\mathbf{v}$. Assume $\mathbf{u}_1$, $\mathbf{v}$, and $\mathbf{u}_s$ formulate a multimodal spurious bias. Then, the model is biased towards predicting $\mathbf{v}$ as c_1 instead of its true class label c_2.

Proof. We first follow Equation (5.24) to expand $p(\mathbf{u}_1|\mathbf{v})$, i.e.,

$$p(\mathbf{u}_1|\mathbf{v}) = \int_{\mathbf{u}'_s} p(\mathbf{u}_1|\mathbf{v}, \mathbf{u}'_s)p(\mathbf{u}'_s|\mathbf{v})d\mathbf{u}'_s \tag{5.28}$$

$$\approx p(\mathbf{u}_1|\mathbf{v}, \mathbf{u}_s)p(\mathbf{u}_s|\mathbf{v}) \tag{5.29}$$

$$= p(\mathbf{u}_s|\mathbf{u}_1)p(\mathbf{u}_1) \tag{5.30}$$

where the approximation in (5.29) uses the definition of multimodal spurious bias in Definition 5.2, and Equation (5.30) can be derived via Bayes' theorem, i.e.,

$$p(\mathbf{u}_1|\mathbf{v}, \mathbf{u}_s) = \frac{p(\mathbf{u}_1, \mathbf{u}_s|\mathbf{v})}{p(\mathbf{u}_s|\mathbf{v})} = \frac{p(\mathbf{u}_s|\mathbf{u}_1)p(\mathbf{u}_1)}{p(\mathbf{u}_s|\mathbf{v})}, \tag{5.31}$$

where the last equality follows the fact that $p(\mathbf{u}_1, \mathbf{u}_s|\mathbf{v}) = p(\mathbf{u}_1, \mathbf{u}_s)$, i.e., $\mathbf{u}_s$ and $\mathbf{u}_1$ do not depend on $\mathbf{v}$, as depicted in Figure 5.7(a). Therefore, we have the following inequality:

$$\frac{p(\mathbf{u}_1|\mathbf{v})}{p(\mathbf{u}_2|\mathbf{v})} \approx \frac{p(\mathbf{u}_s|\mathbf{u}_1)p(\mathbf{u}_1)}{p(\mathbf{u}_2|\mathbf{v})} > 1, \tag{5.32}$$

where the inequality follows from the condition that $\mathbf{u}_1$, $\mathbf{v}$, and $\mathbf{u}_s$ formulate a multimodal spurious bias, i.e., $p(\mathbf{u}_s|\mathbf{u}_1) \approx 1$, $p(\mathbf{u}_2|\mathbf{v}) \approx 0$ given that $p(\mathbf{u}_s|\mathbf{v}) \approx 1$, and $p(\mathbf{u}_1) > 0$ is a constant. Therefore, the model's prediction on $\mathbf{v}$ is biased towards the incorrect label c_1. □

The above theorem proves that strong multimodal spurious biases in a pre-trained CLIP model significantly affect its zero-shot performance. Although the analysis is based on a two-class task, the conclusion generally holds with multiple classes when $p(\mathbf{u}_k|\mathbf{v}), \forall k > 1$ is a small number. An important observation from Theorem 5.3 is that when $p(\mathbf{u}_1|\mathbf{v})$ becomes smaller, i.e., the similarity between $\mathbf{u}_1$ and $\mathbf{v}$ decreases, then $p(\mathbf{u}_s|\mathbf{u}_1)$ also becomes smaller. In other words, the multimodal spurious bias can be mitigated.

Self-Adaptive Prompt Exploration

Based on the previous analysis, mitigating multimodal spurious bias can be achieved by minimizing $p(\mathbf{u}_1|\mathbf{v})$, which, by definition, is proportional to $\mathbf{u}_1^T\mathbf{v}$—the similarity between an image embedding $\mathbf{v}$ and its spuriously associated text representation $\mathbf{u}_1$. However, $\mathbf{u}_1$ is typically unknown and $\mathbf{u}_1^T\mathbf{v}$ cannot be minimized via fine-tuning since downstream task data is inaccessible in the zero-shot setting.

To address these challenges, we first observe that different prompts produce varying zero-shot predictions [41]. This indicates that text representations for different prompt templates differ in their alignment with spurious features. For example, a post hoc analysis on the Waterbirds dataset [3] using CLIP-ViT-L/14 reveals that the template "a drawing of a [CLASS]" is more robust to multimodal spurious bias than "a photo of a [CLASS]", which suggests that the former is less aligned with spurious features. Therefore, prompt selection can be an effective approach to mitigate multimodal spurious bias. However, identifying optimal templates in a zero-shot setting remains challenging. To address this, we introduce a novel self-adaptive prompt exploration method below.

Specifically, given a set of N prompt templates $\mathcal{T} = \{T_i\}_{i=1}^N$, for a K-way zero-shot classification task with K class labels $c_1, \ldots, c_K$, we construct N text descriptions for the k'th class c_k as $\mathcal{D}_k = \{T_i(c_k)\}_{i=1}^N$, where $T_i(c_k)$ denotes the i'th prompt template filled with class c_k. For example, as illustrated in Figure 5.7(b) with $K = 2$ and $N = 3$, when c_1 is "landbird", $T_1(c_1)$ could be "a photo of a landbird". We forward $\mathcal{D}_k$ to the text encoder ψ to obtain the corresponding text representations $\{\mathbf{u}_i^k\}_{i=1}^N$, where $\mathbf{u}_i^k = \psi(T_i(c_k))$.

Prompt Exploration. Following the insights from the previous section, to mitigate a multimodal spurious bias for the current input image, we aim to minimize the similarity between the image representation $\mathbf{v}$ and $\mathbf{u}_1$, which denotes the text representation of a class associated with the bias. Since the exact class linked to the bias is unknown, we explore prompts for all classes. For instance, as shown in Figure 5.7(b), for the landbird class, we identify Prompt 1 as the most distant from $\mathbf{v}$ among the three prompts in the joint embedding space, while for the waterbird class, we select Prompt 3. Formally, for the input image representation $\mathbf{v}$, we select the desired prompt template T_*^k for the class c_k as follows,

$$T_*^k = \arg\min_{T_i^k \in \mathcal{D}_k} \frac{\mathbf{v}^T\mathbf{u}_i^k}{\|\mathbf{v}^T\|_2\|\mathbf{u}_i^k\|_2}, k = 1, \ldots, K. \tag{5.33}$$

The above process is *self-adaptive* because it relies on $\mathbf{v}$, and given $\mathbf{v}$, the selected template T_*^k is entirely determined by the model. By considering prompts across all classes, we ensure effective multimodal spurious bias mitigation tailored to the input image. Furthermore, a prompt that reduces the similarity between the image and text representations for one class generally also reduces similarity with others, which is beneficial for mitigating multimodal spurious biases associated with other classes.

Enhanced Zero-Shot Classification. Then, for each T_*^k, $\forall k = 1, \ldots, K$, we construct a zero-shot classifier with K weights $\{\mathbf{w}_j^k\}_{j=1}^K$, where $\mathbf{w}_j^k = \psi(T_*^k(c_j))$. For example, in the zero-shot classification step shown in Figure 5.7(c), we construct two zero-shot classifiers with the selected prompts. The final prediction $\hat{j}$ is obtained from the classification results averaged over K zero-shot classifiers, i.e.,

$$\hat{j} = \arg\max_j \frac{1}{K} \sum_{k=1}^{K} \frac{\mathbf{v}^T \mathbf{w}_j^k}{\|\mathbf{v}^T\|_2 \|\mathbf{w}_j^k\|_2}. \tag{5.34}$$

Our method, termed **S**elf-**A**dapti**V**e Prompt **E**xploration (SAVE), adaptively selects prompts for each input image from a set of candidates to mitigate multimodal spurious biases in a zero-shot setting. A key advantage of SAVE is that it operates without requiring additional task-specific data or prior knowledge of multimodal spurious biases.

5.3.3 Experiments

Datasets

We experiment on two datasets with **fine-grained spurious correlations**, where each class is correlated with certain spurious features, such as backgrounds and gender.

- **Waterbirds** [3] is an image dataset for recognizing waterbirds and landbirds. It is generated synthetically by combining images of the two kinds of birds from the CUB dataset [163] and the backgrounds, water and land, from the Places dataset [174].
- **CelebA** [149] is a large-scale image dataset of celebrity faces. The task is to identify hair color, non-blond or blond, with male and female as the spurious attributes.

We also experiment on two datasets with **coarse-grained spurious correlations** where classes are associated with domain-specific features.

- **PACS** [211] is a domain generalization dataset that includes four visually different styles: Photo, Art Painting, Cartoon, and Sketch. The task is to identify object categories (dog, elephant, giraffe, guitar, horse, house, person).
- **VLCS** [212] is a domain generalization benchmark composed of four datasets: PASCAL VOC 2007 [213] (V), LabelMe [214] (L), Caltech101 [215] (C), and SUN09 [216] (S). It contains five overlapping classes (bird, car, chair, dog, and person) drawn from each dataset. The main challenge is to learn invariant features that generalize across these distinct domains.

Experimental Setup

Evaluated Methods. For performance comparison in zero-shot classification, we adopt **ZS**, which represents standard zero-shot classification, and **ROBOSHOT** [95], a state-of-the-art method that leverages LLMs to identify spurious attributes and mitigate multimodal spurious bias. In addition to our proposed method, **SAVE**, we also consider its variant, **SAVE-All**. Unlike SAVE, which selects prompt templates that best mitigate multimodal spurious bias, SAVE-All utilizes all available templates. Specifically, given K classes and N templates, SAVE-All constructs $K \times N$ zero-shot classifiers, whereas SAVE constructs only K classifiers.

Models. We use six CLIP-like models with different sizes and architectures, i.e., CLIP-RN-50, CLIP-ViT-B/32, CLIP-ViT-L/14, CLIP-ViT-H/14 [41], ALIGN [204], and AltCLIP [217]. For our method, we use the 80 prompt templates provided by CLIP models [41] and list them in Table A.8.1 in Appendix A.8.

Evaluation Metrics. We evaluate the zero-shot classification performance of a model using average accuracy (AVG), which measures accuracy across all test samples, and worst-group accuracy (WGA), which reflects the lowest accuracy among different test groups (as defined in Table A.8.2 in the Appendix A.8). A model with strong multimodal spurious bias may achieve a high AVG if most test samples contain the spurious correlations the model has learned, but it may achieve a low WGA when these correlations are absent. A robust model should exhibit both high AVG and high WGA.

Main Results

We evaluate the effectiveness of our methods, SAVE and SAVE-All, in mitigating multimodal spurious biases at both the *fine-grained* level, where each class is correlated with specific spurious features, on the Waterbirds and CelebA datasets (Table 5.9), and the *coarse-grained* level, where classes are associated with broader domain-specific features, on the PACS and VLCS datasets (Table 5.10). For

Dataset	Model	ZS		ROBOSHOT		**Ours**			
						SAVE-All		**SAVE**	
		AVG(↑)	WGA(↑)	AVG(↑)	WGA(↑)	AVG(↑)	WGA(↑)	AVG(↑)	WGA(↑)
Waterbirds	CLIP-RN-50	88.7	41.0	72.1	27.6	92.6	**48.8**	91.9	40.0
	CLIP-ViT-B/32	80.4	27.5	74.2	39.3	93.0	32.2	91.3	**43.6**
	CLIP-ViT-L/14	88.6	27.6	79.8	48.1	93.3	37.7	92.8	**49.7**
	CLIP-ViT-H/14	89.0	44.5	76.9	**52.5**	90.0	37.2	88.7	42.7
	ALIGN	72.3	50.0	52.6	38.3	81.6	47.2	79.8	**51.2**
	AltCLIP	90.3	37.2	78.5	**54.2**	88.6	33.6	89.1	45.2
	Average	84.9	38.0	72.4	43.3	89.9	39.5	88.9	**45.4**
CelebA	CLIP-RN-50	81.6	**75.2**	81.6	74.9	76.2	72.5	76.4	71.2
	CLIP-ViT-B/32	78.3	68.9	82.1	75.2	77.5	73.9	79.9	**75.5**
	CLIP-ViT-L/14	80.5	74.0	85.3	**82.2**	79.1	75.3	83.2	80.4
	CLIP-ViT-H/14	83.3	**79.7**	82.7	76.9	78.1	71.5	81.9	77.5
	ALIGN	82.4	78.2	87.0	**84.8**	83.3	81.0	80.1	74.4
	AltCLIP	82.9	80.2	86.1	**80.6**	84.4	79.4	83.0	79.7
	Average	81.5	76.0	84.1	**79.1**	79.8	75.6	80.8	76.5

Table 5.9: Performance on Waterbirds and CelebA with fine-grained spurious correlations. The best worst-group accuracy (WGA) in each model is in **boldface**.

Dataset	Model	ZS		ROBOSHOT		**Ours**			
						SAVE-All		**SAVE**	
		AVG(↑)	WGA(↑)	AVG(↑)	WGA(↑)	AVG(↑)	WGA(↑)	AVG(↑)	WGA(↑)
PACS	CLIP-RN-50	91.8	63.3	92.3	**72.4**	94.0	69.8	93.4	70.2
	CLIP-ViT-B/32	96.6	82.1	96.6	**83.5**	97.9	82.5	97.6	83.1
	CLIP-ViT-L/14	98.1	79.8	98.0	81.3	98.3	**87.4**	97.8	85.2
	CLIP-ViT-H/14	98.9	**90.8**	98.7	89.1	98.9	89.1	98.6	87.8
	ALIGN	95.8	69.6	94.7	63.2	97.2	**82.3**	95.9	76.7
	AltCLIP	98.5	82.5	98.8	**89.4**	98.8	88.4	98.7	85.4
	Average	96.6	78.0	96.5	79.8	97.5	**83.3**	97.0	81.4
VLCS	CLIP-RN-50	75.5	34.1	77.6	**37.6**	80.6	31.5	79.4	24.3
	CLIP-ViT-B/32	75.4	20.5	77.1	**35.2**	79.3	27.3	78.5	30.7
	CLIP-ViT-L/14	72.4	4.1	70.9	12.2	80.2	27.3	79.6	**36.1**
	CLIP-ViT-H/14	70.3	4.2	70.4	13.0	80.0	29.5	80.2	**34.1**
	ALIGN	78.5	34.1	77.4	39.8	80.6	**41.0**	78.7	39.0
	AltCLIP	78.8	22.0	78.3	25.7	81.7	**30.1**	81.0	21.3
	Average	75.2	19.8	75.3	27.3	80.4	**31.1**	79.6	30.9

Table 5.10: Performance on PACS and VLCS with coarse-grained spurious correlations. The best worst-group accuracy (WGA) in each model is in **boldface**.

each dataset, we tested six models and reported the average results across these models to assess the overall effectiveness of each method.

On three out of four datasets (Waterbirds, PACS, and VLCS), our method SAVE, achieves higher average AVGs and WGAs than ROBOSHOT, a state-of-the-art zero-shot debiasing method. This highlights the strong zero-shot debiasing capability of SAVE. Notably, while improving WGAs over ROBOSHOT, SAVE does not compromise AVG. In fact, on the Waterbirds dataset, SAVE outper-

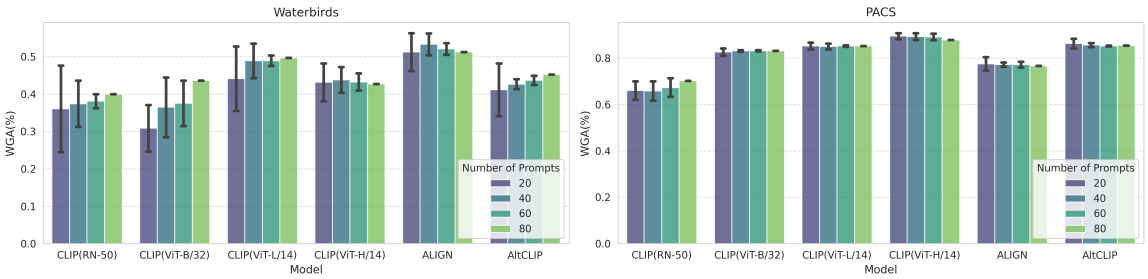


Figure 5.8: Ablation study on the effect of varying prompt numbers in different Models with our proposed method. Standard deviations are marked with dark vertical bars.

forms ROBOSHOT by 16.5% in average accuracy, demonstrating its ability to mitigate multimodal spurious biases in specific data groups without degrading overall predictive performance. Furthermore, SAVE does not require specialized prompt design, making it a convenient, out-of-the-box solution for use with pre-trained CLIP models.

On the CelebA dataset, we observe that ROBOSHOT achieves the highest AVG and WGA scores. We attribute this to the specialized nature of CelebA compared to Waterbirds, PACS, and VLCS, as it consists exclusively of celebrity face images. In zero-shot classification using a pre-trained CLIP model, accurately inferring hair color requires prompts that closely align with the dataset. For example, ROBOSHOT employs prompts such as "a person with dark hair," which are more semantically relevant than generic alternatives like "a photo of dark hair." The latter may struggle to align text descriptions with input images, even in the absence of biases. This highlights the importance of effective prompt design for robust classification on this dataset. Nevertheless, despite relying on a fixed set of 80 templates from Table A.8.1, our method, SAVE, still achieves a higher overall WGA than ZS, which explicitly adopts more aligned prompts.

For mitigating fine-grained multimodal spurious biases on the Waterbirds and CelebA datasets, selectively choosing prompt templates is more effective than ensembling all available templates, as evidenced by the higher average WGAs achieved by SAVE compared with SAVE-All (Table 5.9). However, for mitigating coarse-grained multimodal spurious biases on the PACS and VLCS datasets, SAVE-All slightly outperforms SAVE on average. This suggests that using multiple prompt templates per class is beneficial, as most templates contribute positively to mitigating coarse-grained multimodal spurious biases. Nonetheless, SAVE-All comes with higher computational complexity, requiring the construction of $K \times N$ zero-shot classifiers instead of K, given that there are K classes and N available prompt templates.

Ablation Studies

By default, our method explores all the prompt templates. Intuitively, more prompt templates provide more diverse representations for class labels, which enables our method to explore more broadly and thus to more effectively mitigate multimodal spurious bias. In the following, we analyze how different numbers of prompt templates affect the effectiveness of our method.

We began by randomly sampling 20, 40, 60, and 80 (all) prompt templates from the full set of available templates (Table A.8.1 in Appendix A.8). We then evaluated our method on the Waterbirds and PACS datasets using these sampled templates. For each sample size, we conducted ten independent runs with different random seeds.

Figure 5.8 shows the average worst-group accuracies (WGAs) along with standard deviations across ten runs. Overall, we observe that exploring a larger number of prompt templates tends to improve WGA on the Waterbirds dataset (Figure 5.8, left panel). This improvement is more significant for smaller models, such as CLIP-RN-50 and CLIP-ViT-B/32, whereas larger models, such as CLIP-ViT-H/14, exhibit minimal gains as the number of prompts increases. We hypothesize that this is due to the stronger language understanding capabilities of larger models. As a result, the text representations for the same class label remain more consistent across different prompt templates, reducing the effectiveness of prompt selection in mitigating multimodal spurious bias.

In contrast, on the PACS dataset (Figure 5.8, right panel), we observe that most models achieve similar WGAs regardless of the number of prompt templates used, with significantly lower variance in WGA due to random sampling compared to the Waterbirds dataset. This indicates that our method effectively mitigates multimodal spurious bias at a coarse-grained level (i.e., by addressing classes associated with domain-specific features) as long as a sufficient number of prompt templates are explored. Additionally, the low variance in WGA across different template counts suggests that most prompts contribute positively to bias mitigation. This highlights the potential of an ensemble approach that incorporates all available templates to construct zero-shot classifiers for robust bias reduction, as further validated in Table 5.10. Finally, increasing the number of prompt templates proves particularly effective for CLIP-RN-50, which is more biased than other models (Table 5.10) and thus benefits from a broader range of templates for improved bias mitigation.

Analysis on the Selected Prompt Templates

Our proposed method, SAVE, is designed to adaptively selects prompt templates for each input image. We aim to understand what prompt templates are most frequently selected by our method

Figure 5.9: Most frequently selected prompt templates for each class by our method with CLIP-ViT-B/32 in the Waterbirds dataset.

and how are the selected prompt templates differ across classes. In Figure 5.9, we show the top-10 frequently selected prompt templates along with their selection frequencies for the landbird and waterbird classes in the Waterbirds dataset.

We observe that our method frequently selects the prompt templates "a black and white photo of a [CLASS]" and "a doodle of a [CLASS]" for both the waterbird and landbird classes. This suggests that our method identifies these templates as the most effective for mitigating multimodal spurious biases. The best WGA achieved by our method, as shown in Table 5.9, further validates the effectiveness of this selection strategy. A closer examination of the two templates reveals that the words they contain primarily describe out-of-distribution images. For instance, all images in the Waterbirds dataset are color images. By using "black and white" in the prompt, the corresponding text representation is shifted away from the input image representation, thereby effectively mitigating potential multimodal spurious bias, as discussed in Section 5.3.2. This also suggests that incorporating out-of-distribution words into templates could be a useful approach when constructing customized prompt templates.

Additionally, our method selects distinct prompts within each class (e.g., "a blurry photo of the [CLASS]" for landbird and "a tap of the [CLASS]" for waterbird). This demonstrates that our method is capable of adaptively selecting appropriate prompts to mitigate multimodal spurious biases.

5.3.4 Conclusion

In this paper, we addressed the challenge of mitigating multimodal spurious biases in pre-trained CLIP models for zero-shot classification. We first provided a theoretical definition of multimodal spurious bias and analyzed its impact on zero-shot classification. Based on these insights, we proposed a self-adaptive prompt exploration method that enhances robustness to such biases. Our approach operates out-of-the-box with CLIP models, requiring no additional training data or prior knowledge of biases. It is broadly effective across various model sizes, architectures, and types of spurious correlations. Moreover, it achieves a strong balance between average and worst-group zero-shot classification accuracy, highlighting its practical utility in zero-shot predictions.

Chapter 6

Conclusion and Future Directions

6.1 Conclusion

In this dissertation, we focus on learning robust machine learning (ML) models via spurious bias mitigation. Spurious biases arise when ML models inadvertently learn to use spurious attributes in the training data for predictions and can be revealed when we deploy ML models in new environments with distribution shifts where the learned spurious attributes no longer exist. Based on the available knowledge about distribution shifts, we proposed targeted spurious bias mitigation methods to improve models' out-of-distribution generalization and generalization under subpopulation shifts.

Out-of-distribution generalization requires a model to generalize to new data distributions, such as to novel classes or novel domains, which are not known to the model during training. In this scenario, we proposed to explore new data distributions during model training by synthesizing spurious attributes via data augmentations. In Chapter 3.1, we proposed an adversarial data augmentation and invariant learning framework that synthesizes challenging samples with new spurious attributes, such as images with different rotations, and regularizes the model for learning robust and generalizable features. We showed that the proposed framework significantly improves a model's single domain generalization performance. In Chapter 3.2, we proposed to meta-learn a task transformation layer which transforms samples in an input task to mitigate the risk of learning spurious attributes in the task and to improve generalization to novel concepts with a few labeled samples. This approach can be applied to any meta-learning algorithms to improve few-shot classification performance. In Chapter 3.3, we proposed to meta-learn a dictionary of spurious features with data augmentations and then to mitigate the spurious features synthesized from the learned ones in the

latent space of a model when learning novel classes. We demonstrated that this method outperforms previous baselines in few-shot classification.

Generalization under subpopulation shifts requires a model to perform reliably across data groups with varying spurious attributes, especially when the proportions of these groups differ between training and testing. Models may learn from data groups with the majority of training samples and ignore the information from other data groups. Balancing model performance across data groups is essential; however, it is challenging to acquire group annotations due to costly human annotation efforts. In Chapter 4.1, we proposed to use pre-trained vision language models (VLMs) to detect spurious attributes in images and demonstrated that the extracted attributes are effective in generating challenging classification tasks with subpopulation shifts for evaluating few-shot classifiers. Motivated by the success of using VLMs to extract spurious attributes, in Chapter 4.2, we proposed a fine-grained spurious-attribute-aware classification using the extracted attributes to decouple prediction targets from spurious attributes. In Chapter 4.3, we further proposed to learn spurious-attribute-agnostic representations via meta-learning on the classification tasks with simulated subpopulation shifts based on the spurious attributes extracted from VLMs. These multimodal-assisted approaches improve model robustness against subpopulation shifts without group annotations.

Multimodal-assisted approaches require a relatively long data preprocessing step to extract spurious attributes using pre-trained VLMs. Moreover, the extracted spurious attributes depend on choices of VLMs. In Chapter 5.1, we proposed to probe the latent space of a model to identify prediction shortcuts and use them to regularize model retraining. In Chapter 5.2, we proposed a direct spurious bias mitigation method by suppressing the influence of neurons identified as primarily encoding spurious features. These latent space probing methods are fully self-guided and do not require group annotations. Experiments demonstrated that these methods can effectively improve model robustness against subpopulation shifts. In Chapter 5.3, we extended the idea of latent space probing and proposed prompt selection methods based on their latent representations to improve the zero-shot generalization performance of VLMs under distribution shifts.

6.2 Future Directions

Constructing Diverse Probe Sets. In our experiments, the held-out validation set of a dataset is often used as the probe set for detecting and mitigating spurious biases in a model. In general, any set can be used as the probe set as long as it contains samples with diverse spurious attributes representing distribution shifts from the data used to train the model. Future works may focus

on data-centric approaches to design probe sets that can be used to optimally detect and mitigate spurious biases in a trained model.

Spurious Bias in Large Language Models. Large language models (LLMs), though being powerful, have demonstrated various shortcut learning behaviors, such as using lexical overlap, subsequence, negation, or style for predictions [218, 219, 220, 221]. These shortcut learning behaviors undermine the robustness and generalization capabilities of LLMs. In future works, we may extend our proposed benchmarking system based on images to revealing spurious biases in LLMs using text data. With the data generated from the benchmark system, we may design novel mitigation methods or extend our latent space methods such as NeuronTune to efficiently debias LLMs.

Spurious Bias in Vision-Language Models. Although VLMs have shown strong vision and language understanding, there may exist misalignments between vision and language modalities, such as aligning spurious features in the vision modality with text descriptions for classification, demonstrating multimodal spurious biases in VLMs. We have proposed a spurious bias mitigation method that adaptively selects prompts based on input images to minimize the reliance on spurious attributes in zero-shot classification. In the future, we may analyze spurious biases in other tasks beyond classification, such as visual-question answering [222], and explore spurious mitigation strategies beyond prompt selection, such as efficient fine-tuning, prompt learning, and modality alignment.

References

[1] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, and Antonio Torralba. Learning deep features for discriminative localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2921–2929, 2016.

[2] Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673, 2020.

[3] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2019.

[4] Polina Kirichenko, Pavel Izmailov, and Andrew Gordon Wilson. Last layer re-training is sufficient for robustness to spurious correlations. In *International Conference on Learning Representations*, 2023.

[5] Guangtao Zheng, Wenqian Ye, and Aidong Zhang. Learning robust classifiers with self-guided spurious correlation mitigation. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pages 5599–5607, 2024.

[6] Sara Beery, Grant Van Horn, and Pietro Perona. Recognition in terra incognita. In *Proceedings of the European Conference on Computer Vision*, pages 456–473, 2018.

[7] John R Zech, Marcus A Badgeley, Manway Liu, Anthony B Costa, Joseph J Titano, and Eric Karl Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS Medicine*, 15(11):e1002683, 2018.

[8] Yogesh Balaji, Swami Sankaranarayanan, and Rama Chellappa. Metareg: Towards domain generalization using meta-regularization. *Advances in Neural Information Processing Systems*, 31, 2018.

[9] Da Li, Jianshu Zhang, Yongxin Yang, Cong Liu, Yi-Zhe Song, and Timothy M Hospedales. Episodic training for domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1446–1455, 2019.

[10] Prashant Pandey, Mrigank Raman, Sumanth Varambally, and Prathosh AP. Domain generalization via inference-time label-preserving target projections. *arXiv preprint arXiv:2103.01134*, 2021.

[11] Shiv Shankar, Vihari Piratla, Soumen Chakrabarti, Siddhartha Chaudhuri, Preethi Jyothi, and Sunita Sarawagi. Generalizing across domains via cross-gradient training. *arXiv preprint arXiv:1804.10745*, 2018.

[12] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in Neural Information Processing Systems*, 30, 2017.

[13] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. Meta-learning with differentiable closed-form solvers. In *International Conference on Learning Representations*, 2019.

[14] Kwonjoon Lee, Subhransu Maji, Avinash Ravichandran, and Stefano Soatto. Meta-learning with differentiable convex optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10657–10665, 2019.

[15] Han-Jia Ye, Hexiang Hu, De-Chuan Zhan, and Fei Sha. Few-shot learning via embedding adaptation with set-to-set functions. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8808–8817, 2020.

[16] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Differentiable earth mover's distance for few-shot learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(5):5632–5648, 2022.

[17] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

[18] Dan Hendrycks, Norman Mu, Ekin Dogus Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. Augmix: A simple data processing method to improve robustness and uncertainty. In *International Conference on Learning Representations*, 2019.

[19] Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. In *European Conference on Computer Vision*, pages 566–583. Springer, 2020.

[20] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[21] Qing Lian, Botao Ye, Ruijia Xu, Weilong Yao, and Tong Zhang. Geometry-aware data augmentation for monocular 3D object detection. *arXiv preprint arXiv:2104.05858*, 2021.

[22] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, volume 70, pages 1126–1135, 2017.

[23] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in Neural Information Processing Systems*, 29, 2016.

[24] Yoonho Lee and Seungjin Choi. Gradient-based meta-learning with learned layerwise metric and subspace. In *International Conference on Machine Learning*, 2018.

[25] Andrei A. Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*, 2019.

[26] Yinbo Chen, Xiaolong Wang, Zhuang Liu, Huijuan Xu, and Trevor Darrell. A new meta-baseline for few-shot learning. *arXiv preprint arXiv:2003.04390*, 2020.

[27] Renkun Ni, Micah Goldblum, Amr Sharaf, Kezhi Kong, and Tom Goldstein. Data augmentation for meta-learning. In *International Conference on Machine Learning*, pages 8152–8161. PMLR, 2021.

[28] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pages 5637–5664. PMLR, 2021.

[29] Yuzhe Yang, Haoran Zhang, Dina Katabi, and Marzyeh Ghassemi. Change is hard: A closer look at subpopulation shift. In *International Conference on Machine Learning*, pages 39584–39622. PMLR, 2023.

[30] Yihe Deng, Yu Yang, Baharan Mirzasoleiman, and Quanquan Gu. Robust learning with progressive data expansion against spurious correlation. *arXiv preprint arXiv:2306.04949*, 2023.

[31] Evan Z Liu, Behzad Haghgoo, Annie S Chen, Aditi Raghunathan, Pang Wei Koh, Shiori Sagawa, Percy Liang, and Chelsea Finn. Just train twice: Improving group robustness without training group information. In *International Conference on Machine Learning*, pages 6781–6792. PMLR, 2021.

[32] Shikai Qiu, Andres Potapczynski, Pavel Izmailov, and Andrew Gordon Wilson. Simple and fast group robustness by automatic feature reweighting. In *International Conference on Machine Learning*, pages 28448–28467. PMLR, 2023.

[33] Junhyun Nam, Hyuntak Cha, Sungsoo Ahn, Jaeho Lee, and Jinwoo Shin. Learning from failure: De-biasing classifier from biased classifier. *Advances in Neural Information Processing Systems*, 33:20673–20684, 2020.

[34] Junhyun Nam, Jaehyung Kim, Jaeho Lee, and Jinwoo Shin. Spread spurious attribute: Improving worst-group accuracy with spurious attribute estimation. In *International Conference on Learning Representations*, 2022.

[35] Michael Zhang, Nimit S Sohoni, Hongyang R Zhang, Chelsea Finn, and Christopher Re. Correct-n-contrast: a contrastive approach for improving robustness to spurious correlations. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 26484–26516. PMLR, 17–23 Jul 2022.

[36] Guangtao Zheng, Mengdi Huai, and Aidong Zhang. Advst: Revisiting data augmentations for single domain generalization. In *Proceedings of the AAAI conference on artificial intelligence*, 2024.

[37] Guangtao Zheng, Qiuling Suo, Mengdi Huai, and Aidong Zhang. Learning to learn task transformations for improved few-shot classification. In *Proceedings of the 2023 SIAM International Conference on Data Mining*, pages 784–792. SIAM, 2023.

[38] Guangtao Zheng and Aidong Zhang. Knowledge-guided semantics adjustment for improved few-shot classification. In *2022 IEEE International Conference on Data Mining*, pages 1347–1352. IEEE, 2022.

[39] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[40] NLP Connect. vit-gpt2-image-captioning (revision 0e334c7), 2022. URL `https://huggingface.co/nlpconnect/vit-gpt2-image-captioning`.

[41] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[42] Guangtao Zheng, Wenqian Ye, and Aidong Zhang. Benchmarking spurious bias in few-shot image classifiers. In *European Conference on Computer Vision*, pages 346–364. Springer, 2024.

[43] Guangtao Zheng, Wenqian Ye, and Aidong Zhang. Spuriousness-aware meta-learning for learning robust classifiers. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4524–4535, 2024.

[44] Guangtao Zheng, Wenqian Ye, and Aidong Zhang. Shortcutprobe: Probing prediction shortcuts for learning robust models. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, 2025.

[45] Guangtao Zheng, Wenqian Ye, and Aidong Zhang. Neurontune: Towards self-guided spurious bias mitigation. In *Proceedings of the Forty-Second International Conference on Machine Learning*, 2025.

[46] Wenqian Ye, Guangtao Zheng, Xu Cao, Yunsheng Ma, and Aidong Zhang. Spurious correlations in machine learning: A survey. *ICML Workshop on Data-Centric Machine Learning Research*, 2024.

[47] Nicholas Baker, Hongjing Lu, Gennady Erlikhman, and Philip J Kellman. Deep convolutional networks do not classify based on global object shape. *PLoS Computational Biology*, 14(12): e1006613, 2018.

[48] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.

[49] Kai Yuanqing Xiao, Logan Engstrom, Andrew Ilyas, and Aleksander Madry. Noise or signal: The role of image backgrounds in object recognition. In *International Conference on Learning Representations*, 2021.

[50] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[51] Riccardo Volpi, Hongseok Namkoong, Ozan Sener, John Duchi, Vittorio Murino, and Silvio Savarese. Generalizing to unseen domains via adversarial data augmentation. In *Advances in Neural Information Processing Systems*, pages 5339–5349, 2018.

[52] Long Zhao, Ting Liu, Xi Peng, and Dimitris Metaxas. Maximum-entropy adversarial data augmentation for improved generalization and robustness. *Advances in Neural Information Processing Systems*, 2020.

[53] Zhun Zhong, Yuyang Zhao, Gim Hee Lee, and Nicu Sebe. Adversarial style augmentation for domain generalized urban-scene segmentation. *Advances in Neural Information Processing Systems*, 35:338–350, 2022.

[54] Yabin Zhang, Bin Deng, Ruihuang Li, Kui Jia, and Lei Zhang. Adversarial style augmentation for domain generalization. *arXiv preprint arXiv:2301.12643*, 2023.

[55] Fengchun Qiao, Long Zhao, and Xi Peng. Learning to learn single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12556–12565, 2020.

[56] Zijian Wang, Yadan Luo, Ruihong Qiu, Zi Huang, and Mahsa Baktashmotlagh. Learning to diversify for single domain generalization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 834–843, 2021.

[57] Lei Li, Ke Gao, Juan Cao, Ziyao Huang, Yepeng Weng, Xiaoyue Mi, Zhengze Yu, Xiaoya Li, and Boyang Xia. Progressive domain expansion network for single domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 224–233, 2021.

[58] Jeet Mohapatra, Tsui-Wei Weng, Pin-Yu Chen, Sijia Liu, and Luca Daniel. Towards verifying robustness of neural networks against a family of semantic perturbations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 244–252, 2020.

[59] Hossein Hosseini and Radha Poovendran. Semantic adversarial examples. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1695–16955, 2018.

[60] Anand Bhattad, Min Jin Chong, Kaizhao Liang, Bo Li, and D. A. Forsyth. Unrestricted adversarial examples via semantic manipulation. In *International Conference on Learning Representations*, 2020.

[61] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial autoaugment. In *International Conference on Learning Representations*, 2019.

[62] Tejas Gokhale, Abhishek Chaudhary, Pratyay Banerjee, Chitta Baral, and Yezhou Yang. Semantically distributed robust optimization for vision-and-language inference. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1493–1513, 2022.

[63] Zhongqi Yue, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. Interventional few-shot learning. *Advances in Neural Information Processing Systems*, 33:2734–2746, 2020.

[64] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Frank Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *International Conference on Learning Representations*, 2019.

[65] Yaqing Wang, Quanming Yao, James T Kwok, and Lionel M Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53(3):1–34, 2020.

[66] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? *arXiv preprint arXiv:2003.11539*, 2020.

[67] Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, and Hugo Larochelle. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*, 2020.

[68] Chi Zhang, Yujun Cai, Guosheng Lin, and Chunhua Shen. Deepemd: Few-shot image classification with differentiable earth mover's distance and structured classifiers. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2020.

[69] Huaxiu Yao, Long-Kai Huang, Linjun Zhang, Ying Wei, Li Tian, James Zou, Junzhou Huang, and Zhenhui Li. Improving generalization in meta-learning via task augmentation. In *International Conference on Machine Learning*, pages 11887–11897. PMLR, 2021.

[70] Huaxiu Yao, Linjun Zhang, and Chelsea Finn. Meta-learning with fewer tasks through task interpolation. In *International Conference on Learning Representations*, 2022.

[71] Ekin Dogus Cubuk, Barret Zoph, Dandelion Mané, Vijay Vasudevan, and Quoc V. Le. Autoaugment: Learning augmentation strategies from data. *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019.

[72] Sungbin Lim, Ildoo Kim, Taesup Kim, Chiheon Kim, and Sungwoong Kim. Fast AutoAugment. In *Advances in Neural Information Processing Systems*, 2019.

[73] Ekin Dogus Cubuk, Barret Zoph, Jon Shlens, and Quoc Le. RandAugment: Practical automated data augmentation with a reduced search space. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18613–18624. Curran Associates, Inc., 2020.

[74] Ryuichiro Hataya, Jan Zdenek, Kazuki Yoshizoe, and Hideki Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020.

[75] Xinyu Zhang, Qiang Wang, Jian Zhang, and Zhao Zhong. Adversarial AutoAugment. In *International Conference on Learning Representations*, 2020.

[76] Aoming Liu, Zehao Huang, Zhiwu Huang, and Naiyan Wang. Direct differentiable augmentation search. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12219–12228, 2021.

[77] Soravit Changpinyo, Wei-Lun Chao, Boqing Gong, and Fei Sha. Synthesized classifiers for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5327–5336, 2016.

[78] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546, 2019.

[79] Yin Cui, Menglin Jia, Tsung-Yi Lin, Yang Song, and Serge Belongie. Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9268–9277, 2019.

[80] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[81] Jonathon Byrd and Zachary Lipton. What is the effect of importance weighting in deep learning? In *International Conference on Machine Learning*, pages 872–881. PMLR, 2019.

[82] Elliot Creager, Jörn-Henrik Jacobsen, and Richard Zemel. Environment inference for invariant learning. In *International Conference on Machine Learning*, pages 2189–2200. PMLR, 2021.

[83] Tyler LaBonte, Vidya Muthukumar, and Abhishek Kumar. Towards last-layer retraining for group robustness with fewer annotations. *Advances in Neural Information Processing Systems*, 36, 2024.

[84] Christopher Clark, Mark Yatskar, and Luke Zettlemoyer. Don't take the easy way out: Ensemble based methods for avoiding known dataset biases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4069–4082, 2019.

[85] Meike Nauta, Ricky Walsh, Adam Dubowski, and Christin Seifert. Uncovering and correcting shortcut learning in machine learning models for skin cancer diagnosis. *Diagnostics*, 12(1):40, 2021.

[86] Besmira Nushi, Ece Kamar, and Eric Horvitz. Towards accountable ai: Hybrid human-machine analyses for characterizing system failure. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, volume 6, pages 126–135, 2018.

[87] Jiawei Zhang, Yang Wang, Piero Molino, Lezhi Li, and David S Ebert. Manifold: A model-agnostic framework for interpretation and diagnosis of machine learning models. *IEEE Transactions on Visualization and Computer Graphics*, 25(1):364–373, 2018.

[88] Gregory Plumb, Marco Tulio Ribeiro, and Ameet Talwalkar. Finding and fixing spurious patterns with explanations. *Transactions on Machine Learning Research*, 2022. ISSN 2835-8856. Expert Certification.

[89] Abubakar Abid, Mert Yuksekgonul, and James Zou. Meaningfully debugging model mistakes using conceptual counterfactual explanations. In *International Conference on Machine Learning*, pages 66–88. PMLR, 2022.

[90] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? In *International Conference on Learning Representations*, 2021.

[91] Yannic Neuhaus, Maximilian Augustin, Valentyn Boreiko, and Matthias Hein. Spurious features everywhere–large-scale detection of harmful spurious features in imagenet. *arXiv preprint arXiv:2212.04871*, 2022.

[92] Shirley Wu, Mert Yuksekgonul, Linjun Zhang, and James Zou. Discover and cure: Concept-aware mitigation of spurious correlation. *arXiv preprint arXiv:2305.00650*, 2023.

[93] Yunhao Ge, Jie Ren, Andrew Gallagher, Yuxiao Wang, Ming-Hsuan Yang, Hartwig Adam, Laurent Itti, Balaji Lakshminarayanan, and Jiaping Zhao. Improving zero-shot generalization and robustness of multi-modal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11093–11101, 2023.

[94] Ching-Yao Chuang, Varun Jampani, Yuanzhen Li, Antonio Torralba, and Stefanie Jegelka. Debiasing vision-language models via biased prompts. *arXiv preprint arXiv:2302.00070*, 2023.

[95] Dyah Adila, Changho Shin, Linrong Cai, and Frederic Sala. Zero-shot robustification of zero-shot models. In *International Conference on Learning Representations*, 2024.

[96] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5542–5550, 2017.

[97] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[98] Jose Blanchet and Karthyek Murthy. Quantifying distributional model risk via optimal transport. *Mathematics of Operations Research*, 44(2):565–600, 2019.

[99] Rui Gao and Anton Kleywegt. Distributionally robust stochastic optimization with wasserstein distance. *Mathematics of Operations Research*, 2022.

[100] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673, 2020.

[101] Ruidi Chen and Ioannis Ch Paschalidis. Distributionally robust learning. *arXiv preprint arXiv:2108.08993*, 2021.

[102] Hamed Rahimian and Sanjay Mehrotra. Distributionally robust optimization: A review. *arXiv preprint arXiv:1908.05659*, 2019.

[103] Daniel Kuhn, Peyman Mohajerin Esfahani, Viet Anh Nguyen, and Soroosh Shafieezadeh-Abadeh. Wasserstein distributionally robust optimization: Theory and applications in machine learning. In *Operations Research & Management Science in the Age of Analytics*, pages 130–166. INFORMS, 2019.

[104] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189. PMLR, 2015.

[105] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Baolin Wu, Andrew Y Ng, et al. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*. Granada, 2011.

[106] John S Denker, WR Gardner, Hans Peter Graf, Donnie Henderson, Richard E Howard, W Hubbard, Lawrence D Jackel, Henry S Baird, and Isabelle Guyon. Neural network recognizer for hand-written zip code digits. In *Advances in Neural Information Processing Systems*, pages 323–331. Citeseer, 1989.

[107] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1406–1415, 2019.

[108] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski. Kornia: An open source differentiable computer vision library for pytorch. In *Winter Conference on Applications of Computer Vision*, 2020. URL `https://arxiv.org/pdf/1910.02190.pdf`.

[109] Saeid Motiian, Marco Piccirilli, Donald A Adjeroh, and Gianfranco Doretto. Unified deep supervised domain adaptation and generalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5715–5725, 2017.

[110] Xiang Xu, Xiong Zhou, Ragav Venkatesan, Gurumurthy Swaminathan, and Orchid Majumder. d-sne: Domain adaptation using stochastic neighborhood embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2497–2506, 2019.

[111] Fabio M Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2229–2238, 2019.

[112] Jiayi Chen and Aidong Zhang. Hetmaml: Task-heterogeneous model-agnostic meta-learning for few-shot learning across modalities. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 191–200, 2021.

[113] Jiayi Chen and Aidong Zhang. Topological transduction for hybrid few-shot learning. In *Proceedings of the ACM Web Conference 2022*, pages 3134–3142, 2022.

[114] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. *arXiv preprint arXiv:1805.09501*, 2018.

[115] Riccardo Volpi and Vittorio Murino. Addressing model vulnerability to distributional shifts over image transformation sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7980–7989, 2019.

[116] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *International Conference on Learning Representations*, 2018.

[117] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, October 2019.

[118] Ilke Cugu, Massimiliano Mancini, Yanbei Chen, and Zeynep Akata. Attention consistency on visual corruptions for single-source domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 4165–4174, June 2022.

[119] Thomas M Cover. *Elements of information theory*. John Wiley & Sons, 1999.

[120] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[121] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning*, pages 1139–1147. PMLR, 2013.

[122] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[123] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[124] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

[125] F. Sung, Y. Yang, L. Zhang, T. Xiang, P. H. S. Torr, and T. M. Hospedales. Learning to compare: Relation network for few-shot learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[126] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6023–6032, 2019.

[127] Brenden M. Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua B. Tenenbaum. One shot learning of simple visual concepts. *Cognitive Science*, 33, 2011.

[128] Mengye Ren, Sachin Ravi, Eleni Triantafillou, Jake Snell, Kevin Swersky, Josh B. Tenenbaum, Hugo Larochelle, and Richard S. Zemel. Meta-learning for semi-supervised few-shot classification. In *International Conference on Learning Representations*, 2018.

[129] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2018.

[130] Boris Oreshkin, Pau Rodríguez López, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[131] Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Few-shot learning with embedded class models and shot-free meta training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 331–339, 2019.

[132] Ruibing Hou, Hong Chang, MA Bingpeng, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *Advances in Neural Information Processing Systems*, pages 4003–4014, 2019.

[133] Qianru Sun, Yaoyao Liu, Tat-Seng Chua, and Bernt Schiele. Meta-transfer learning for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 403–412, 2019.

[134] Bin Liu, Yue Cao, Yutong Lin, Qi Li, Zheng Zhang, Mingsheng Long, and Han Hu. Negative margin matters: Understanding margin in few-shot classification. In *European Conference on Computer Vision*, pages 438–455. Springer, 2020.

[135] Christian Simon, Piotr Koniusz, Richard Nock, and Mehrtash Harandi. Adaptive subspaces for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4136–4145, 2020.

[136] Nanyi Fei, Zhiwu Lu, Tao Xiang, and Songfang Huang. MELR: Meta-learning via modeling episode-level relationships for few-shot learning. In *International Conference on Learning Representations*, 2021.

[137] Yizhao Gao, Nanyi Fei, Guangzhen Liu, Zhiwu Lu, Tao Xiang, and Songfang Huang. Contrastive prototype learning with augmented embeddings for few-shot learning. *arXiv preprint arXiv:2101.09499*, 2021.

[138] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant risk minimization. *arXiv preprint arXiv:1907.02893*, 2019.

[139] Shiori Sagawa, Pang Wei Koh, Tatsunori B Hashimoto, and Percy Liang. Distributionally robust neural networks. In *International Conference on Learning Representations*, 2020.

[140] Faruk Ahmed, Yoshua Bengio, Harm Van Seijen, and Aaron Courville. Systematic generalisation with group invariant predictions. In *International Conference on Learning Representations*, 2020.

[141] Pierre Stock and Moustapha Cisse. Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases. In *Proceedings of the European Conference on Computer Vision*, pages 498–512, 2018.

[142] Soumya Suvra Ghosal and Yixuan Li. Are vision transformers robust to spurious correlations? *International Journal of Computer Vision*, 132(3):689–709, 2024.

[143] Yihao Xue, Ali Payani, Yu Yang, and Baharan Mirzasoleiman. Eliminating spurious correlations from pre-trained models via data mixing. *arXiv preprint arXiv:2305.14521*, 2023.

[144] Saeid Motiian, Quinn Jones, Seyed Iranmanesh, and Gianfranco Doretto. Few-shot adversarial domain adaptation. *Advances in Neural Information Processing Systems*, 30, 2017.

[145] Xiangyu Yue, Zangwei Zheng, Hari Prasanna Das, Kurt Keutzer, and Alberto Sangiovanni Vincentelli. Multi-source few-shot domain adaptation. *arXiv preprint arXiv:2109.12391*, 2021.

[146] Micah Goldblum, Liam Fowl, and Tom Goldstein. Adversarially robust few-shot learning: A meta-learning approach. *Advances in Neural Information Processing Systems*, 33:17886–17895, 2020.

[147] Junhao Dong, Yuan Wang, Jian-Huang Lai, and Xiaohua Xie. Improving adversarially robust few-shot image classification with generalizable representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9025–9034, 2022.

[148] Yue He, Zheyan Shen, and Peng Cui. Towards non-iid image classification: A dataset and baselines. *Pattern Recognition*, 110:107383, 2021.

[149] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3730–3738, 2015.

[150] Zhenguo Li, Fengwei Zhou, Fei Chen, and Huang Li. Meta-SGD: Learning to learn quickly for few shot learning. *ArXiv*, abs/1707.09835, 2017.

[151] Y. Wang, R. Girshick, M. Hebert, and B. Hariharan. Low-shot learning from imaginary data. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7278–7286, 2018.

[152] Sebastian Flennerhag, Andrei A. Rusu, Razvan Pascanu, Francesco Visin, Hujun Yin, and Raia Hadsell. Meta-learning with warped gradient descent. In *International Conference on Learning Representations*, 2020.

[153] Beier Zhu, Yulei Niu, Yucheng Han, Yue Wu, and Hanwang Zhang. Prompt-aligned gradient for prompt tuning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15659–15669, 2023.

[154] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Maple: Multi-modal prompt learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19113–19122, 2023.

[155] Hung-Yu Tseng, Hsin-Ying Lee, Jia-Bin Huang, and Ming-Hsuan Yang. Cross-domain few-shot classification via learned feature-wise transformation. In *International Conference on Learning Representations*, 2020.

[156] Xiaorong Qin, Xinhang Song, and Shuqiang Jiang. Bi-level meta-learning for few-shot domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15900–15910, 2023.

[157] Min Zhang, Haoxuan Li, Fei Wu, and Kun Kuang. Metacoco: A new few-shot classification benchmark with spurious correlation. In *The Twelfth International Conference on Learning Representations*, 2024.

[158] Sahil Singla, Besmira Nushi, Shital Shah, Ece Kamar, and Eric Horvitz. Understanding failures of deep networks via robust feature extraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12853–12862, 2021.

[159] Eric Wong, Shibani Santurkar, and Aleksander Madry. Leveraging sparse linear layers for debuggable deep networks. In *International Conference on Machine Learning*, pages 11205–11216. PMLR, 2021.

[160] Sahil Singla and Soheil Feizi. Salient imagenet: How to discover spurious features in deep learning? In *International Conference on Learning Representations*, 2022.

[161] Wenbin Li, Ziyi Wang, Xuesong Yang, Chuanqi Dong, Pinzhuo Tian, Tiexin Qin, Jing Huo, Yinghuan Shi, Lei Wang, Yang Gao, et al. Libfewshot: A comprehensive library for few-shot learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):14938–14955, 2023.

[162] Yu Yang, Besmira Nushi, Hamid Palangi, and Baharan Mirzasoleiman. Mitigating spurious correlations in multi-modal models during fine-tuning. *arXiv preprint arXiv:2304.03916*, 2023.

[163] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

[164] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *International Conference on Learning Representations*, 2020.

[165] Jaehoon Oh, Hyungjun Yoo, ChangHwan Kim, and Se-Young Yun. Boil: Towards representation change for few-shot learning. In *International Conference on Learning Representations*, 2020.

[166] Wenbin Li, Lei Wang, Jinglin Xu, Jing Huo, Yang Gao, and Jiebo Luo. Revisiting local descriptor based image-to-class measure for few-shot learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7260–7268, 2019.

[167] Dahyun Kang, Heeseung Kwon, Juhong Min, and Minsu Cho. Relational embedding for few-shot classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8822–8833, 2021.

[168] Leann Myers and Maria J Sirois. Spearman correlation coefficients, differences between. *Encyclopedia of Statistical Sciences*, 12, 2004.

[169] Shiori Sagawa, Aditi Raghunathan, Pang Wei Koh, and Percy Liang. An investigation of why overparameterization exacerbates spurious correlations. In *International Conference on Machine Learning*, pages 8346–8356. PMLR, 2020.

[170] Pavel Izmailov, Polina Kirichenko, Nate Gruver, and Andrew G Wilson. On feature learning in the presence of spurious correlations. *Advances in Neural Information Processing Systems*, 35:38516–38532, 2022.

[171] Daniel Levy, Yair Carmon, John C Duchi, and Aaron Sidford. Large-scale methods for distributionally robust optimization. *Advances in Neural Information Processing Systems*, 33: 8847–8860, 2020.

[172] Yoonho Lee, Huaxiu Yao, and Chelsea Finn. Diversify and disambiguate: Out-of-distribution robustness via disagreement. In *International Conference on Learning Representations*, 2022.

[173] Saeid Asgari, Aliasghar Khani, Fereshte Khani, Ali Gholami, Linh Tran, Ali Mahdavi Amiri, and Ghassan Hamarneh. Masktune: Mitigating spurious correlations by forcing to explore. *Advances in Neural Information Processing Systems*, 35:23284–23296, 2022.

[174] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2017.

[175] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

[176] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15262–15271, 2021.

[177] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

[178] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

[179] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. *arXiv preprint arXiv:1811.12231*, 2018.

[180] Remi Cadene, Corentin Dancette, Matthieu Cord, Devi Parikh, et al. Rubi: Reducing unimodal biases for visual question answering. *Advances in Neural Information Processing Systems*, 32, 2019.

[181] Hyojin Bahng, Sanghyuk Chun, Sangdoo Yun, Jaegul Choo, and Seong Joon Oh. Learning debiased representations with biased representations. In *International Conference on Machine Learning*, pages 528–539. PMLR, 2020.

[182] Tan Wang, Chang Zhou, Qianru Sun, and Hanwang Zhang. Causal attention for unbiased visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3091–3100, 2021.

[183] Nayeong Kim, Sehyun Hwang, Sungsoo Ahn, Jaesik Park, and Suha Kwak. Learning debiased classifier with biased committee. *Advances in Neural Information Processing Systems*, 35: 18403–18415, 2022.

[184] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European Conference on Computer Vision*, pages 3–19, 2018.

[185] Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Logan Engstrom, Brandon Tran, and Aleksander Madry. Adversarial examples are not bugs, they are features. *Advances in Neural Information Processing Systems*, 32, 2019.

[186] Haoyue Bai, Rui Sun, Lanqing Hong, Fengwei Zhou, Nanyang Ye, Han-Jia Ye, S-H Gary Chan, and Zhenguo Li. Decaug: Out-of-distribution generalization via decomposed feature representation and semantic augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6705–6713, 2021.

[187] Rishabh Tiwari and Pradeep Shenoy. Overcoming simplicity bias in deep networks using a feature sieve. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 34330–34343. PMLR, 23–29 Jul 2023.

[188] David Krueger, Ethan Caballero, Joern-Henrik Jacobsen, Amy Zhang, Jonathan Binas, Dinghuai Zhang, Remi Le Priol, and Aaron Courville. Out-of-distribution generalization via risk extrapolation (rex). In *International Conference on Machine Learning*, pages 5815–5826. PMLR, 2021.

[189] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[190] Rishabh Tiwari and Pradeep Shenoy. Overcoming simplicity bias in deep networks using a feature sieve. In *International Conference on Machine Learning*, pages 34330–34343. PMLR, 2023.

[191] Simone Bombari and Marco Mondelli. How spurious features are memorized: Precise analysis for random and ntk features. In *International Conference on Machine Learning*, 2024.

[192] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, Chris Chute, Henrik Marklund, Behzad Haghgoo, Robyn Ball, Katie Shpanskaya, et al. CheXpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 590–597, 2019.

[193] Vladimir N Vapnik. An overview of statistical learning theory. *IEEE Transactions on Neural Networks*, 1999.

[194] Adina Williams, Nikita Nangia, and Samuel R Bowman. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*, 2017.

[195] Daniel Borkan, Lucas Dixon, Jeffrey Sorensen, Nithum Thain, and Lucy Vasserman. Nuanced metrics for measuring unintended bias with real data for text classification. In *Companion Proceedings of the 2019 World Wide Web Conference*, pages 491–500, 2019.

[196] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[197] Yihe Deng, Yu Yang, Baharan Mirzasoleiman, and Quanquan Gu. Robust learning with progressive data expansion against spurious correlation. *Advances in Neural Information Processing Systems*, 36, 2024.

[198] Kirill Bykov, Mayukh Deb, Dennis Grinwald, Klaus Robert Muller, and Marina MC Höhne. Dora: Exploring outlier representations in deep neural networks. *Transactions on Machine Learning Research*, 2023.

[199] Haotian Ye, James Zou, and Linjun Zhang. Freeze then train: Towards provable representation learning under spurious correlations and feature noise. In *International Conference on Artificial Intelligence and Statistics*, pages 8968–8990. PMLR, 2023.

[200] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[201] Gaotang Li, Jiarui Liu, and Wei Hu. Bias amplification enhances minority group performance. *Transactions on Machine Learning Research*, 2024.

[202] Seonguk Seo, Joon-Young Lee, and Bohyung Han. Unsupervised learning of debiased representations with pseudo-attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16742–16751, 2022.

[203] Nimit Sohoni, Jared Dunnmon, Geoffrey Angus, Albert Gu, and Christopher Ré. No subclass left behind: Fine-grained robustness in coarse-grained classification problems. *Advances in Neural Information Processing Systems*, 33:19339–19352, 2020.

[204] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pages 4904–4916. PMLR, 2021.

[205] Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in Neural Information Processing Systems*, 34:9694–9705, 2021.

[206] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. In *International Conference on Learning Representations*, 2022.

[207] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022.

[208] Chenyu You, Yifei Mint, Weicheng Dai, Jasjeet S Sekhon, Lawrence Staib, and James S Duncan. Calibrating multi-modal representations: A pursuit of group robustness without annotations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26140–26150. IEEE, 2024.

[209] Sepehr Dehdashtian, Lan Wang, and Vishnu Boddeti. Fairerclip: Debiasing clip's zero-shot predictions using functions in rkhss. In *International Conference on Learning Representations*, 2024.

[210] Jie Zhang, Xiaosong Ma, Song Guo, Peng Li, Wenchao Xu, Xueyang Tang, and Zicong Hong. Amend to alignment: Decoupled prompt tuning for mitigating spurious correlation in vision-language models. In *International Conference on Machine Learning*, 2024.

[211] Kaiyang Zhou, Yongxin Yang, Timothy Hospedales, and Tao Xiang. Deep domain-adversarial image generation for domain generalisation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

[212] Chen Fang, Ye Xu, and Daniel N Rockmore. Unbiased metric learning: On the utilization of multiple datasets and web images for softening bias. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1657–1664, 2013.

[213] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88:303–338, 2010.

[214] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International Journal of Computer Vision*, 77:157–173, 2008.

[215] Monika Bansal, Munish Kumar, Monika Sachdeva, and Ajay Mittal. Transfer learning for image classification using VGG19: Caltech-101 image data set. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–12, 2023.

[216] Myung Jin Choi, Joseph J Lim, Antonio Torralba, and Alan S Willsky. Exploiting hierarchical context on a large database of object categories. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 129–136. IEEE, 2010.

[217] Zhongzhi Chen, Guang Liu, Bo-Wen Zhang, Qinghong Yang, and Ledell Wu. Altclip: Altering the language encoder in clip for extended language capabilities. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8666–8682, 2023.

[218] Yu Yuan, Lili Zhao, Kai Zhang, Guangting Zheng, and Qi Liu. Do llms overcome shortcut learning? an evaluation of shortcut challenges in large language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 12188–12200, 2024.

[219] Ruixiang Tang, Dehan Kong, Longtao Huang, and Hui Xue. Large language models can be lazy learners: Analyze shortcuts in in-context learning. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4645–4657, 2023.

[220] Zechen Sun, Yisheng Xiao, Juntao Li, Yixin Ji, Wenliang Chen, and Min Zhang. Exploring and mitigating shortcut learning for generative large language models. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 6883–6893, 2024.

[221] Yuqing Zhou, Ruixiang Tang, Ziyu Yao, and Ziwei Zhu. Navigating the shortcut maze: A comprehensive analysis of shortcut learning in text classification by language models. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 2586–2614, 2024.

[222] Wenqian Ye, Guangtao Zheng, Yunsheng Ma, Xu Cao, Bolin Lai, James Matthew Rehg, and Aidong Zhang. Mm-spubench: Towards better understanding of spurious biases in multimodal llms. In *Workshop on Responsibly Building the Next Generation of Multimodal Foundational Models*, 2024.

[223] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

[224] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[225] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[226] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[227] Jin-Woo Seo, Hong-Gyu Jung, and Seong-Whan Lee. Self-augmentation: Generalizing deep networks to unseen classes for few-shot learning. *Neural Networks*, 138:140–149, 2021.

[228] Yuning Lu, Liangjian Wen, Jianzhuang Liu, Yajing Liu, and Xinmei Tian. Self-supervision can be a good few-shot learner. In *European Conference on Computer Vision*, pages 740–758. Springer, 2022.

[229] Huiwon Jang, Hankook Lee, and Jinwoo Shin. Unsupervised meta-learning via few-shot pseudo-supervised contrastive learning. In *International Conference on Learning Representations*, 2023.

[230] Stylianos Poulakakis-Daktylidis and Hadi Jamali-Rad. Beclr: Batch enhanced contrastive few-shot learning. In *International Conference on Learning Representations*, 2024.

[231] Shell Xu Hu, Pablo Garcia Moreno, Yang Xiao, Xi Shen, Guillaume Obozinski, Neil Lawrence, and Andreas Damianou. Empirical bayes transductive meta-learning with synthetic gradients. In *International Conference on Learning Representations*, 2020.

[232] Georg Heinze, Christine Wallisch, and Daniela Dunkler. Variable selection–a review and recommendations for the practicing statistician. *Biometrical Journal*, 60(3):431–449, 2018.

Appendix

A.1 AdvST: Adversarial Learning with Semantics Transformations

A.1.1 Proof of Proposition 1

We first show that the inner maximization in Equation (3.10) satisfies strong duality condition [98] and that the dual problem involves optimization over a one-dimensional dual variable. Lemma A.1 gives the useful result for *any* distribution $\mathcal{Q}$ satisfying $W_c(Q, P) \leq \delta$. We omit the proof since it is a minor adaptation of Proposition 1 in [98].

Lemma A.1. Let $\ell : \Theta \times \mathcal{X} \times \mathcal{Y} \to [0, \infty)$ denote the loss function which is upper semi-continuous and integrable. The transportation cost function $c : \Xi \times \Xi \to [0, \infty)$ with $\Xi = \mathcal{X} \times \mathcal{Y}$ is a lower semi-continuous function satisfying $c(\xi, \xi) = 0$ for $\xi \in \Xi$. For any distribution Q and any $\delta \geq 0$, let $s_\lambda(\theta; (x, y)) = \sup_{\xi \in \Xi}(\ell(\theta; \xi) - \lambda c(\xi, (x, y)))$. Then, for any given P and $\delta > 0$, it holds that

$$\sup_{Q \in \mathcal{Q}} \mathbb{E}_Q[\ell(\theta; x, y)] = \inf_{\lambda \geq 0} \{\lambda\delta + \mathbb{E}_P[s_\lambda(\theta; (x, y))]\} \tag{A.1}$$

and for any $\lambda \geq 0$, we have

$$\sup_{Q \in \mathcal{Q}} \{\mathbb{E}_Q[\ell(\theta; x, y)] - \lambda W_c(Q, P)\} = \mathbb{E}_P[s_\lambda(\theta; (x, y))]. \tag{A.2}$$

where $\mathcal{Q} = \{Q : W_c(Q, P) \leq \delta\}$.

Note that in our AdvST framework, the distribution Q is semantics-induced and is defined as a mixture of M distributions as shown in Equation (3.6). To get a tractable learning objective through Lemma A.1, let $Q_i = \int_\xi p(\xi' | \tau_i, \xi, \omega_i) dP$, and we have the following

$$\begin{aligned}
&\mathbb{E}_{Q_\psi}[\ell(\theta; x, y)] - \lambda W_c(Q_\psi, P) && \text{(A.3)}\\
&= \mathbb{E}_{\tau_i \sim G}\big[\mathbb{E}_{Q_i}\big(\ell(\theta; x, y)\big)\big] - \lambda \mathbb{E}_{\tau_i \sim G}[W_c(Q_i, P)] && \text{(A.4)}\\
&= \mathbb{E}_{\tau_i \sim G}\big[\mathbb{E}_{Q_i}\big(\ell(\theta; x, y)\big) - \lambda W_c(Q_i, P)\big] && \text{(A.5)}\\
&= \mathbb{E}_{\tau_i \sim G}\mathbb{E}_P\big[\sup_{\xi \in \Xi_i}(\ell(\theta; \xi) - \lambda c_\theta(\xi, (x, y)))\big], && \text{(A.6)}
\end{aligned}$$

where Equation (A.6) is the result of applying Lemma A.1 to the inner term of Equation (A.5), and $\Xi_i = \{(x', y) | x' = \tau_i(x; \omega_i), \xi \in \Xi_0, \omega_i \subset \psi\}$ is the support of Q_i with Ξ_0 being the support of P. Finding the supreme over Ξ_i that maximizes the inner term in Equation (A.6) is equivalent to finding ω_i.

A.1.2 Experimental Details

Semantics Transformations. The semantics transformations used in the experiments are constructed from the 12 standard image transformations described in Table A.1.1. These standard

transformations are designed with domain knowledge about image transformations. Each standard transformation manipulates a particular kind of semantics of an image with a few learnable parameters controlling the transformation magnitude. For example, HSV perturbs an image in the HSV color space with three learnable parameters, and Translate changes an object's position in an image with two learnable parameters. Some standard transformations do not have any learnable parameters because they do not need any parameters, such as Equalize, or they are just non-differentiable functions, such as Posterize. For the latter case, we randomly sample values for parameters from their valid ranges and treat the corresponding function as an identity function during back-propagation. We design a semantics transformation as the concatenation of $L_{\max}$ ($L_{\max} = 3$ in the experiments) standard transformations to manipulate multiple kinds of semantics in an image.

Standard transformations	Description	Number of Parameters
HSV	Perturb in the HSV color space	3
Contrast	Perturb the contrast of an image	1
Invert	Invert pixel values at a given threshold	1
Sharpness	Perturb the sharpness of an image	1
Shear	Shear an image in horizontal and vertial directions	2
Translate	Move an image in horizontal and vertial directions	2
Rotate	Rotate an image	1
Scale	Change the size of an image	1
Solarize	Reverse the tone of an image	1
Equalize	Improve global contrast of an image via equalization	None
Posterize	Reduce the number of bits for each color channel	None
Cutout	Produce occlusions in an image	None

Table A.1.1: Standard data augmentations used in experiments.

Contrastive Regularizer. The regularizer uses a contrastive loss to facilitate learning domain-invariant features from samples in $\mathcal{D}$, which stores generated samples. The loss ensures that samples with the same label are moved close to each other, and those with different labels are moved away from each other. Concretely, we denote the index of a sample in a batch as i, the set of all indexes as $\mathcal{I}_{\mathcal{B}}$, the index set excluding i as $\mathcal{I}_{\mathcal{B}}(i) = \mathcal{I}_{\mathcal{B}} \backslash \{i\}$, and the indexes of samples with label y_i as $\mathcal{P}(i) = \{p \in \mathcal{I}_{\mathcal{B}}(i) | y_i = y_p\}$. Then, our contrastive regularizer is given as follows:

$$\ell_{sc}(\theta; \mathcal{B}) = \sum_{i \in \mathcal{I}_{\mathcal{B}}} \frac{-1}{|\mathcal{P}(i)|} \sum_{p \in \mathcal{P}(i)} \log \frac{\exp(u_i^T u_p)}{\sum_{a \in \mathcal{I}_{\mathcal{B}}(i)} \exp(u_i^T u_a)}, \tag{A.7}$$

where $u_i = \phi(v_i)$ is the projection of the embedding v_i of the input x_i with $v_i = f_\theta(x_i)$, and ϕ is a projection function. Choices of ϕ are given in the experimental details for specific datasets.

Entropy Regularizer. The regularizer uses output entropy to penalize overly confident predictions and to learn good decision boundaries that benefit model generalization. Specifically, it calculates the average output entropy for a batch of samples $\mathcal{B}$ as follows

$$\ell_{ent}(\theta; \mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{I}_{\mathcal{B}}} \sum_{j=1}^{C} -p_{ij} \log p_{ij}, \tag{A.8}$$

where $\mathcal{I}_{\mathcal{B}}$ is the index set of samples in $\mathcal{B}$, C is the number of classes/outputs, p_{ij} is the jth element of $p_i = \text{softmax}(f_\theta(x_i))$. By definition, a confident prediction, which has a very high value on a particular class, has a low output entropy; while a less confident prediction has a larger output entropy. For a single sample, we have $l_{ent}(\theta; x_i, y_i) = \sum_{j=1}^{C} -p_{ij} \log p_{ij}$.

The regularizer used in the minimization (Equation (3.4)) $\ell_{reg}(\theta; \mathcal{B})$ is defined as the combination of a contrastive and entropy loss terms, i.e., $\ell_{reg}(\theta; \mathcal{B}) = \ell_{sc}(\theta; \mathcal{B}) - \eta \ell_{ent}(\theta; \mathcal{B})$, where η is a nonnegative regularization parameter.

Experiments on Digits. The backbone network (i.e., the whole model except the last classification layer) has two 5×5 convolutional layers. The two layers have 64 and 128 channels, respectively. Each convolutional layer is followed by a 2×2 max pooling layer. After the two convolutional layers, there are two fully-connected layers with a size of 1024. The classification layer is a linear layer with 1024 inputs and 10 outputs. To calculate the contrastive loss, we design the projection function ϕ in Equation (A.7) as a linear layer (1024 inputs and 128 outputs) followed by a normalization layer. Specifically, ϕ first projects the embeddings from the backbone network and then normalizes the projections to have a unit length. We set $\eta = 10$ and $\epsilon = 10$. To speed up training, we adopt early stopping in the maximization procedure, i.e., if the difference between the previous loss and current loss is smaller than 0.1, then we exit the maximization procedure.

Experiments on PACS. We use a ResNet-18 [200] as the backbone network. To facilitate knowledge transfer, we pre-train the network on ImageNet and fix the batch normalization statistics of all its batch normalization layers during fine-tuning. The classification layer is a linear layer with 512 inputs and 7 outputs. To calculate the contrastive loss, we design the projection function ϕ in Equation (A.7) as a linear layer with 512 inputs and 128 outputs followed by a normalization layer. We set $\eta = 0.1$ and $\epsilon = 1$. We adopt the same early stop technique in the maximization procedure as in the Digits experiments.

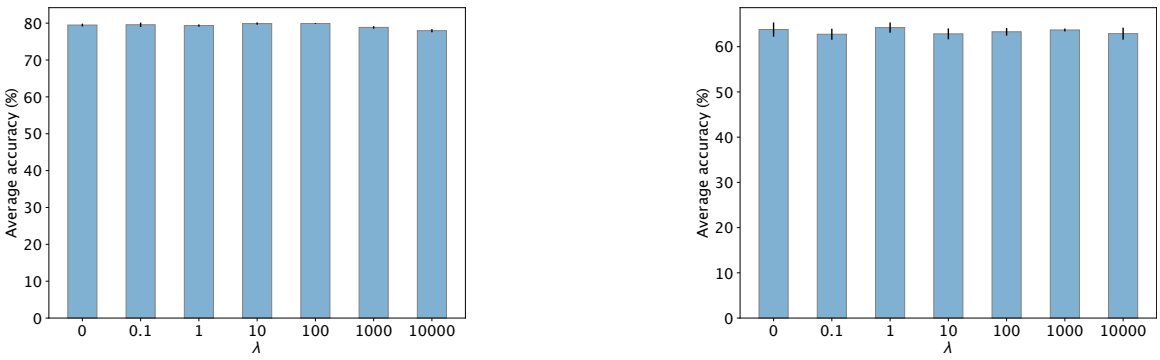
Experiments on DomainNet. We use a ResNet-18 [200] as the backbone network. To speed up training, we use a ResNet-18 network pre-trained on ImageNet as the initialization for the backbone network. The classification layer is a linear layer with 512 inputs and 345 outputs. To calculate the contrastive loss, we design the projection function ϕ in Equation (A.7) as a linear layer with 512 inputs and 128 outputs followed by a normalization layer. We set $\eta = 10$ and $\epsilon = 10$. We adopt the same early stop technique in the maximization procedure as in the PACS experiments.

Method for Obtaining Figure 3.1. We first train a model using ERM on the MNIST domain. The backbone (all layers before the last classification layer) of the trained model will be used to get embeddings of all the sampled images. Then, we sample 1000 images from the source domain MNIST, the four target domains, and the generated images obtained by a DRO-based method, respectively. We get the embeddings of all the samples using the backbone network. To visualize the embeddings, we use UMAP [223] with `n_neighbors=100` and `min_dist=0.9` to get two-dimensional representations of the embeddings.

A.1.3 Additional Experimental Results

Sensitivity Analysis on λ

The parameter λ in Equation (3.1) controls the size of the uncertainty set. A small λ allows the uncertainty set to have distributions with large distributional shifts from the source. With semantics transformations, the average performance of a AdvST-trained model does not change too much under different values of λ on the Digits (a maximum drop of 1.94% in Figure A.1.1 (a)) and the PACS (a maximum drop of 1.47% in Figure A.1.1 (b)) datasets. In practice, using a small λ, e.g., $\lambda = 1$ or $\lambda = 10$, works well.



(a) Digits (b) PACS

Figure A.1.1: Sensitivity analysis on λ. We train models with AdvST under different values of λ. For each λ, we report average classification accuracy (blue bars) and its standard deviation (vertical black bars) over all target domains for each dataset.

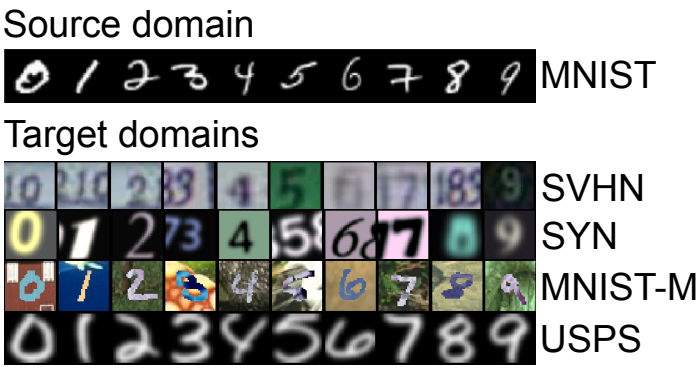


Figure A.1.2: Examples of the Digits dataset.

Semantics Transformations

We analyze how the 12 standard transformations, which are used to construct semantics transformations, affect the generalization performance of a model on target domains. We adopt the leave-one-out strategy to evaluate the contribution of each standard transformation. Specifically, we remove only one standard transformation at a time and train the model using the semantics transformations constructed with the remaining 11 standard transformations. Then, we calculate the difference in classification accuracy on target domains between the model and the model trained using all 12 standard transformations.

We use AdvST-ME and the Digits dataset in this experiment and obtain the heatmap of classification accuracy change in Figure A.1.3. We observe that removing any standard transformation results in a drop in performance. In particular, without `Translate`, the model has the most drop in average accuracy due to the significant performance drop in the SYN domain, indicating that translational invariance is important for generalizing to the SYN domain. However, removing `Translate` benefits generalizing to the MNIST-M domain. The contradictory effect of `Translate` in MNIST-M and SYN domains explains the performance tradeoff between AdvST and AdvST-ME on the two domains observed in Table 3.3. We also observe a similar contradictory effect of `Scale` in SVHN and USPS domains and the corresponding performance tradeoff on the two domains in Table 3.3. Moreover, we identify that `Contrast`, `HSV`, `Translate`, and `Scale` are the most important standard transformations for generalizing to the SVHN, MNIST-M, SYN, and USPS domains, respectively. For example, Scale is beneficial for generalizing to the USPS domain since digits in this domain are enlarged compared to those in the MNIST domain (see examples in Figure A.1.2).

The above analysis highlights the key advantage of our method: semantics transformations can bring domain knowledge, such as translational invariance or scale invariance, that benefits generalization on unseen target domains. Adding more semantics transformations could benefit generalization on target domains; however, it may also bring undesired semantics transformations that have adverse effects on specific target domains.

In-Distribution Accuracy

We show the in-distribution accuracy comparison between ERM and our methods on three datasets in Table A.1.2 below. Our approach does not hurt the nominal accuracy and slightly improves it.

Method	MNIST	Photo (PACS)	Real (DomainNet)
ERM	98.8	98.5	76.0
AdvST (ours)	**99.0**	99.7	**76.7**
AdvST-ME (ours)	98.9	**99.9**	76.5

Table A.1.2: In-distribution accuracy comparison. Models are trained and tested in the same domain.

Figure A.1.3: Heatmap of classification accuracy change on the four target domains and average accuracy change after removing one standard transformation (shown as column name).

Figure A.1.4: Visualization of the images generated by AdvST for the MNIST domain.

Visualization of Generated Samples

We visualize the images generated by AdvST. Figure A.1.4 shows the images generated for the MNIST domain. Figure A.1.5 shows the images generated for the four domains in the PACS dataset. Figure A.1.6 shows the images generated for the Real domain in the DomainNet dataset. From the three figures, we observe diverse variations in the generated images. For example, images from the Sketch domain in the PACS dataset all have white background and black strokes, while the generated images have various background and stroke colors.

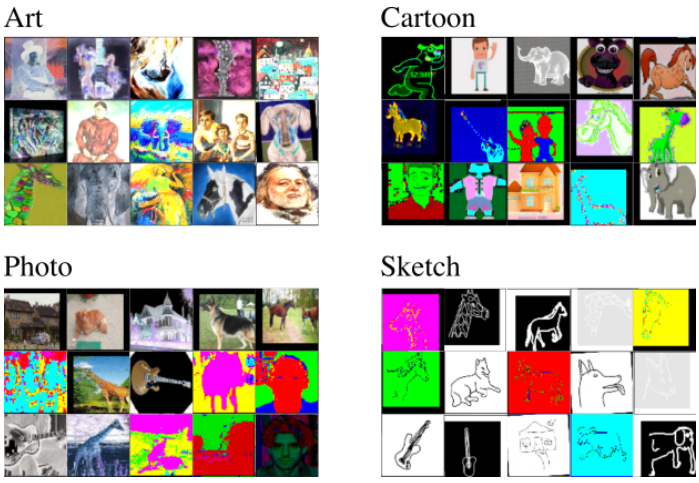


Figure A.1.5: Visualization of the images generated by AdvST for the four domains in the PACS dataset.

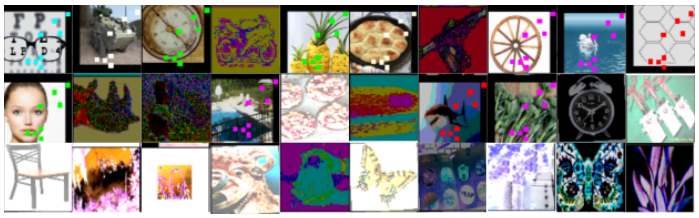


Figure A.1.6: Visualization of the images generated by AdvST for the Real domain in the DomainNet dataset.

Experiments on OfficeHome

We evaluate our two implementations, AdvST and AdvST-ME on OfficeHome [224] which contains four domains (Art, Clipart, Product,and Real) with 65 classes. This is one of the canonical domain adaptation/generalization benchmarks. We use a ResNet-18 [200] as the backbone network which is pre-trained on ImageNet. The classification layer is a linear layer with 512 inputs and 65 outputs. To calculate the contrastive loss, we design the projection function ϕ in Equation (A.7) as a linear layer with 512 inputs and 128 outputs followed by a normalization layer. We set $\eta = 0.01$ and $\epsilon = 1$. Other settings are the same as in the PACS experiments. The SDG results are shown in Table A.1.3.

Source		ADA	ME-ADA	L2D	AdvST	AdvST-ME
OfficeHome	Art	48.3	49.5	52.1	**52.1±0.5**	51.6±0.3
	Clipart	46.1	46.9	51.2	**52.3±0.3**	52.0±0.3
	Product	43.9	44.3	49.4	**49.6±0.3**	49.2±0.3
	Real	53.6	54.6	58.2	**60.1±0.2**	59.9±0.4
	Avg.	48.0	48.8	52.7	**53.5±0.1**	53.2±0.2

Table A.1.3: Single domain generalization results on OfficeHome. We report the average classification accuracy over the remaining domains when one domain is used as the source domain.

A.2 Learning to Learn Task Transformations for Improved Few-Shot Classification

A.2.1 Meta-Learning Settings

Meta-Learning Algorithms. We use three metric-based meta-learning algorithms and one gradient-based meta-learning algorithm in the experiments. A metric-based algorithm first uses a feature extractor to get the embedding of each sample, and then learns a specifically designed classification head to classify samples based on their embeddings. A gradient-based algorithm uses gradient descent to learn a new classifier to classify samples. The four algorithms are described as follows:

- ProtoNet[12] is a metric-based meta-learning algorithm. It first calculates a class prototype from the support set of a task by averaging embeddings of the samples with the same label. Then, it searches the nearest class prototype for each sample in the query set and predicts the sample to have the same class label as the nearest class prototype.
- R2D2 [13] is the abbreviation for Ridge Regression Differentiable Discriminator, and it is a metric-based meta-learning algorithm. It adopts ridge regression as the classification head which is differentiable and has closed form solutions.
- MetaOptNet [14] is a metric-based meta-learning algorithm. It adopts an SVM classifier as the classification head in few-shot classification.
- MAML [22] is a gradient-based meta-learning algorithm. It aims to learn a model that can quickly generalize to new concepts with a few gradient descent steps.

Meta-Model Architectures. We call the meta-model architectures and backbones interchangeably. Essentially, they are feature extractors that convert inputs to their vector representations. In the experiments, we use the ResNet-12 backbone adopted in [130] and the four-layer convolutional backbone (CNN64) with 64 filters in each layer adopted in [12].

A.2.2 Implementation Details

Meta-Learning. We follow the implementations in [27] to implement the above meta-learning algorithms and the backbones. Similar to [130], we adopt a learnable scaler to scale the outputs of each of the classification heads from the three metric-based meta-learning algorithms. For MAML, we adopt its first-order approximation in the experiments to achieve a good tradeoff between computational complexity and few-shot classification performance. Moreover, we use a 5-step gradient descent (10 steps in evaluation) with a learning rate of 0.01 in the inner loop of MAML, an Adam optimizer with a learning rate of 0.001 in the outer loop of MAML, and a cosine annealing scheduler with the minimum learning rate of 1×10^{-5} to control the learning rate in the outer loop.

Differentiable Image Operations. The image operations/functions used in our method are listed in Table A.2.1. Each function has its description in the "Description" column and its own transformation magnitude listed in the "Magnitude" column. We implement each image operation as a differentiable function in the sense that its output is differentiable with respect to the input. However, for functions that are not inherently differentiable, e.g., the `posterize` function, we use the straight-through estimator [74, 225] to approximate the corresponding gradient. Specifically, given an operation $O(x) = g(x; m)$ with an input x and a transformation magnitude m, we implement the operation as $O(x) = \text{StopGrad}(g(x; m) - x - m) + x + m$, where StopGrad is a stop gradient operation and treats its operand as a constant in the backward pass. We find that this method is simple and works well in our experiments.

L2TT Algorithm. The details of L2TT are shown in Algorithm 3.

Data Augmentation Methods. We describe the two data augmentation methods used in our experiments: AutoAugment [71] and MetaDA [27].

Function	Magnitude	Description	Function	Magnitude	Description
Shift_r	[0,1]	Change red channel value	Shift_x	[0,0.5]	Shift horizontally
Shift_g	[0,1]	Change green channel value	Shift_y	[0,0.5]	Shift vertically
Shift_b	[0,1]	Change blue channel value	Scale	[0.1,10]	Scale images
Brightness	[-1,1]	Change brightness	Self_mix	None	Self-mix up an image
Contrast	[1,10]	Change contrast	Posterize	[0,8]	Reduce number of bits for each color channel
Solarize	[0,1]	Invert pixels under a threshold value	Equalize	None	Equalize the histogram of an image
Hflip	None	Horizontally flip	Cutout_fixed1	[0,1]	Cutout 8 regions with a random size in an image
Vflip	None	Vertically flip	Cutout_fixed2	[0,8]	Cutout random number of regions with fixed size
Rotate	[-180,180]	Rotate an image	Sample_pairing	[0,1.0]	Combine two different images with random weights

Table A.2.1: Image operations used in our method.

Algorithm 3 L2TT

Input: training tasks distribution $p(\mathcal{T})$, parameters of the task transformation distribution ω, a meta-learning algorithm $\mathcal{E} = \{\mathcal{A}, \mathcal{B}\}$, a meta-model f_θ
Output: θ

//Outer loop
while in algorithm $\mathcal{B}$ **do**
  Randomly construct a task $\mathcal{T}$ such that $\mathcal{T} \sim p(\mathcal{T})$
  Sample τ from $p_\tau(x;\omega)$
  $\mathcal{T}' = \{x' | x' = \tau(x), x \in \mathcal{T}\}$
  Split $\mathcal{T}'$ such that $\mathcal{T}' = \{\mathcal{S}', \mathcal{Q}'\}$
  //Inner loop
  $f_{\hat{\theta}} = \mathcal{A}(f_\theta, \mathcal{S}')$
  //Task loss
  $\mathcal{L}(\mathcal{T}') = \frac{1}{n_Q} \sum_{(x,y)\in\mathcal{Q}'} \ell(f_{\hat{\theta}}(x), y)$
  Use $\mathcal{B}$ to jointly optimize θ and ω with respect to $\mathcal{L}(\mathcal{T}')$.
end while
return θ, ω

- **AutoAugment.** There are 25 policies optimized for a selected dataset. Each policy is a sequence of transformations. In AutoAugment, each policy is designed to have 2 transformations. For meta-models trained on the CIFAR-FS dataset, we use the set of policies optimized for CIFAR-10. For meta-models trained on the miniImageNet dataset, we use the set of policies optimized for ImageNet [226]. To use AutoAugment in training, we first apply the sequential transformation `RandomCrop`→`RandomHorizontalFlip` to an image. Then, we randomly sample a policy from the selected set of AutoAugment policies and apply it to the transformed image. Finally, we apply Cutout [17] with 16x16 pixels [71] to the image obtained from the previous step.

- **MetaDA.** There are four basic augmentation functions in MetaDA: SelfMix, Cutmix, random erase, and rotation. SelfMix [227] replaces a patch of an image with another patch from the same image. Cutmix [126] cuts an image patch from one image and pastes the patch to another image to construct a mixed image with the ground truth labels of the two original images mixed proportionally to reflect the area of the image patch in the mixed image. Rotation rotates an

image with a degree which is a multiple of 90. Random erase (RE) randomly erases patches from an image. For each task, the augmentation functions can be applied to the support set (S), the query set (Q), or the whole task (T). We use the large-size pool defined in [27] as the set of augmentation policies. These policies are: Q-cutmix, Q-RE, S-RE, T-Rotation, Q-cutmix→T-rotation, Q-RE→T-Rotation, Q-RE→S-RE, Q-cutmix→Q-RE, Q-cutmix→S-RE.

Note that MetaDA includes Q-cutmix (QC) in the set of augmentation policies. Since QC changes the learning objective from minimizing classification loss to additionally predicting the area of an image patch in a mixed image, we cannot directly compare MetaDA with AutoAugment. For fair comparison, we append QC to the end of the policy sampled from AutoAugment.

Hyperparameter Settings. In Table A.2.2, we give the settings for the two important hyperparameters L and ϵ used in different meta-learning settings listed in Table 3.6. We select the optimal values for each meta-learning setting based on the validation performance. For metric-based meta-learning algorithms, we observe that L is larger for the settings with a ResNet-12 backbone than the one with a CNN64 backbone. In general, a task transformation with a large L indicates large variations in the images of a transformed task, and the transformed task can be considered as a "hard" task. Moreover, we observe that for MAML with the same CNN64 backbone, the optimal L is even larger than those for the metric-based algorithms with the deeper ResNet-12 backbone. This indicates that MAML is easier to suffer from overfitting when compared with the three metric-based meta-learning algorithms. For training MAML on the miniImageNet dataset, we set $\epsilon = \infty$ which means uniformly sampling task transformations. This is because the task transformation distribution $p_\tau(\tau; \omega)$ tends to concentrate its probability mass on certain trivial task transformations that make little change to the images in a task, resulting in inferior performance. Hence, we set ϵ to infinity to circumvent this problem in this meta-learning setting.

For the experiments in Table 3.7, because of the introduction of QC, we find that setting $L = 1$ works well for all the meta-learning settings.

Architecture	Meta-learning algorithm	CIFAR		miniImageNet	
		L	ϵ	L	ϵ
ResNet-12	R2D2	3	20	3	20
ResNet-12	ProtoNet	3	20	3	20
ResNet-12	MetaOptNet	3	20	3	20
CNN64	ProtoNet	2	20	2	20
CNN64	MAML	5	40	4	∞

Table A.2.2: The values of L and ϵ used in Table 3.6.

A.2.3 Additional Results

Meta-Learned Task Transformations

At the end of meta-training, we show the most probable task transformation with different lengths for ProtoNet-CNN64 and ProtoNet-ResNet12 meta-trained on the CIFAR-FS dataset. The results are shown in Table A.2.3. These task transformations reflect the difference between different meta-learning settings. For the shallow network CNN64, the meta-learned task transformations are not as diverse as the ones meta-learned with the deep network ResNet-12. For example, when $L = 1$, the maximum probability of sampling an operation is 0.09 in the ProtoNet-CNN64 setting, while the value decreases to 0.07 in the ProtoNet-ResNet12 setting. Since the probabilities of all the operations add up to 1, this means that operations other than Equalize are more likely to be sampled in the ProtoNet-ResNet12 setting than those in the ProtoNet-CNN64 setting. We observe the same trend for other values of L. With more operations involved in a task transformation, we can create "harder" tasks with more variations in the images of the tasks. From the meta-learned most probable task transformations, we can conclude that in order to obtain a well-trained meta-model, we need

"harder" tasks for meta-training in the ProtoNet-ResNet12 setting than in the ProtoNet-CNN64 setting.

Architecture	Meta-learning algorithm	L	Task Transformation ($\cdots \rightarrow$Operation(Probability, Magnitude)$\rightarrow \cdots$)
CNN64	ProtoNet	1	Equalize(0.09,N/A)
		2	Equalize(0.10,N/A)→RandomContrast(0.06,0.24)
		3	Equalize(0.10,N/A)→RandomContrast(0.06,0.32)→Equalize(0.06,N/A)
ResNet-12	ProtoNet	1	Equalize(0.07,0.50)
		2	Equalize(0.08,N/A)→Posterize(0.06,0.50)
		3	Equalize(0.09,N/A)→Posterize(0.06,0.44)→Posterize(0.06,0.50)

Table A.2.3: Meta-learned task transformations with different lengths. We describe a task transformation as a sequence of operations. Each operation has a probability and a magnitude. For operations that do not have magnitudes, such as Equalize and Hflip, we set their magnitudes to "N/A".

Performance Comparison on 10-way Tasks

We evaluate how well our method (L2TT-QC) generalizes to a high-way setting by comparing the performance of L2TT-QC, AutoAugment-QC, and MetaDA on 10-way tasks generated from the CIFAR-FS and miniImageNet datasets. We exclude the results of MAML since it learns a fixed-way classifier during meta-training, and it cannot be directly evaluated in this setting. All the models are meta-trained on 5-way 5-shot tasks. The few-shot classification accuracy results are shown in Table A.2.4. We observe that our method generalizes well to this challenging setting and achieves the best performance in all the testing cases.

Architecture	Meta-learning algorithm	Data augmentation method	CIFAR-FS		miniImageNet	
			1-shot	5-shot	1-shot	5-shot
ResNet-12	R2D2	AutoAugment-QC	63.01±0.08	76.66±0.06	48.16±0.07	65.11±0.06
		MetaDA	62.46±0.08	77.27±0.06	47.52±0.07	65.33±0.05
		L2TT-QC	**63.78±0.08**	**77.36±0.06**	**51.07±0.07**	**68.64±0.05**
ResNet-12	ProtoNet	AutoAugment-QC	62.08±0.08	76.33±0.06	45.25±0.07	63.12±0.06
		MetaDA	61.74±0.08	77.01±0.06	44.68±0.07	64.40±0.06
		L2TT-QC	**62.66±0.08**	**77.22±0.06**	**46.42±0.07**	**64.79±0.06**
ResNet-12	MetaOptNet	AutoAugment-QC	59.41±0.08	76.20±0.06	48.41±0.07	65.73±0.05
		MetaDA	59.54±0.08	77.00±0.06	47.23±0.07	66.08±0.05
		L2TT-QC	**62.61±0.08**	**77.48±0.06**	**48.70±0.07**	**67.84±0.05**
CNN64	ProtoNet	AutoAugment-QC	48.35±0.08	67.55±0.06	34.34±0.06	54.45±0.06
		MetaDA	48.74±0.08	69.09±0.06	34.25±0.06	56.75±0.06
		L2TT-QC	**49.28±0.08**	**69.11±0.06**	**34.90±0.06**	**57.46±0.05**

Table A.2.4: Few-shot classification accuracy comparison on 10-way tasks.

A.2.4 Visualization of Transformed Images

Figure A.2.1 shows three sets of images obtained by applying three task transformations with variable numbers of image operations to the original images in a sampled task. With more image operations, the obtained images show more variations than those obtained with less image operations.

A.2.5 Visualization of Task Embeddings

We visualize the original tasks and their transformed versions using t-SNE. We obtain each task embedding by averaging all the embeddings of the images in the same task. We sample 2000 original tasks from the training split of the CIFAR-FS dataset. For each task, we transform it with a task transformation sampled from $p_\tau(\tau;\omega)$. We use the backbone network meta-trained in the

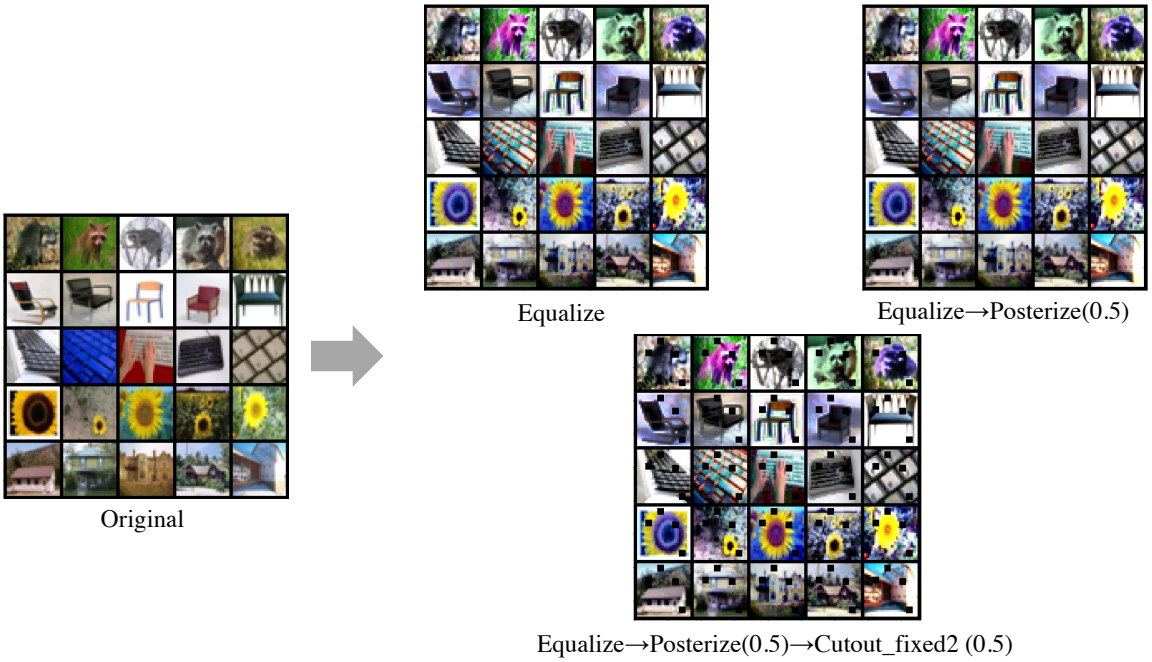


Figure A.2.1: Visualization of three task transformations with variable numbers of image operations.

ProtoNet-CNN64 setting on the CIFAR-FS dataset to extract image embeddings. The visualization is shown in Figure A.2.2. We see that the meta-learned task transformations can generate not only tasks that are close to the original ones, but also tasks that are distant in the embedding space.

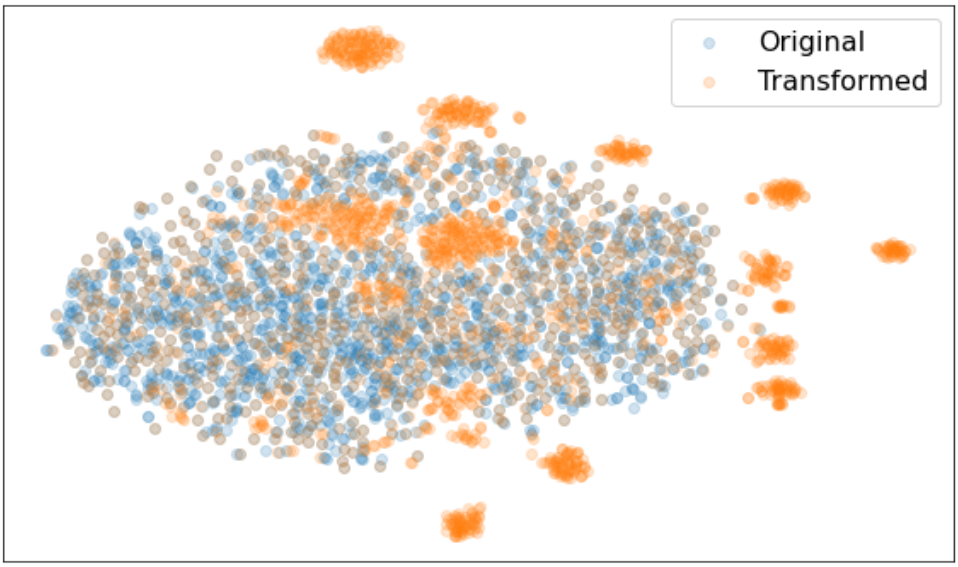


Figure A.2.2: Visualization of original and transformed tasks via t-SNE.

A.3 Benchmarking Spurious Bias Using Vision-Language Models

The appendix in this section is organized as follows: we introduce the ten FSC algorithms adopted in the paper in Chapter A.3.1. Then, we give the details of the evaluation metrics used in the main paper in Chapter A.3.2. In Chapter A.3.3, we show statistics of the datasets used in this paper along with detailed training settings. In Chapter A.3.4, we analyze different methods for constructing the support and query sets in a FewSTAB task (Chapter A.3.4), show the scatter plots of wAcc-A versus Acc from all the training settings (Chapter A.3.4), present more results on the effectiveness of FewSTAB (Chapter A.3.4), and demonstrate the robustness of FewSTAB with different VLMs (Chapter A.3.4). Finally, we give more examples of the tasks constructed by FewSTAB in Chapter A.3.5.

A.3.1 Few-Shot Classification Algorithms

ANIL (Almost No Inner Loop) [164]: ANIL is an optimization-based meta-learning method and follows a similar optimization procedure to MAML [22] whose few-shot adaptation algorithm $\mathcal{O}$ is to update the whole model using gradient descent with a few learning samples. ANIL does not update the whole model and instead only updates the classifier in the last layer.
BOIL (Body Only update in Inner Loop) [165]: BOIL is another optimization-based meta-learning method. Its adaptation algorithm $\mathcal{O}$ freezes the update of the classifier and only updates the embedding backbone.
LEO (Latent Embedding Optimization) [25]: LEO is similar to MAML. But instead of directly optimizing high-dimensional model parameters, its adaptation algorithm $\mathcal{O}$ learns a generative distribution of model parameters and optimizes the model parameters in a low-dimensional latent space.
ProtoNet (Prototypical Networks) [12]: ProtoNet is a metric-based meta-learning method. Its adaptation algorithm $\mathcal{O}$ first calculates a prototype representation for each class as the mean vector of each support class, and then uses a nearest-neighbor classifier created with the class prototypes and the Euclidean distance function to predict a query image.
DN4 (Deep Nearest Neighbor Neural Network) [166]: DN4 is a metric-based meta-learning method, which does not use attributes after pooling for classification. Instead, DN4 uses the local attributes before pooling and employs a local descriptor based image-to-class measure for classification.
R2D2 (Ridge Regression Differentiable Discriminator) [13]: R2D2 is a metric-based meta-learning method and adopts ridge regression as the few-shot adaptation algorithm $\mathcal{O}$. The advantage of R2D2 is that ridge regression enjoys a closed-form solution and can learn efficiently with a few training samples.
CAN (Cross Attention Network) [132]: CAN is a metric-based meta-learning method and calculates the cross attention between each pair of class and query features so as to exploit and learn discriminative features for predictions.
RENet (Relational Embedding Network) [167]: RENet is a metric-based meta-learning method. It uses a self-correlational representation module and a cross-correlational attention module to learn relational patterns within and between images, respectively.
RFS (Rethinking Few-Shot) [66]: RFS is a transfer learning method. It first trains an embedding network using base classes. Then, instead of fine-tuning the last fully-connected classification layer, it learns a new logistic regression classifier with L2-normalized feature vectors from a few samples of novel classes.
Baseline++ [64]: Baseline++ is a transfer learning method. It first pretrains an embedding network using samples from base classes. Then, it fine-tunes the last fully-connected layer with a few samples of novel classes but replaces the standard inner product with a cosine distance between input features and the weight vectors of the layer.

A.3.2 Evaluation Metrics

Standard Accuracy (Acc): Acc measures on average how a few-shot classifier generalizes to different tasks with novel classes not seen before. We define Acc as follows,

$$\text{Acc} = \frac{1}{N_T} \sum_{t=1}^{N_T} \sum_{c=1}^{C} M_c(\mathcal{T}_t; f_\theta, \mathcal{O}), \tag{A.9}$$

where N_T is the number of test tasks, C is the number of classes per task, $\mathcal{T}_t$ is the t-th C-way N_S-shot task with N_Q query samples per class, f_θ is a few-shot classifier, $\mathcal{O}$ is the few-shot adaptation algorithm associated with f_θ, M_c denotes the classification accuracy on the *query samples* of the class c. This metric is used in Figure 4.5.

Class-Wise Worst Classification Accuracy (wAcc): wAcc characterizes the performance limit of f_θ in learning novel classes, and we calculate wAcc as the average of the smallest per-class classification accuracy on query samples over N_T tasks, i.e.,

$$\text{wAcc} = \frac{1}{N_T} \sum_{t=1}^{N_T} \min_{c=1,\ldots,C} M_c(\mathcal{T}_t; f_\theta, \mathcal{O}). \tag{A.10}$$

Depending on what kinds of tasks are used for evaluation, we have the following two types of wAcc:

- **wAcc-R:** If the test tasks are randomly sampled in Equation (A.10), then we get wAcc-R on N_T randomly sampled tasks. This metric is used in Table 4.3 as a baseline for highlighting the effectiveness of our FewSTAB in revealing the spurious bias in few-shot classifiers.
- **wAcc-A:** If the N_T test tasks in Equation (A.10) are constructed by our FewSTAB, then we get wAcc-A, which characterizes the robustness of a few-shot classifier to spurious bias. This metric is the main metric used in the experiments.

Accuracy Gap between wAcc-R and wAcc-A: We obtain the wAcc-R and wAcc-A of a model by testing it with tasks randomly sampled and with tasks constructed by FewSTAB, respectively. The accuracy gap is calculated as the wAcc-R minus the wAcc-A. A large gap indicates the effectiveness of FewSTAB in revealing the robustness of a few-shot classifier to spurious bias. This metric is used in Figure 4.4 and Table 4.5.

Accuracy Gap between wAcc-A of Models Trained with Different Shots: We train a few-shot classifier with C-way (e.g. 5-way) 5-shot and 1-shot training tasks from $\mathcal{D}_{train}$, respectively. Then, we test the obtained two classifiers with the *same* tasks created by FewSTAB and calculate the accuracy gap as the wAcc-A of the model trained with 5-shot tasks minus the wAcc-A of the model trained with 1-shot tasks. A large accuracy gap indicates that increasing training shots can improve a few-shot classifier's robustness to spurious bias. This metric is used in Figure 4.6.

Split	miniImageNet	tieredImageNet	CUB-200
$\mathcal{D}_{train}$	64 (38.4k)	351 (448.7k)	130 (7.6k)
$\mathcal{D}_{val}$	16 (9.6k)	97 (124.3k)	20 (1.2k)
$\mathcal{D}_{test}$	20 (12k)	160 (206.2k)	50 (3.0k)

Table A.3.1: Numbers of classes along with numbers of samples (in parentheses) in each split of the three datasets.

A.3.3 Experimental Settings

We conducted experiments using three datasets: miniImageNet, tieredImageNet, and CUB-200. Each of these datasets has training ($\mathcal{D}_{train}$), validation ($\mathcal{D}_{val}$), and test ($\mathcal{D}_{test}$) sets. Numbers of classes and samples in the three sets of the three datasets are shown in Table A.3.1.

Method	Mode	Learning rate	LR scheduler	Optimizer	Epochs	Training episodes	Episode size
ANIL	T (5w1s)	0.001	-	Adam	100	2000	4
	T (5w5s)	0.001	-	Adam	100	2000	4
LEO	T (5w1s)	0.0005	CosineAnnealingLR	Adam	100	2000	1
	T (5w5s)	0.001	CosineAnnealingLR	Adam	100	2000	1
BOIL	T (5w1s)	0.0006	-	Adam	100	2000	4
	T (5w5s)	0.0006	-	Adam	100	2000	4
ProtoNet	T (5w1s)	0.001	StepLR(20, 0.5)	Adam	100	200	1
	T (5w5s)	0.001	StepLR(20, 0.5)	Adam	100	2000	1
DN4	T (5w1s)	0.001	StepLR(50, 0.5)	Adam	100	2000	1
	T (5w5s)	0.001	StepLR(50, 0.5)	Adam	100	2000	1
R2D2	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	2000	4
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	2000	4
CAN	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	2000	8
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	2000	4
RENet	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	300	1
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	300	1
Baseline++	B (128)	0.01	CosineAnnealingLR	SGD	100	-	-
RFS	B (64)	0.05	MultiStepLR([60, 80], 0.1)	SGD	100	-	-

Table A.3.2: Training configurations and hyperparameters for training on the miniImageNet dataset. "-" denotes not applicable.

Method	Mode	Learning rate	LR scheduler	Optimizer	Epochs	Training episodes	Episode size
ANIL	T (5w1s)	0.001	-	Adam	100	5000	4
	T (5w5s)	0.001	-	Adam	100	5000	4
LEO	T (5w1s)	0.0005	CosineAnnealingLR	Adam	100	5000	1
	T (5w5s)	0.001	CosineAnnealingLR	Adam	100	5000	1
BOIL	T (5w1s)	0.0006	-	Adam	100	5000	4
	T (5w5s)	0.0006	-	Adam	100	5000	4
ProtoNet	T (5w1s)	0.001	StepLR(20, 0.5)	Adam	100	5000	1
	T (5w5s)	0.001	StepLR(20, 0.5)	Adam	100	5000	1
DN4	T (5w1s)	0.001	StepLR(50, 0.5)	Adam	200	5000	1
	T (5w5s)	0.001	StepLR(50, 0.5)	Adam	200	5000	1
R2D2	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	5000	4
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	5000	4
CAN	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	2000	4
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	2000	4
RENet	T (5w1s)	0.1	MultiStepLR([60, 80], 0.05)	SGD	100	1752	1
	T (5w5s)	0.1	MultiStepLR([40, 50], 0.05)	SGD	60	1752	1
Baseline++	B (128)	0.01	CosineAnnealingLR	SGD	100	-	-
RFS	B (128)	0.1	CosineAnnealingLR	SGD	100	-	-

Table A.3.3: Training configurations and hyperparameters for training on the tieredImageNet dataset. "-" denotes not applicable.

We trained eight meta-learning based FSC methods with the ResNet-12 backbone using 5-way 1-shot or 5-way 5-shot tasks from each $\mathcal{D}_{train}$ of the three datasets, resulting in a total of 48 models. For the two transfer learning based methods, RFS and Baseline++, we trained them on each $\mathcal{D}_{train}$ of the three datasets using mini-batch stochastic gradient descent. As a result, we trained a total of 54 models.

To facilitate reproducibility and further research, the training configurations and hyperparameters are provided in Tables A.3.2, A.3.3, and A.3.4 for training on the miniImageNet, tieredImageNet, and CUB-200 datasets, respectively. We closely followed the settings in [161] to train these models. In the "Mode" column of these tables, "T(5w1s)" denotes that we trained the corresponding model using 5-way 1-shot tasks, "T(5w5s)" denotes that we trained the corresponding model using 5-way 5-shot tasks, and "B (128)" denotes that we trained the corresponding model using mini-batch stochastic gradient descent with a batch size of 128. In the "LR scheduler" column, "CosineAnnealingLR" denotes a cosine annealing learning rate scheduler, "StepLR(20, 0.5)" denotes a learning rate scheduler which decreases the learning rate after every 20 epochs by multiplying it with 0.5, and "MultiStepLR([60, 80], 0.1)" denotes a learning rate scheduler which decreases the learning rate after 60 epochs and 80 epochs by multiplying it with 0.1 each time. The "Training episodes" column

Method	Mode	Learning rate	LR scheduler	Optimizer	Epochs	Training episodes	Episode size
ANIL	T (5w1s)	0.001	-	Adam	100	2000	4
	T (5w5s)	0.001	-	Adam	100	2000	4
LEO	T (5w1s)	0.0005	CosineAnnealingLR	Adam	100	2000	4
	T (5w5s)	0.001	CosineAnnealingLR	Adam	100	2000	1
BOIL	T (5w1s)	0.0006	-	Adam	100	2000	4
	T (5w5s)	0.0006	-	Adam	100	2000	4
ProtoNet	T (5w1s)	0.001	StepLR(20, 0.5)	Adam	100	2000	1
	T (5w5s)	0.001	StepLR(20, 0.5)	Adam	100	2000	1
DN4	T (5w1s)	0.001	StepLR(50, 0.5)	Adam	100	2000	1
	T (5w5s)	0.001	StepLR(50, 0.5)	Adam	100	2000	1
R2D2	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	2000	4
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	2000	4
CAN	T (5w1s)	0.01	-	Adam	100	100	4
	T (5w5s)	0.01	-	Adam	100	100	4
RENet	T (5w1s)	0.1	CosineAnnealingLR	SGD	100	300	1
	T (5w5s)	0.1	CosineAnnealingLR	SGD	100	600	1
Baseline++	B (128)	0.01	CosineAnnealingLR	SGD	100	-	-
RFS	B (64)	0.05	MultiStepLR([60, 80], 0.1)	SGD	100	-	-

Table A.3.4: Training configurations and hyperparameters for training on the CUB-200 dataset. "-" denotes not applicable.

Method	miniImageNet				tieredImageNet				CUB-200			
	wAcc-R	wAcc-A/Acc. gap			wAcc-R	wAcc-A/Acc. gap			wAcc-R	wAcc-A/Acc. gap		
		SC1	SC2	SC3		SC1	SC2	SC3		SC1	SC2	SC3
ANIL	25.37	19.69	15.64	14.83	30.60	19.55	14.53	13.72	45.47	38.73	32.39	31.63
		5.68	9.73	**10.54**		11.05	16.07	**16.88**		6.74	13.08	**13.84**
LEO	41.33	34.33	28.02	26.31	57.22	40.28	30.23	29.49	59.76	48.04	43.07	46.62
		7.00	13.31	**15.02**		16.94	26.99	**27.73**		11.72	**16.69**	13.14
BOIL	15.21	13.88	13.46	13.09	18.55	15.82	15.11	14.90	21.33	18.42	17.84	19.17
		1.33	1.75	**2.12**		2.73	3.44	**3.65**		2.91	**3.49**	2.16
ProtoNet	51.95	43.37	33.40	32.07	62.53	44.23	31.61	30.95	75.68	67.12	59.72	60.06
		8.58	18.55	**19.88**		18.30	30.92	**31.58**		8.56	**15.96**	15.62
DN4	42.68	36.74	28.62	27.60	40.63	24.32	16.50	16.07	73.58	66.07	58.32	59.25
		5.94	14.06	**15.08**		16.31	24.13	**24.56**		7.51	**15.26**	14.33
R2D2	50.84	44.01	36.47	35.37	61.08	43.34	31.79	31.12	75.20	65.12	56.88	58.66
		6.83	14.37	**15.47**		17.74	29.29	**29.96**		10.08	**18.32**	16.54
CAN	54.23	46.66	37.82	36.44	64.19	45.53	32.23	31.17	61.61	53.91	44.91	41.31
		7.57	16.41	**17.79**		18.66	31.96	**33.02**		7.70	16.70	**20.30**
RENet	56.52	47.48	37.60	36.19	63.49	44.23	31.04	30.27	71.82	63.03	53.27	52.93
		9.04	18.92	**20.33**		19.26	32.45	**33.22**		8.79	18.55	**18.89**
Baseline++	44.94	37.70	30.47	29.52	59.06	40.95	30.74	30.01	29.84	24.21	19.55	16.86
		7.24	14.47	**15.42**		18.11	28.32	**29.05**		5.63	10.29	**12.98**
RFS	55.66	48.17	38.33	36.85	62.71	44.35	31.94	31.15	74.33	64.54	54.41	54.98
		7.49	17.33	**18.81**		18.36	30.77	**31.56**		9.79	**19.92**	19.35
Average drop	-	6.67	13.89	**15.05**	-	15.75	25.43	**26.12**	-	7.94	**14.83**	14.72

Table A.3.5: Comparison between different techniques used by FewSTAB for constructing the support sets in 5-way 5-shot FSC test tasks. Values in the shaded areas are the accuracy gaps defined as wAcc-R minus wAcc-A. Average drop is the average of accuracy gaps over the ten FSC methods. "-" denotes not applicable.

in these tables denotes the number of tasks used in each epoch. The "Episode size" column of these tables denotes the number of tasks jointly used to do a model update.

A.3.4 Additional Experimental Results

Ablation Studies

Support Set Construction Methods. To construct the support set in an FSC test task, FewSTAB randomly selects samples that have *mutually exclusive* spurious attributes across the randomly selected classes, which is illustrated in Figure 4.2(a) and formally described in Chapter 4.1.4. To further show the effectiveness of this construction method, we keep the techniques for constructing

Method	miniImageNet				tieredImageNet				CUB-200			
	wAcc-R	wAcc-A/Acc. gap			wAcc-R	wAcc-A/Acc. gap			wAcc-R	wAcc-A/Acc. gap		
		QC1	QC2	QC3		QC1	QC2	QC3		QC1	QC2	QC3
ANIL	25.37	21.61	16.37	14.83	30.60	21.95	14.00	13.72	45.47	39.77	32.58	31.63
		3.76	9.00	**10.54**		8.65	16.60	**16.88**		5.70	12.89	**13.84**
LEO	41.33	36.04	28.36	26.31	57.22	46.94	31.93	29.49	59.76	56.73	48.21	46.62
		5.29	12.97	**15.02**		10.28	25.29	**27.73**		3.03	11.55	**13.14**
BOIL	15.21	14.57	13.70	13.09	18.55	17.61	15.37	14.90	21.33	20.49	19.85	19.17
		0.64	1.51	**2.12**		0.94	3.18	**3.65**		0.84	1.48	**2.16**
ProtoNet	51.95	44.28	34.17	32.07	62.53	49.58	33.56	30.95	75.68	70.33	61.81	60.06
		7.67	17.78	**19.88**		12.95	28.97	**31.58**		5.35	13.87	**15.62**
DN4	42.68	39.25	28.91	27.60	40.63	28.28	17.63	16.07	73.58	70.37	60.61	59.25
		3.43	13.77	**15.08**		12.35	23.00	**24.56**		3.21	12.97	**14.33**
R2D2	50.84	45.68	36.99	35.37	61.08	48.96	33.83	31.12	75.20	69.78	60.34	58.66
		5.16	13.85	**15.47**		12.12	27.25	**29.96**		5.42	14.86	**16.54**
CAN	54.23	47.83	38.16	36.44	64.19	50.58	33.63	31.17	61.61	54.32	42.88	41.31
		6.40	16.07	**17.79**		13.61	30.56	**33.02**		7.29	18.73	**20.30**
RENet	56.52	49.80	38.31	36.19	63.49	49.86	32.76	30.27	71.82	64.26	54.48	52.93
		6.72	18.21	**20.33**		13.63	30.73	**33.22**		7.56	17.34	**18.89**
Baseline++	44.94	39.51	31.26	29.52	59.06	47.08	31.86	30.01	29.84	27.47	18.62	16.86
		5.43	13.68	**15.42**		11.98	27.20	**29.05**		2.37	11.22	**12.98**
RFS	55.66	48.87	39.48	36.85	62.71	49.99	33.64	31.15	74.33	67.60	56.60	54.98
		6.79	16.18	**18.81**		12.72	29.07	**31.56**		6.73	17.73	**19.35**
Average drop	-	5.13	13.30	**15.05**	-	10.92	24.19	**26.12**	-	4.75	13.27	**14.72**

Table A.3.6: Comparison between different techniques used by FewSTAB for constructing the query sets in 5-way 5-shot FSC test tasks. Values in the shaded areas are the accuracy gaps defined as wAcc-R minus wAcc-A. Average drop is the average of accuracy gaps over the ten FSC methods. "-" denotes not applicable.

the query set in an FSC test task, and report in Table A.3.5 the results of two alternatives for constructing the support set: randomly selecting samples of the selected classes (**SC1**) and randomly selecting samples with targeted attributes for selected classes with no further constraints on the selected samples (**SC2**). We also include the results of the proposed one: randomly selecting samples with mutually exclusive targeted attributes across the selected classes (**SC3**) in Table A.3.5. A larger average drop in Table A.3.5 indicates that the corresponding support set construction method is more effective in revealing robustness of few-shot classifiers to spurious bias. We observe that the third technique SC3, which is used by FewSTAB, achieves the largest average accuracy drop among the techniques compared on the miniImageNet and tieredImageNet datasets and achieves a comparable drop to SC2 on the CUB-200 dataset due to the limited number of detected attributes in this dataset.

Query Set Construction Methods. There are three techniques used by FewSTAB to construct the query set in a task: the intra-class attribute-based sample selection (**QC1**), the inter-class attribute-based sample selection (**QC2**), which is a special case of the intra-class attribute-based sample selection, and the query sample selection (**QC3**). We have done an ablation study on the effectiveness of the three techniques in Table 4.5 using the miniImageNet dataset. Here, we include the results on all the three datasets in Table A.3.6. We observe that all the three proposed techniques in FewSTAB are effective with positive accuracy drops for all the ten FSC methods on the three datasets. Moreover, using the inter-class attribute-based sample selection significantly improves the average drops of the intra-class attribute-based sample selection, with 8.17%, 13.26%, and 8.52% absolute gains on the miniImageNet, tieredImageNet, and CUB-200 datasets, respectively.

Scatter Plots of wAcc-A versus Acc

We show the scatter plots of wAcc-A versus Acc (standard accuracy) of the ten FSC methods when they are tested with FewSTAB and randomly constructed FSC test tasks, respectively, on the three datasets in Figure A.3.1 (exact values are shown in Table A.3.7). We observe that an FSC method having a higher Acc does not necessarily have a higher wAcc-A. For example, in Figure A.3.1(a), BOIL has a higher Acc but a lower wAcc-A than ProtoNet, LEO, and Baseline++. Moreover, we observe that in Figures A.3.1(b) and A.3.1(d), for methods that achieve high standard

accuracies, e.g., for the top-5 methods in terms of Acc, their relative increments in wAcc-A are small (with differences smaller than 1%) compared with their relative increments in Acc. In other words, methods with higher standard accuracies do not necessarily learn more robust decision rules, since their wAcc-A values remain comparable to those with lower Acc values.

The values of Acc and wAcc-A on the fine-grained dataset CUB-200 in Figures A.3.1(e) and A.3.1(f) show a different pattern from those in Figures A.3.1(c) and A.3.1(d). More specifically, methods that achieve high Acc values, e.g., R2D2, ProtoNet, DN4, RENet, and RFS, tend to have comparable relative increments in wAcc-A compared with their relative increments in Acc. This indicates that on a fine-grained dataset, which does not have many spurious attributes, an FSC method with a higher Acc also tends to have a higher wAcc-A or improved robustness to spurious bias.

In summary, our framework, FewSTAB, reveals new robustness patterns of FSC methods in different evaluation settings.

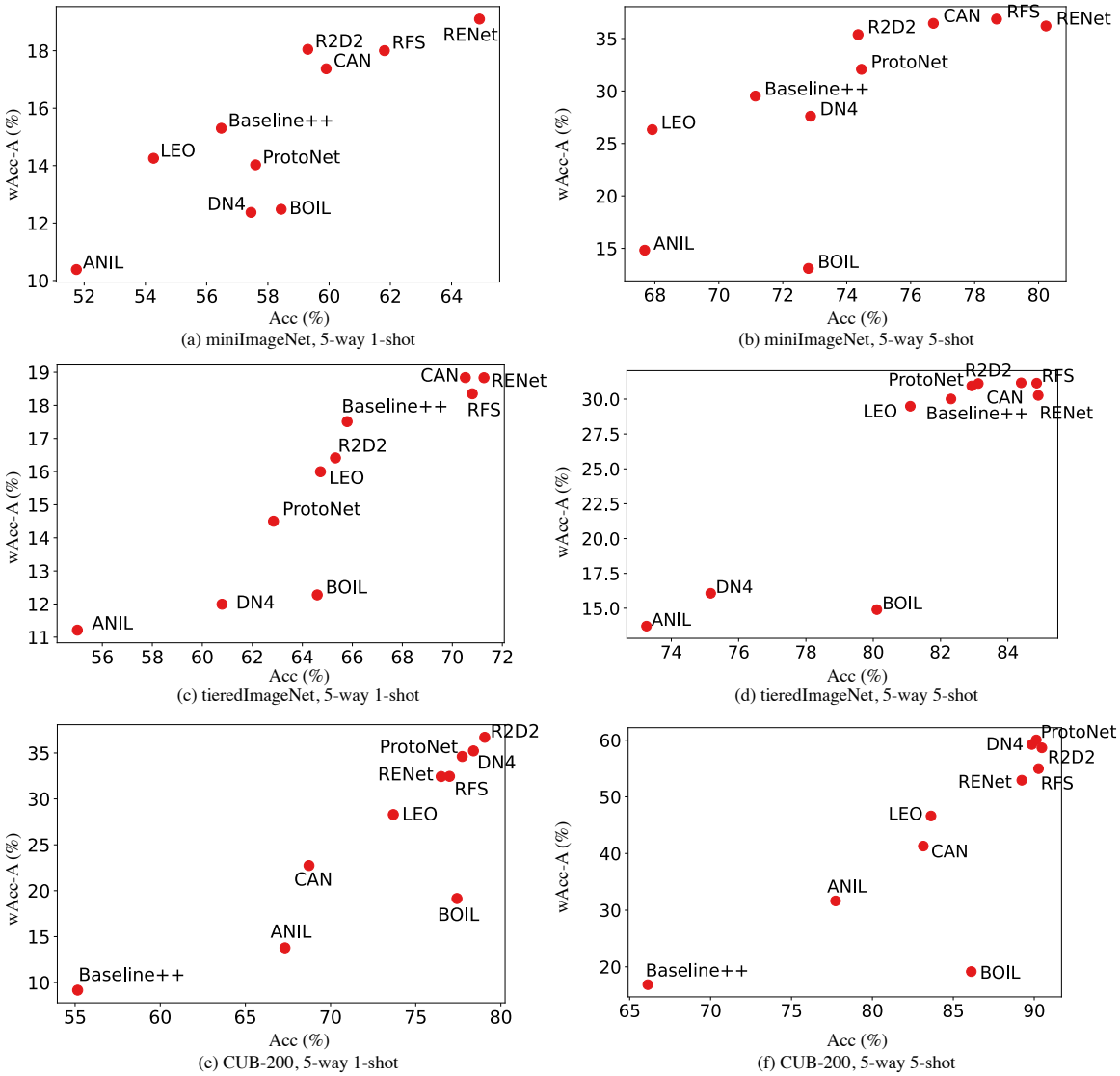


Figure A.3.1: Scatter plots of wAcc-A versus Acc of the ten FSC methods tested with 5-way 1/5-shot FewSTAB and randomly constructed tasks from the miniImageNet, tieredImageNet, and CUB-200 datasets, respectively. All methods are trained and tested with the same shot number.

Shot	Method	miniImageNet		tieredImageNet		CUB-200	
		Acc	wAcc-A	Acc	wAcc-A	Acc	wAcc-A
1	ANIL	51.75±0.39	10.38±0.30	55.00±0.45	11.21±0.30	67.32±0.45	13.78±0.40
	LEO	54.27±0.38	14.26±0.46	64.73±0.46	16.00±0.55	73.68±0.42	28.29±0.80
	BOIL	58.43±0.39	12.48±0.23	64.60±0.43	12.27±0.21	77.42±0.39	19.15±0.29
	ProtoNet	57.60±0.38	14.03±0.49	62.85±0.44	14.50±0.50	77.73±0.39	34.62±0.85
	DN4	57.45±0.36	12.37±0.46	60.79±0.42	11.99±0.47	78.39±0.38	35.22±0.86
	R2D2	59.30±0.39	18.05±0.53	65.33±0.44	16.41±0.54	79.05±0.38	36.70±0.90
	CAN	59.91±0.38	17.37±0.53	70.52±0.43	18.84±0.60	68.73±0.41	22.74±0.72
	RENet	64.91±0.38	19.10±0.57	71.27±0.42	18.83±0.61	76.49±0.36	32.43±0.81
	Baseline++	56.48±0.37	15.30±0.48	65.79±0.42	17.51±0.54	55.15±0.44	9.17±0.47
	RFS	61.81±0.35	18.00±0.53	70.80±0.42	18.35±0.60	76.99±0.35	32.45±0.80
5	ANIL	67.68±0.33	14.83±0.40	73.26±0.35	13.72±0.39	77.72±0.34	31.63±0.55
	LEO	67.92±0.32	26.31±0.59	81.10±0.34	29.49±0.72	83.62±0.30	46.62±0.82
	BOIL	72.80±0.29	13.09±0.22	80.11±0.32	14.90±0.22	86.11±0.26	19.17±0.28
	ProtoNet	74.46±0.28	32.07±0.58	82.93±0.31	30.95±0.70	90.13±0.21	60.06±0.74
	DN4	72.87±0.29	27.60±0.58	75.17±0.36	16.07±0.62	89.85±0.21	59.25±0.77
	R2D2	74.36±0.29	35.37±0.59	83.12±0.30	31.12±0.72	90.47±0.21	58.66±0.82
	CAN	76.71±0.28	36.44±0.65	84.40±0.29	31.17±0.76	83.14±0.27	41.31±0.74
	RENet	80.23±0.26	36.19±0.63	84.90±0.28	30.27±0.76	89.23±0.21	52.93±0.82
	Baseline++	71.14±0.30	29.52±0.57	82.31±0.31	30.01±0.72	66.12±0.35	16.86±0.52
	RFS	78.69±0.26	36.85±0.64	84.86±0.29	31.15±0.76	90.26±0.20	54.98±0.81

Table A.3.7: Standard accuracies (Acc) and class-wise worst accuracies obtained with FewSTAB (wAcc-A) with 95% confidence intervals of the ten FSC methods on miniImageNet, tieredImageNet, and CUB datasets. Numbers in the Shot column indicate that the models are both trained (if applicable) and tested on 5-way 1- or 5-shot tasks. Darker colors indicate higher values.

Effectiveness of FewSTAB: More Results

Results on More Recent Methods. Note that our method selection in Table 4.3 aims to cover *diverse* methods and allow for *rigorous* comparison in the *same* setting. Importantly, our method is general and can continue to evaluate emerging methods. To demonstrate, we provide results on recent methods, namely UniSiam [228], PsCo [229], and BECLR [230]. FewSTAB uncovers that, even the state-of-the-art methods still suffer from spurious bias as we observe large gaps between wAcc-R and wAcc-A (Table A.3.9), when we explicitly construct the test tasks to have spurious correlations. This also shows that FewSTAB is effective for various FSC methods.

Results on IFSL. Interventional few-shot learning (IFSL) [63] is a method that specifically addresses spurious correlations in few-shot classification. We follow the settings in [63] and report the results of MAML [22], MN [23], SIB [231], and MTL [133] in Table A.3.10, where "Base" refers to one of the four methods, "+IFSL" denotes using IFSL on top of "Base", and the better performance between the two is in bold. Overall, IFSL is effective in mitigating spurious bias in few-shot classifiers except for some methods, e.g. SIB. This shows that FewSTAB can reveal the improvement made to mitigate spurious bias.

Robustness of FewSTAB with Different VLMs

We instantiated our FewSTAB with a pre-trained ViT-GPT2 and a pre-trained BLIP, respectively. We calculated the wAcc-A on FSC test tasks constructed by FewSTAB with the two VLMs on the miniImageNet, tieredImageNet, and CUB-200 datasets, respectively.

Effects on Individual and Relative Measurements. We observe from Table A.3.8 that FewSTAB with BLIP produces lower wAcc-A than with ViT-GPT2 on the miniImageNet and tieredImageNet datasets. This indicates that FewSTAB with BLIP is more effective in uncovering the robustness of few-shot classifiers to spurious bias. We reason that BLIP can identify more attributes than ViT-GPT2 (Table 4.2) and therefore more spurious correlations can be formulated by our

Shot	Method	miniImageNet		tieredImageNet		CUB-200	
		ViT-GPT2	BLIP	ViT-GPT2	BLIP	ViT-GPT2	BLIP
1	ANIL	10.38±0.30	10.39±0.29	11.21±0.30	10.76±0.29	13.78±0.40	14.74±0.41
	LEO	14.26±0.46	14.38±0.45	16.00±0.55	14.34±0.52	28.29±0.80	31.06±0.80
	BOIL	12.48±0.23	12.51±0.22	12.27±0.21	11.65±0.21	19.15±0.29	20.35±0.29
	ProtoNet	14.03±0.49	13.50±0.46	14.50±0.50	13.25±0.50	34.62±0.85	38.63±0.81
	DN4	12.37±0.46	12.86±0.46	11.99±0.47	11.21±0.46	35.22±0.86	39.51±0.82
	R2D2	18.05±0.53	17.66±0.51	16.41±0.54	15.01±0.53	36.70±0.90	40.61±0.84
	CAN	17.37±0.53	16.89±0.51	18.84±0.60	17.43±0.61	22.74±0.72	24.23±0.71
	RENet	19.10±0.57	18.80±0.54	18.83±0.61	17.29±0.60	32.43±0.81	36.12±0.82
	Baseline++	15.30±0.48	15.06±0.46	17.51±0.54	15.60±0.52	9.17±0.47	10.42±0.50
	RFS	18.00±0.53	17.43±0.50	18.35±0.60	16.81±0.57	32.45±0.80	35.43±0.79
5	ANIL	14.83±0.40	13.67±0.38	13.72±0.39	12.57±0.37	31.63±0.55	33.01±0.56
	LEO	26.31±0.59	24.79±0.57	29.49±0.72	27.92±0.70	46.62±0.82	49.97±0.81
	BOIL	13.09±0.22	12.79±0.22	14.90±0.22	14.63±0.22	19.17±0.28	20.03±0.27
	ProtoNet	32.07±0.58	29.28±0.57	30.95±0.70	28.51±0.68	60.06±0.74	64.67±0.64
	DN4	27.60±0.58	25.28±0.57	16.07±0.62	14.98±0.58	59.25±0.77	65.61±0.67
	R2D2	35.37±0.59	31.81±0.59	31.12±0.72	29.50±0.68	58.66±0.82	64.02±0.77
	CAN	36.44±0.65	33.81±0.62	31.17±0.76	29.28±0.72	41.31±0.74	43.10±0.73
	RENet	36.19±0.63	33.76±0.63	30.27±0.76	28.71±0.72	52.93±0.82	60.29±0.74
	Baseline++	29.52±0.57	27.17±0.55	30.01±0.72	28.20±0.70	16.86±0.52	17.25±0.53
	RFS	36.85±0.64	34.72±0.62	31.15±0.76	29.29±0.72	54.98±0.81	62.33±0.69

Table A.3.8: Comparison between wAcc-A calculated over 5-way 1/5-shot tasks obtained using Vit-GPT2 and using BLIP. We calculated wAcc-A for ten FSC methods on miniImageNet, tieredImageNet, and CUB datasets. Numbers in the Shot column indicate that the models are both trained (if applicable) and tested on 1- or 5-shot tasks. Darker colors indicate higher values.

Method	Shot	wAcc-R	wAcc-A (V)	wAcc-A (B)
UniSiam	1	$21.26_{\pm 0.48}$	$13.52_{\pm 0.43}$	$13.49_{\pm 0.42}$
PsCo	1	$21.50_{\pm 0.47}$	$14.30_{\pm 0.40}$	$12.46_{\pm 0.37}$
BECLR	1	$35.57_{\pm 0.80}$	$23.60_{\pm 0.83}$	$22.42_{\pm 0.82}$
UniSiam	5	$45.60_{\pm 0.52}$	$27.76_{\pm 0.57}$	$25.42_{\pm 0.56}$
PsCo	5	$42.15_{\pm 0.52}$	$25.54_{\pm 0.52}$	$22.64_{\pm 0.49}$
BECLR	5	$55.20_{\pm 0.49}$	$37.32_{\pm 0.66}$	$33.42_{\pm 0.68}$

Table A.3.9: Results on the miniImageNet dataset. V: ViT-GPT2, B: BLIP. All input images are resized to 84×84.

FewSTAB. However, on the fine-grained CUB-200 dataset, which contains different bird classes, FewSTAB with BLIP is less effective than with ViT-GPT2. Although BLIP can identify more attributes than ViT-GPT2 in this fine-grained dataset, it may also detect more attributes related to classes. To validate this, we first found a set of attributes $\mathcal{U}_{\text{BLIP}}$ unique to BLIP from all the attributes $\mathcal{A}_{\text{BLIP}}$ detected by BLIP, and a set of attributes $\mathcal{U}_{\text{ViT-GPT2}}$ unique to ViT-GPT2 from all the attributes $\mathcal{A}_{\text{ViT-GPT2}}$ detected by ViT-GPT2. Specifically, we have $\mathcal{U}_{\text{BLIP}} = \mathcal{A}_{\text{BLIP}} - \mathcal{A}_{\text{ViT-GPT2}}$, and $\mathcal{U}_{\text{ViT-GPT2}} = \mathcal{A}_{\text{ViT-GPT2}} - \mathcal{A}_{\text{BLIP}}$. Then, we found in $\mathcal{U}_{\text{BLIP}}$ and $\mathcal{U}_{\text{ViT-GPT2}}$ how many attributes contain "bird", "beak", "wing", "breast", "tail", or "mouth", which are all related to the concept of a bird. We found that there are 11 attributes, or 8.5% of total attributes in $\mathcal{U}_{\text{BLIP}}$ that are related to a bird. While there is only 1 attribute (2.4% of total attributes) in $\mathcal{U}_{\text{ViT-GPT2}}$ that is related to a bird. Due to the limited capability of BLIP, these class-related attributes cannot be detected in all the images. Hence, although these attributes are not spurious, they are treated as spurious attributes and used by FewSTAB to construct FSC test tasks. In this case, FewSTAB becomes ineffective in revealing the spurious bias in few-shot classifiers since the classifiers can exploit spurious correlations in the tasks to achieve high accuracies. Nevertheless, from the perspective of comparing

Method	1-shot		5-shot	
	Base	+IFSL	Base	+IFSL
MAML	**13.29**$_{\pm 0.55}$	12.05$_{\pm 0.56}$	28.70$_{\pm 0.69}$	**29.82**$_{\pm 0.76}$
MN	17.40$_{\pm 0.62}$	**17.72**$_{\pm 0.63}$	30.48$_{\pm 0.73}$	**31.51**$_{\pm 0.75}$
SIB	**30.09**$_{\pm 1.04}$	27.10$_{\pm 0.97}$	**46.73**$_{\pm 0.96}$	46.66$_{\pm 0.95}$
MTL	37.29$_{\pm 0.57}$	**40.22**$_{\pm 0.57}$	49.49$_{\pm 0.58}$	**52.66**$_{\pm 0.58}$

Table A.3.10: wAcc-A comparison (%) on the miniImageNet dataset.

VLM	Detection accuracy		Spearman's rank correlation coefficient	
	ViT-GPT2	BLIP	1-shot	5-shot
miniImageNet	34.46	31.42	0.98	1.0
tieredImageNet	35.04	32.00	1.0	0.99
CUB-200	70.12	59.28	1.0	0.98

Table A.3.11: Detection accuracies of the ViT-GPT2 and BLIP along with the Spearman's rank correlation coefficients between the results based on the two VLMs.

the robustness of different FSC methods to spurious bias, the test tasks constructed by FewSTAB using different VLMs can reveal consistent ranks in terms of wAcc-A for different FSC methods (Table 4.6).

Detection Accuracies of VLMs. Using different VLMs may generate different sets of attributes. Some sets of attributes may not exactly reflect the data being described, resulting in low detection accuracies. For example, some attributes are not identified by a VLM or the identified attributes do not match with the ground truth attributes. To analyze how the detection accuracy of a VLM affects our framework, we show in Table A.3.11 the detection accuracies of the two VLMs that we used in Chapter 4.1.5 along with the Spearman's rank correlation coefficients between the evaluation results on the ten FSC methods based on the two VLMs. To calculate the detection accuracy of a VLM without the labor-intensive human labeling, we use the outputs of another VLM as the ground truth. Specifically, for the i'th image, we have two detected sets of attributes, $\mathcal{A}^i_{query}$ and $\mathcal{A}^i_{ref}$, representing the attributes from a VLM being evaluated and the ones from another VLM serving as the ground truth attributes. The detection accuracy is calculated as follows:

$$Acc(VLM_{query}, VLM_{ref}) = \frac{1}{|\mathcal{D}_{test}|} \sum_{i=1}^{N_{test}} \frac{|\mathcal{A}^i_{query} \cap \mathcal{A}^i_{ref}|}{|\mathcal{A}^i_{ref}|}, \tag{A.11}$$

where $N_{test} = |\mathcal{D}_{test}|$, and $|\cdot|$ denotes the size of a set. For example, to calculate the detection accuracy of ViT-GPT2, we set VLM_{query} =ViT-GPT2 and VLM_{ref} =BLIP. From Table A.3.11, we observe that the detection accuracies of the two VLMs are not high, indicating that the attributes identified by the two VLMs are very different. However, the two VLMs are well-established in practice and can identify many attributes from images (Table 4.2). The correlation coefficients in Table A.3.11 indicate that for well-established VLMs, the detection accuracies have little impact on the comparison of robustness to spurious bias between different FSC methods.

A.3.5 Tasks Constructed by FewSTAB

FewSTAB does not construct tasks based on a specific model. Hence, FewSTAB is a fair evaluation framework for different FSC methods, and the tasks constructed by FewSTAB can be used to reveal few-shot classifiers' varied degrees of robustness to spurious bias.

We show a 5-way 1-shot task constructed by FewSTAB using samples from the tieredImageNet and CUB-200 datasets in Figures A.3.2 and A.3.3, respectively. Query samples for each class are constructed such that they do not contain the spurious attribute from the support set sample of the same class but contain spurious attributes from support set samples of other classes. For example,

in Figure A.3.2, the class `malamute` has a support set sample with a `rocky` background, but most of its query samples have a `bike` which is the spurious attribute from the support set sample of the `valley` class. Moreover, in Figure A.3.3, the class `Mallard` has a support set sample with a `sandy` background, but its query samples all have a `water` background similar to that in the support set sample of the `Baltimore Oriole` class. Note that the sample selection may not be ideal due to the limited capacity of VLMs. For example, in Figure A.3.2, some query images of the class `eggnog` have the spurious attribute `cup` which also appears in the support set image of the class, leading to a high accuracy on these query samples for a model that relies on this spurious attribute. However, this does not affect our evaluation of different FSC methods on their robustness to spurious bias since the same set of tasks is used to evaluate different FSC methods. Moreover, our metric, wAcc-A, measures the worst per-class classification accuracy over FSC tasks, making our evaluation robust to the sampling noise caused by a VLM.

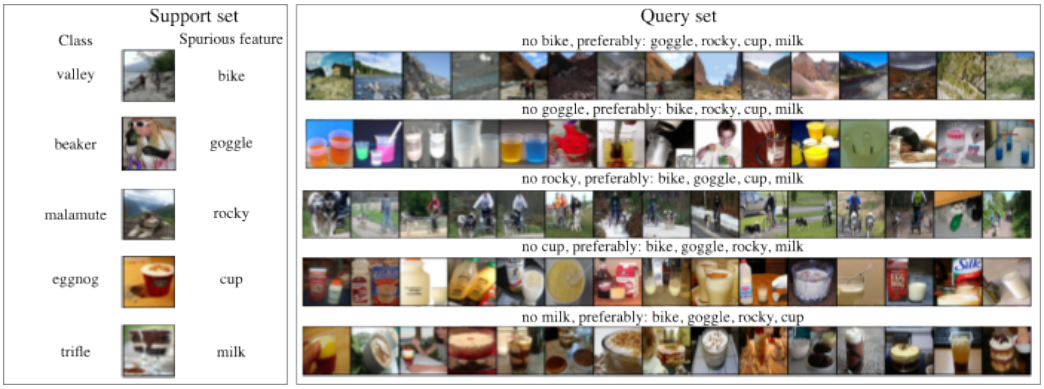


Figure A.3.2: A 5-way 1-shot task constructed by our FewSTAB using samples from the tieredImageNet dataset. Note that due to the limited capacity of a VLM, the attributes may not well align with human understandings.

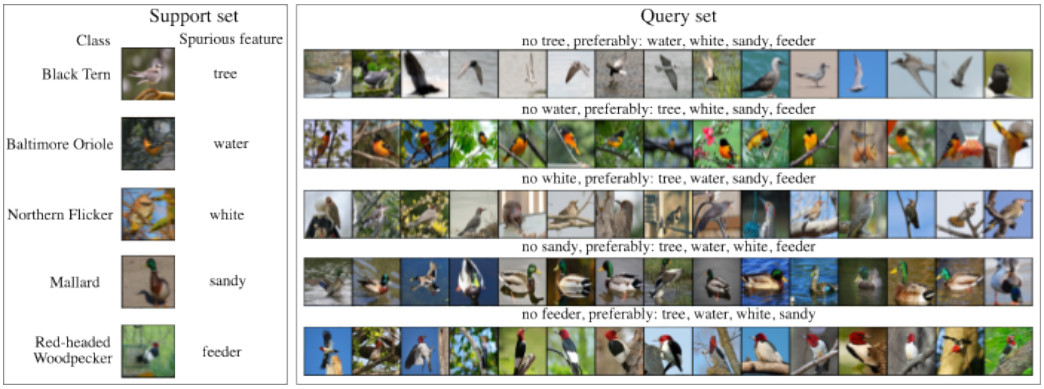


Figure A.3.3: A 5-way 1-shot task constructed by our FewSTAB using samples from the CUB-200 dataset. Note that due to the limited capacity of a VLM, the attributes may not well align with human understandings.

A.4 Learning Robust Classifiers with Self-Guided Spurious Correlation Mitigation

A.4.1 Learning Algorithm

The whole learning procedure of our proposed LBC is shown in Algorithm 4. We iteratively retrain a model adapted from an ERM-trained model using relabeled (Section 4.2.3) and balanced (Section 4.2.3) training data. Our relabeling does not alter the class membership of the training data; instead, it creates fine-grained labels within classes. Therefore, although the classification head of $\tilde{\theta}$ keeps changing in each training epoch, the model's ability to recognize different classes keeps improving after each training epoch. Even when the generated fine-grained labels are noisy, the backbone of $\tilde{\theta}$ is still encouraged to recognize different classes. We select the best model based on its performance on the validation data.

Time complexity. The first step of our algorithm, i.e., building the attribute set $\mathcal{A}$, is only needed once for each dataset. Thus, its time complexity is negligible once $\mathcal{A}$ has been generated. Generating the spurious scores needs a forward pass of all the N training samples. Therefore, the time complexity is $O(N)$ with a scaling constant τ_{sc} representing the average complexity over N samples. The KMeans clustering step has a time complexity of $O(KNT)$ with a scaling constant τ_{clu}, where K is the number of clusters, T is the number of iterations for the clustering, and τ_{clu} denotes the complexity for computing the Euclidean distance between two vectors. The complexity of optimizing $\tilde{f}_{\tilde{\theta}}$ is $O(N)$ with a scaling constant τ_{opt} denoting the complexity for a backward pass of the model. Typically, $\tau_{sc} \ll \tau_{opt}$, $\tau_{clu} \ll \tau_{opt}$, and $K \cdot T$ is typicall small. Therefore, the overall complexity of our algorithm is approximately $O(EN)$, where E is the number of training epochs.

Algorithm 4 Learning beyond classes (LBC)

Input: Training dataset $\mathcal{D}_{tr}$, an ERM trained model f_θ, number of clusters K, a pre-trained VLM ϕ, an attribute extraction procedure ψ, and the number of training iterations E.
Output: Learned weights $\tilde{\theta}$

Build the attribute set $\mathcal{A} = \cup_{(x,y)\in\mathcal{D}_{tr}} \psi(\phi(x))$
Transform f_θ into $\tilde{f}_{\tilde{\theta}}$
for $e = 1, \ldots, E$ **do**
    Generate spuriousness scores using Equation (4.8)
    Get cluster labels $p_K(x, y)$ with Equation (4.10)
    Relabeling with $g_K(x, y) = p_K(x, y) + (y - 1) \cdot K$
    Optimize $\tilde{f}_{\tilde{\theta}}$ using Equation (4.12)
end for
return $\tilde{\theta}$

A.4.2 Datasets

Table A.4.1 depicts detailed statistics for all datasets. For Waterbirds and CelebA datasets, we give the number of training, validation, and test images in each group specified by classes and attributes. For example, the group ⟨landbird, land⟩ in the Waterbirds dataset has 3498 training images which are all landbirds and have land backgrounds. The NICO dataset uses multiple contexts as spurious attributes which are listed in Table A.4.2. The ImageNet-9 and ImageNet-A datasets do not have clear group partitions specified by the class and attribute associations.

In the NICO dataset [182], the training set consists of 7 context classes per object class, and there are 10 object classes. Images in the training set are long-tailed distributed in the sense that an object class has exponentially decreasing numbers of images that correlate with the 7 contexts. Table A.4.2 gives the contexts for each of the 10 classes. The contexts of a class are arranged based on the number of images they have in the class, and the first context has the most images. We follow the setting in [182], where for each class, the number of images having the context t is proportional to the ratio IR^{i_t}, where $i_t (0 \leq i_t \leq 6$ denotes the index of the context t for the corresponding class

Dataset	Number of classes	⟨class, attribute⟩	Number of images		
			Train	Val	Test
Waterbirds	2	⟨landbird, land⟩	3,498	467	2,255
		⟨landbird, water⟩	184	466	2,255
		⟨waterbird, land⟩	56	133	642
		⟨waterbird, water⟩	1,057	133	642
CelebA	2	⟨non-blond, female⟩	71,629	8,535	9,767
		⟨non-blond, male⟩	66,874	8,276	7,535
		⟨blond, female⟩	22,880	2,874	2,480
		⟨blond, male⟩	1,387	182	180
NICO	10	-	2840	1299	1299
ImageNet-9	9	-	54,600	2,100	-
ImageNet-A	9	-	-	-	1087

Table A.4.1: Detailed statistics of the 5 datasets. ⟨class, attribute⟩ represents a spurious correlation between a class and a spurious attribute. "-" denotes not applicable.

in Table A.4.2, and the imbalance ratio IR is 0.02. The validation and test sets contain images from the 10 classes, and each class has a equal number of images from its 7 associated context classes and 3 new contexts not seen in the training.

Class	Contexts
dog	on_grass, in_water, in_cage, eating, on_beach, lying, running; at home, in street, on snow
cat	on_snow, at_home, in_street, walking, in_river, in_cage, eating; in water, on grass, on tree
bear	in_forest, black, brown, eating_grass, in_water, lying, on_snow; on ground, on tree, white
bird	on_ground, in_hand, on_branch, flying, eating, on_grass, standing; in water, in cage, on shoulder
cow	in_river, lying, standing, eating, in_forest, on_grass, on_snow; at home, aside people, spotted
elephant	in_zoo, in_circus, in_forest, in_river, eating, standing, on_grass; in street, lying, on snow
horse	on_beach, aside_people, running, lying, on_grass, on_snow, in_forest; at home, in river, in street
monkey	sitting, walking, in_water, on_snow, in_forest, eating, on_grass; in cage, on beach, climbing
rat	at_home, in_hole, in_cage, in_forest, in_water, on_grass, eating; lying, on snow, running
sheep	eating, on_road, walking, on_snow, on_grass, lying, in_forest; aside people, in water, at sunset

Table A.4.2: Classes and their associated contexts in the NICO datasets. Contexts after the semi-colons are unseen in the training set.

The ImageNet-9 dataset [181] is a subset of ImageNet. It has 9 super-classes, i.e., Dog, Cat, Frog, Turtle, Bird, Primate, Fish, Crab, Insect, which are obtained by merging similar classes from ImageNet. ImageNet-A contains real-world images that are challenging to the image classifiers trained on standard ImageNet. We extract images of the 9 super-classes from the ImageNet-A dataset and use these images as the test data. To calculate the Unbiased accuracy on the validation

set of ImageNet-9, we use the cluster labels provided in [181] that partition the validation data into groups and calculate the average accuracy over these groups.

A.4.3 Implementation Details

Spurious Attribute Detection

We generate text descriptions for images using a pre-trained ViT-GPT2 model [40]. Figure A.4.1 shows four images from the ImageNet-9 dataset along with their descriptions generated by the ViT-GPT2 model. After generating text descriptions, we use Spacy (https://spacy.io/) to automatically extract nouns and adjectives from the descriptions. Then, we add the extracted words to $\mathcal{A}$, forming a set of detected attributes which are potentially spurious. We additionally filter out attributes with frequencies less than 10 to remove rare words that represent too few images and potential annotation noise. Table A.4.3 shows the numbers of detected attributes as well as the numbers of average detected attributes per image for the four datasets which we used during training. We did not detect attributes on the ImageNet-A dataset since it was only used for testing.

Figure A.4.1: Examples of the generated text descriptions for images in the ImageNet-9 dataset.

Dataset	Number of detected attributes	Average number of attributes per image
Waterbirds	144	4.314
CelebA	345	4.291
NICO	199	3.995
ImageNet-9	442	4.311

Table A.4.3: Statistics of the attributes detected from the Waterbirds, CelebA, NICO, and ImageNet-9 datasets.

Dataset	Backbone	Initialization	Learning Rate	Learning Rate Scheduler	Batch Size	Epochs
Waterbirds	ResNet-50	ImageNet pre-trained	3e-3	Cosine Annealing	32	100
CelebA	ResNet-50	ImageNet pre-trained	3e-3	Cosine Annealing	100	20
NICO	ResNet-18	ImageNet pre-trained	-	-	-	-
ImageNet-9	ResNet-18	ImageNet pre-trained	1e-3	Cosine Annealing	128	100
ImageNet-9	ResNet-18	Random	5e-2	MultiStepLR([50, 80, 100],0.2)	256	100

Table A.4.4: Details for training ERM models on the four datasets. MultiStepLR([epoch1, epoch2, epoch3], r) denotes a learning rate scheduler which decays the learning rate at specified epochs with a multiplication factor r, and '-' denotes no training.

Dataset	Learning rate	Batch size	Number of Batches Per Epoch	Training Epochs	K	Model Selection Metric
Waterbirds	1e-4	128	20	50	3	PU-ValAcc
CelebA	1e-4	128	20	50	3	PU-ValAcc
NICO	1e-4	128	50	50	3	PU-ValAcc
ImageNet-9	1e-4	128	200	100	4	Validation accuracy

Table A.4.5: Hyperparameter settings and model selection criteria for LBC training on the Waterbirds, CelebA, NICO, and ImageNet-9 datasets. PU-ValAcc denotes pseudo unbiased validation accuracy.

Non-self-explanatory attributes are still informative. We use two detected attributes, `christmas tree` and `phone`, to select samples from the CelebA dataset and show four samples for each of the attribute in Figure A.4.2. We observe that `christmas tree` and `phone` are not self-explanatory in representing the common features shared among the samples because of the limited capacity of the pre-trained vision-language model (VLM) used to generate text descriptions. However, samples selected by each attribute do have some characteristics shared in common. For the samples selected based on `christmas tree`, they all have background colors that are related to a Christmas tree, e.g., red colors are recognized as some decorations on a Christmas tree by the pre-trained VLM. In the samples selected based on `phone`, the people all hold their hand close to their faces.

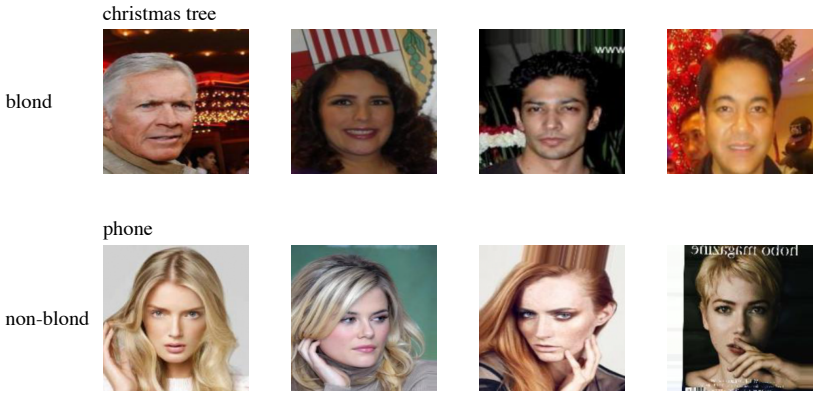


Figure A.4.2: Samples selected based on the two detected attributes, `christmas tree` and `phone`. Although these attributes are not self-explanatory in representing the selected samples, samples selected by them have some common characteristics.

Training ERM Models

Our method starts with an ERM model and retrains it in an adapted form using the proposed techniques. Table A.4.4 shows the detailed settings for training ERM models on the four datasets. Note that for the NICO dataset, since the training data is limited, we did not use the training data to train an ERM model; instead, we followed the setting in [183] to only initialize a model for the later LBC training with ImageNet pre-trained weights. For fair comparison with existing methods, we adopted ResNet-50 as the backbone for experiments on the Waterbirds and CelebA datasets and adopted ResNet-18 as the backbone for experiments on the NICO and ImageNet-9 datasets. All images are resized to 224×224 resolution. Standard data augmentations, i.e., RandomResizedCrop and RandomHorizontalFlip, were used in training these models. Models that achieved the best validation accuracy were saved as the final ERM models.

Training LBC Models

To train our LBC models, we used a stochastic gradient descent optimizer with a momentum of 0.9 and a weight decay of 10^{-4}. The key hyperparameter settings and model selection criteria for training on the Waterbirds, CelebA, NICO, and ImageNet-9 datasets are shown in Table A.4.5. We used the ERM-trained models to initialize our LBC models. Standard data augmentations are used during training. The pseudo unbiased validation accuracy exploits detected attributes and is defined in Section 4.2.3. In each training epoch, we generated training batches by sampling with replacement in case we could not find enough samples under our within- and cross-class balancing techniques proposed in Section 4.2.3.

Time Costs for Extracting Attributes

The VLM and the attribute extractor are only used once for offline data preparation on the training and validation splits of a dataset. The attribute extractor performs a single pass on the texts to find informative words with a linear time complexity. Thus, the overall time complexity wouldn't be a major concern, compared with costly human annotations. Specifically, the total processing time using ViT-GPT2 on a single NVIDIA RTX 8000 GPU for each of the datasets is in the table below.

Datasets	Time
Waterbirds	9.7min
CelebA	4.6h
NICO	16.2min
ImageNet-9	1.5h

Table A.4.6: Time costs for extracting attributes from the four datasets.

Dataset	Metric	$\tanh(\text{Abs}(\log(\eta)))$	$\tanh(\log(\eta))$	$\text{Abs}(\log(\eta))$	$\log(\eta)$	$\text{Abs}(\delta)$	δ
Waterbirds	Pseudo unbiased validation accuracy	**95.1**	94.3	94.9	95.0	94.7	94.4
	Average test accuracy	**93.2**	89.8	91.3	92.2	92.1	91.0
	Worst-group test accuracy	**87.3**	79.2	82.9	85.1	85.0	81.7
CelebA	Pseudo unbiased validation accuracy	**94.6**	94.5	94.3	94.4	94.3	94.3
	Average test accuracy	92.2	92.9	93.0	**93.3**	92.8	93.3
	Worst-group test accuracy	**81.2**	78.1	79.1	79.7	80.8	78.8

Table A.4.7: Comparison between different designs of spuriousness scores. We ran experiments using different scores for 5 times on the Waterbirds and CelebA datasets and calculated the average performance under different metrics.

A.4.4 Attributes with High Spuriousness Scores

We give 10 spurious attributes with the highest spuriousness scores for each class of the CelebA, NICO, and ImageNet-9 datasets. As discussed in Section 4.3.3 in the main paper, not all of these attributes are self-explanatory; some of them may represent features that cannot be described by themselves. In general, these spurious attributes are not directly related to their corresponding classes.

CelebA.

Non-blond hair: sun, umbrella, pretty, flag, lady, sky, blonde, ear, long, tooth

Blond hair: apple, flag, right, animal, blow dryer, blow, dryer, bottle, hand, scarf.

NICO.

Dog: ground, snow, white, green, lush, road, grassy, grass, sheep, side.

Cat: food, painting, snow, feeder, yellow, bird feeder, wood, colorful, parrot, seagull.

Bear: grass, floor, animal, wire fence, bear, wire, person, hand, piece, branch.

Bird: rock, bunch, grass, man, animal, bowl, large, room, chair, table.
Cow: snow, beach, brown, top, woman, field, sandy, back, white, people. *Elephant*: leave, river, middle, tree, herd, man, stage, fence, body, baby elephant.
Horse: group, banana, window, rock, picture, pile, people, plate, face, sign.
Monkey: cat, beach, ground, hand, animal, snow, small, white, body, water.
Rat: elephant, snow, man, body, herd, cow, cattle, water, field, white.
Sheep: cement, bunch, plant, gray, wooden, banana, teddy, teddy bear, post, monkey.
ImageNet-9.
Dog: cell phone, phone, cell, right, desk, plant, cage, hand, picture, log.
Cat: dirt road, statue, road, dirt, laptop, man, woman, bear, cat, large.
Frog: woman, young, flower pot, boy, little, bunch, flower, girl, pot, body.
Turtle: painting, boat, group, leave, dead, pile, animal, body, picture, beach.
Bird: forest, mouth, middle, bird, white, duck, water, colorful, hummingbird, feeder
Primate: trash can, trash, collage, can, parrot, flock, air, squirrel, dirt road, baby.
Fish: fence, surfboard, shot, dog, right, person, fire hydrant, hydrant, fire, hand.
Crab: view, beach scene, scene, flower, group, people, body, close, water, bunch.
Insect: face, woman, front, knife, object, banana, hand, dog, person, animal.

A.4.5 Different Designs of Spuriousness Score

We show the performance comparison of six variants of spuriousness score on the Waterbirds and CelebA datasets in Table A.4.7, where

$$\delta = M(\mathcal{D}_{tr}^{\langle c,a\rangle}; f_\theta) - M(\mathcal{D}_{tr}^{\langle c,\hat{a}\rangle}; f_\theta), \tag{A.12}$$

$$\eta = M(\mathcal{D}_{tr}^{\langle c,a\rangle}; f_\theta)/M(\mathcal{D}_{tr}^{\langle c,\hat{a}\rangle}; f_\theta), \tag{A.13}$$

and $M(\cdot;\cdot)$ is the accuracy measure used in Equation (4.17). The models used for testing are selected based on the pseudo unbiased validation accuracy defined in Section 4.2.3. We observe that taking the simple difference between the accuracies $M(\mathcal{D}_{tr}^{\langle c,a\rangle}; f_\theta)$ and $M(\mathcal{D}_{tr}^{\langle c,\hat{a}\rangle}; f_\theta)$) is not as effective as taking the logarithm of their ratio. Therefore, adding non-linearity into our design of spuriousness score is beneficial. Moreover, tanh and Abs together further improve the average and worst-group test accuracies of our proposed method on the Waterbirds dataset. On the CelebA dataset, the default score, i.e., $\tanh(\text{Abs}(\log(\eta)))$, achieves the best pseudo unbiased validation accuracy, which favors a model that achieves the best worst-group test accuracy. Overall, our spuriousness score works well with the pseudo unbiased validation accuracy in selecting a model that is most robust to spurious correlations in terms of worst-group test accuracy and has competitive average performance.

A.4.6 Analysis Based on Spuriousness Score

We additionally show the spuriousness scores of the attributes detected within the non-blond and blond classes in the CelebA dataset before (denoted as ERM) and after applying our proposed LBC. The high maximum score in Figure A.4.3(b) shows that for the ERM model, predicting the blond class heavily relies on spurious correlations, while predicting the non-blond class is relatively robust to spurious correlations as the maximum score in Figure A.4.3(a) is small. This also aligns with our empirical observation that the ERM model struggles in predicting the blond class. Figure A.4.3(c) shows that some of the prediction behaviors (orange points) for predicting images from the blond class are similar to those leading to the non-blond class, offering insights into why the ERM model performs poorly on predicting the blond class.

After our LBC retraining, as shown in Figure A.4.3(e), the reliance on spurious correlations is significantly reduced. However, as a side effect, the reliance on spurious correlations increases for predicting the non-blond class, as observed in Figure A.4.3(d). As a result, for images in the non-blond class, we observe dense clusters in Figure A.4.3(f) with each cluster representing similar prediction behaviors which use certain spurious correlations for predictions. Interestingly, we observe that images in the blond class are more concentrated in the spuriousness embedding space after

our LBC retraining, indicating more consistent prediction behaviors on the class. This improved consistency comes at the cost of increased inconsistency in the predictions of the non-blond class images, as we observe that several non-blond class images (blue points) are close to the orange cluster. Given that the non-blond class is the majority class, while the blond class is the minority class, the increased consistency in predicting blond class images improves the performance on the blond class images reflected by the increased worst-group accuracy. At the same time, the average accuracy dominated by the non-blond class images decreases due to the increased inconsistency in the predictions of the non-blond class images. This average and worst-group accuracy tradeoff is commonly observed in Table 4.7 in the main paper across different methods, and our spuriousness score can effectively reveal this tradeoff.

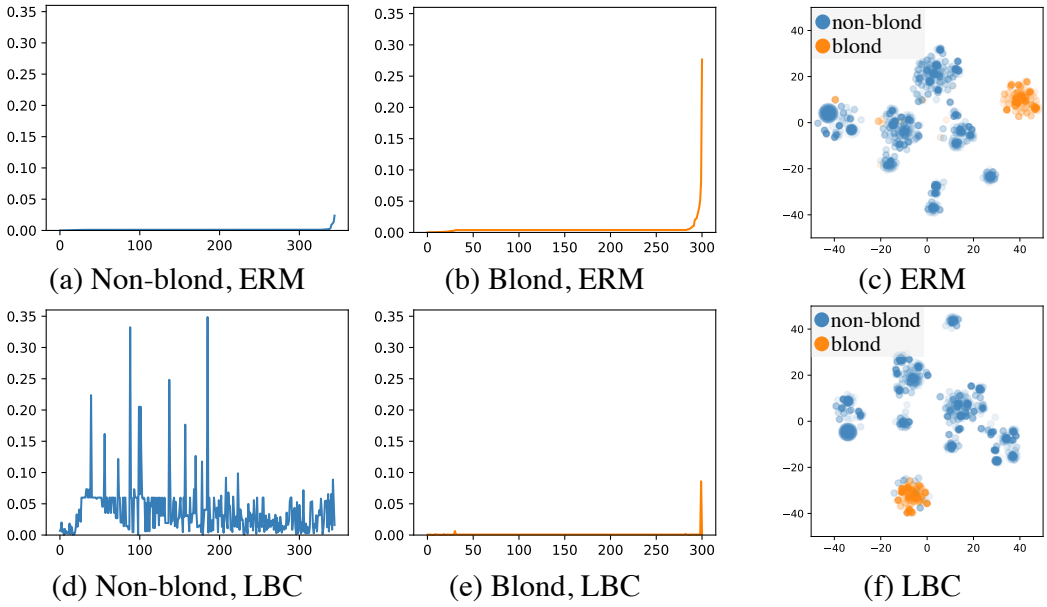


Figure A.4.3: (a) and (b): Spuriousness scores for the attributes detected from non-blond and blond based on an ERM model. (d) and (e): Spuriousness scores based on our LBC model. (c) and (f): Spurious embeddings of the images in the CelebA dataset based on the ERM and LBC model, respectively.

A.4.7 Analysis on Using ERM-Trained Models

Our method starts training using the initialization of an ERM-trained model. To investigate how different initializations affect the performance of our method, we tested three kinds of models used by our method: (1) a randomly initialized model, (2) an ERM model trained from scratch, and (3) an ERM model trained with ImageNet pre-trained weights. Table A.4.8 shows that LBC with a randomly initialized model does not perform well on the three evaluation metrics, because the randomly initialized model gives noisy information on the spuriousness of the detected attributes. LBC with an ERM model trained from scratch performs better than the first one thanks to the good initialization provided by the ERM-trained model. The ImageNet pre-trained weights contain knowledge about recognizing multiple objects and patterns. Therefore, when the ERM model is trained with ImageNet pre-trained weights, LBC performs the best on the three metrics.

A.4.8 Does the Performance Gain Come from the Attributes?

Since we used a VLM to detect attributes from training data, it is naturally to ask whether the performance gain comes from the detected attributes. We showed that the performance gain mainly comes from our proposed learning algorithm. Specifically, we added an *additional* layer after the

Method	ERM Model	ImageNet-9		ImageNet-A
		Validation (↑)	Unbiased (↑)	Test (↑)
LBC	Random initialization	46.38	43.92	15.73
LBC	Trained from scratch	93.71	92.14	39.65
LBC	Trained with ImageNet pre-trained weights	96.97	96.03	40.63

Table A.4.8: Performance comparison (%) between different choices of model initializations used in our method LBC on the ImageNet-9 and ImageNet-A datasets.

backbone to predict attributes for each image, and we trained the whole model on the Waterbirds and CelebA datasets, respectively. In other words, we added an additional attribute prediction loss term in Equation (4.15) for each image. Essentially, the attributes act as a regularization for the classifier. If the attributes contain information effective in improving a classifier's robustness to spurious correlations, we would observe improved performance after training.

The worst-group accuracies on the Waterbirds and CelebA datasets are 71.7% and 47.2%, respectively. Although this approach is slightly better than ERM, but it falls far behind our proposed LBC algorithm. Therefore, the detected attributes from the VLM alone do not contain information effective for improving a classifier's robustness to spurious correlations. In contrast, LBC directly identifies highly dependent spurious attributes for a classifier and mitigates the classifier's reliance on them, effectively improving the classifier's robustness to spurious correlations.

A.5 Spuriousness-Aware Meta-Learning for Learning Robust Classifiers

A.5.1 Datasets

Table A.5.1 depicts detailed statistics for all datasets. For Waterbirds and CelebA datasets, we give the number of training, validation, and test images in each group specified by classes and attributes. For example, the group label (landbird, land) in the Waterbirds dataset has 3498 training images which are all landbird and have land backgrounds. NICO provides context labels as spurious attributes. ImageNet-9 and ImageNet-A datasets do not have clear group partitions specified by the class and attribute associations.

Dataset	Number of classes	⟨class, attribute⟩	Number of images		
			Train	Val	Test
Waterbirds	2	⟨landbird, land⟩	3,498	467	2,255
		⟨landbird, water⟩	184	466	2,255
		⟨waterbird, land⟩	56	133	642
		⟨waterbird, water⟩	1,057	133	642
CelebA	2	⟨non-blond, female⟩	71,629	8,535	9,767
		⟨non-blond, male⟩	66,874	8,276	7,535
		⟨blond, female⟩	22,880	2,874	2,480
		⟨blond, male⟩	1,387	182	180
NICO	10	⟨object, context ⟩	10298	642	894
ImageNet-9	9	-	54,600	2,100	-
ImageNet-A	9	-	-	-	1087

Table A.5.1: Detailed statistics of the 5 datasets. ⟨class, attribute⟩ represents a spurious correlation between a class and a spurious attribute. "-" denotes not applicable.

Class	Contexts	
	Validation	Test
dog	running	in_street
cat	on_tree	in_street
bear	on_tree	white
bird	on_shoulder	in_hand
cow	spotted	standing
elephant	in_circus	in street
horse	running	in_street
monkey	climbing	sitting
rat	running	in_hole
sheep	at_sunset	on_road

Table A.5.2: Classes and their associated contexts in the NICO datasets. Contexts not shown in the table are used in the training set.

NICO [148] is a real-world dataset for out-of-distribution robustness. We used its Animal subset containing 10 object classes and 33 context labels. Following the setting in [186, 190], we split the dataset into training, validation, and test sets with each set having unique contexts. Table A.5.2 gives the allocation of the contexts for the 10 classes.

Dataset	Learning rate	Learning rate scheduler	Number of tasks per epoch	Training epochs	τ	Model selection metric
Waterbirds	1e-3	Cosine Annealing	80	100	5	Acc_{pu}
CelebA	1e-3	Cosine Annealing	80	100	5	Acc_{pu}
NICO	5e-3	Cosine Annealing	80	50	10	Validation accuracy
ImageNet-9	1e-3	Cosine Annealing	80	50	50	Validation accuracy

Table A.5.3: Hyperparameter settings and model selection criteria for SPUME training on the Waterbirds, CelebA, NICO, and ImageNet-9 datasets. Acc_{pu} denotes pseudo unbiased validation accuracy.

The ImageNet-9 dataset [181] is a subset of ImageNet. It has 9 super-classes, i.e., Dog, Cat, Frog, Turtle, Bird, Primate, Fish, Crab, Insect, which are obtained by merging similar classes from ImageNet. ImageNet-A contains real-world images that are challenging to the image classifiers trained on standard ImageNet. We extract images of the 9 super-classes from the ImageNet-A dataset and use these images as the test data.

A.5.2 Experimental Details

VLM Settings. For both ViT-GPT2 and BLIP, we set the maximum length of the sequence to be generated as 16 and the number of beams for beam search to 4.

Training Details. We initialize ResNet-50 and ResNet-18 using ImageNet pre-trained weights. Standard data augmentations, i.e., `RandomResizedCrop` and `RandomHorizontalFlip` are used during model training. We use an SDG optimizer with a momentum of 0.9 and a weight decay of 10^{-4} during meta-training. The detailed training configurations are shown in Table A.5.3.

A.5.3 Baselines

We briefly summarize and describe the baselines which are compared in the experiments:
Group DRO [3] proposes to train the models on the worst-case loss over a set of predefined groups.
ReBias [181] proposes a novel framework to train a de-biased representation by encouraging it to be different from a set of biased representations.
REx [188] proposes a min-max algorithm to optimize for the worst linear combination of risks on different environments.
LfF [33] proposes a failure-based debiasing scheme by training a pair of neural networks: the first network to be biased by repeatedly amplifying its "prejudice" and debias the training of the second network by focusing on samples that counter the first network.
CVaR DRO [171] is an algorithm for distributionally robust optimization of convex losses with conditional value at risk (CVaR) and χ^2 divergence uncertainty sets.
JTT [31] proposes a simple two-stage approach that first trains a standard ERM model and then trains a second model by upweighting the training examples misclassified by the first model.
DFR [4] retrains the last linear layer on a small held-out dataset with balanced groups of data.
CaaM [182] learns causal features that are robust in any confounding context and self-annotates the confounders in an unsupervised fashion.
LWBC / SSL+ERM [183] employs a committee of classifiers as an auxiliary module that identifies bias-conflicting data and assigns large weights to them when training the main classifier. SSL+ERM is another approach proposed in this paper that uses self-supervised representation as the frozen backbone of the committee and the main classifier.
MaskTune [173] employs an interpretation-based masking strategy that mitigates over-reliance on spurious features. It forces the trained model to explore new features during a single epoch fine-tuning by masking previously discovered features.

DivDis [172] is a simple two-stage framework for identifying and resolving ambiguity in data. It first learns a diverse set of hypotheses and then disambiguates them by selecting one of the discovered functions using additional information (e.g. target labels).
JiGen [111] jointly classifies objects and solves unsupervised jigsaw tasks.
Mixup [189] trains a neural network on convex combinations of pairs of examples and their labels to alleviate memorization and sensitivity to adversarial examples in deep neural networks.
CNBB [148] is a non-independent and identically distributed (Non-I.I.D) learning method that is based on batch balancing inspired by causal inference.
DecAug [186] proposes a semantic augmentation and feature decomposition approach to disentangle context features from category-related features.
SIFER [187] automatically identifies and suppresses easily-computable spurious features in lower layers of the network and allows the higher layers of the network to extract and utilize more meaningful representations.

A.5.4 Analyzing the Effects of Using VLMs

Using the Outputs of VLMs as Regularization. We added a linear layer with weights $\mathbf{W}_A \in \mathbb{R}^{|\mathcal{A}|\times D}$ and bias $\mathbf{b}_A \in \mathbb{R}^{|\mathcal{A}|}$ after the backbone to predict the detected attributes for each image, i.e.,

$$\tilde{\theta} = \arg\min_{\theta} \mathbb{E}_{(x,y)\in\mathcal{D}_{\text{tr}}} \ell(f_\theta(x), y) + \sum_{a\in\psi(\phi(x))} \ell'(f'_{\theta'}(x), a) \tag{A.14}$$

where $f'_{\theta'}(x) = \mathbf{W}_A h_{\theta_1}(x) + \mathbf{b}_A$, and $\ell'(\cdot,\cdot)$ is the binary entropy loss function. We trained the whole model on the Waterbirds and CelebA datasets, respectively. If the attributes contain information effective in improving a classifier's robustness to spurious correlations, we will observe improved performance after training. However, the worst-group accuracies on the Waterbirds and CelebA datasets are 71.7% and 47.2%, respectively, which are only slightly better than those of ERM and fall far behind the results of SPUME. Therefore, the detected attributes from the VLM alone do not contain information effective for improving a classifier's robustness to spurious correlations.

Directly Using VLMs for Predictions. Although the goal of this paper is to learn a classic and resource-light classifier that is robust to spurious correlations, we explored the scenario when BLIP is directly used for prediction with modifications on the inference paradigm. Specifically, we used text embeddings of the sentences with the template "a photo of `class_label`" ("a person with `hair_color` hair" for CelebA) from BLIP as the classifier weights and calculated the cosine similarity between an image embedding and these weights in the shared embedding space of BLIP. We predicted the label such that its corresponding sentence has the highest similarity to the image embedding. The worst group accuracies on the Waterbirds and CelebA datasets are 1.17% and 29.71% respectively. The average accuracies on the NICO, ImageNet-9, and ImageNet-A datasets are 14.30%, 13.43%, and 9.20%, respectively. Directly using the VLM without carefully tuning the inference pipeline performs much worse than our proposed method. In contrast, our proposed method SPUME exploits the attributes provided by VLMs in a novel way for significant improvement in the robustness of a classifier to spurious correlations.

A.6 ShortcutProbe: Probing Prediction Shortcuts for Learning Robust Models

A.6.1 Proof of Proposition 1

The proposition uses the result from Lemma 1 in our main paper, which is a restatement of Equation (4.4) in [191]. To prove the proposition, we first show that $\mathbf{OO} = \mathbf{O}$. Note that $\mathbf{O} = \mathbf{I} - \mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V}$, thus we have

$$\begin{aligned}\mathbf{OO} &= \big(\mathbf{I} - \mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V}\big)\big(\mathbf{I} - \mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V}\big) \\ &= \mathbf{I} - 2\mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V} + \mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V} \\ &= \mathbf{I} - \mathbf{V}^T(\mathbf{V}\mathbf{V}^T)^{-1}\mathbf{V} = \mathbf{O}. \end{aligned} \tag{A.15}$$

Next, we expand $\varphi(\tilde{x})^T\mathbf{O}\varphi(x)$ as follows

$$\varphi(\tilde{x})^T\mathbf{O}\varphi(x) = \varphi(\tilde{x})^T\mathbf{O}\mathbf{O}\varphi(x) \tag{A.16}$$

$$= (\mathbf{O}\varphi(\tilde{x}))^T(\mathbf{O}\varphi(x)) \tag{A.17}$$

$$\leq \|\mathbf{O}\varphi(\tilde{x})\|_2\|\mathbf{O}\varphi(x)\|_2, \tag{A.18}$$

where Equation (A.17) uses the fact that $\mathbf{O}^T = \mathbf{O}$, and (A.18) is the result of Cauchy–Schwarz inequality.

The inequality (A.18) holds in general. However, the equality actually holds in our setting. To show this, we need to prove that the vectors $\mathbf{O}\varphi(\tilde{x})$ and $\mathbf{O}\varphi(x)$ are independent. We first note that a spurious sample $\tilde{x}$ is independent of an original sample x. For example, in the Waterbirds dataset [3], let $\tilde{x}$ represent an image showing only a water background, and $\tilde{x}$ is independent of x, as $\tilde{x}$ may be obtained by removing core objects from images of landbirds or waterbirds with waterbird backgrounds. Thus, the corresponding feature vectors $\varphi(x)$ and $\varphi(\tilde{x})$ are independent. If we assume that $\mathbf{O}\varphi(x)$ and $\mathbf{O}\varphi(\tilde{x})$ are dependent with $\mathbf{O}\varphi(x) = \eta\mathbf{O}\varphi(\tilde{x})$, where η is a non-zero constant, then we have $\varphi(x) = \eta\mathbf{O}^{-1}\mathbf{O}\varphi(\tilde{x}) = \eta\varphi(\tilde{x})$, which contradicts the fact that $\varphi(x)$ and $\varphi(\tilde{x})$ are independent. Therefore, $\mathbf{O}\varphi(x)$ and $\mathbf{O}\varphi(\tilde{x})$ are independent. Consequently, we have the following equality,

$$\varphi(\tilde{x})^T\mathbf{O}\varphi(x) = \|\mathbf{O}\varphi(\tilde{x})\|_2\|\mathbf{O}\varphi(x)\|_2. \tag{A.19}$$

Finally, we reinterpret the feature alignment γ_φ as follows,

$$\gamma_\varphi = \mathbb{E}_{\tilde{x},x}\Big[\frac{\varphi(\tilde{x})^T\mathbf{O}\varphi(x)}{\|\mathbf{O}\varphi(x)\|_2^2}\Big] \tag{A.20}$$

$$= \mathbb{E}_{\tilde{x},x}\Big[\frac{\|\mathbf{O}\varphi(\tilde{x})\|_2 \cdot \|\mathbf{O}\varphi(x)\|_2}{\|\mathbf{O}\varphi(x)\|_2 \cdot \|\mathbf{O}\varphi(x)\|_2}\Big] \tag{A.21}$$

$$= \frac{\mathbb{E}_{\tilde{x}}[\|\mathbf{O}\varphi(\tilde{x})\|_2]}{\mathbb{E}_x[\|\mathbf{O}\varphi(x)\|_2]}, \tag{A.22}$$

where Equation (A.22) results from the fact that the random variables x and $\tilde{x}$ are independent.

A.6.2 Learning Algorithm

We show the detailed training process of our proposed method, ShortcutProbe, in Algorithm 5. The algorithm is a two-step procedure. In the first step, we train a shortcut detector, and in the second step, we use the prediction shortcuts detected by the shortcut detector to mitigate spurious biases in the model.

Complexity Analysis. Given that the time complexity for obtaining $\mathcal{D}^y_{\text{cor}}$, $\mathcal{D}^y_{\text{pre}}$, and $\mathcal{D}^y_{\text{mis}}$ is C_{data}, the time complexity for each batch update during the shortcut detector training is C_{det},

Algorithm 5 ShortcutProbe

Input: Probe set $\mathcal{D}_{\text{prob}}$, parameters of an ERM-trained model θ including θ_1 of the feature extractor and θ_2 of the classifier, parameters of the shortcut detector $\psi = \mathbf{A}$ with K base vectors, batch size B, number of batches per epoch N_B, number of training epochs for the first step E_1, learning rate α used in the first step, number of training epochs for the second step E_2, learning rate β used in the second step, regularization strengths η and λ.

Output: the classifier's weights θ_2

Obtain $\mathcal{D}^y_{\text{cor}}$, $\mathcal{D}^y_{\text{pre}}$, and $\mathcal{D}^y_{\text{mis}}$ for each class y from $\mathcal{D}_{\text{prob}}$ using Equation (5.2) and Equation (5.3), respectively
//Learn shortcut detector
for $e = 1, \ldots, E_1$ **do**
  for $b = 1, \ldots, N_B$ **do**
    Sample $\mathcal{B}^y_{\text{cor}} \subset \mathcal{D}^y_{\text{cor}}$ and $\mathcal{B}^y_{\text{pre}} \subset \mathcal{D}^y_{\text{pre}}$, $\forall y \in \mathcal{Y}$, with $|\mathcal{B}^y_{\text{cor}}| = |\mathcal{B}^y_{\text{pre}}|$ and $\sum_{y\in\mathcal{Y}}(|\mathcal{B}^y_{\text{pre}}| + |\mathcal{B}^y_{\text{cor}}|) = B$
    Calculate $\psi = \psi - \alpha\nabla_\psi(\mathcal{L}_{\text{det}} + \eta\mathcal{L}_{\text{reg}})$ using $\mathcal{B}^y_{\text{cor}}$ and $\mathcal{B}^y_{\text{pre}}$
  end for
end for
//Mitigate spurious biases
for $e = 1, \cdots, E_2$ **do**
  for $b = 1, \cdots, N_B$ **do**
    Sample $\mathcal{B}^y_{\text{cor}} \subset \mathcal{D}^y_{\text{cor}}$ and $\mathcal{B}^y_{\text{mis}} \subset \mathcal{D}^y_{\text{mis}}$, $\forall y \in \mathcal{Y}$, with $|\mathcal{B}^y_{\text{cor}}| = |\mathcal{B}^y_{\text{mis}}|$ and $\sum_{y\in\mathcal{Y}}(|\mathcal{B}^y_{\text{mis}}| + |\mathcal{B}^y_{\text{cor}}|) = B$
    Calculate $\theta_2 = \theta_2 - \beta\nabla_{\theta_2}\lambda\mathcal{L}_{\text{ori}}/\mathcal{L}_{\text{spu}}$ using $\mathcal{B}^y_{\text{cor}}$ and $\mathcal{B}^y_{\text{mis}}$
  end for
end for
return θ_2

and the time complexity for each batch update during classifier retraining is C_{ret}, the overall time complexity is $O(C_{\text{data}} + E_1 N_B C_{\text{det}} + E_2 N_B C_{\text{ret}})$.

Notably, the sets $\mathcal{D}^y_{\text{cor}}$, $\mathcal{D}^y_{\text{pre}}$, and $\mathcal{D}^y_{\text{mis}}$ can be precomputed before training and need to be constructed only once, allowing C_{data} to be omitted once these sets are available. Additionally, C_{det} and C_{ret} are typically very small due to the lightweight design of the shortcut detector and the retraining process, which only involves the model's final linear layer. Consequently, ShortcutProbe is highly computation-efficient. We provide a run-time comparison between different debiasing methods in Table A.6.1 below.

JTT	DFR	AFR	ShortcutProbe
1440	162	230	210

Table A.6.1: Training time (s) comparison on the Waterbirds dataset.

A.6.3 Datasets

Table A.6.2 gives detailed statistics for all the eight datasets. We give the number of training, validation, and test images in each group specified by classes and attributes for the Waterbirds, CelebA, MultiNLI, and CivilComments datasets. For example, the group label ⟨landbird, land⟩ in the Waterbirds dataset has 3498 training images which are all landbirds and have land backgrounds.

NICO [148] is a real-world dataset for evaluating a method's out-of-distribution generalization performance. NICO provides context labels as spurious attributes. We used its Animal subset containing 10 object classes and 33 context labels. Following the setting in [186, 190], we split the dataset into training, validation, and test sets with each set having unique contexts. Table A.6.3 gives the allocation of the contexts for the 10 classes.

Dataset	Number of classes	⟨class, attribute⟩	Number of images		
			Train	Val	Test
Waterbirds [3]	2	⟨landbird, land⟩	3,498	467	2,255
		⟨landbird, water⟩	184	466	2,255
		⟨waterbird, land⟩	56	133	642
		⟨waterbird, water⟩	1,057	133	642
CelebA [149]	2	⟨non-blond, female⟩	71,629	8,535	9,767
		⟨non-blond, male⟩	66,874	8,276	7,535
		⟨blond, female⟩	22,880	2,874	2,480
		⟨blond, male⟩	1,387	182	180
NICO [148]	10	⟨object, context ⟩	10298	642	894
ImageNet-9 [185]	9	N/A	54,600	2,100	N/A
ImageNet-A [176]	9	N/A	N/A	N/A	1087
CheXpert [192]	2	⟨diagnose, race+gender ⟩	167093	22280	33419
MultiNLI [194]	3	⟨contradiction, no negation⟩	57498	22814	34597
		⟨contradiction, negation⟩	11158	4634	6655
		⟨entailment , no negation⟩	67376	26949	40496
		⟨entailment, negation⟩	1521	613	886
		⟨neither, no negation⟩	66630	26655	39930
		⟨neither, negation⟩	1992	797	1148
CivilComments [195]	2	⟨neutral , no identity⟩	148186	25159	74780
		⟨neutral , identity⟩	90337	14966	43778
		⟨toxic , no identity⟩	12731	2111	6455
		⟨toxic , identity⟩	17784	2944	8769

Table A.6.2: Detailed statistics of the 8 datasets. ⟨class, attribute⟩ represents a spurious correlation between a class and a spurious attribute. "N/A" denotes not applicable.

ImageNet-9 [181] is a subset of ImageNet, and ImageNet-A contains real-world images that are challenging to the image classifiers trained on standard ImageNet. Both datasets do not have clear group partitions specified by the class and attribute associations. ImageNet-9 has 9 super-classes, i.e., Dog, Cat, Frog, Turtle, Bird, Primate, Fish, Crab, Insect, obtained by merging similar classes from ImageNet. We extract images of the 9 super-classes from the ImageNet-A dataset and use these images for testing.

The CheXpert dataset [192] is a chest X-ray dataset from the Stanford University Medical center. There are six spurious attributes in the dataset, each of them is the combination of race (White, Black, Other) and gender (Male, Female). Two diagnose results, i.e., "No Finding" (positive) and "Finding" (negative) are the labels.

A.6.4 Training Details

ERM Training. This step trains ERM models which serve as the base models used in our framework for detecting prediction shortcuts and mitigating spurious biases. The training hyperparameters as well as the optimizer and learning rate scheduler used for each dataset are given in Table A.6.4. For vision models, we initialized them with ImageNet-pretrained weights. For text models, we initialized them with weights pretrained on Book Corpus and English Wikipedia data.

Training ShortcutProbe Models. We provide hyperparameter settings for the experiments on the Waterbirds, CelebA, CheXpert, MultiNLI, CivilComments, ImageNet-9, and NICO datasets in Table A.6.5. We used an SGD optimizer with a momentum of 0.9 and a weight decay of

Class	Contexts	
	Validation	Test
dog	running	in_street
cat	on_tree	in_street
bear	on_tree	white
bird	on_shoulder	in_hand
cow	spotted	standing
elephant	in_circus	in street
horse	running	in_street
monkey	climbing	sitting
rat	running	in_hole
sheep	at_sunset	on_road

Table A.6.3: Classes and their associated contexts in the NICO datasets. Contexts not shown in the table are used in the training set.

Dataset	Batch size	Epochs	Initial learning rate	Weight decay	Learning rate scheduler	Optimizer
Waterbirds	32	100	0.003	0.0001	CosineAnnealing	SGD
CelebA	128	20	0.003	0.0001	CosineAnnealing	SGD
CheXpert	128	20	0.003	0.0001	CosineAnnealing	SGD
MultiNLI	16	10	0.00001	0.0001	Linear	AdamW
CivilComments	16	10	0.00001	0.0001	Linear	AdamW
NICO	128	100	0.003	0.0001	CosineAnnealing	SGD
ImageNet-9	128	100	0.003	0.0001	CosineAnnealing	SGD

Table A.6.4: Training settings for training ERM models on different datasets.

1×10^{-4} in training the shortcut detector and retraining the classifier. We chose K from $\{2, 4, 6, 8\}$, η from $\{0.1, 1.0, 5.0, 10.0\}$, λ from $\{0.1, 1.0, 5.0, 10.0, 50.0\}$, and N_B from $\{50, 100, 200\}$, β from $\{0.0001, 0.0005, 0.001, 0.003, 0.01\}$, and r from $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. We selected the best hyperparameters based on the performance on the whole validation set if the training data was used to construct the probe set or the remaining validation set if part of the validation set was used for the construction. The remaining hyperparameters were determined based on our empirical observations considering both dataset size and the convergence of training.

Training Baseline Models. For JTT [31], we combined the training data with half of the validation data to create a new training set for training JTT models. For DFR [4] and AFR [32], we applied these methods to the same ERM-trained model to ensure a fair comparison. We adhered to the hyperparameter settings recommended in the respective original papers.

A.6.5 Additional Results

ImageNet-9 and ImageNet-A. We presents performance comparison on the ImageNet-9 and ImageNet-A datasets in Table A.6.7. The validation accuracy measures the in-distribution performance of a model, while the accuracy gap measures the performance drop from ImageNet-9 to ImageNet-A. Images in the ImageNet-A dataset represent distribution shifts from the training images in the ImageNet-9 dataset. Thus, the accuracy on the ImageNet-A dataset measures a model's performance under distribution shifts. As shown in Table A.6.7, our method achieves the best on the ImageNet-A dataset, demonstrating its robustness to distribution shifts. It also exhibits a good tradeoff between in-distribution and out-of-distribution performance by achieving the best accuracy gap.

Dataset	K	η	λ	E_1	E_2	B	N_B	α	β	r
Waterbirds	2	5.0	5.0	50	50	32	200	0.0001	0.001	0.3
CelebA	2	5.0	5.0	50	50	128	100	0.0001	0.001	0.1
CheXpert	6	10.0	50.0	50	50	128	50	0.0001	0.003	0.1
MultiNLI	2	5.0	5.0	50	50	128	100	0.0001	0.001	0.1
CivilComments	6	1.0	1.0	50	50	128	100	0.0001	0.003	0.1
NICO	8	1.0	1.0	50	50	128	200	0.0001	0.001	-
ImageNet-9	4	1.0	1.0	50	50	128	200	0.0001	0.001	-

Table A.6.5: Hyperparameter settings for experiments on the seven datasets. K: number of base vectors; η: regularization strength for the semantic similarity constraint in Equation (5.5); λ: regularization strength used in the training objective in Equation (5.10); E_1: number of training epochs for learning the shortcut detector; E_2: number of training epochs for retraining the classifier; B: batch size; N_B: number of batches sampled in each epoch; α: learning rate for learning the shortcut detector; β: learning rate for retraining the classifier; r: proportion of samples used to construct the probe set. When r is not specified ("-"), it means using the training data to construct the probe set.

ResNet-152 and ViT Backbones. Our method can be easily applied to larger backbone networks beyond ResNet-50, such as ResNet-152 and ViT. We evaluated our method with ResNet-152 and ViT-B/32 on three vision datasets and provide a performance comparison with baseline methods in Table A.6.6 below. We observe that our method remains highly effective on large-scale models.

Backbone	Method	Waterbirds	CelebA	Chexpert
ResNet-152	ERM	17.4	60.6	18.6
	DFR	30.3	68.3	66.0
	AFR	31.4	70.7	63.8
	Ours	**34.7**	**80.0**	**68.3**
ViT-B/32	ERM	69.5	52.2	20.9
	DFR	87.6	63.0	73.4
	AFR	86.6	79.5	64.5
	Ours	**88.0**	**83.7**	**75.1**

Table A.6.6: Comparison of worst-group accuracy (%) across last-layer retraining methods using ResNet-152 and ViT backbones.

A.6.6 Qualitative Analysis of Learned Prediction Shortcuts

To qualitatively analyze the detected prediction shortcuts, we aim to interpret the learned base vectors in the matrix $\mathbf{A}$, as prediction shortcuts are obtained by linearly combining these vectors. To achieve this, for each base vector, we gave the top-5 images whose prediction shortcuts are most similar to the base vector. Specifically, for each learned base vector, we first calculated the embeddings of training samples, and following Equation (5.1), we extracted prediction shortcuts in those samples as projected vectors by projecting the embeddings to the subspace spanned by the learned base vectors. We ranked the images based on the similarity of their prediction shortcuts to the base vector. A large similarity score signals a strong existence of the feature the base vector represents in the corresponding image.

As shown in Figure A.6.1(a), on the Waterbirds dataset, the two learned base vectors are most similar to images with land backgrounds and water backgrounds, respectively. In Figure A.6.1(b), the two learned base vectors are most similar to images of male celebrities and female celebrities, respectively. These results show that our shortcut detector can learn spurious attributes that well align with the biases in the datasets which models tend to capture during training.

Method	ImageNet-9 (↑)	ImageNet-A (↑)	Acc. gap (↓)
ERM	$90.8_{\pm 0.6}$	$24.9_{\pm 1.1}$	65.9
ReBias [181]	$91.9_{\pm 1.7}$	$29.6_{\pm 1.6}$	62.3
LfF [33]	86.0	24.6	61.4
CaaM [182]	95.7	32.8	62.9
SSL+ERM [183]	$94.2_{\pm 0.1}$	$34.2_{\pm 0.5}$	60
LWBC[183]	$94.0_{\pm 0.2}$	$36.0_{\pm 0.5}$	58
SIFER [187]	$\mathbf{97.8}_{\pm 0.1}$	$40.0_{\pm 0.8}$	57.8
ShortcutProbe (Ours)	$96.9_{\pm 0.2}$	$\mathbf{45.3}_{\pm 1.2}$	**51.6**

Table A.6.7: Comparison of average accuracy (%) and accuracy gap (%) on the ImageNet-9 and ImageNet-A datasets.

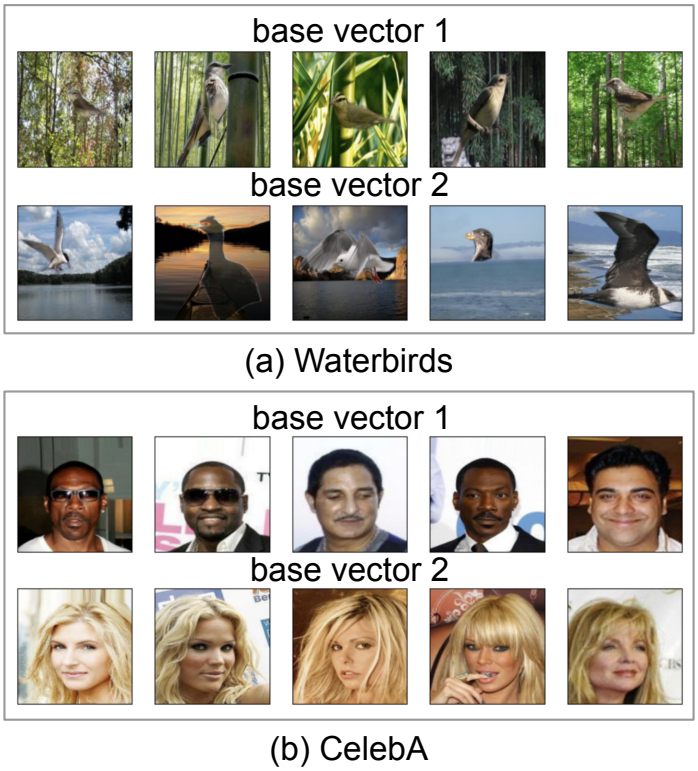


Figure A.6.1: Visualization of the top-5 images that are most similar to the learned base vectors from the (a) Waterbirds and (b) CelebA datasets.

A.7 NeuronTune: Towards Self-Guided Spurious Bias Mitigation

The appendix is organized as follows:

A.7.1 Details of the Synthetic Experiment

Data Model. Without loss of generality, we considered an input $\mathbf{v} \in \mathbb{R}^3$ to simulate a latent embedding before the last prediction layer, which consists of three dimensions: a core dimension with the core component $v^c \in \mathbb{R}$, a spurious dimension with the spurious component $v^s \in \mathbb{R}$, and a noise dimension with the noise component v^ϵ. We considered a dataset $\mathcal{D}^{\text{syn}} = \{(\mathbf{v}_i, y_i)\}_{i=1}^N$ of N sample-label pairs, where $y_i \in \{-1, +1\}$, $v_i^c = y_i + n_c$, and v^ϵ and n_c are zero-mean Gaussian noises with variances σ_ϵ^2 and σ_c^2, respectively. When $y_i = -1$, $v_i^s = 0 + n_s$ with the probability α and $v_i^s = 1 + n_s$ with the probability $1 - \alpha$; when $y_i = +1$, $v_i^s = 1 + n_s$ with the probability α and $v_i^s = 0 + n_s$ with the probability $1 - \alpha$, where n_s is an independent zero-mean Gaussian noise with the variance σ_s^2. To facilitate developing the spurious bias of using the correlation between v_i^s and y_i for predictions, we generated a training set $\mathcal{D}_{\text{train}}^{\text{syn}}$ with easy-to-learn spurious attributes by setting $\sigma_c^2 > \sigma_s^2$ and $\alpha \approx 1$ [169]. Thus, the correlations between v_i^s and y_i are predictive of αN labels. To demonstrate, we set $\sigma_c^2 = 0.6$, $\sigma_s^2 = 0.1$, $\sigma_\epsilon^2 = 0.1$, $\alpha = 0.95$, and $N = 5000$. We generated a test set $\mathcal{D}_{\text{test}}^{\text{syn}}$ with the same set of parameters except $\alpha = 0.1$. Now, spurious correlations between v_i^s and y_i are only predictive of a small portion of the test samples. Figure 5.5 shows four data groups along with their respective proportions in each class.

Classification Model. We considered a logistic regression model $\phi_{\tilde{\mathbf{w}}}(\mathbf{v}) = 1/(1+\exp\{-(\mathbf{w}^T\mathbf{v} + b)\})$, where $\tilde{\mathbf{w}} = [\mathbf{w}, b]$. The model predicts $+1$ when $\phi_{\tilde{\mathbf{w}}}(\mathbf{v}) > 0.5$ and -1 otherwise. We trained $\phi_{\tilde{\mathbf{w}}}$ on $\mathcal{D}_{\text{train}}^{\text{syn}}$ and tested it on $\mathcal{D}_{\text{test}}^{\text{syn}}$.

Spurious Bias. We observed a high average accuracy of 95.4% but a WGA of 66.2% (Figure 5.5(a) in the main paper) on the training data. The results show that the model heavily relies on the correlations that exist in the majority of samples and exhibits strong spurious bias. As expected,

the performance on the test data is significantly lower (Figure 5.5(a), right). The decision boundary (Figure 5.5(a), black lines) learned from the training data does not generalize to the test data.

Mitigation Strategy. Without group labels, it is challenging to identify and mitigate spurious bias in the model. We tackled this challenge by first finding that the distributions of values of an input dimension, together with the prediction outcomes for a certain class, provide discriminative information regarding the spuriousness of the dimension. (1) When the values for misclassified samples at the dimension are high, while values for the correctly predicted samples are low, this indicates that the absence of the dimension input does not significantly affect the correctness of predictions, while the presence of the dimension input does not generalize to certain groups of data. Therefore, the dimension tends to be a biased dimension. The plots in Figure 5.5(b) illustrate the value distributions of the first and second dimensions of input embeddings when $y_i = -1$. (2) In contrast, if the absence of the dimension input results in misclassification, then the dimension tends to represent a core attribute. The left plot of Figure 5.5(b) represents the first dimension of input embeddings when $y_i = -1$. Next, we retrained the model while suppressing the second and third dimensions. As a result, the retrained model has learned to balance its performance on both the training and test data with a significant increase in WGA on the test data (Figure 5.5(c)).

A.7.2 Theoretical Analysis

Preliminary

For the ease of readability, we restate the data model specified by (5.14) and (5.15) in the following

$$\mathbf{x} = \mathbf{x}_{\text{core}} \oplus \mathbf{x}_{\text{spu}} \in \mathbb{R}^{D\times 1},\ y = \boldsymbol{\beta}^T \mathbf{x}_{\text{core}} + \varepsilon_{\text{core}}, \tag{A.23}$$

and

$$\mathbf{x}_{\text{spu}} = (2a-1)\boldsymbol{\gamma} y + \boldsymbol{\varepsilon}_{\text{spu}}, a \sim \text{Bern}(p), \tag{A.24}$$

where $(2a-1) \in \{-1, +1\}$, $a \sim \text{Bern}(p)$ is a Bernoulli random variable, p is close to 1, $\varepsilon_{\text{core}}$ is a zero-mean Gaussian random variable with the variance η^2_{core}, and each element in $\boldsymbol{\varepsilon}_{\text{spu}}$ follows a zero-mean Gaussian distribution with the variance η^2_{spu}. We set $\eta^2_{\text{core}} \gg \eta^2_{\text{spu}}$ to facilitate the learning of spurious attributes. The model $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$ in Section 5.2.3 can be further expressed as follows,

$$\hat{y} = \sum_{i=1}^{M} b_i(\mathbf{x}^T_{\text{core}}\mathbf{w}_{\text{core},i} + \mathbf{x}^T_{\text{spu}}\mathbf{w}_{\text{spu},i}) = \mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} + \mathbf{x}^T_{\text{spu}}\mathbf{u}_{\text{spu}}, \tag{A.25}$$

where $\mathbf{w}_i^T \in \mathbb{R}^{1\times D}$ is the i'th row of $\mathbf{W}$, $\mathbf{w}_i^T = [\mathbf{w}^T_{\text{core},i}, \mathbf{w}^T_{\text{spu},i}]$ with $\mathbf{w}_{\text{core},i} \in \mathbb{R}^{D_1\times 1}$ and $\mathbf{w}_{\text{spu},i} \in \mathbb{R}^{D_2\times 1}$, $\mathbf{u}_{\text{core}} = \sum_{i=1}^{M} b_i\mathbf{w}_{\text{core},i}$, and $\mathbf{u}_{\text{spu}} = \sum_{i=1}^{M} b_i\mathbf{w}_{\text{spu},i}$. The loss function which we use to optimize $\mathbf{W}$ and $\mathbf{b}$ is

$$\ell_{\text{tr}}(\mathbf{W}, \mathbf{b}) = \frac{1}{2}\mathbb{E}_{(\mathbf{x},y)\in\mathcal{D}_{\text{train}}}\|f(\mathbf{x}) - y\|_2^2. \tag{A.26}$$

With the above definitions, the following lemma gives the optimal coefficients $\mathbf{u}^*_{\text{core}}$ and $\mathbf{u}^*_{\text{spu}}$ based on the training data.

Proof of Lemma 1

Lemma 1. Given a training dataset $\mathcal{D}_{\text{train}}$ with p defined in (A.24) satisfying $1 \geq p \gg 0.5$, the optimized weights in the form of $\mathbf{u}^*_{\text{core}}$ and $\mathbf{u}^*_{\text{spu}}$ are

$$\mathbf{u}^*_{\text{core}} = \frac{(2-2p)\eta^2_{\text{core}} + \eta^2_{\text{spu}}}{\eta^2_{\text{core}} + \eta^2_{\text{spu}}}\boldsymbol{\beta}, \tag{A.27}$$

and

$$\mathbf{u}^*_{\text{spu}} = \frac{(2p-1)\eta^2_{\text{core}}}{\eta^2_{\text{core}} + \eta^2_{\text{spu}}}\boldsymbol{\gamma}, \tag{A.28}$$

respectively. When $p = 0.5$, the training data is unbiased and we obtain an unbiased classifier with weights $\mathbf{u}^*_{\text{core}} = \boldsymbol{\beta}$ and $\mathbf{u}^*_{\text{spu}} = 0$.

Proof. Note that $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x} = \mathbf{x}^T\mathbf{v} = \mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} + \mathbf{x}^T_{\text{spu}}\mathbf{u}_{\text{spu}}$, then we have

$$\ell_{\text{tr}}(W, b) = \frac{1}{2}\mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} + \mathbf{x}^T_{\text{spu}}\mathbf{u}_{\text{spu}} - y\|^2_2 \tag{A.29}$$

$$= \frac{1}{2}\mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} + \big[(2a-1)\boldsymbol{\gamma}y + \boldsymbol{\varepsilon}_{\text{spu}}\big]^T\mathbf{u}_{\text{spu}} - y\|^2_2 \tag{A.30}$$

$$= \frac{1}{2}\mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - \big[1 - (2a-1)\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}\big]y\|^2_2 + \frac{1}{2}\eta^2_{\text{spu}}\|\mathbf{u}_{\text{spu}}\|^2_2 \tag{A.31}$$

$$= \frac{1}{2}(pE_1 + (1-p)E_2) + \frac{1}{2}\eta^2_{\text{spu}}\|\mathbf{u}_{\text{spu}}\|^2_2, \tag{A.32}$$

where $E_1 = \|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})y\|^2_2$ when $a = 1$ and $E_2 = \|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})y\|^2_2$ when $a = 0$. We first calculate the lower bound for E_1 as follows

$$E_1 = \mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})(\boldsymbol{\beta}^T\mathbf{x}_{\text{core}} + \varepsilon_{\text{core}})\|^2_2 \tag{A.33}$$

$$= \mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})\boldsymbol{\beta}^T\mathbf{x}_{\text{core}} + (1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})\varepsilon_{\text{core}})\|^2_2 \tag{A.34}$$

$$= \mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}\|^2_2 + \eta^2_{\text{core}}(1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 \tag{A.35}$$

$$\geq \eta^2_{\text{core}}(1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2. \tag{A.36}$$

Similarly, we have

$$E_2 = \mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})(\boldsymbol{\beta}^T\mathbf{x}_{\text{core}} + \varepsilon_{\text{core}})\|^2_2 \tag{A.37}$$

$$= \mathbb{E}\|\mathbf{x}^T_{\text{core}}\mathbf{u}_{\text{core}} - (1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}\|^2_2 + \eta^2_{\text{core}}(1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 \tag{A.38}$$

$$\geq \eta^2_{\text{core}}(1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2. \tag{A.39}$$

Then, plug in (A.36) and (A.39) into (A.32), we obtain the following

$$\ell_{\text{tr}}(W, b) \geq \frac{1}{2}\Big(p\eta^2_{\text{core}}(1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + (1-p)\eta^2_{\text{core}}(1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + \eta^2_{\text{spu}}\|\mathbf{u}_{\text{spu}}\|^2_2\Big) \tag{A.40}$$

$$= \frac{1}{2}\Big(p\eta^2_{\text{core}}(1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + (1-p)\eta^2_{\text{core}}(1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + \eta^2_{\text{spu}}\|\boldsymbol{\gamma}\|^2_2\|\mathbf{u}_{\text{spu}}\|^2_2\Big) \tag{A.41}$$

$$\geq \frac{1}{2}\Big(p\eta^2_{\text{core}}(1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + (1-p)\eta^2_{\text{core}}(1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + \eta^2_{\text{spu}}\|\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}\|^2_2\Big), \tag{A.42}$$

where (A.41) uses the fact that $\boldsymbol{\gamma}$ has a unit norm, and the inequality (A.42) exploits the Cauchy–Schwarz inequality. Let $z = \boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}$, we have $\ell(z) = p\eta^2_{\text{core}}(1-z)^2 + (1-p)\eta^2_{\text{core}}(1+z)^2 + \eta^2_{\text{spu}}z^2$. Let $\frac{\partial\ell(z)}{\partial z} = 0$, we obtain

$$z^* = \gamma^T\mathbf{u}^*_{\text{spu}} = \frac{(2p-1)\eta^2_{\text{core}}}{\eta^2_{\text{core}} + \eta^2_{\text{spu}}}.$$

Given $\mathbf{u}^*_{\text{spu}}$, we can obtain the optimal $\mathbf{u}'_{\text{core}}$ for minimizing E_1 in (A.35) as $\mathbf{u}'_{\text{core}} = (1-z^*)\boldsymbol{\beta}$; similarly, we can obtain the optimal $\mathbf{u}''_{\text{core}}$ for minimizing E_2 in (A.38) as $\mathbf{u}''_{\text{core}} = (1+z^*)\boldsymbol{\beta}$. Via proof by contradiction, only $\mathbf{u}'_{\text{core}}$ or $\mathbf{u}''_{\text{core}}$ is the solution for $\mathbf{u}^*_{\text{core}}$. Since $p \gg 0.5$, E_1 contributes to the majority error of (A.35). Thus, $\mathbf{u}^*_{\text{core}} = (1-z^*)\boldsymbol{\beta}$, i.e.,

$$\mathbf{u}^*_{\text{core}} = (1-z^*)\boldsymbol{\beta} = \frac{(2-2p)\eta^2_{\text{core}} + \eta^2_{\text{spu}}}{\eta^2_{\text{core}} + \eta^2_{\text{spu}}}\boldsymbol{\beta}.$$

□

Proof of Corollary 1

Lemma 1 gives the optimal model weights under a given training dataset $\mathcal{D}_{\text{train}}$ with the parameter p controlling the strength of spurious correlations. Lemma 1 generalizes the result in [199] where $p = 1$. Importantly, we obtain the following corollary for unbiased models:

Corollary 1. The unbiased model $f(\mathbf{x}) = \mathbf{u}^T\mathbf{x} = \mathbf{x}_{\text{core}}^T\mathbf{u}_{\text{core}} + \mathbf{x}_{\text{spu}}^T\mathbf{u}_{\text{spu}}$ is achieved when $\mathbf{u}_{\text{core}} = \mathbf{u}_{\text{core}}^*$ and $\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}} = 0$.

Proof. Plug $\boldsymbol{\gamma}^T\mathbf{u}_{\text{core}} = 0$ into (A.35) and (A.38), then we observe that $\mathbf{u}_{\text{core}}$ minimizes errors from both the majority ($a = 1$) and minority ($a = 0$) groups of data. □

If we could obtain a set of unbiased training data with $p = 0.5$, then we obtain an unbiased model with $\mathbf{u}_{\text{spu}}^* = 0$ and $\mathbf{u}_{\text{core}}^* = \boldsymbol{\beta}$. However, in practice, it is challenging to obtain a set of unbiased training data, i.e., it is challenging to control the value of p.

Proof of Proposition 5.2

Proposition 5.2 (**Principle of NeuronTune**). Given the model $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$ trained with data generated under the data model specified in (A.23) and (A.24), it captures spurious correlations when $\boldsymbol{\gamma}^T\mathbf{w}_{\text{spu},i} < 0, i \in \{1, \ldots, M\}$. The principle of NeuronTune is to suppress neurons containing negative $\gamma^T\mathbf{w}_{\text{spu},i}$.

Proof. Consider the i'th neuron e_i ($i = 1, \ldots, M$) before the last layer. We first expand it based on our data model specified by (A.23) and (A.24) as follows:

$$
\begin{aligned}
e_i &= \mathbf{x}_{\text{core}}^T\mathbf{w}_{\text{core},i} + \mathbf{x}_{\text{spu}}^T\mathbf{w}_{\text{spu},i} && \text{(A.43)}\\
&= \mathbf{x}_{\text{core}}^T\mathbf{w}_{\text{core},i} + [(2a-1)\boldsymbol{\gamma}y + \boldsymbol{\varepsilon}_{\text{spu}}]^T\mathbf{w}_{\text{spu},i} && \text{(A.44)}\\
&= \mathbf{x}_{\text{core}}^T\mathbf{w}_{\text{core},i} + (2a-1)[\beta^T\mathbf{x}_{\text{core}} + \varepsilon_{\text{core}}]\gamma^T\mathbf{w}_{\text{spu},i} + \boldsymbol{\varepsilon}_{\text{spu}}^T\mathbf{w}_{\text{spu},i} && \text{(A.45)}\\
&= \mathbf{x}_{\text{core}}^T\mathbf{w}_{\text{core},i} + (2a-1)\beta^T\mathbf{x}_{\text{core}}\gamma^T\mathbf{w}_{\text{spu},i} + \varepsilon_{\text{rem}}, && \text{(A.46)}
\end{aligned}
$$

where $\varepsilon_{\text{rem}} = \varepsilon_{\text{core}}\gamma^T\mathbf{w}_{\text{spu},i} + \boldsymbol{\varepsilon}_{\text{spu}}^T\mathbf{w}_{\text{spu},i}$. In (A.46), if $\gamma^T\mathbf{w}_{\text{spu},i} \geq 0$, the model handles the spurious component correctly. Specifically, when $a = 1$, the spurious component positively correlates with the core component and contributes to the output, whereas when $a = 0$, its correlation with the core component breaks with a negative one and has a negative contribution to the output. In contrast, if $\gamma^T\mathbf{w}_{\text{spu},i} < 0$ and $a = 1$, then the model still utilizes the spurious component even the correlation breaks, demonstrating a strong reliance on the spurious component instead of the core component. Therefore, the principle of selective activation is to find neurons containing negative $\gamma^T\mathbf{w}_{\text{spu},i}$ so that suppress them improves the model's generalization. □

Proof of Theorem 5.1

The following theorem validates our neuron selection method.

Theorem 5.1 (**Metric for Neuron Selection**). Given the model $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$, we cast it to a classification model by training it to regress $y \in \{-\mu, \mu\}$ ($\mu > 0$) on $\mathbf{x}$ based on the data model specified in (A.23) and (A.24), where $\mu = \mathbb{E}[\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}]$. The metric δ_i^y defined in the following can identify neurons with spurious correlations when $\delta_i^y > 0$:

$$\delta_i^y = \text{Med}(\bar{\mathcal{V}}_i^y) - \text{Med}(\hat{\mathcal{V}}_i^y),$$

where $\bar{\mathcal{V}}_i^y$ and $\hat{\mathcal{V}}_i^y$ are the sets of activation values for misclassified and correctly predicted samples with the label y from the i'th neuron, respectively; an activation value is defined as $\mathbf{x}_{\text{core}}^T\mathbf{w}_{\text{core},i} + \mathbf{x}_{\text{spu}}^T\mathbf{w}_{\text{spu},i}$; and $\text{Med}(\cdot)$ returns the median of an input set of values.

Proof. We start by obtaining the set of correctly predicted samples $\hat{\mathcal{D}}_y$ and the set of incorrectly predicted samples $\bar{\mathcal{D}}_y$ as $\hat{\mathcal{D}}_y = \{\mathbf{x}|f(\mathbf{x}) \geq 0, (\mathbf{x}, y) \in \mathcal{D}_{\text{Ide}}\}$ and $\bar{\mathcal{D}}_y = \{\mathbf{x}|f(\mathbf{x}) < 0, (\mathbf{x}, y) \in \mathcal{D}_{\text{Ide}}\}$, where $\mathcal{D}_{\text{Ide}}$ is the set of identification data. Then, we have $\hat{\mathcal{V}}_i^y = \{e_i|\mathbf{x} \in \hat{\mathcal{D}}_y\}$, and $\bar{\mathcal{V}}_i^y = \{e_i|\mathbf{x} \in \bar{\mathcal{D}}_y\}$, where e_i is the i'th neuron activation defined in (A.46). Expanding e_i following (A.46), we obtain

$$e_i = \mathbf{x}_{\text{core}}^T \mathbf{w}_{\text{core},i} + (2a-1)\beta^T \mathbf{x}_{\text{core}} \gamma^T \mathbf{w}_{\text{spu},i} + \varepsilon_{\text{rem}}.$$

Note that $\mathbf{x}_{\text{core}}^T \mathbf{w}_{\text{core},i}$ and ε_{rem} exist for all the samples, regardless of the ultimate prediction results, and all e_i follows a Gaussian distribution given a. Then, among all the correctly predicted samples with the label y, according the Lemma 2, we have $\text{Med}(\hat{\mathcal{V}}_i^y) \approx \mathbb{E}[\mathbf{x}_{\text{core}}^T \mathbf{w}_{\text{core},i}] + \mu\gamma^T \mathbf{w}_{\text{spu},i}$. Similarly, among all the incorrectly predicted samples with the label y, we have $\text{Med}(\bar{\mathcal{V}}_i^y) \approx \mathbb{E}[\mathbf{x}_{\text{core}}^T \mathbf{w}_{\text{core},i}] - \mu\gamma^T \mathbf{w}_{\text{spu},i}$. Then, the difference between the two is

$$\delta_i^y \approx -2\mu\gamma^T \mathbf{w}_{\text{spu},i}.$$

When $\delta_i^y > 0$, we have $\gamma^T \mathbf{w}_{\text{spu},i} < 0$. According Proposition 5.2, using $\delta_i^y > 0$ indeed selects neurons that have strong reliance on spurious components. □

Proof of Theorem 5.2

Theorem 5.2 (**NeuronTune Mitigates Spurious Bias**). Consider the model $f^*(\mathbf{x}) = \mathbf{x}^T \mathbf{u}^*$ trained on the biased training data with $p \gg 0.5$, with $\mathbf{u}_{\text{core}}^*$ and $\mathbf{u}_{\text{spu}}^*$ defined in (A.27) and (A.28), respectively. Under the mild assumption that $\boldsymbol{\beta}^T \mathbf{w}_{\text{core},i} \approx \boldsymbol{\gamma}^T \mathbf{w}_{\text{spu},i}, \forall i = 1, \ldots, M$, then applying NeuronTune to $f^*(\mathbf{x})$ produces a model that is closer to the unbiased one.

Proof. Consider $f^*(\mathbf{x})$ as the base model. We aim to prove that the retrained model obtained with NeuronTune produces model parameters that is closer to the unbiased model defined in Corollary 1 than the base model.

First, the assumption that $\boldsymbol{\beta}^T \mathbf{w}_{\text{core},i} \approx \boldsymbol{\gamma}^T \mathbf{w}_{\text{spu},i}, \forall i = 1, \ldots, M$ generally holds for a biased model as the model has learned to associate spurious attributes with the core attributes.

Then, we denote the retrained parameters obtained with NeuronTune as $\mathbf{u}_{\text{core}}^\dagger$ and $\mathbf{u}_{\text{spu}}^\dagger$. We start with calculating $\mathbf{u}_{\text{spu}}^\dagger$. Focusing on (A.42) and following the derivation in Lemma 1, we obtain $\mathbf{u}_{\text{spu}}^\dagger = \sum_{i \in \mathcal{I}_+} b_i \mathbf{w}_{\text{spu},i} = \mathbf{u}_{\text{spu}}^*$, where $\mathcal{I}_+$ denotes the set of neuron indexes satisfying $\boldsymbol{\gamma}^T \mathbf{w}_{\text{spu},i} > 0$. Note that NeuronTune is a last-layer retraining method; thus we only optimize b_i here and $\mathbf{w}_{\text{spu},i}$ is the same as in $f^*(\mathbf{x})$. Left multiplying $\mathbf{u}_{\text{spu}}^\dagger$ with $\boldsymbol{\gamma}^T$, we have

$$\begin{aligned} \boldsymbol{\gamma}^T \mathbf{u}_{\text{spu}}^\dagger &= \sum_{i \in \mathcal{I}_+} b_i^\dagger \boldsymbol{\gamma}^T \mathbf{w}_{\text{spu},i} \\ &= z^* = \frac{(2p-1)\eta_{\text{core}}^2}{\eta_{\text{core}}^2 + \eta_{\text{spu}}^2} > 0. \end{aligned} \tag{A.47}$$

Note that $\boldsymbol{\gamma}^T \mathbf{w}_{\text{spu},i} > 0, \ \forall i \in \mathcal{I}_+$ because of NeuronTune. Hence, we have $b_i^\dagger > 0, \ \forall i \in \mathcal{I}_+$. Moreover, we observe that $\mathbf{u}_{\text{spu}}^\dagger$ is the same as $\mathbf{u}_{\text{spu}}^*$ as long as $\mathcal{I}_+$ is non-empty. This shows that NeuronTune is not able to optimize parameters related to the spurious components in the input data.

According to the Corollary 1, the unbiased model is achieved when $p = 0.5$ and $\mathbf{u}_{\text{core}} = \boldsymbol{\beta}$. The Euclidean distance between $\boldsymbol{\beta}$ and the biased solution $\mathbf{u}_{\text{core}} = (1 - z^*)\boldsymbol{\beta}$ is $\|\mathbf{u}_{\text{core}}^* - \boldsymbol{\beta}\| = z^*$. Based

on (A.47), we estimate the distance between our NeuronTune solution $\mathbf{u}^{\dagger}_{\text{core}}$ and $\boldsymbol{\beta}$ as follows

$$\begin{aligned}
\|\mathbf{u}^{\dagger}_{\text{core}} - \boldsymbol{\beta}\|_2 &= \|\boldsymbol{\beta}^T(\mathbf{u}^{\dagger}_{\text{core}} - \boldsymbol{\beta})\|_2 && \text{(A.48)}\\
&= \|\boldsymbol{\beta}^T\mathbf{u}^{\dagger}_{\text{core}} - 1\|_2 && \text{(A.49)}\\
&= \|\sum_{i\in\mathcal{I}_+} b_i^{\dagger}\boldsymbol{\beta}^T\mathbf{w}_{\text{core},i} - 1\|_2 && \text{(A.50)}\\
&\approx \|\sum_{i\in\mathcal{I}_+} b_i^{\dagger}\boldsymbol{\gamma}^T\mathbf{w}_{\text{spu},i} - 1\|_2 && \text{(A.51)}\\
&= \|z^* - 1\|, && \text{(A.52)}
\end{aligned}$$

where (A.49) uses the fact that $\boldsymbol{\beta}^T\boldsymbol{\beta} = 1$, and (A.50) uses the condition $\boldsymbol{\beta}^T\mathbf{w}_{\text{core},i} \approx \boldsymbol{\gamma}^T\mathbf{w}_{\text{spu},i}, \forall i = 1,\ldots,M$. Note that z^* is achieved on the training data with $p \gg 0.5$ and $\eta^2_{\text{core}} \gg \eta^2_{\text{spu}}$, hence we have $z^* \approx 1$ and $\|\mathbf{u}^{\dagger}_{\text{core}} - \boldsymbol{\beta}\|_2 \approx 0$. In other words, NeuronTune can bring model parameters closer to the optimal and unbiased solution than the parameters of the biased model.

□

Proof of Lemma 2

Lemma 2 (**Majority of Samples among Different Predictions**). Given the model $f(\mathbf{x}) = \mathbf{b}^T\mathbf{W}\mathbf{x}$ trained on $y \in \{-\mu, \mu\}$ ($\mu > 0$) with $\mu = \mathbb{E}[\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}]$, and the conditions that $p > 3/4$ and $\eta^2_{\text{core}} \gg \eta^2_{\text{spu}}$, we have the following claims:

- Among the set of all correctly predicted samples with the label y, more than half of them are generated with $a = 1$;
- Among the set of all incorrectly predicted samples with the label y, more than half of them are generated with $a = 0$.

Proof. With the two regression targets, $-\mu$ and μ, the optimal decision boundary is 0. Without loss of generality, we consider $y = \mu$. Then, the set of correctly predicted samples $\hat{\mathcal{D}}_y$ is

$$\hat{\mathcal{D}}_y = \{\mathbf{x}|f(\mathbf{x}) \geq 0, (\mathbf{x}, y) \in \mathcal{D}_{\text{Ide}}\},$$

and the set of incorrectly predicted samples $\hat{\mathcal{D}}_y$ is

$$\bar{\mathcal{D}}_y = \{\mathbf{x}|f(\mathbf{x}) < 0, (\mathbf{x}, y) \in \mathcal{D}_{\text{Ide}}\}.$$

The probability of a sample with the label y that is correctly predicted is

$$\begin{aligned}
P(\mathbf{x} \in \hat{\mathcal{D}}_y|y) &= P(a=1)P(f(\mathbf{x}) \geq 0|a=1,y) + P(a=0)P(f(\mathbf{x}) \geq 0|a=0,y)\\
&= pP(f(\mathbf{x}) \geq 0|a=1,y) + (1-p)P(f(\mathbf{x}) \geq 0|a=0,y).
\end{aligned}$$

Similarly, the probability of a sample with the label y that is incorrectly predicted is

$$P(\mathbf{x} \in \bar{\mathcal{D}}_y|y) = pP(f(\mathbf{x}) < 0|a=1,y) + (1-p)P(f(\mathbf{x}) < 0|a=0,y).$$

To calculate $P(f(\mathbf{x}) \geq 0|a=1,y)$, we expand $f(\mathbf{x})$ as follows:

$$\begin{aligned}
f(\mathbf{x}) &= \mathbf{x}^T_{\text{core}}\mathbf{u}^*_{\text{core}} + \mathbf{x}^T_{\text{spu}}\mathbf{u}^*_{\text{spu}}\\
&= \mathbf{x}^T_{\text{core}}\boldsymbol{\beta}(1-z^*) + (\boldsymbol{\gamma}(\boldsymbol{\beta}^T\mathbf{x}_{\text{core}} + \varepsilon_{\text{core}}) + \boldsymbol{\varepsilon}_{\text{spu}})^T\mathbf{u}^*_{\text{spu}}\\
&= \mathbf{x}^T_{\text{core}}\boldsymbol{\beta}(1-z^*) + \mathbf{x}^T_{\text{core}}\boldsymbol{\beta}\boldsymbol{\gamma}^T\mathbf{u}^*_{\text{spu}} + \boldsymbol{\gamma}^T\mathbf{u}^*_{\text{spu}}\varepsilon_{\text{core}} + \boldsymbol{\varepsilon}^T_{\text{spu}}\mathbf{u}^*_{\text{spu}}\\
&= \mathbf{x}^T_{\text{core}}\boldsymbol{\beta} + z^*\varepsilon_{\text{core}} + \boldsymbol{\varepsilon}^T_{\text{spu}}\mathbf{u}^*_{\text{spu}}
\end{aligned}$$

The output of $f(\mathbf{x})$ follows a Gaussian distribution, with the mean $\mu_1 = \mathbb{E}[f(\mathbf{x})] = \mu$, and the variance $\sigma_1^2 = Var(\mathbf{x}_{\text{core}}^T\boldsymbol{\beta}) + \eta_{\text{core}}^2(z^*)^2 + \eta_{\text{spu}}^2(z^*)^2$. Therefore, we have

$$P(f(\mathbf{x}) \geq 0|a=1,y) = P(\mathbf{x} \in \hat{\mathcal{D}}_y|a=1,y) = 1 - \Phi(\frac{0-\mu}{\sigma_1}) = \Phi(\frac{\mu}{\sigma_1}), \tag{A.53}$$

$$P(f(\mathbf{x}) < 0|a=1,y) = P(\mathbf{x} \in \bar{\mathcal{D}}_y|a=1,y) = 1 - \Phi(\frac{\mu}{\sigma_1}) = \Phi(\frac{-\mu}{\sigma_1}). \tag{A.54}$$

Similarly, to calculate $P(f(\mathbf{x}) \geq 0|a=0,y)$, we expand $f(\mathbf{x})$ as follows:

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{x}_{\text{core}}^T\boldsymbol{\beta}(1-z^*) - \mathbf{x}_{\text{core}}^T\boldsymbol{\beta}\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}^* - \boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}^*\varepsilon_{\text{core}} + \boldsymbol{\varepsilon}_{\text{spu}}^T\mathbf{u}_{\text{spu}}^* \\ &= \mathbf{x}_{\text{core}}^T\boldsymbol{\beta}(1-2z^*) - z^*\varepsilon_{\text{core}} + \boldsymbol{\varepsilon}_{\text{spu}}^T\mathbf{u}_{\text{spu}}^*. \end{aligned}$$

The output of $f(\mathbf{x})$ follows a Gaussian distribution, with the mean $\mu_0 = \mathbb{E}[f(\mathbf{x})] = \mu(1-2z^*)$, and the variance $\sigma_0^2 = (1-2z^*)^2 Var(\mathbf{x}_{\text{core}}^T\boldsymbol{\beta}) + \eta_{\text{core}}^2(z^*)^2 + \eta_{\text{spu}}^2(z^*)^2$. Therefore, we have

$$P(f(\mathbf{x}) \geq 0|a=0,y) = P(x \in \hat{\mathcal{D}}_y|a=0,y) = 1 - \Phi(\frac{0-\mu_0}{\sigma_0}) = \Phi(\frac{(1-2z^*)\mu}{\sigma_0}), \tag{A.55}$$

$$P(f(\mathbf{x}) < 0|a=0,y) = P(x \in \bar{\mathcal{D}}_y|a=0,y) = 1 - \Phi(\frac{\mu_0}{\sigma_0}) = \Phi(\frac{-(1-2z^*)\mu}{\sigma_0}). \tag{A.56}$$

Therefore, we have the probabilities for correctly and incorrectly predicted samples with the label y, i.e.,

$$P(\mathbf{x} \in \hat{\mathcal{D}}_y|y) = p\Phi(\frac{\mu}{\sigma_1}) + (1-p)\Phi(\frac{(1-2z^*)\mu}{\sigma_0}), \tag{A.57}$$

and

$$P(\mathbf{x} \in \bar{\mathcal{D}}_y|y) = p\Phi(\frac{-\mu}{\sigma_1}) + (1-p)\Phi(\frac{-(1-2z^*)\mu}{\sigma_0}) \tag{A.58}$$

Next, we seek to determine whether the majority of samples in the correctly (incorrectly) predicted set $\hat{\mathcal{D}}_y$ ($\bar{\mathcal{D}}_y$) is generated with $a=0$ or $a=1$. To achieve this, in the set of correctly predicted samples, we use the Bayesian theorem based on (A.57), i.e.,

$$\begin{aligned} P(a=1|\mathbf{x} \in \hat{\mathcal{D}}_y, y) &= \frac{P(\mathbf{x} \in \hat{\mathcal{D}}_y|a=1,y)P(a=1)}{P(\mathbf{x} \in \hat{\mathcal{D}}_y|y)} \\ &= \frac{p\Phi(\mu/\sigma_1)}{p\Phi(\mu/\sigma_1) + (1-p)\Phi((1-2z^*)\mu/\sigma_0)}, \end{aligned} \tag{A.59}$$

and

$$\begin{aligned} P(a=0|\mathbf{x} \in \hat{\mathcal{D}}_y, y) &= 1 - P(a=1|\mathbf{x} \in \hat{\mathcal{D}}_y, y) \\ &= \frac{(1-p)\Phi((1-2z^*)\mu/\sigma_0)}{p\Phi(\mu/\sigma_1) + (1-p)\Phi((1-2z^*)\mu/\sigma_0)}. \end{aligned} \tag{A.60}$$

Similarly, in the set of incorrectly predicted samples, we have

$$\begin{aligned} P(a=1|\mathbf{x} \in \bar{\mathcal{D}}_y, y) &= \frac{P(\mathbf{x} \in \bar{\mathcal{D}}_y|a=1,y)P(a=1)}{P(\mathbf{x} \in \bar{\mathcal{D}}_y|y)} \\ &= \frac{p\Phi(-\mu/\sigma_1)}{p\Phi(-\mu/\sigma_1) + (1-p)\Phi(-(1-2z^*)\mu/\sigma_0)}, \end{aligned} \tag{A.61}$$

and

$$P(a=0|\mathbf{x}\in\bar{\mathcal{D}}_y,y) = 1-P(a=1|\mathbf{x}\in\bar{\mathcal{D}}_y,y) \\ = \frac{(1-p)\Phi(-(1-2z^*)\mu/\sigma_0)}{p\Phi(-\mu/\sigma_1)+(1-p)\Phi(-(1-2z^*)\mu/\sigma_0)}. \tag{A.62}$$

Under the assumption that $p > 3/4$ and $\eta_{\text{core}}^2 \gg \eta_{\text{spu}}^2$, we have $1-2z^* = \big((3-4p)\eta_{\text{core}}^2+\eta_{\text{spu}}^2\big)/(\eta_{\text{core}}^2+\eta_{\text{spu}}^2) < 0$. Hence, $\Phi(-(1-2z^*)\mu/\sigma_0) < 1/2$ and $P(a=1|\mathbf{x}\in\hat{\mathcal{D}}_y,y) > 1/2$; in other words, **among the set of all correctly predicted samples with the label y, more than half of them are generated with $a=1$.**

Moreover, under the assumption that $\Phi(-\mu/\sigma_1)\approx 0$, i.e., predictions of the model have a high signal-to-noise ratio, then $P(a=0|\mathbf{x}\in\bar{\mathcal{D}}_y,y) > 1/2$, i.e., **among the set of all incorrectly predicted samples with the label y, more than half of them are generated with $a=0$.** This assumption is generally true, as $\sigma_1^2 = Var(\mathbf{x}_{\text{core}}^T\boldsymbol{\beta}) + \eta_{\text{core}}^2(z^*)^2 + \eta_{\text{spu}}^2(z^*)^2$ is typically very small when z^* approaches zero given $p > 3/4$ and $\eta_{\text{core}}^2 \gg \eta_{\text{spu}}^2$. □

Proof of Lemma 3

Lemma 3. Consider the model $f(\mathbf{x}) = \mathbf{x}^T\mathbf{u}$ with $\mathbf{u} = [\mathbf{u}_{\text{core}}, \mathbf{u}_{\text{spu}}]$, the optimal solution for $\mathbf{u}_{\text{spu}}$ that can be achieved by last-layer retraining on the retraining data with p_{re} is $\mathbf{u}_{\text{spu}}^r$, which is defined as

$$\mathbf{u}_{\text{spu}}^r = \frac{(2p_{\text{re}}-1)\eta_{\text{core}}^2}{\eta_{\text{core}}^2+\eta_{\text{spu}}^2}\boldsymbol{\gamma}. \tag{A.63}$$

Proof. First, we have $f(\mathbf{x}) = \mathbf{x}^T\mathbf{u} = \mathbf{b}^T\mathbf{W}\mathbf{x}$. For last-layer retraining, $\mathbf{b}$ is optimized. Following the derivation in Lemma 1, we similarly obtain the inequality in (A.42) with $p = p_{\text{re}}$, i.e.,

$$\ell(\mathbf{b}) \geq \frac{1}{2}\Big(p_{\text{re}}\eta_{\text{core}}^2(1-\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + (1-p_{\text{re}})\eta_{\text{core}}^2(1+\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + \eta_{\text{spu}}^2\|\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}\|_2^2\Big), \tag{A.64}$$

Note that the terms on the right side of the inequality are independent of any manipulation of the retraining data, such as reweighting. Then, taking the derivative to the sum of these terms with respect to $\mathbf{b}$, we obtain the following equation

$$\boldsymbol{\gamma}^T\mathbf{W}_{\text{spu}}\mathbf{b} = \frac{(2p_{\text{re}}-1)\eta_{\text{core}}^2}{\eta_{\text{core}}^2+\eta_{\text{spu}}^2}, \tag{A.65}$$

where $\mathbf{u}_{\text{spu}} = \mathbf{W}_{\text{spu}}\mathbf{b}$. Since $\boldsymbol{\gamma}^T\boldsymbol{\gamma} = 1$, then we have $\mathbf{u}_{\text{spu}} = \mathbf{u}_{\text{spu}}^r$. We finally verify that $\mathbf{u}_{\text{spu}}^r$ indeed minimizes the sum of the terms on the right hand side of (A.64). If p_{re} equals to p for the training data, then $\mathbf{u}_{\text{spu}}^r = \mathbf{u}_{\text{spu}}^*$ defined in (A.28). □

A.7.3 Connection to Last-Layer Retraining Methods

Although our method shares a similar setting to last-layer retraining methods, such as AFR [32] and DFR [4], our method is fundamentally different from these methods in how spurious bias is mitigated. Take AFR for an example. It, in essence, is a sample-level method and adjusts the weights of the last layer indirectly via retraining on samples with loss-related weights. Our method directly forces the weights identified as affected by spurious bias to zero, while adjusting the remaining weights with retraining.

The advantage of NeuronTune can be explained more formally in our theoretical analysis framework. First, consider the training loss in (A.32), we can express it as the sum of following terms for brevity,

$$\ell_{tr}(\mathbf{W},\mathbf{b}) = \frac{1}{2}p\mathbb{E}[\psi_1(\mathbf{u}_{\text{core}},\mathbf{u}_{\text{spu}})] + \frac{1}{2}(1-p)\mathbb{E}[\psi_2(\mathbf{u}_{\text{core}},\mathbf{u}_{\text{spu}})] + \frac{1}{2}\psi_3(\mathbf{u}_{\text{spu}}), \tag{A.66}$$

where p is the data generation parameter and is fixed, and ψ_1, ψ_2, and ψ_3 are defined as

$$\psi_1(\mathbf{u}_{\text{core}}, \mathbf{u}_{\text{spu}}) = \mathbb{E}\|\mathbf{x}_{\text{core}}^T\mathbf{u}_{\text{core}} - (1 - \boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}\|_2^2,$$

$$\psi_2(\mathbf{u}_{\text{core}}, \mathbf{u}_{\text{spu}}) = \mathbb{E}\|\mathbf{x}_{\text{core}}^T\mathbf{u}_{\text{core}} - (1 + \boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})\boldsymbol{\beta}^T\mathbf{x}_{\text{core}}\|_2^2,$$

and

$$\psi_3(\mathbf{u}_{\text{spu}}) = p\eta_{\text{core}}^2(1 - \boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + (1-p)\eta_{\text{core}}^2(1 + \boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}})^2 + \eta_{\text{spu}}^2\|\boldsymbol{\gamma}^T\mathbf{u}_{\text{spu}}\|_2^2,$$

respectively. Based on Lemma 3, for last-layer retraining methods in general, the optimal solution for $\mathbf{u}_{\text{spu}}$ is $\mathbf{u}_{\text{spu}}^*$, given that the retraining data follows the same distribution as the training data.

AFR changes the distribution within the first two expectation terms $\psi_1(\mathbf{u}_{\text{core}}, \mathbf{u}_{\text{spu}})$ and $\psi_2(\mathbf{u}_{\text{core}}, \mathbf{u}_{\text{spu}})$ and jointly updates $\mathbf{u}_{\text{core}}$ and $\mathbf{u}_{\text{spu}}$, while there is no optimality guarantee for $\mathbf{u}_{\text{spu}}$ ($\psi_3(\mathbf{u}_{\text{spu}})$ is not considered in AFR). By contrast, according to Theorem 5.2, NeuronTune first ensures that $\mathbf{u}_{\text{spu}}$ is optimal, then it moves $\mathbf{u}_{\text{core}}$ close the the unbiased solution.

A.7.4 Comparison between Models Selected with Worst-Class Accuracy

We compared our approach with AFR [32] and JTT [31] to demonstrate the challenges of the unsupervised setting for semi-supervised methods. These methods were tuned using worst-class accuracy [29] on the validation set instead of WGA. As shown in Table A.7.1, our method exhibits larger performance gains over AFR and JTT compared to their results presented in Tables 5.4 and 5.5.

Method	Waterbirds	CelebA
JTT	$84.2_{\pm 0.5}$	$52.3_{\pm 1.8}$
AFR	$89.0_{\pm 2.6}$	$68.7_{\pm 1.7}$
NeuronTune	$91.8_{\pm 0.8}$	$83.0_{\pm 2.8}$

Table A.7.1: WGA comparison when models selected by the worst-class accuracy on the validation set.

A.7.5 Complexity Analysis

We analyze the computational complexity of our method, NeuronTune, alongside representative reweighting-based methods, including AFR [32], DFR [4], and JTT [31]. Let the number of identification samples be N_{Ide}, the number of retraining samples be N_{ret}, the total number of training samples be N, the number of latent dimensions be D, and the number of training epochs be E. Additionally, denote the time required for inference as τ_{fw}, for last-layer retraining as τ_{ll}, and for optimizing the entire model as τ_{opt}. The computational complexities of these methods are summarized in Table A.7.2.

Among the methods, JTT has the highest computational complexity since $\tau_{\text{opt}} \gg \tau_{\text{ll}}$, requiring full model optimization. DFR is much faster due to its reliance on last-layer retraining, though it requires group annotations. AFR extends DFR by additionally precomputing sample losses, increasing its computational cost slightly. NeuronTune, while requiring more time than AFR to identify biased dimensions across all D embedding dimensions, remains computationally efficient. This is because τ_{fw}, the time required for forward inference, is typically very small. As a result, NeuronTune offers an effective balance between computational efficiency and robust spurious bias mitigation.

A.7.6 Advantages over Variable Selection Methods

Although the identification of biased dimensions in (5.19) may resemble traditional variable selection methods [232], our approach extends beyond simply selecting a subset of variables that optimally

Method	Time complexity
JTT [31]	$O(NE\tau_{\text{opt}})$
AFR [32]	$O(N_{\text{Ide}}\tau_{\text{fw}} + EN_{\text{ret}}E\tau_{\text{ll}})$
DFR [4]	$O(EN_{\text{ret}}E\tau_{\text{ll}})$
NeuronTune	$O(E(N_{\text{Ide}}D\tau_{\text{fw}} + N_{\text{ret}}E\tau_{\text{ll}}))$

Table A.7.2: Computation complexity comparison with different reweighting methods.

explain the target variable. Instead, it specifically addresses spurious bias—an issue often neglected in traditional variable selection.

Traditional variable selection methods, such as L1 regularization, do not distinguish whether variables represent spurious or core attributes. Since spurious attributes are often predictive of target labels in the training data and are easier for models to learn [190, 199], these methods may mistakenly prioritize spurious attributes, thereby amplifying spurious bias. In contrast, our method explicitly targets dimensions influenced by spurious bias and re-balances the model's reliance on features, reducing the model's dependency on spurious information.

Furthermore, unlike many variable selection methods that require explicit supervision (e.g., labels or statistical relationships) to mitigate spurious bias, NeuronTune operates in an unsupervised setting where group labels indicative of spurious attributes are unavailable. By leveraging misclassification signals to estimate spuriousness scores, our method is better suited for scenarios where group annotations are costly or infeasible, offering a practical and scalable solution to the challenge of spurious bias mitigation.

A.7.7 Dataset Details

Table A.7.3 gives the details of the two image and two text datasets used in the experiments. Additionally, the ImageNet-9 dataset [49] has 54600 and 2100 training and validation images, respectively. The ImageNet-A [176] dataset has 1087 images for evaluation.

A.7.8 Training Details

Table A.7.4 and Table A.7.5 give the hyperparameter settings for ERM and NeuronTune training, respectively.

A.7.9 Visualizations of Unbiased and Biased Dimensions

We provide visualizations of the neuron activation value distributions for the identified unbiased and biased dimensions in Figures A.7.1 to A.7.4. The biased and unbiased dimensions selected for visualizations are obtained by first sorting the dimensions based on their spuriousness scores and then selecting three biased dimensions that have the largest scores and three unbiased dimensions that have the smallest scores. Note that a dimension does not exclusively represent a core or spurious attribute; it typically represents a mixture of them.

On the CelebA dataset, as shown in Figure A.7.1, samples that highly activate the unbiased dimensions have both males and females; thus, the unbiased dimensions do not appear to have gender bias. For samples that highly activate the identified biased dimensions, all of them are females, demonstrating a strong reliance on the gender information. In Figure A.7.2, samples that highly activate the identified biased dimensions (right side of Figure A.7.2) tend to have slightly darker hair colors or backgrounds, as compared with samples that highly activate the identified unbiased dimensions (left side of Figure A.7.2). With the aid of the heatmaps, we observe that these biased dimensions mostly represent a person's face, which is irrelevant to target classes.

On the Waterbirds dataset, as shown in Figure A.7.3, for the landbirds class, the identified unbiased dimensions mainly represent certain features of a bird and land backgrounds. For the identified biased dimensions, they mainly represent water backgrounds, which are irrelevant to the landbirds class based on the training data. For the waterbirds class, as shown in Figure A.7.4, the

Class	Spurious attribute	Train	Val	Test
	`Waterbirds`			
landbird	land	3498	467	2225
landbird	water	184	466	2225
waterbird	land	56	133	642
waterbird	water	1057	133	642
	`CelebA`			
non-blond	female	71629	8535	9767
non-blond	male	66874	8276	7535
blond	female	22880	2874	2480
blond	male	1387	182	180
	`MultiNLI`			
contradiction	no negation	57498	22814	34597
contradiction	negation	11158	4634	6655
entailment	no negation	67376	26949	40496
entailment	negation	1521	613	886
neither	no negation	66630	26655	39930
neither	negation	1992	797	1148
	`CivilComments`			
neutral	no identity	148186	25159	74780
neutral	identity	90337	14966	43778
toxic	no identity	12731	2111	6455
toxic	identity	17784	2944	8769

Table A.7.3: Numbers of samples in different groups and different splits of the four datasets.

Hyperparameters	Waterbirds	CelebA	ImageNet-9	MultiNLI	CivilComments
Initial learning rate	3e-3	3e-3	1e-3	1e-5	1e-3
Number of epochs	100	20	120	10	10
Learning rate scheduler	CosineAnnealing	CosineAnnealing	MultiStep[40,60,80]	Linear	Linear
Optimizer	SGD	SGD	SGD	AdamW	AdamW
Backbone	ResNet50	ResNet50	ResNet18	BERT	BERT
Weight decay	1e-4	1e-4	1e-4	1e-4	1e-4
Batch size	32	128	128	16	16

Table A.7.4: Hyperparameters for ERM training.

identified unbiased dimensions mostly represent certain features of a bird and water backgrounds, while the identified biased dimensions mainly represent land backgrounds.

Hyperparameters	Waterbirds	CelebA	ImageNet-9	MultiNLI	CivilComments
Learning rate	1e-3	1e-3	1e-3	1e-5	1e-3
Number of batches per epoch	200	200	200	200	200
Number of epochs	40	40	1	60	60
Optimizer	SGD	SGD	SGD	AdamW	AdamW
Batch size	128	128	128	128	128

Table A.7.5: Hyperparameters for NeuronTune.

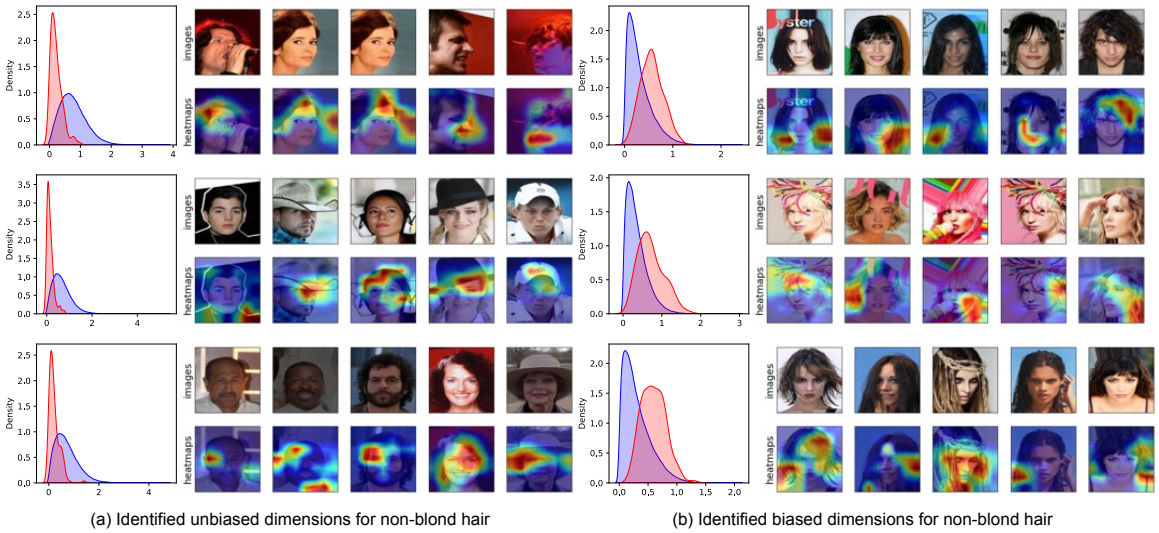


Figure A.7.1: Value distributions of the correctly (blue) and incorrectly (red) predicted samples for unbiased (a) and biased (b) dimensions, along with the representative samples, respectively, based on the non-blond hair samples in the CelebA dataset.

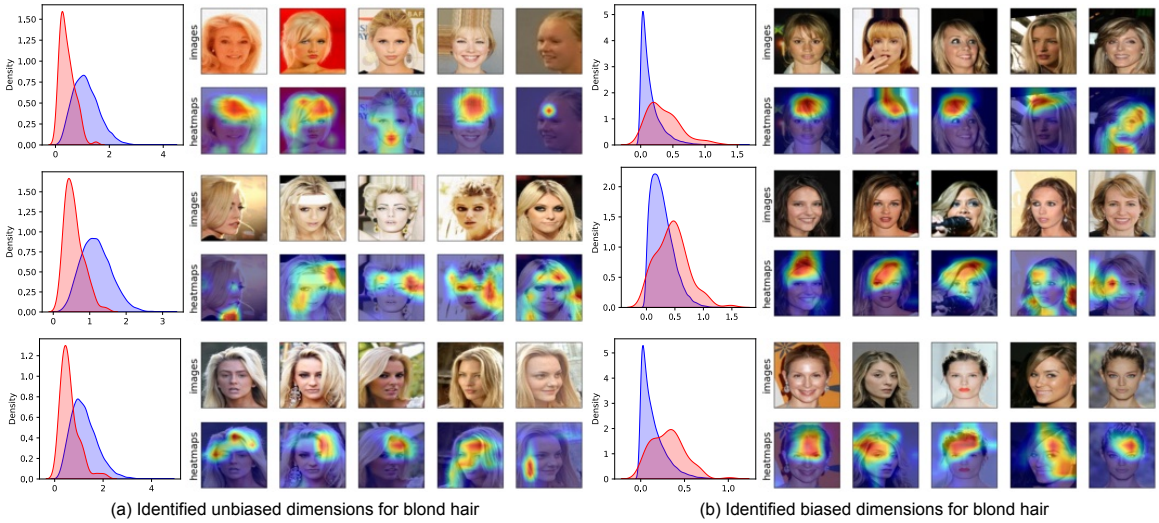


Figure A.7.2: Value distributions of the correctly (blue) and incorrectly (red) predicted samples for unbiased (a) and biased (b) dimensions, along with the representative samples, respectively, based on the blond hair samples in the CelebA dataset.

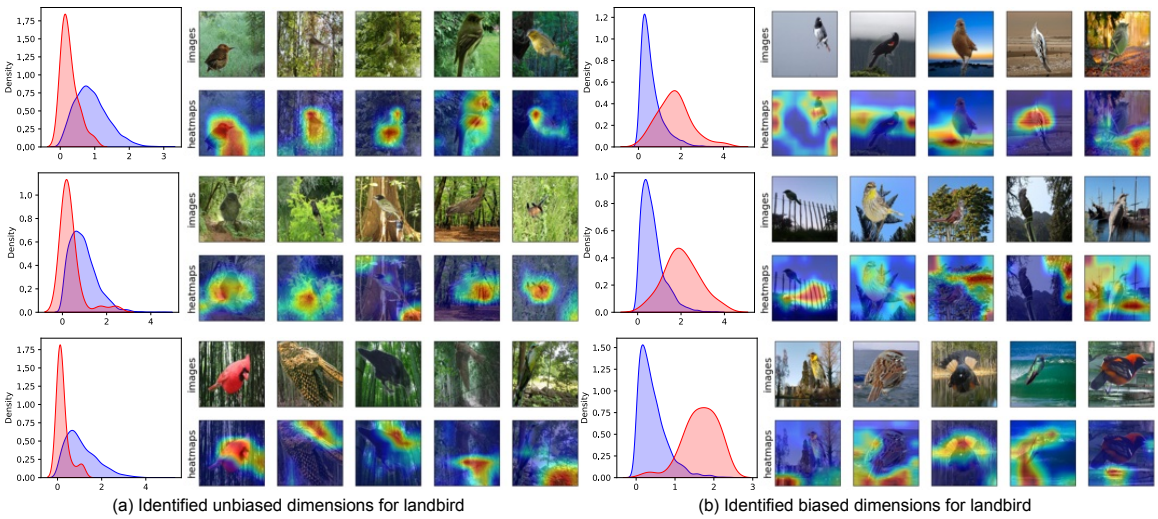


Figure A.7.3: Value distributions of the correctly (blue) and incorrectly (red) predicted samples for unbiased (a) and biased (b) dimensions, along with the representative samples, respectively, based on the landbirds samples in the Waterbirds dataset.

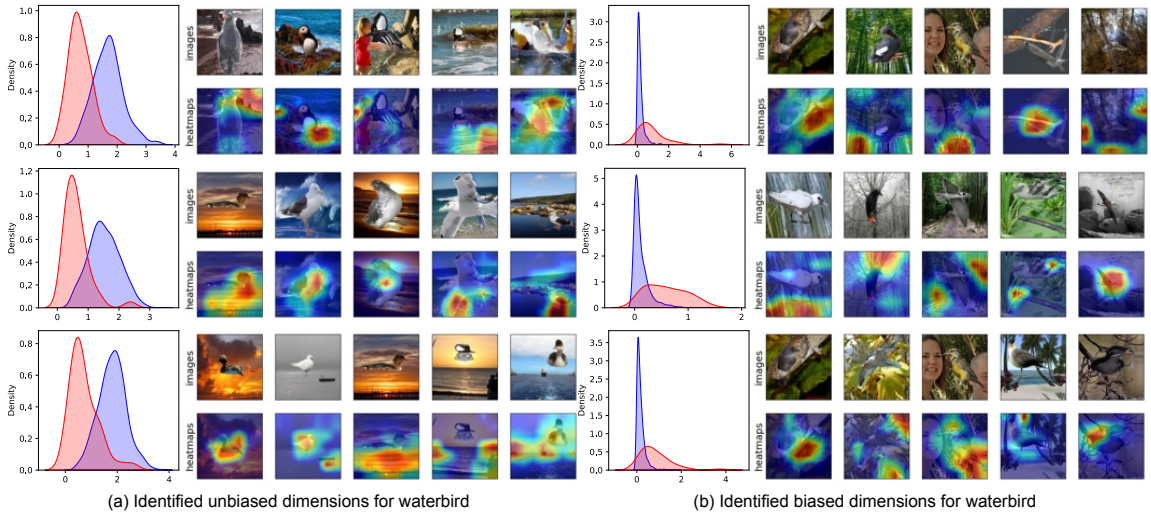


Figure A.7.4: Value distributions of the correctly (blue) and incorrectly (red) predicted samples for unbiased (a) and biased (b) dimensions, along with the representative samples, respectively, based on the waterbirds samples in the Waterbirds dataset.

A.8 Self-Adaptive Prompt Exploration for Zero-Shot Spurious Bias Mitigation in Vision-Language Models

A.8.1 Prompt Templates

We provide the prompt templates used in the experiments in Table A.8.1. There are a total of 80 templates. The special symbol "[CLASS]" is a placeholder, which will be replaced with actual class labels in zero-shot classification.

For the vanilla zero-shot classification method, we followed the prompts used in [95]. Specifically, on the Waterbirds dataset, we used "an image of landbird" and "an image of waterbird"; on the CelebA dataset, we used "person with dark hair" and "person with blond hair"; on the PACS and VLCS datasets, we directly used the class names as the input text descriptions.

A.8.2 Dataset Details

The details of the four datasets used in the experiments are shown in Table A.8.2, including groups, total samples, number of classes, and class labels. As we focus on the zero-shot setting, only the information regarding the test set in each dataset is shown in Table A.8.2.

A.8.3 Limitations and Future Works

While our proposed method demonstrates significant robustness, the performance of SAVE is contingent on the diversity and quality of the predefined prompt templates. A more diverse and task-relevant set of prompt templates could enhance the method's ability to select optimal prompts for mitigating multimodal spurious biases. Furthermore, SAVE operates within the framework of zero-shot debiasing, meaning it does not incorporate any training techniques for vision-language models (VLMs). Although this ensures the approach remains entirely out-of-the-box, future work could explore integrating SAVE with small labeled datasets to further refine and improve model performance. Lastly, while we evaluated SAVE across multiple datasets, extending its evaluation to a broader range of tasks and bias types would provide deeper insights into its generalizability and broader applicability.

A.8.4 More Selected Prompt Templates

In the same setting as Section 5.3.3, we show more prompt templates selected by our method in Figures A.8.1, A.8.2, and A.8.3. We observe that, in general, the most frequently selected template is different across classes and datasets. One defining characteristic of those frequently selected templates is that they typically contain words that describe out-of-distribution images. For example, "black and white" occurs frequently when images all have colors. This could provide useful insights into the design of customized and more effective prompt templates for mitigating multimodal spurious bias.

Prompt Templates	Prompt Templates
a bad photo of a [CLASS].	a photo of many [CLASS].
a sculpture of a [CLASS].	a photo of the hard to see [CLASS].
a low resolution photo of the [CLASS].	a rendering of a [CLASS].
graffiti of a [CLASS].	a bad photo of the [CLASS].
a cropped photo of the [CLASS].	a tattoo of a [CLASS].
the embroidered [CLASS].	a photo of a hard to see [CLASS].
a bright photo of a [CLASS].	a photo of a clean [CLASS].
a photo of a dirty [CLASS].	a dark photo of the [CLASS].
a drawing of a [CLASS].	a photo of my [CLASS].
the plastic [CLASS].	a photo of the cool [CLASS].
a close-up photo of a [CLASS].	a black and white photo of the [CLASS].
a painting of the [CLASS].	a painting of a [CLASS].
a pixelated photo of the [CLASS].	a sculpture of the [CLASS].
a bright photo of the [CLASS].	a cropped photo of a [CLASS].
a plastic [CLASS].	a photo of the dirty [CLASS].
a jpeg corrupted photo of a [CLASS].	a blurry photo of the [CLASS].
a photo of the [CLASS].	a good photo of the [CLASS].
a rendering of the [CLASS].	a [CLASS] in a video game.
a photo of one [CLASS].	a doodle of a [CLASS].
a close-up photo of the [CLASS].	a photo of a [CLASS].
the origami [CLASS].	the [CLASS] in a video game.
a sketch of a [CLASS].	a doodle of the [CLASS].
an origami [CLASS].	a low resolution photo of a [CLASS].
the toy [CLASS].	a rendition of the [CLASS].
a photo of the clean [CLASS].	a photo of a large [CLASS].
a rendition of a [CLASS].	a photo of a nice [CLASS].
a photo of a weird [CLASS].	a blurry photo of a [CLASS].
a cartoon [CLASS].	art of a [CLASS].
a sketch of the [CLASS].	an embroidered [CLASS].
a pixelated photo of a [CLASS].	itap of the [CLASS].
a jpeg corrupted photo of the [CLASS].	a good photo of a [CLASS].
a plushie [CLASS].	a photo of the nice [CLASS].
a photo of the small [CLASS].	a photo of the weird [CLASS].
the cartoon [CLASS].	art of the [CLASS].
a drawing of the [CLASS].	a photo of the large [CLASS].
a black and white photo of a [CLASS].	the plushie [CLASS].
a dark photo of a [CLASS].	itap of a [CLASS].
graffiti of the [CLASS].	a toy [CLASS].
itap of my [CLASS].	a photo of a cool [CLASS].
a photo of a small [CLASS].	a tattoo of the [CLASS].

Table A.8.1: List of prompt templates.

Dataset	Groups	Statistics		Classes
		Total Samples	# Classes	
Waterbirds	landbird in land, landbird in water, waterbird on land, waterbird on water	5794	2	landbird, waterbird
CelebA	male & not blond, female & not blond, male & blond, female & blond	19962	2	not blond, blond
PACS	art, cartoons, photos, sketches	9991	7	dogs, elephant, giraffe, guitar, house, person
VLCS	Caltech101, LabelMe, SUN09, VOC2007	10725	5	bird, car, chair, dog, person

Table A.8.2: Dataset statistics including groups, total samples, number of classes, and class labels.

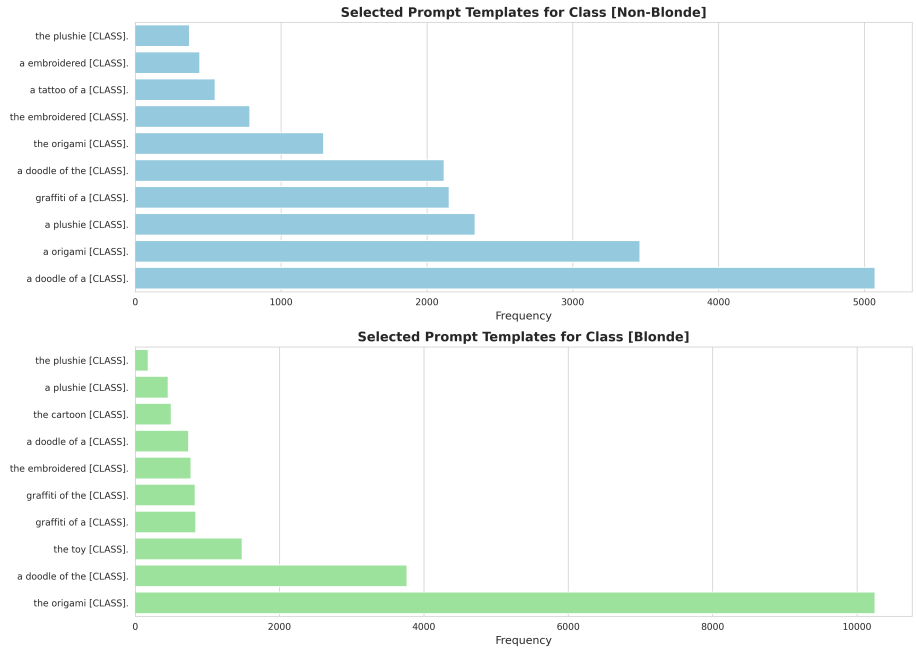


Figure A.8.1: Top-10 most frequently selected prompt templates by our method for each class with CLIP-ViT-B/32 in the CelebA dataset.

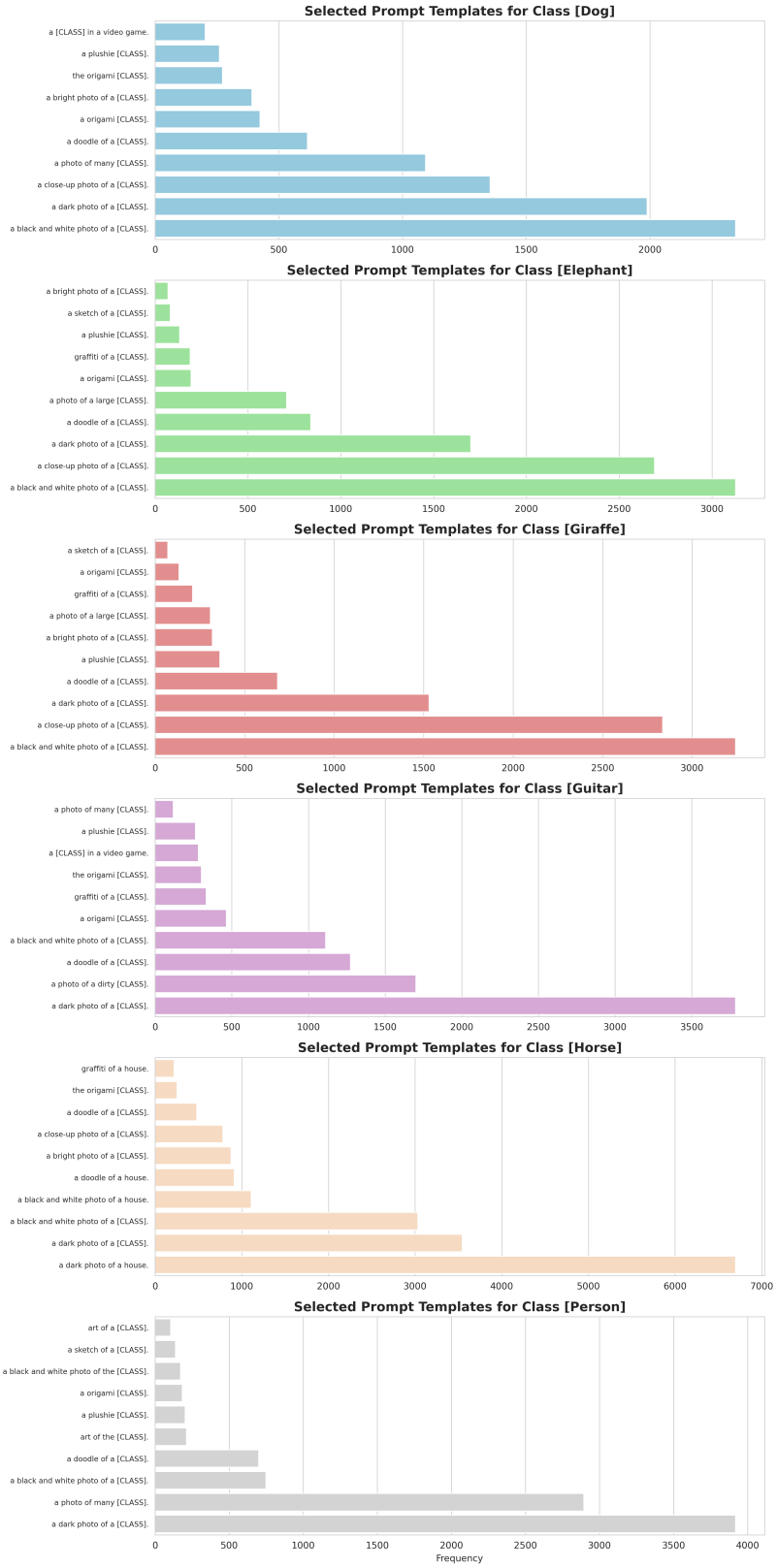


Figure A.8.2: Top-10 most frequently selected prompt templates by our method for each class with CLIP-ViT-B/32 in the PACS dataset.

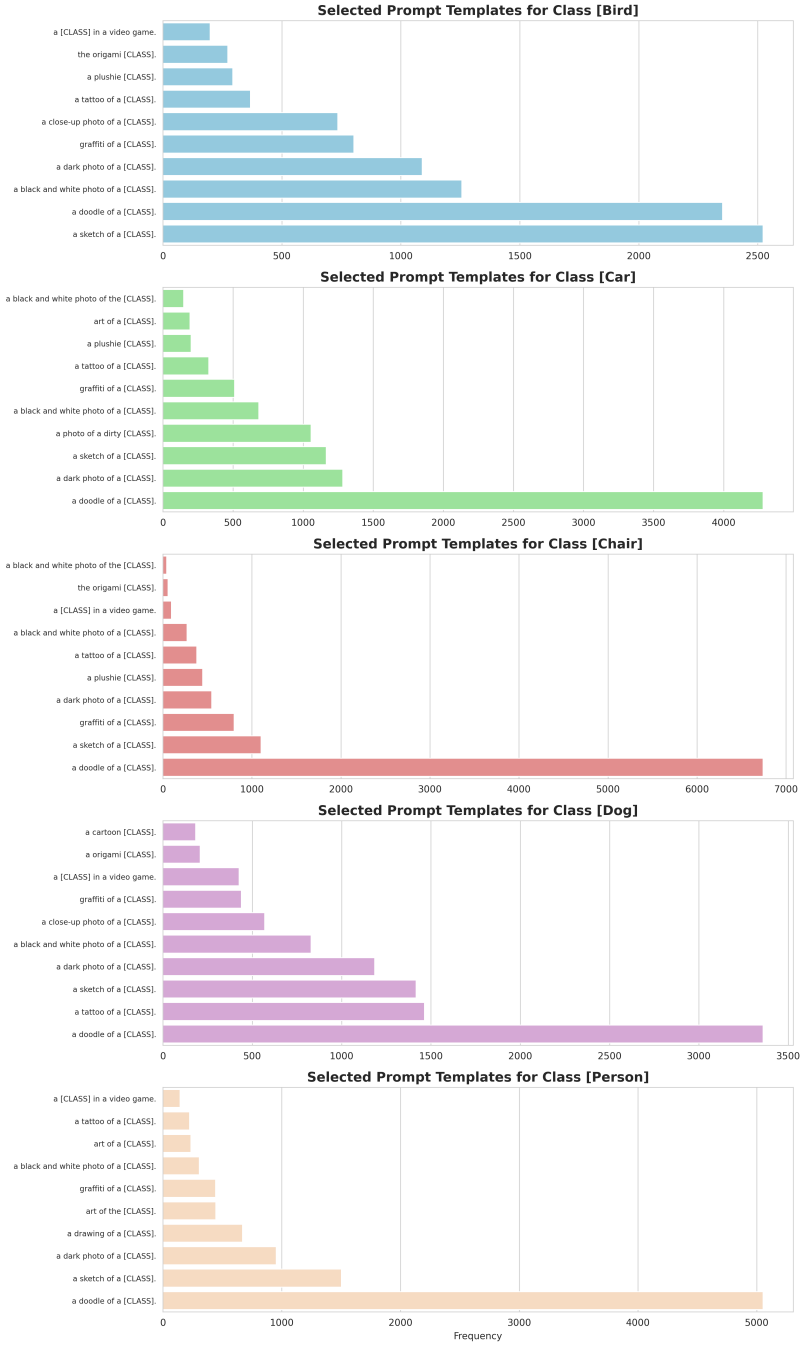


Figure A.8.3: Top-10 most frequently selected prompt templates by our method for each class with CLIP-ViT-B/32 in the VLCS dataset.