# Addressing Realisms Faced by Deep Learning Models in Cyber Physical Systems

Thesis by

Ashley Gao

In Partial Fulfillment of the Requirements for the

Degree of

Doctor of Philosophy


UNIVERSITY OF VIRGINIA

Charlottesville, Virginia

2023

Defended April 19 2023

© 2023

Ashley Gao
ORCID: 0000-0003-3979-8710

All rights reserved

# ACKNOWLEDGEMENTS

For science and truth.

# ABSTRACT

In recent years, applications in the cyber physical systems (CPS) area have greatly benefited from deep learning (DL)'s success. However, there exists an intrinsic problem with directly applying a trained deep learning model on a CPS application: CPS applications have constraints arising from realisms, whereas the training of deep learning models often does not take any or enough realisms into consideration. In the context of this thesis, a realism is defined as the reality as a result of the DL models' interaction with the CPS. Among the multitude of realisms, there are the following types of realisms that are most important. The first realism is task-specific, resulting from the interaction between the deep learning model and the environment in which the deep learning model is deployed. In this thesis, we address this type of realism in a specific acoustic application where audio samples are environmentally distorted in real cyber physical systems (smart homes). The second realism is non-targeted samples, which are defined as samples whose classes are not seen by the DL models during their training. In the same acoustic application, we incorporate a Mahalanobis distance-based out-of-distribution (OOD) detection technique to prevent OOD audio samples from being passed to the classifier trained on in-distribution data. As a result, the classifier is less prone to make mistakes as fewer OOD samples are passed to it. The third realism is that many data-driven deep learning models are not robust against even minor changes. In this thesis, we use an attention-enhanced graph neural network (GNN) architecture coupled with real-world knowledge, using both the GNN architecture and the real-world knowledge as ways to boost the robustness of the DL models. Importantly, the underlying problem with the aforementioned realisms is the problem of domain adaptation (DA): how do we make sure the DL models trained in one domain (clean samples free of these realisms) can perform adequately on another domain (samples tainted by realisms)? In this thesis, we develop two novel unsupervised domain adaptation (UDA) algorithms that are superior to the state-of-the-art UDA algorithms. The final realism addressed in this thesis has nothing to do with the training of the DL models. Instead, it is the realism that comes during the process when the deep learning model is deployed, such as caused by human behavior. In this thesis, we provide a comprehensive analysis of the case study in which we deploy several acoustics-based deep learning models in six smart homes, where we present and evaluate various techniques for deploying the deep learning models in CPS.

# CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

*Chapter 1*

# INTRODUCTION

Deep learning has emerged to be very successful in addressing problems arising in the setting of cyber physical systems (CPS) which require solutions for things such as anomaly detection, knowledge acquisition, and various tasks that use natural language processing (NLP) and computer vision (Sonntag et al., 2017; Esteva et al., 2021; Szegedy, Ioffe, et al., 2017; Y. Zhu and Newsam, 2017; Purwins et al., 2019; Noda et al., 2015; L. Wu et al., 2021; Wahab et al., 2021). However, CPS applications have constraints that arise because of real-world realism, whereas the training of deep learning models often do not take realism into consideration. We define a realism as the reality as a result of the DL models' interaction with the CPS. There is a multitude of realisms, and in this thesis we focus on four most important ones. The first realism is task-specific, resulting from the interaction between the deep learning model and the particular environment in which the deep learning model is deployed. For example, a deep learning model that detects emotions based on people's speech must deal with environmental distortions such as reverberation. The second realism is non-targeted samples. For example, an image classifier is trained on datasets with explicit finite classes, but in the real world, it may encounter samples that belong to none of its classes; hence it is bound to misclassify them. The third realism is that many data-driven deep learning models are not robust against even minor changes. The classic example is that an image classifier may correctly identify a stop sign under ideal conditions, but it may misclassify it as something else if even a few pixels are changed. The fourth realism is the realism that comes during the process when the deep learning model is deployed. For example, deploying a deep learning model in a participant's home requires the participant to set up the equipment on which the deep learning model is installed. This realism is dependent on factors such as the participant's familiarity with (setting up) technologies.

Addressing these four realisms is important to CPS applications because CPS applications usually involve samples to be processed by the deep learning model that has these four types of realisms. For instance, a self-driving car that has a deep learning model incorporated into it to detect objects on the road must be robust to perturbations. It is common for the objects on the road to be blurry or partially blocked from the view of the camera. Adequately addressing these realisms will greatly aid the performance of deep learning models in more realistic scenarios instead of the clean, unaltered data with which they are traditionally trained.

There have been state-of-the-art works (Zeya Chen, Mohsin Y Ahmed, et al., 2019a) that address the first type of realism. For example, Chen et al. (Zeya Chen, Mohsin Y Ahmed, et al., 2019a) propose a speaker identification model that is trained on samples in which environmental distortions such as reverberation are incorporated. There have also been state-of-the-art works on using voice to detect emotions

(Dickerson et al., 2014; Vrebčević, Mijić, and Petrinović, 2019). Dickerson et al. (Dickerson et al., 2014) incorporate reverberation into their training samples, attempting to deal with the first type of realism, but they do not incorporate the deamplification effect into the samples, which renders their solution not effective in addressing this type of realism. Similarly, Vrebvcevic et al. (Vrebčević, Mijić, and Petrinović, 2019) incorporate background noise into the samples, but they do not deal with the deamplification effect. Not enough attention has been paid to the second type of realism, despite that there exist ways to potentially deal with it, i.e., the out-of-distribution (OOD) detection. To deal with the third challenge, people have been working in the area of physics-guided deep learning (Daw, Karpatne, et al., 2017; Elhamod et al., 2020). One way ((Daw, Karpatne, et al., 2017)) to do physics-guided deep learning is to incorporate the physics equation into the loss function by adding a differential equation to the terms in the original loss function. However, some constraints, called properties, are too complex and there is no single physical equation for them. To the best of our knowledge, there has not been any work that directly addresses the fourth realism. The most related works such as Ngaruiya et al. (Ngaruiya, Orwa, and Waiganjo, 2017) demonstrate that real deployment of technology is highly dependent on the familiarity of the user with technologies.

In this thesis, we address these four types of realism for two different CPS applications: smart health and smart cities. To deal with the first type of realism, which is the consequence as the result of the interaction between the deep learning model and the environment, the challenge is that deep learning models trained on clean data will not work on samples collected in such environments. To address the challenge, we use a specific acoustic application where the audio samples passed to the deep learning model for classification is environmentally distorted (for example, they are contaminated with background noise). In this application, when we train the deep learning model, we train it with audio clips that are also environmentally distorted, so that the distributions of the samples used for training and the samples used for testing are the same, hence preventing the scenario in which a deep learning classifier trained on clean samples perform poorly on distorted samples. To deal with the second realism, non-targeted samples, the challenge is that dealing with non-targeted samples is not straightforward - traditionally, deep learning samples are not trained to differentiate the samples not should classify and samples they should not. To address the challenge, we add a Mahalanobis distance-based OOD detection to filter samples that do not belong to the classes on which the deep learning classifier is trained, and in the specific application of audio classification, we have empirically proved the efficacy of this strategy. The third realism is that many deep learning models are purely data-driven and as a result, they are not robust against even minor changes. A challenge with dealing with this realism is that state-of-the-art works do not consider real-world knowledge when they train their deep learning models. To deal with this realism, we use an attention-enhanced graph neural network (GNN) architecture empowered by real-world knowledge, using both GNN and real-world knowledge as ways to increase the robustness of the deep learning models. Stepping

back, we realize that the fundamental problem with these three types of realisms is that the distribution in the training data is different from the distribution of the testing data that are collected in CPS. This fundamental problem is the premise of domain adaptation (DA), which deals with the distribution shift that takes place when trying to use a deep learning model on samples with one distribution to samples with a different distribution. Having realized this, we invented two novel (unsupervised) domain adaptation algorithms, E-ADDA and MiddleGAN, and we have empirically shown that they achieve new state-of-the-art performance on popular DA benchmarks. Last but not least, there is a fourth type of realism, which has nothing to do with the training of the deep learning models - rather, it is a direct result of the outside world, such as COVID which forbids in-person contact so the developers can't go to the smart homes where the participants live to deploy the DL models. The challenge to addressing this realism is that the participants are not tech experts which makes it hard for them to deploy the deep learning models all by themselves. In this thesis, we present a comprehensive analysis of a case study in which we deploy the eight acoustics-based deep learning models in eight smart homes and evaluate various techniques to improve the deployment time success of deep learning models in CPS.

## 1.1 Thesis Statement

Our hypothesis is that by addressing the four most prominent realisms that deep learning models face in cyber physical systems that include environmentally determined distortions, non-targeted samples, the brittleness of deep learning models, and the out-of-the-box deployment of such models by layman participants by using a combination of data augmentation, real-world knowledge-informed graph neural networks, and unsupervised domain adaptation, we can significantly improve the success of deep learning models in real world cyber physical systems.

## 1.2 Contributions

The main contributions of this thesis include:

- We identify four types of important realisms to be addressed in order to deploy deep learning models in cyber physical systems, for each of which we have presented one or more solutions.
- We identify the unique perspective that the handling of non-targeted samples is in essence an out-of-distribution (OOD) detection problem, and we provide a novel solution that incorporates OOD to successfully detect non-targeted samples.
- We identify the unique perspective that the first three realisms that deal with the distribution shift during the testing time are, in essence, the premise of domain adaptation, and we present two domain adaptation algorithms that achieve state-of-the-art performance on popular domain adaptation benchmarks.

- We invent a novel attention-enhanced integration of the diffusion theory, real-world knowledge, and GNN to counter the brittleness of deep learning models that are purely data-driven.
- We conduct a comprehensive evaluation that includes real-world in-home deployments for smart health for 6 homes each for 4 months, and a smart city application that is informed by real-world datasets.
- We invent a set of solutions for successful no-contact out-of-the-box deployments of deep learning models as a useful experience to other research teams facing similar challenges.

Overall, this thesis moves the state of the art towards more robust and realistic solutions that combine machine learning and physical systems, i.e., CPS.

### 1.3 Thesis Outline

In Chapter 3, we deal with realisms that arise in a specific application: detecting emotions from people's voices. There are two realisms in this Chapter. The first realism arises from the interaction between the deep learning model and the environment; in this case, the realism is acoustical environmental distortions to the audio samples such as reverberation, deamplification, and background noise. The second realism is the non-targeted class: people may exhibit more emotions than the total number of classes of emotions that the deep learning model is trained to classify.

Chapter 4 also deals with the task-specific realism that arises from the interaction between the deep learning model and the complex environment in which it is deployed: deammplification, reverberation, and noise-contamination of audio signals that occur in this environment.

In Chapters 4 and 6, we develop two unsupervised domain adaptation algorithms to deal with the realisms: Recall that we have listed four realisms, three of which have to do with the development stage of the DL models, and domain adaptation is effective at dealing with these three realisms, because these realisms are the result of the fact that the training samples (clean, undistorted) have a different distribution than the samples collected in the CPS where they are environmentally distorted.

In Chapter 5, we deal with the realism that the data-driven deep learning models are not robust. This is because purely data-driven models to not incorporate real-world knowledge into them. In Chapter 5, we first acknowledge that there have been attempts at addressing the brittleness of deep learning models by using graph neural networks on graph-structured data, but we discover that their results could be further improved if we add another kind of property, on top of the properties being enforced by GNN. This kind of property is called the points of interest (POIs), and we shall elaborate on what POIs are and how to incorporate them during the training of the deep learning model for traffic speed prediction in the Sections in this Chapter.

In Chapter 7, we re-confirm the claims demonstrated in Chapter 3 and 4: if you are aware of the realism (in this case, the acoustical environmental distortion) beforehand, integrating the realism (environmental distortion) into the training samples can ensure deployment time success.

In Chapter 8, we deal with the realism which has nothing to do with the training of the deep learning models - rather, it is a direct result of the outside world, such as COVID which does not allow for in-person contact so the developers can't go to the smart homes where the participants live to deploy the deep learning models.

*Chapter 2*

# RELATED WORKS

## 2.1 Detect Emotions in Realistic Home Environments

There have been studies on detecting emotions in smart-home environments. Many of these do not use speech. For example, Fernandez et al. propose a proof-of-concept architecture that detects a user's emotion by analyzing their facial expression and physiological signals (Fernández-Caballero et al., 2016). Mano et al. propose to use the facial expression of elderly people to detect their emotions; the emotion detection classifier can identify when an elderly person needs urgent help (Mano, 2018). Zhao et al. develop EQ-Radio that transmits an RF signal that will bounce off a person's body and back to the analyzing device. The RF signal carries the information of heartbeat and respiration signals that can be teased out and used for emotion recognition (M. Zhao, Adib, and Katabi, 2016). However, it is not always feasible to deploy such systems.

Other in-home systems such as Alexa and Siri are providing voice assistance for access to health-care services and support for reminders, detecting colds, etc. To date, they do not assess mood. In addtion, there have been works that use a smart home assistant to detect mood, such as Chatterjee et al. (Chatterjee et al., 2021), but this work is evaluated only on clean datasets that are not environmentally distorted. Callisto et al. (Castillo et al., 2018) also propose to use a smart home assistant robot (and available smart home speakers can fill this role) to detect emotions in smart homes, but to date this work has been only a proposal.

The works that have focused on using the acoustic modality to detect emotions on samples that are under various degrees of environmental distortions are by Triantafyllopoulos et al. (Triantafyllopoulos et al., 2019), Dickerson et al. (Dickerson et al., 2014), and Vrebcevic et al. (Vrebčević, Mijić, and Petrinović, 2019). The Deep Residual Network (Triantafyllopoulos et al., 2019) is a scalable deep learning network that can be used to detect emotions in previously unseen environments due to its noise-removal procedure. RESONATE (Dickerson et al., 2014) is a reverberation compensation approach that add reverberation (but not noise or deamplification effects) to a training corpus in order for a model trained on this corpus to be able to account for reverberation. After adapting Alex-Net so it is suitable to process sound, Vrebcevic et al. (Vrebčević, Mijić, and Petrinović, 2019) train the classifier on samples that are contaminated with ambient noise (but there are no reverberation or deamplification effects).

The works that have focused on detecting emotions in real-time include Lech et al. (Lech et al., 2020) that adapts AlexNet for speech processing but does not deal with environmental distortions, Cen et al. (Cen et al., 2016) that develops a system consisting of voice activity detection and emotion recognition. However, Cen et al.

| Work | Approach | Performance | Environmental Distortion |
|---|---|---|---|
| (Datcu and Rothkrantz, 2005) | GentleBoost | 86.3% | × |
| (Lugger and Yang, 2007) | Linear Discriminant Analyses | 81.8% | × |
| (Altun and Polat, 2007) | Support Vector Machine | 85.5% | × |
| (Danisman and Alpkocak, 2008) | Ensemble of SVM | 86.3% | × |
| (K. Wang et al., 2015) | Fourier Transform + SVM | 79.51% | × |
| (J. Deng et al., 2017) | Autoencoder-Based Domain Adaptation | 62.0% | × |
| (Alex, Babu, and Mary, 2018) | Convolutional Neural Network | 69% | × |
| (Triantafyllopoulos et al., 2019) | Deep Residual Network | 86.3% (unweighted aver. recall) | ✓ |
| (Dickerson et al., 2014) | RESONATE | 80% (approximate) | ✓ |
| (Vrebčević, Mijić, and Petrinović, 2019) | Alex-Net + Data Augmentation | 34.03% | ✓ |
| (Burkhardt et al., 2005) | Evaluation by Humans | 86.0 % | × |

Table 2.1: Evaluation of previous works on EMO-DB. The performance, if not otherwise noted, is measured by accuracy. Three of the works attempt to deal with environmental distortions. The definition of environmental distortions is defined differently in different works.

(Cen et al., 2016) is evaluated on a simulated online learning environment so the testing samples are absent of environmental distortions. Not taking environmental distortions explicitly into account is a very prevalent problem with works on emotion detection in real-time or time-sensitive tasks; works that suffer from this problem also include Stolar et al. (Stolar et al., 2017), Fayek et al. (Fayek, Lech, and Cavedon, 2015), and Bahreini (Bahreini, Nadolski, and Westera, 2016).

A published summary (Eyben et al., 2015) shows acoustic features or parameters that have been used for emotion detection, encompassing frequency, time, and amplitude. Since a huge variety of features are used in different previous works and different works uses different set of these features, direct comparison and cross-examination on the effectiveness of the features prove challenging. The summary (Eyben et al., 2015) proposes that emotions from the datasets analyzed can be represented by two features: valence, which represents the degree of positivity or negativity of an expressed emotion, and arousal, which represents the energy or intensity of the expressed emotion. Trigeorgis et al. (Trigeorgis et al., 2016) use valence and arousal for emotion detection from speech. The usage of acoustics-based emotion recognition using valence and arousal extends from emotion classification from speech to classifying emotions based on music (Grekow, 2018), which provides a set of features that describe valence and arousal in a musical piece. However, when it comes to realistic deployments and detecting emotions from speech in the wild, the features extracted from the raw audio signal must be resistant to both background noise and various combinations of reverberation factors. The demonstration that the features descriptive of valence and arousal will not be sensitive to environmental distortions remains lacking.

Other popular datasets of acted emotional speech include the Surrey Audio-Visual Expressed Emotion Database (SAVEE) (Haq and Jackson, 2010), the Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS), (Livingstone and Russo, 2018) and the Crowd-sourced Emotional Multimodal Actors Dataset (CREMA-D) (Cao et al., 2014). These three datasets consist of both audio-video and audio-only samples. Other works (Beard et al., 2018), (Ghaleb, Popa, and Aste-

| Work | Dataset(s) | Approach | Accuracy | Distortions |
|---|---|---|---|---|
| (Huang and S. Narayanan, 2018) | RAVDESS, SAVEE | Shake-Shake Regularization | 60.8 % | × |
| (Beard et al., 2018) | SAVEE, CREMA-D | LSTM encoder | 41.2 % | × |
| (Ghaleb, Popa, and Asteriadis, 2019) | RAVDESS | Temporal DNN | 67.7 % | × |
| (Ghaleb, Popa, and Asteriadis, 2019) | CREMA-D | Temporal DNN | 74.0 % | × |
| (Wijayasingha and Stankovic, 2021) | RAVDESS | Noise Mitigation + CNN | 81.0 % | ✓ |
| (Shchetinin et al., 2020) | RAVDESS | Noise Mitigation | 80.0 % | ✓ |
| Our baseline | RAVDESS, EMA, CREMA-D, TESS, SAVEE | Hierarhical Classifier | 94.7 % | ✓ |
| Our baseline | RAVDESS, EMA, CREMA-D, TESS, SAVEE | Hierarhical Classifier | 88.4 % | × |

Table 2.2: Evaluation of works on SAVEE, CREMA-D, RAVDESS, EMA, TESS. The performance, if not otherwise noted, is measured by accuracy. The definition of environmental distortions is defined differently in different works. Except for the hierarchical classifier, the other items are published state-of-the-art works on emotion recognition on (a subsets of) the datasets that our solution is trained on. The hierarchical classifier is developed by us to serve as a baseline that we compare against our solution because it is trained on the same datasets that our solution is trained on. On clean samples, our baseline achieves an accuracy of 94.7% which outperforms the first three baselines (two of which use the same algorithm) evaluated on clean speech. On environmentally distorted samples, our baseline achieves an accuracy of 88.47%, which out-performs the last three baselines that are tested with environmental distortions.

riadis, 2019), (Barros and Wermter, 2016), (Noroozi et al., 2017) seek to exploit the multi-modality of the datasets, while works such as (Huang and S. Narayanan, 2018) only use the audio clips from SAVEE and RAVDESS. Despite the high accuracy achieved by some of the works on one or more of the three multi-modal datasets, they still haven't explicitly addressed the issue of environmental distortions. Salekin et al. (Salekin et al., 2017) publish the Distant Emotion Recognition in which they select features that are robust to distance and only extract those features when processing a speech signal. However this work does not evaluate its model on speech that are contaminated with background noise, so it is hard to tell if it is robust to this particular kind of environmental distortion. Another work (Wijayasingha and Stankovic, 2021) improves the robustness of speech-based emotion recognition by considering the magnitude spectrogram and the modified group delay spectrogram. This work considers both noise and reverberation, but they simply show that their work is robust noise-contaminated speech samples and there is no evaluation on how their model fares when different combinations of reverberation factors, such as the decay factor and diffusion, are combined.

Since confounding emotions are largely overlooked in research papers by the acoustic-based emotion detection community, we are unable to find any research that indicates the effort of filtering out confounding emotions so that such samples will not be sent to the classifier and result in a wrong classification. A possible approach to discern an emotional utterance sample that is of emotions that the classifier is not trained to classify is to detect if this sample is within the distribution of the training samples. Hendrycks et al. discovers that, when wrongly classified, an out-of-distribution sample results in a generally smaller softmax probability than an in-distribution sample that is correctly classified (Hendrycks and Gimpel, 2016).

Based on this observation, they propose to detect an out-of-distribution sample by comparing it within the context of the statistics of the softmax probability scores by all samples (in the testing set). This approach's performance is improved by Liang et al. (Liang, Yixuan Li, and Srikant, 2017) who use temperature scaling and input perturbing for out of distribution detection. Both Liang et al. and Hendrycks et al.'s approaches are out-perormed by a Mahalanobis distance-based approach: Lee et al. (K. Lee et al., 2018) proposes a way to detect out-of-distribution samples to prevent them from being sent to the softmax layer of a deep neural network and result in a wrong classification. For each training sample, Lee et al. calculate the activation of the penultimate layer (the layer that will forward its activation to the output layer) to get the distribution of the training samples. After training, when a previously unseen sample is sent to the classifier for classification, the activation of the penultimate layer of the neural network is calculated before being sent to the output layer. The Mahalanobis distance (Mahalanobis, 1936) from the activation of the penultimate layer by this previously unseen sample to the distribution of the activations of the penultimate layers by training samples thisen be calculated. The Mahalanobis distance measures how many standard deviations a data point is away from the distribution of a group of data points.

## 2.2 Detect Conflict in Realistic Home Environments

### Unsupervised Domain Adaptation

A large amount of work has been done on UDA by minimizing the dissimilarity between the distributions of the source and target domains. The common measurements of domain dis-similarity include KL divergence, and maximum mean discrepancy (MMD). Extensive research on transfer learning is dedicated to minimizing the dis-similarity measurements (S. Zhao, Qiu, and Y. He, 2021). The minimization of dis-similarity measurements is also used with other measurements, such as classification loss on the source to find features that both discriminate and are domain-invariant (Tzeng, Hoffman, N. Zhang, et al., 2014). However, the minimization of MMD of domains jeopardizes the locality structure of samples and potentially reduces the effectiveness of transfer learning (L. Zhang et al., 2019). Also, feature discriminability is also decreased due to the unintentional minimization of joint variance of features from source and target sets (Wei Wang et al., 2020).

Adversarial-based UDA has been a popular sub-field of UDA (Hoffman et al., 2018; M. Xu et al., 2020; Tzeng, Hoffman, Saenko, et al., 2017; Ganin, Ustinova, et al., 2016; Tang and K. Jia, 2020). Adversarial-based UDA can be grouped into generative and non-generative categories. The methods of the generative category attempt at generating samples to aid the final classification of the target samples. For example, CyCADA (Hoffman et al., 2018) adapts source samples to appear as if they are from the target domain, and then trains a category classifier on these adapted source images with their true labels to classify the target data. Similarly, DM-ADA trains the generated auxiliary images that are source-like and the category classifier together from the embeddings of the source and the target domains. The non-

generative methods attempt to achieve domain confusion, which usually requires a generator/encoder and a domain label discriminator to engage in a mini-max game. The discriminator attempts at recognizing the domain label of a given sample, and the generator/encoder attempts at masking the source images to be target-like, or vice versa. For example, ADDA (Tzeng, Hoffman, Saenko, et al., 2017) makes the generator/encoder on the target samples train against a domain label discriminator, and the goal is to obtain a target generator/encoder that can successfully mask the target samples as if they were from the source domain. Consequently, a category classifier trained on the source samples and their true labels can be used to classify these encoded target samples. RSDA's (Gu, J. Sun, and Z. Xu, 2020) idea on how to achieve UDA is similar to the vanilla non-generative idea with the mini-max game, with a twist that they define the neural networks in the spherical feature space. Our E-ADDA is in the non-generative category.

**Out-Of-Distribution Detection**

A lot of attention has been paid to detecting abnormal samples so that they can be intercepted before being sent to a neural network. Specifically, (Hendrycks and Gimpel, 2016), (K. Lee et al., 2018), and (Liang, Yixuan Li, and Srikant, 2017) are three state-of-the-art approaches to detect out-of-distribution samples. Liang et al. (Liang, Yixuan Li, and Srikant, 2017) observe that fabricating small perturbations into samples as well as using temperature scaling can separate the softmax scores of in-distribution and out-of-distribution samples. Lee et al. (K. Lee et al., 2018) use Mahalanobis distance to separate in-distribution samples from out-of-distribution ones.

Lee et al. (K. Lee et al., 2018) provide a comparison of the three approaches and the performances of the three approaches are indicated in Table 2.3, from which we observe that the Mahalanobis distance based approach outperforms Softmax Probability (Hendrycks and Gimpel, 2016) and ODIN (Liang, Yixuan Li, and Srikant, 2017). Therefore, in the rest of the paper, we use the Mahalanobis distance-based approach for out-of-distribution detection. For details, see Section 3.3.

| | Softmax Probability | Mahalanobis | ODIN |
|---|---|---|---|
| Acc. | 85.06% | **95.75%** | 91.08% |

Table 2.3: The performances of the three state-of-the-art out-of-distribution detection algorithms. The metric is accuracy. The performances are obtained when training ResNet on CIFAR-10 and SVHN samples are used as out-of-distribution samples.

There have not been enough works on incorporating out-of-distribution detection with transfer learning or domain adaptation. Perera et al. (Perera and Patel, 2019) use an out-of-distribution dataset to improve the performance of a classifier on in-distribution samples, which is the only work that intends to combine the two

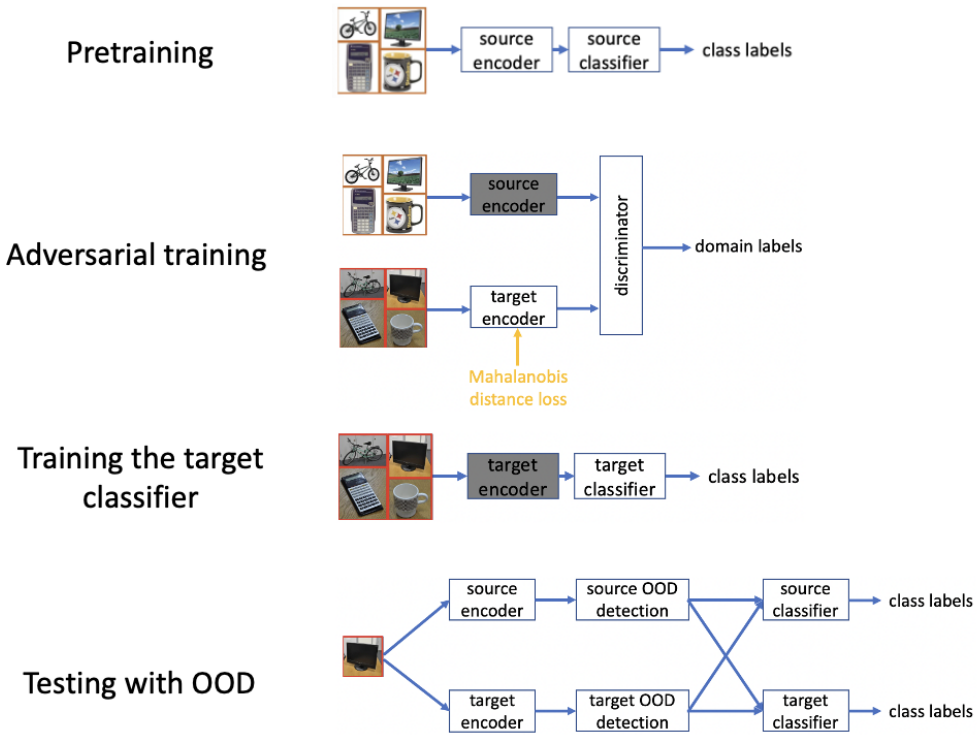



Figure 2.1: The flowchart of E-ADDA: the pretraining, adversarial training, target category classifier training and testing phases. In the pretraining phase, the source encoder $E_s$ and the source category classifier $F_s$ are trained end-to-end using the source samples and their labels. In the adversarial training phase, we freeze the source encoder $E_s$ and train the target encoder $E_t$ and the discriminator $D$ adversarially by engaging them in a mini-max game. To train $E_t$, in addition to the adversarial loss, we incorporate the Mahalanobis distance loss defined in Equation 4.4. To train the target category classifier $F_t$, we freeze the adversarially trained $E_t$ and train $F_t$ using its outputs on the target samples. Note that $F_t$ is trained using the pseudo-labels of the target domain samples. During the testing phase, each sample $x$ (in the testing set of) the target domain, $E_s(x)$ and $E_t(x)$ are calculated to determine if the domain confusion is successful. If the domain confusion is not successful, $E_t(x)$ is sent to the target category classifier $F_t$ instead of $F_s$.

knowledge fields. Our work, E-ADDA, is one of the first approaches that use out-of-distribution to improve the performance of (unsupervised) domain adaptation.

### 2.3 Incorporate Properties into Models that Are Modeling Cyber Physical Systems

Traffic forecasting has seen a lot of interest in recent years (X. Wang et al., 2020; Sutskever, Vinyals, and Le, 2014; Yaguang Li et al., 2017; Z. Wu, Pan, G. Long, Jiang, and C. Zhang, 2019). In this section, we first talk about the statistical methods and the methods using classical machine learning such as the Support Vector Regression (SVR) model. Then, we talk about attempts to use neural networks. Last but not least, we talk about the recent advances in using graph neural networks (GNNs) to model the spatiotemporal dependencies that exist inherently in the traffic data.

Traditional statistical methods for traffic prediction include the historical average (HA) model. It divides the traffic flow into periods and applies the weighted average from the previous periods as the result for future prediction. Its drawback is that it ignores the spatial and temporal dependencies in the traffic data and treats each period as a stationary and unchanging entity. Classical machine learning methods such as the SVR model have also shown promise in traffic prediction; however, deep learning-based methods such as FC-LSTM (Sutskever, Vinyals, and Le, 2014), which is an improvement over the LSTM model by adding to the LSTM model hidden units, shows improvement over the classical machine learning methods.

In recent years, researchers have been paying a lot of attention to GNN-inspired methods, such as the Diffusion Convolutional Recurrent Neural Network (DCRNN) (Yaguang Li et al., 2017). DCRNN has been proposed. It interprets the traffic flow as information infusing to each node and diffusing from that node. Similarly, Graph WavNet (Z. Wu, Pan, G. Long, Jiang, and C. Zhang, 2019) is proposed, using the stack of the gated TCN and GCN layers that tease out the temporal and spatial dependencies respectively. MTGNN (Z. Wu, Pan, G. Long, Jiang, Chang, et al., 2020) is a direct improvement on Graph WavNet in which the authors incorporated a novel mix-hop propagation layer to further capture the temporal dependencies. Last but not least, GNN-based methods include ASTGCN (Guo et al., 2019) which adds the attention mechanism to the vanilla GNN architecture, STSGCN (Song et al., 2020) which pays more attention to the heterogeneity of the spatiotemporal correlations in traffic flow data, GMAN (Zheng et al., 2020) which stacks up the spatial attention and temporal attention mechanism, and DGCRN (F. Li et al., 2021) that interprets the graph as a dynamic entity in which the spatial and temporal correlations are captured by the dynamic graph convolutional recurrent model.

### 2.4 Use GAN to Generate Domain Agnostic Samples

In this Section we present two important areas of research related to our work: the Generative Adversarial Nets (GANs), and recent advances in domain adaptation.

**Generative Adversarial Nets**

Goodfellow et al. (Goodfellow, Pouget-Abadie, et al., 2014) propose the original GAN which consists of two neural nets: the discriminator and the generator. The two nets engage in a minimax game where the generator attempts to generate images from noise to fool the discriminator, while the discriminator attempts to distinguish generated images from real ones. Inspired by this work, works such as the CGAN (Mirza and Osindero, 2014) and ACGAN (Odena, Olah, and Shlens, 2017) attempt to regulate the classes of the generated images. The CGAN (Mirza and Osindero, 2014) is constructed in the way that, during training, the label information is fed to both the generator and the discriminator. The discriminator of the ACGAN (Odena, Olah, and Shlens, 2017) has two objective functions: to maximize the log likelihood that a given sample is of the correct source (generated or real), and to maximize the log likelihood that the label (which class is this sample from) of the sample is correct. Moving past the GANs that leverage label information, another set of GANs focus on cycle consistency of the generated images. For example, the CycleGAN (J.-Y. Zhu et al., 2017) employs two generators, one to translate an image from the source to the target and the other to translate back a translated image by the first generator. A cycle consistency loss is added to minimize the discrepancy between an original, unaltered image, and the image translated by the first generator and then translated back by the second generator. StarGAN (Choi et al., 2018) addresses the scalability issue that different GAN models need to be created for all pairs of domains. Unlike the previous works (Mirza and Osindero, 2014; Odena, Olah, and Shlens, 2017) which use only one generator, the MiddleGAN employs two discriminators and one generator and aims to generate samples that are similar to both the source domain samples and the target domain samples. MiddleGAN is a GAN designed specifically for Domain Adaptation while the other previous works on GAN mentioned in this section seek to generate realistic samples or achieve style transfer.

**(Unsupervised) Domain Adaptation Using Adversarial Approaches**

There are two types of (unsupervised) domain adaptation methods using generative approaches.

The first type is non-generative, which aims to achieve domain confusion via adversarial training. The MiddleGAN is not non-generative, but it is adversarial-based, so it is only fit that we elaborate on the advances of this type of methods. The general idea behind this type of methods is that an encoder-generator and a domain discriminator engage in a mini-max game in which the domain discriminator is trained to differentiate the origin of a sample (source or target) and the encoder-generator is trained to mask the domain labels of samples. RSDA (Gu, J. Sun, and Z. Xu, 2020) proposes to engage the encoder-generator and the domain discriminator in the mini-max game, but their novelty is that the mini-max game happens in the spherical space. Specifically, RSDA consists of a spherical classifier to predict class labels and a spherical domain discriminator to predict domain labels. ADDA (Tzeng, Hoffman, Saenko, et al., 2017) is another non-generative adversarial approach - it achieves domain confusion by encoding the source and target samples and training

the two encoder-generators with the domain discriminator in the mini-max game. In DANN (Ganin and Lempitsky, 2015), the domain discriminator is connected to a gradient reversal layer (GRL), which multiplies the gradient with a negative number during the back-propagation process. This results in the maximization of the discriminator loss and ensures the feature representations of the source and target domains are similar. In DADA (Tang and K. Jia, 2020), the authors develop an adversarial objective that results in the inhibitory relationships of the class and domain predictions. In other words, the domain output and the true class output compete with each other, which results in the explicit alignment of the joint distribution and consequently improves the performance of predicting the target data. The second type is generative, which aims at generating samples to aid in the cause of a better performance in the target data classification. For example, DM-ADA (M. Xu et al., 2020) trains the target classifier while generating samples that are source-like from the embeddings learned of both the source and the target domains. The generated samples are theoretically compelled to appear as if they were samples from the source domain whereas the input samples' class information is preserved. Another example, CYCADA (Hoffman et al., 2017) attempts to obtain adapted pixel-level representations and feature-level representations. Structural consistency is enforced both locally and globally via cycle-consistency loss and semantic loss. As a result, CYCADA adapts the source samples to appear as if they are sampled from the target domain. The MiddleGAN is similar to DM-ADA and CYCADA because it also generates new samples to aid the process of training a target label classifier. However, existing generative adversarial solutions aim at directly adapting source samples to be target-like (CYCADA), or the other way around (DM-ADA), meanwhile the MiddleGAN aims to generate samples that are, distribution-wise, in the middle of the source and the target domains.

## 2.5 Real Deployments in which Realisms Are Present

In this Related Works section, we discuss not only the state-of-the-art of conflict detection but also the components required to make deploying the conflict detection model possible: the state-of-the-art on both VAD and SID.

### Voice Activity Detection

There have been lots of work on voice activity detection (VAD) so we only discuss the recent advances in the field. MarbleNet (F. Jia, Majumdar, and Ginsburg, 2021) uses deep residual network consisting of blocks of 1-D time-channel separable convolution, able to achieve the state-of-the-art performance with the advantage that the number of their parameters is significantly smaller. The robustness of MarbleNet is also extensively studied to demonstrate that it is robust to real-world acoustical distortions. Using teacher-student training, Dinkel et al. (Dinkel et al., 2021) also strive to train a model that is robust to real-world acoustic distortions. Dinkel et al. identify that traditional VAD algorithms are trained on data devoid of such acoustic distortions, and therefore their usage is limited to data without the acoustic distortions that are inevitable in the real world, rending them unable to perform well

in real-world settings. Other works on VAD include Wang et al. (Weiqing Wang, Qin, and M. Li, 2022) that uses a cross channel attention based model to achieve voice activity detection in the M2met challenge, Braun et al. (Braun and Tashev, 2021) that is specifically concerned about dealing with the robustness issue of many state-of-the-art models. What is worth-noting is that, some works developed for other purposes such as transcription, can be used as voice activity detection models. For example, the Google speech Recognition (GSR), a transcription service, outputs the transcribed sentence from an audio clip if that audio clip is speech, and it will throw an exception is the audio clip is silence. It is worth noting that although works such as MarbleNet (F. Jia, Majumdar, and Ginsburg, 2021) and Dinkel et al. (Dinkel et al., 2021) attempt to ensure that they work on datasets that account for realism to be encountered in real, designated environments in which the algorithms are to be deployed, they do not evaluate their post-deployment performances in the real, designated environments. Realisms that the VAD model deals with usually arise from background noise such as footsteps, and the VAD model needs to differentiate not only silence from human speech but also those background noises from speech. The realisms that the VAD model faces is simpler than the models that we discuss in the later sections, the speaker identification (SID) model, the emotion detection model, and the conflict detection model, which needs to deal with the tv sound as the speech from the tv could affect the classification performance of these models.

**Speaker Identification**

Again, the works in the field of speaker identification (SID) are abundant, so we only discuss the recent advances in the field. Chen et al. (L. Chen, Ravichandran, and Stolcke, 2021) introduce a graph-based speaker identification model that is reliant on speaker label inference. It is particularly concerned with the task of SID in household scenarios. WavLM (S. Chen et al., 2022) recognizes that the speech content by by speakers contains multi-faceted information such as the identities of the speakers, the content of the speech, and paralinguistics. WavLM is propsoed as a pre-trained model that can be used to be fine-tuned for the purpose of various speech recognition tasks such as speaker identification. Snyder et al. (Snyder et al., 2018) proposes an xvector, the results of mapping variable-length spoken clips to fixed-dimensional embeddings. Again, works such as Chen et al. (L. Chen, Ravichandran, and Stolcke, 2021) evaluate their algorithms on datasets in which the realism to be encountered in real, designated environments in which the algorithms are to be deployed, but no post-deployment evaluation is presented in such works to show if their approaches to deal with the realism are successful. Realisms that the SID model faces arise from background noise, especially the tv sounds. Note that the realisms that the SID model needs to deal with are more complex than the realisms that the VAD model needs to deal with, as voice from the tv could confound the model from correctly identifying the identity of the speaker in an audio clip. In other words, the SID model needs to be able to deal with more complex background noise (more complex acoustical realisms) than the VAD model.

**Conflict Detection**

There have been several attempts to detect verbal conflict using sound signals that a microphone picks up from the ambient environment. A work (Lefter and Jonker, 2017) creates verbal conflict between pairs of a student and an actor who act out conflict. From the generated conflict episode, it is observed that overlapped speech is an important indicator of interpersonal conflict (Lefter and Jonker, 2017). However, they did not create a model of automatic conflict detection based on their conclusion. Based on the fact that repetition of parts of speech, such as syllables, phrases and words, is indicative of interpersonal conflict, another work (Letcher et al., 2018) develops a repetition detection model that uses the audio files collected by the on-body sensors of police officers to detect conflict. However, the interpersonal conflicts that police officers encounter during their jobs are not the same as every-day interpersonal conflicts that take place in households between arguing family members. The state-of-the-art modules on automatic conflict detection using speech (Caraty and Montacié, 2015; Grezes, Richards, and Rosenberg, 2013), achieve satisfactory performance on their respective datasets, but their approaches are not evaluated to demonstrate if acoustic distortions of noise, distance, and reverberation affect the results. As a result, the automatic detection of every-day harmful interpersonal conflicts among people in home environments remains unsolved. Again, in addition to the realisms such as reverberation, common indoor background noise, and deamplification, the conflict detection model, just as the emotion detection model, needs to deal with the realisms that are the tv sound: the characters on the tv might be in a verbal conflict (as the background sound for the participants whose conflict we want to monitor), which can confound the conflict detection model.

*Chapter 3*

# DETECT EMOTIONS IN REALISTIC HOME ENVIRONMENTS

Chapter 3 deals with realisms that arise in a specific application: detecting emotions from people's voices. The realism in this Chapter is that in a complex cyber physical environment such as a smart home, the acoustic signals are reverberated, deamplified, and contaminated with background noise. A second realism in this Section is that people exhibit an many different emotions, but a deep learning model is only trained on a dataset with finite classes.

There are two constraints posed by the realism that arise in a smart home if we want to detect the emotions of a registered speaker. The first constraint is environment distortions that consist of room reverberation, deamplification effect, and background noise. Reverberation occurs when audio signals bounce off of the furniture in the room. Deamplification effect occurs when the speaker is not right next to the microphone. Background noise such as cutlery sounds and object impact sounds is an inevitable part of human daily life. These three types of environmental distortions make the audio samples' quality deteriorate. The deteriorated audio samples have a different distribution compared to the clean samples collected in controlled lab environments with high-end acoustic equipment that maximizes the mitigation of the effect of environmental distortions on audio samples. The second constraint is that a deep learning classifier is trained on a dataset with a finite number of classes of emotions, whereas humans are capable of expressing a lot more emotions than the emotion categories that appear in the dataset on which the deep learning model is trained. When the speaker demonstrates an emotion not registered by the dataset, the deep learning model has no choice but to classify it as one of the classes in the dataset, therefore making a mistake. The two constraints must be dealt with in order for a deep learning model to detect emotions in realistic home environments.

## 3.1 Introduction

Many research papers from the field of psychology (Gross and Muñoz, 1995), (Cheng, Friesen, and Adekola, 2019) (Gross, H. Uusberg, and A. Uusberg, 2019), (Cai et al., 2018) have shown that emotional health is a crucial part of one's well-being. The rapid development of machine learning in the field of acoustic signal processing has resulted in a surge of interest in detecting emotions from speech. Recent publications (Jalili et al., 2018), (Fernandes et al., 2018), (Choudhury et al., 2018), (Zamil et al., 2019) have classified emotions based only on acoustics with no visual images. These acoustics-based emotion detection algorithms have the potential to monitor people's emotions 24 hours a day and consequently play a vital role in maintaining people's emotional well-being. For example, in one important application, family or informal caregivers for persons with dementia can benefit from these algorithms. These caregivers are twice as likely to experience emotional

difficulties compared to the caregivers of patients living with diseases other than dementia (Gaugler et al., 2019). Because long-term untreated emotional difficulties are linked to mental health disorders, helping the caregivers of persons with dementia become aware of their emotions is of paramount importance. However, it is impractical to appoint a human professionally trained to detect the onset of various emotions in the household of each pair of dementia caregiver and care recipient dyad. Therefore, in this application area, the need for emotion detection algorithms is pressing. It is also well known that many other in-home and in-office applications can also benefit from detecting emotions.

The development of machine learning in the field of speech-based emotion recognition has produced solutions such as (Alex, Babu, and Mary, 2018), (Triantafyllopoulos et al., 2019), (K. Wang et al., 2015). The early works in speech-based emotion detection were developed for clean and acted speech and for a fixed set of emotions such as happiness, anger, sadness, and neutrality. One caveat with using such datasets to develop an emotion detection algorithm is that the datasets and algorithms assume that people only exhibit one of those emotions. Recent works have considered the reality of speech-based emotion recognition by taking into account the effect of environmental distortions such as de-amplification, background noise, and reverberation caused by sound signals bouncing on objects. However, these solutions often consider one dataset at a time and consider that all emotions are accounted for in the model developed on that one dataset.

In spite of the progress made on emotion detection from speech, challenges remain to more accurately handle in the wild situations and to identify emotions of interest among the vast array of human emotions exhibited by individuals in every day life. Out of the state-of-the-art algorithms published on emotion recognition (Huang and S. Narayanan, 2018), (Beard et al., 2018), (Ghaleb, Popa, and Asteriadis, 2019), (Salekin et al., 2017), (Wijayasingha and Stankovic, 2021), (Shchetinin et al., 2020), in Table 2.2, only three of them deals with environmental distortions such as deamplification, background noise, and reverberation. Out of the three works that deals with environmental distortions, there is no evaluation on how their algorithm fares when different combination of reverberation factors, such as the decay factor and diffusion, are combined.

Based on the state-of-the-art in this field, **Key Challenges** for speech-based emotion detection are:

- Many speech-based emotion detection algorithms are developed on datasets of either clean speech or speech that are environmentally distorted in various degrees, but they assume that people will only exhibit one of the emotions accounted for in the dataset.
- The effect that environmental distortions has on the classification of different emotions is not well studied, because, despite that there exist works (Shchetinin et al., 2020), (Wijayasingha and Stankovic, 2021) that take envi-

| Dataset | HAP | ANG | NEU | SAD | SUR | FEA | DIS | CAL |
|---|---|---|---|---|---|---|---|---|
| CREMA-D | ✓ | ✓ | ✓ | ✓ | × | ✓ | ✓ | × |
| RAVDESS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| SAVEE | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | × |
| TESS | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | × |
| EMA | ✓ | ✓ | ✓ | ✓ | × | × | × | × |

Table 3.1: The components of the original datasets: CREMA-D, SAVEE, RAVDESS, TESS, and EMA. The emotions in this Table are: Happiness, Anger, Neutrality, Sadness, Surprise, Fear, Disgust, and Calm.

ronmental distortions into account, they don't study to what extent does each environmental distortion affect the classification of different emotions.

**The main contributions** of this work are:

- We created a combined CNN and out of data distribution (OOD) solution that performs in the range of 90% accuracy even in the presence of non-targeted emotions, 11 realistic home noises, deamplification due to distance from the microphones, and reverberations due to different room types. This is a significant improvement over a state-of-the-art model, the hierarchical classifier described in Section 5.2 whose accuracy over the same testing set (that includes environmentally distorted samples and samples of non-targeted class) is 56.2%. The hierarchical classifier outperforms five state-of-the-art models on samples with and without environmental distortions (See table 2.2).
- To address the first challenge, we show that explicitly training on non-interested confounding emotions where you have data plus employing an OOD technique for non-interested emotions where you don't have data outperforms using just an OOD for all non-interested emotions by more than 10%.
- To address the second challenge, we demonstrate how different environmental distortions affect the classification results: background noise and deamplification have the most impact on the decrease of classification accuracy by 7.3%, followed by room reverberation that results in the decrease of classification accuracy by 4.5%.
- Accurate mood detection by our solution allows the emotions of users in a smart home to be automatically detected in a passive manner. Upon detection, while not part of this thesis, relaxation and mindfulness techniques that have been shown to alleviate unhealthy mood can be recommended to the users.

### 3.2 Synthetic Datasets

There are 5 well known datasets used for emotion detection: CREMA-D, SAVEE, RAVDESS, TESS, and EMA. Table 3.1 shows what emotional utterances are in each of the datasets. Even combining these datasets results in too limited data and that data would not account well for noise, reverberation and deamplification. Consequently, we not only combine these 5 datasets to provide more samples and to be more general, but we also develop multiple synthetic datasets. These datasets include clean speech (to serve as a baseline), noisy speech, reverberated speech, and combining all the factors and adding deamplification. This enables the evaluation to show the effects of each factor as well as the overall situation that closely represents realistic environments. The following subsections describe how we built the synthetic datasets from the five publicly available sets of emotional speech. We also describe how we create out of distribution samples to test the case when humans exhibit emotions that are not accounted for in the training.

The fact that all five of the publicly available datasets have happy, angry, neutral, and sad classes suggests that these four emotions are acknowledged to be commonplace, but distinctive enough to be separated from other emotions. As a result, these four classes of emotions are also included in our synthetic datasets as emotions of interest. In addition to these four classes, we also have a class of confounding emotions, which are emotions that are distinct enough to not be confused with any of the four commonplace emotions. Because surprise is similar to happiness (for example, TESS lists the surprised emotional utterances are speech samples of "pleasant surprise") (Dupuis and Pichora-Fuller, 2010), and boredom is similar to neutrality, surprise and boredom are not considered confounding emotions, rather variations of happiness and neutrality. This leaves us with two confounding emotions - fear and disgust, and one class of out-of-distribution emotion, calmness. The class of confounding emotions in our synthetic datasets consists of fearful and disgusted speech samples.

**Generate padded samples that are otherwise not environmentally distorted, denoted as $\mathcal{D}_1$**

After getting rid of corrupted and otherwise unusable samples from the five sets of emotional speech, we end up with the set of clean audio samples of various lengths. Since most of the audio clips are less than or equal to 4 seconds, we pad each of them into a 5-second window. In order to do the padding, we first generate a 5-second segment of pure silence audio clip and randomly decide the index of a frame in the 5-second segment and overlay a sample of emotional speech with the silence with that frame index as the starting point. This strategy of padding results in a more diverse set of samples of emotional speech in comparison to the strategy in which each sample of emotional speech is overlaid with the silence segment starting universally at a fixed index. Our padding strategy also results in a more realistic dataset that resembles the set of emotional speech segments collected in the wild, as it is unreasonable to expect that each speech segment is perfectly captured by the microphone right when the first syllable of the segment is spoken. By providing

| Event | Instances |
|---|---|
| (object) rustling | 60 |
| (object) snapping | 57 |
| cupboard | 40 |
| cutlery | 76 |
| dishes | 151 |
| drawers | 51 |
| glass jingling | 36 |
| object impact | 250 |
| people walking | 54 |
| washing dishes | 84 |
| water tap running | 47 |

Table 3.2: Events that are present in the background noise collected from real homes from the dataset (Mesaros, Heittola, and Virtanen, 2016). All of them are covered in the process of contaminating audio samples with background noise.

a more diverse and more realistic way of padding, we increase the diversity of our dataset and this contributes to the robustness of the classifier.

The complete set of generated clean padded samples is referred as $\mathcal{D}_1$. In $\mathcal{D}_1$, there are 1792 happy samples, 1793 angry samples, 1573 neutral samples, 1793 sad samples, and 1837 samples of confounding speech. In total, there are 8788 samples.

**Generate de-amplified, noise-contaminated samples, denoted as $\mathcal{D}_2$**
For each audio clip in $\mathcal{D}_1$, we create two copies of them. For each of the two copies, we randomly de-amplify it with $m$ decibels such that $m \leq 12$. Then, we randomly choose among the dataset of background noise collected in real homes which are around 5-minutes long; within a certain chosen clip of background noise in real home environments, we randomly take a 5-second audio segment from it, and overlay it with the de-amplified sample. Table 7.2 illustrates the events that are present in these audio clips of home environments. The way we overlay the background noise clips with the duplicate samples ensures that the household events in Table 7.2 are present in the resulted audio clips of the overlaying.

The set of de-amplified, noise-contaminated samples is referred as $\mathcal{D}_2$. In $\mathcal{D}_2$, there are 3584 happy samples, 3586 angry samples, 3146 neutral samples, 3586 sad samples, and 3674 samples of confounding speech. In total, there are 17576 samples.

**Generate reverberated samples, denoted as $\mathcal{D}_3$**
For each audio clip in $\mathcal{D}_1$, we duplicate it once. The duplication is then reverberated. The reverberation effect is generated by the combination of three reverberation factors: the wet/dry ratio $r$, diffusion $d$, and the decay factor $f$. Each time an

audio sample is reverberated, a random set of values for the wet/dry ratio, diffusion, and the decay factor is generated. The choice of the reverberation parameters and the set of their values is also used in DER (Distant Emotion Recognition) (Zeya Chen, Mohsin Y. Ahmed, et al., 2019b) which seeks to solve the problem of speaker identification by generating different reverberation models to represent difficult rooms in which a speaker might be present. Their approach yields almost zero error rate when tested in real-time when human participants were speaking in a room where a microphone array was present. The performance achieved by DER indicates that the different combinations of the given reverberation parameters and their given ranges are able to describe typical indoor environments. Therefore, we adopt the same reverberation model to change the audio samples as if they were collected in the indoor environments described by the different combinations of the reverberation parameters.

The set of reverberated samples is referred as $\mathcal{D}_3$. In $\mathcal{D}_3$, there are 1792 happy samples, 1793 angry samples, 1573 neutral samples, 1793 sad samples, and 1542 samples of confounding speech. In total, there are 8493 samples.

**Generate samples that are de-amplified, noise-contaminated, and reverberated, denoted as $\mathcal{D}_4$**

For each audio clip in $\mathcal{D}_2$ that is not of a confounding emotion, we duplicate it once. For each duplication, we reverberate it in the same way as we obtain the reverberated samples in $\mathcal{D}_3$. The set of samples that are de-amplified, noise-contaminated, and reverberated is denoted as $\mathcal{D}_4$. In $\mathcal{D}_4$, there are 3584 happy samples, 3586 angry samples, 3146 neutral samples, and 3586 sad samples. In total, there are 13902 samples.

$$\mathcal{D} = \cup_{i=1}^{n} \mathcal{D}_i, n = 4 \tag{3.1}$$

$$\cap_{i=1}^{n} \mathcal{D}_i = \varnothing, n = 4 \tag{3.2}$$

**Training and testing sets, denoted as $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$**

After generating the above synthetic datasets we take the union of them for overall evaluation. We split the union of the synthetic datasets into training and testing sets. To ensure that all the emotion classes are equally represented in both the training and testing sets, we randomly select 80% of confounding samples, 80% of happy samples, 80% of angry samples, 80% of neutral samples, 80% of sad samples and use them for training, while the rest of the samples are used for testing. The 80% of samples selected from an emotion consists of samples from $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$, $\mathcal{D}_4$ so the environmental distortions and their combination are accounted for the testing set.

Let $\mathcal{D}_{train}$ denote the training set and $\mathcal{D}_{test}$ denote the testing set. Equation 3.3 describes the relationship between the entirety of the synthetic datasets, $\mathcal{D}$, and $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$. The mutual exclusion of $\mathcal{D}_{train}$ and $\mathcal{D}_{test}$ is described in Equation 3.4.

$$\mathcal{D} = \mathcal{D}_{test} \cup \mathcal{D}_{train} \tag{3.3}$$

$$\mathcal{D}_{test} \cap \mathcal{D}_{train} = \varnothing \tag{3.4}$$

Equations 3.5 and 3.6 describe the subsets of the training and testing sets based on the subsets' relationship to $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$, and $\mathcal{D}_4$. Each of $\mathcal{D}_i, i \in \{1, 2, 3, 4\}$ shares some members with the training set. Similarly, it also shares some members with the testing set. For example, $\mathcal{D}_{train,1}$ denotes the set of samples in the training set that are not distorted in anyway, and $\mathcal{D}_{test,3}$ denotes the set of samples in the testing set that are reverberated, but are neither deamplified nor contaminated with noise.

$$\mathcal{D}_{train,i} = \mathcal{D}_{train} \cap \mathcal{D}_i, i \in \{1, 2, 3, 4\} \tag{3.5}$$

$$\mathcal{D}_{test,i} = \mathcal{D}_{test} \cap \mathcal{D}_i, i \in \{1, 2, 3, 4\} \tag{3.6}$$

**Generate samples that are out of the distribution of the training and testing sets**

We have addressed the importance of considering confounding emotions. However, the number of confounding emotion classes to be included in the training and testing sets is finite. If an emotional speech segment is not within the distribution of the samples of interested and confounding emotions in the training set, this speech segment does not belong to any of the interested or confounding classes. Classifying such a sample will be pointless, as the classifier will make mistakes on classifying it. Therefore, out-of-distribution detection is crucial, and it will further assist in filtering out emotional utterances that are not of the interested classes.

We generate out-of-distribution samples to test the performance of the out-of-distribution technique. We use calm speech segments from TESS as out-of-distribution samples, because in-distribution samples will be happy, angry, neutral, sad, or confounding (disgusted or fearful). We generate synthetic calm samples in the same way that we generate synthetic samples of other emotions. First, we pad them to 5-second window, following the exact step described in Section 3.1. This group of padded but not otherwise environmentally distorted calm samples is represented by $O_1$. Then, we create three copies of them. For the two copies of each padded calm sample, we de-amplify them and contaminate them with noise, as described in Section 3.2. This group of calm samples is represented by $O_2$. The other copy of the padded calm sample is reverberated, following the same procedure described in Section 3.3. This group of calm samples is represented by $O_3$. Then, we make a copy of the calm samples that are de-amplified, noise-contaminated, but not reverberated. For all the copies, we reverberate them, so they become de-amplified, noise-contaminated, and reverberated. This group of calm speech segments is represented by $O_4$. Collectively, the entire set of calm speech segments is represented

| Low-Level Descriptors (LLD's) | Amounts |
|---|---|
| Mel-Frequency cepstral coefficients (MFCC) 1-13 | 104 |
| Delta coefficients for MFCC 1-13 | 104 |
| Zero-crossing rates | 8 |
| Delta coefficients for zero-crossing rates | 8 |
| Root-mean-square signal frame energy | 8 |
| Delta coefficients for root-mean-square signal frame energy | 8 |
| Spectral centroid features | 8 |
| Delta coefficients for the spectral centroid related features | 8 |
| Pitch-related features | 8 |
| Delta coefficients for the pitch-related features | 8 |
| Total amount | 272 |

Table 3.3: The 272 low-level descriptor features

by $O$, which is the union of $O_1$, $O_2$, $O_3$, and $O4$. Note that these calm speech segments are in neither training nor testing sets.

### 3.3 Feature Selection

In this work we use 272 low-level descriptor features associated with emotions because these are common to many previous solutions such as (Salekin et al., 2017). Table 3.3 provides a summary of these features.

### 3.4 Solution

**Overview**

To illustrate the importance of considering confounding emotions, we have developed two emotion detection algorithms. One of the algorithms serves as a baseline and is a hierarchical structure that consists of 3 CNN's and this algorithm only has four classes (section 5.4). It is able to achieve high accuracy even in the presence of background noise, de-amplification, and reverberation, under the condition that all audio clips passed to this structure are happy, angry, neutral, or sad utterances. However, as the Evaluation section shows, the accuracy significantly drops once the condition no longer holds - in other words, the structure fails to perform adequately if speech samples of confounding emotions are added which is the typical situation in the real world.

The second algorithm (sections 5.2 and 5.3) is our solution. Due to the importance of considering confounding emotions, our solution is a CNN classifier that has five classes - confounding emotions (emotions that we are not interested in, but for which we have data), happiness, anger, neutral, and sadness and to these 5 classes we add an OOD component to capture confounding emotions where we do not have data.

Our solution works with 5-second audio clips. If the input is shorter than 5-seconds, we pad it with the padding algorithm described in the synthetic datasets section.

Note that the authors of (Salekin et al., 2017) show that 5-second audio samples obtained from padding the original samples from datasets yields high accuracy across various speaker-to-microphone distances. After making the input to be exactly 5-seconds long, we slice it into 48 small, overlapping frames, each of which lasts for 25 ms. From each small frame, we obtain 272 LLD features. As a result, the input of our classifier is a tensor of 48 frames $\times$ 272 channels per frame. The hyper-parameters we choose are provided in (Salekin et al., 2017), as these values for the hyper-parameters are shown to achieve good classification accuracy.

**The 5-Class CNN Classifier**

Although neural networks sometimes yield good performance with minimally tuned hyper-parameters, (Goodfellow, Bengio, and Courville, 2016) suggests that performance will significantly increase if a thorough tuning of hyper-parameters is performed. Grid search is recommended by (Goodfellow, Bengio, and Courville, 2016) as an approach to perform hyper-parameter optimization when there are relatively few hyper-parameters. When performing grid search over a set of hyper-parameters, the programmer assigns a small, finite set of values for each hyper-parameter. The grid search algorithm loops over the combinations of the specified hyper-parameters and trains the model multiple times to iterate through the combinations. The combination of the hyper-parameters that results in the model that yields the least validation set error is the final choice of the hyper-parameters.

The hyper-parameters that we choose to tune are a standard choice for tuning CNN's. Due to the relatively small number of the hyper-parameters we have, we choose to perform grid search on them, instead of random search, another hyper-parameter optimization algorithm that is quicker but less thorough. We have performed grid search over the following hyper-parameters: epoch, batch size, the number of convolutional layers, the number of kernels in each layer, the size of kernels in each layer, the stride size of max-pooling, the activation function (ReLU and Leaky ReLU), the optimizer, the learning rate, the decay ratio, and the size of the dense layers after the convolutional neural nets are flattened. Our final model consists of four convolutional layers. After the output of the final CNN is flattened, three dense layers of 2048 neurons are attached. We choose the adam optimizer with the learning rate as 1e-4 and decay ratio as 0. The optimal batch size is 128. The optimal epoch is 1000. The optimal filter size is 3 (filters are measured by the number of small frames).

The classifier is implemented with Keras, a neural network API using Tensorflow backend and trained on the samples in the training set, $\mathcal{D}_{train}$. For each sample $\boldsymbol{X}$ in $\mathcal{D}_{train}$, we slice them into 48 overlapping small frames $x_1 ... x_n, n = 48$ with hop length of 5. Hop length is defined as the amount of samples between two small frames. Librosa, a python package on acoustic signal analysis, is used to extract the 272 features for each small frames $s_i$ such that $i \in \{1, ..., 48\}$. As we have described in Table 3.3, the features can be categorized into (1) MFCC's 1-13, (2) the delta coefficients of MFCC's 1-13, (3) zero-crossing rates, (4) the delta-coefficients of zero-crossing rates, (5) RMSE, (6) delta-coefficients of RMSE, (7) spectral centroids, (8) delta-coefficient of spectral centroids, (9) pitch-related

| | Softmax Distribution | ODIN | Mahalanobis |
|---|---|---|---|
| Accuracy | 85.0% | 91.0% | 95.7 |

Table 3.4: The detection accuracy of three out-of-distribution detection algorithms. The in-distribution samples are CIFAR-10 samples. Samples of SVHN are out-of-distribution samples (K. Lee et al., 2018). The metric is accuracy.

features, and (10) their delta-coefficients. For each of these categories, we calculate the minimum, the maximum, the median, the mean, the standard deviation, the variability, the skewness, and the kurtosis. This indicates that, for the first category, we have $13 \times 8$ features, for the second category, we have $13 \times 8$, for the rest of the categories, each category has $1 \times 8$ features. In total, we have $2 \times 13 \times 8 + 8 \times 1 \times 8 = 272$ features.

**The Detection of Out-Of-Distribution Samples**

The fifth class (the confounding emotions class) of the classifier described above is to prevent disgusted and fearful emotional speech segments (as examples of emotions for which we are not interested in but have data) from being classified as happiness, anger, neutrality, or sadness. However, this classifier can handle only six most basic emotions (happiness, anger, neutrality, sadness, fear, and disgust), and human beings are capable of expressing other emotions. Table 3.1 shows that the TESS dataset considers calmness to be an emotion category that is different from the aforementioned six emotions.

In order to further filter out emotional speech segments that are not included by the 5 emotions, we use the out-of-distribution technique (K. Lee et al., 2018) using Mahalanobis distance. We select this technique because it outperforms two other major out-of-distribution detection algorithms - softmax Distribution and ODIN as indicated in Table 3.4. In the following paragraphs, we describe in detail the Mahalanobis distance-based OOD detection technique.

Let $\boldsymbol{x}_i$ and $y_i$ represent a sample in the training set and its label. Let $N_c$ represent the number of all samples in the training set whose label is of class $c$. Equation 3.7 defines the empirical mean of a class $c$ and $f$ represents the activation of the penultimate layer of our 5-class CNN by the input $\boldsymbol{x}_i$. $\hat{\mu}_c$ is the mean of the activations of the penultimate layer of our 5-class CNN by all samples in the training set whose label is of class $c$.

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i:y_i=c}^{N} f(\boldsymbol{x}_i) \tag{3.7}$$

Equation 3.8 (K. Lee et al., 2018) defines the empirical covariance matrix, an important part required to represent the distribution of the training set. After the empirical means for all $c \in C$ are calculated, whereas $C$ is the set of all possible

labels, $(f(\boldsymbol{x}_i) - \hat{\mu}_c)$ is used to measure how each samples in the training set deviates from the empirical mean of all samples in the same class, $c$. $(f(\boldsymbol{x}_i) - \hat{\mu}_c)(f(\boldsymbol{x}_i) - \hat{\mu}_c)^\top$ results in a symmetric matrix with numbers of the rows and columns equal to the number of neurons of in the penultimate layer of our 5-class CNN. By looping through all classes, the empirical covariance matrix of the training set is calculated.

$$\hat{\Sigma} = \frac{1}{N} \sum_{c}^{C} \sum_{i:y_i=c}^{N} (f(\boldsymbol{x}_i) - \hat{\mu}_c)(f(\boldsymbol{x}_i) - \hat{\mu}_c)^\top \tag{3.8}$$

After the empirical means for all classes and the covariance matrix for the training set are computed, we have obtained the two crucial components that collectively describe the distribution of the training set so that a new incoming sample that is two many standard deviations away from the this distribution is considered abnormal and should be prevented from being sent to the output layer of the classifier. Equation 3.9 (K. Lee et al., 2018) describes how to calculate the Mahalanobis distance of a previously unseen sample $\boldsymbol{x}$ to our 5-class CNN after the training phase of the CNN. Since there are multiple classes, we need to calculate the Mahalanobis distance for each class $c$ based on its $\mu c$. After the calculation of the Mahalanobis distances for all the classes, we pick the class such that the Mahalanobis distance from the sample $\boldsymbol{x}$ is the smallest.

$$C_M(\boldsymbol{x}) = \underset{c}{\operatorname{argmin}} \ (f(\boldsymbol{x}) - \hat{\mu}_c)^\top \hat{\Sigma}^{-1} (f(\boldsymbol{x}) - \hat{\mu}_c) \tag{3.9}$$

After we obtain the Mahalanobis distance from $x$, the previously unseen sample that we want to classify, we check if its Mahalanobis distance is in a threshold obtained during the validation phase with direct experiment. If yes, the this sample is in the distribution of the training set, indicating that it is of happiness, anger, neutrality, sadness, disgust, or fear. Since it is in distribution, the activation of the penultimate layer when given $x$ will be forwarded to the output layer of the 5-class CNN. If the sample is out of the distribution of the training set, then it is not of happiness, anger, neutrality, sadness, disgust and fear. Therefore, forwarding it to the output layer of our 5-class CNN will result in a wrong classification no matter what, so we do not forward the penultimate layer's activation of this sample to the output layer.

**The Hierarchical Structure of Classifiers**

As mentioned above, to further demonstrate the value of our solution, we compare it to a hierarchical set of 3 classifiers as a baseline. Hierarchical structures attempt to leverage the information given by the higher-level classifiers to reduce the complexity of the problem that lower-level classifiers need to solve (Bennett and Nguyen, 2009). Our hierarchical set of three classifiers is trained on $\mathcal{D}_{train}$, the same training set on which our solution is trained, for the purpose of a better comparison on the performance of the hierarchical structure and our solution. This baseline demonstrates how a false sense of accuracy can be achieved by only testing on

targeted emotions when other emotions also exist. We intend to use the hierarchical structure to prove the hypothesis that a mood detection algorithm that achieves high accuracy on interested emotions - in our case, happiness, anger, neutrality and sadness, will not perform adequately in a scenario where confounding emotions are present.

The hierarchy baseline consists of 3 binary CNN classifiers: $\boldsymbol{C}_1$, a classifier that separates happy and angry audio clips from the neutral and sad audio clips, $\boldsymbol{C}_2$, the classifier that separates happy and angry audio clips, and $\boldsymbol{C}_3$ the classifier that separates neutral and sad audio clips. The three classifiers are trained with the same batch size, which is 128, the same optimizer, adam, and the same learning rate (1e-4) and decay (0), and the same kernel size (3 small frames). All of them are trained in 1000 epochs with an early stop of 50 epochs.

**The classifier that separates happy and angry audio clips from the neutral and sad audio clips, represented as $\boldsymbol{C}_1$.** This $\boldsymbol{C}_1$ decides if a given audio input is in either of the two classes: (1) the input is of happy or angry speech; (2) the input is of neutral or sad speech. This classifier returns a vector of size 2, whereas the first value represents the score computed by the CNN that this input of emotions is happiness or anger and the second value, the score for neutrality or sadness. Based on these scores, the audio input is passed to $\boldsymbol{C}_2$ or $\boldsymbol{C}_3$. $\boldsymbol{C}_1$ is trained on samples of happiness, anger, neutrality, and sadness in $\mathcal{D}_{train}$.

**The classifier that classifies happy and angry audio clips, represented as $\boldsymbol{C}_2$.** If $\boldsymbol{C}_1$ decides that a given input is either a happy speech or an angry speech, that input is passed to this classifier which further determines if the speaker that produced the audio clip is happy or angry. The output is a vector of size 2, whereas the first value represents the score that the input is a happy speech and the second value represents the score that the input is an angry speech. $C_2$ is trained on samples of happiness and anger in $D_{train}$.

**The classifier that classifies neutral and sad audio clips, represented as $\boldsymbol{C}_3$.** If $\boldsymbol{C}_1$ decides that a given input is either a neutral speech or a sad speech, the input is passed to this classifier which further determines if the speaker that produced the audio clip was neutral or sad. The output is a vector of size 2, whereas the first value represents the score that the input is a neutral speech and the second value represents the score that the input is sad speech. $C_3$ is trained on samples of neutrality and sadness in $D_{train}$.

### 3.5 Evaluation

After training, the evaluation is conducted on $\mathcal{D}_{test}$ for both our 5-class CNN solution and our baseline, the hierarchical classifier.

#### Evaluation to Show the Necessity of Adding Environmental Distortions to Training Samples

We place an emphasis on the importance/necessity of adding reverberation, background noise, and deamplification effect into samples in the training set. Before

| Dataset | f1 score |
|---|---|
| Clean samples in testin set | 73.84% |
| VoxCeleb | 17.51% |

Table 3.5: The left column consists of the two datasets that the classifier trained on only the clean samples is evaluated on. The difference in the f1 score demonstrates the necessity of adding environmental distortions. The metric is f1 score.

we start evaluating our solution, the 5-class classifier, and the baseline we built, which is the hierarchical classifier, we present experimental results to demonstrate the importance/necessity of adding environmental distortions; see Table 3.5.

In this experiment, we have trained a classifier using *only clean samples* of the five classes (happiness, anger, neutrality, sadness, fear/disgust) from the synthetic training dataset. For this classifier, we evaluate it in two ways. First, we evaluate it on samples that are not environmentally distorted. These clean samples are the samples from the testing set that have not been environmental distorted in any way. Second, we evaluate it on a subset of a real-life dataset, VoxCeleb.

VoxCeleb (Nagrani, Chung, and Zisserman, 2017) contains utterances extracted from YouTube videos in which celebrities give talks or attend interviews. We evaluate the clean classifier on a subset of VoxCeleb. Voxceleb is a very imbalanced dataset, for there are only 216 angry samples in the entire testing set. Since we want to have a more balanced dataset, the subset of VoxCeleb we choose consists of: 500 happy samples, 216 angry samples, 500 neutral samples, 500 sad samples, and 500 samples that are not of the happy, angry, neutral, and sad emotions As seen in Table 3.5, the clean classifier yields an f1 score of 73.84% on the not environmentally distorted samples, but it drops to 17.51% on the subset of the real-life dataset. The decrease in the performance is by 56.33%. This demonstrates that the classifier trained on clean speech does not maintain the same level of performance on real-life dataset. **As a result, adding environmental distortions are necessary.**

**Evaluation on the Hierarchical Structure of CNN's, the Baseline**

The purpose of the hierarchical structure solution is to serve as the baseline, as stated before. It's performance is superior to the state-of-the-art on clean samples without non-targeted emotions. This baseline achieves an accuracy of 94.7%, and it outperforms 5 state-of-art algorithms on mood detection (see Table 2.2) on clean samples that do not include non-targeted samples.

We now show that the hierarchy of classifiers, only trained on happy, angry, neutral, and sad examples, are able to achieve very high accuracy when only evaluated on these emotions. However, it will always make mistakes if the input is of a confounding emotion, such as disgust. This is the case of many state-of-the-art classifiers on emotion detection; they are able to perform accurately only on certain

| Classifier | Happy | Angry | Neutral | Sad | Overall |
|---|---|---|---|---|---|
| $\boldsymbol{C}_1$ | 99.7% | 99.6% | 100% | 100% | 99.8% |
| $\boldsymbol{C}_2$ | 95.5% | 95.0% | | | 95.3% |
| $\boldsymbol{C}_3$ | | | 92.9% | 96.1% | 94.6% |

Table 3.6: The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_1$ that contains only samples of interested emotions that are not environmentally distorted. On the entire testing set, the hierarchical structure achieves an accuracy of 94.7%. The metric is accuracy.

| Classifier | Happy | Angry | Neutral | Sad | Overall |
|---|---|---|---|---|---|
| $\boldsymbol{C}_1$ | 95.9% | 98.4% | 98.4% | 98.5% | 98.2% |
| $\boldsymbol{C}_2$ | 95.4% | 94.3% | | | 94.8% |
| $\boldsymbol{C}_3$ | | | 92.2% | 92.8% | 92.5% |

Table 3.7: The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_2$ that consists of only samples of interested emotions that are de-amplified and mixed with noise. The metric is accuracy.

emotions, but they are not expected to be as accurate as they are in environments where different kinds of emotions are omnipresent. In addition, by evaluating the hierarchical classifier, we have further confirmed that environmental distortions affect the classifier's performance.

**The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_1$**

Table 3.6 shows the performance of the hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_1$. Recall that $\mathcal{D}_{test}$ is the testing set, $C$ is the set of every samples of confounding emotions, and $\mathcal{D}_1$ is the set of samples that are not distorted. Thus $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_1$ is the subset of the testing set that contains only happy, angry, neutral, and sad samples that are not distorted. The three CNN classifiers in the hierarchy achieve 99.8%, 95.3%, and 94.6% of accuracy respectively, which suggests that, without environmental distortions, the hierarchy can achieve a very high level of accuracy, as many state-of-the-art algorithms on mood detection do.

**The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_2$**

$(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_2$ is the subset of the testing set that contains only happy, angry, neutral, and sad samples that are de-amplified and then mixed with background noise. Table 3.7 describes the performance of $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, and $\boldsymbol{C}_3$ when evaluated only on samples of interested emotions that are de-amplified and contaminated with background noise. The three CNN classifiers in the hierarchy achieve 98.2%, 94.8%, and 92.5% of accuracy respectively. The accuracy of the three classifier drops by 1.6%, 0.5%, and 2.1% when compared to their performance on samples of targeted

| Classifier | Happy | Angry | Neutral | Sad | Overall |
|---|---|---|---|---|---|
| $\boldsymbol{C}_1$ | 99.4% | 99.3% | 99.6% | 99.6% | 99.5% |
| $\boldsymbol{C}_2$ | 88.9% | 89.2% | | | 89.1% |
| $\boldsymbol{C}_3$ | | | 86.1% | 93.2% | 89.6% |

Table 3.8: The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_3$ that contains only samples of interested emotions that are reverberated. The metric is accuracy.

| Classifier | Happy | Angry | Neutral | Sad | Overall |
|---|---|---|---|---|---|
| $\boldsymbol{C}_1$ | 95.9% | 98.1% | 98.4% | 98.1% | 97.6% |
| $\boldsymbol{C}_2$ | 92.3% | 89.8% | | | 91.0% |
| $\boldsymbol{C}_3$ | | | 88.4% | 91.9% | 90.3% |

Table 3.9: The hierarchical structure evaluated on $\mathcal{D}_{test} \setminus C$, that contains all samples of interested emotions. The metric is accuracy.

emotions that are not distorted, demonstrating that de-amplification and noise have impact on the performance of the hierarchy but the hierarchical structure is still able to compete with state-of-the-art mood detection algorithms in terms of accuracy.

**The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C) \cap \mathcal{D}_3$.**

Table 3.8 shows the hierarchy's performance in terms of weighted accuracy on the samples of interested emotions that are reverberated. Compared to their performance on samples of interested emotions that are not environmentally distorted, the accuracy of the three classifiers all drop: $\boldsymbol{C}_1$'s accuracy drops to 99.5% by 0.3%, $\boldsymbol{C}_2$'s accuracy drops to 89.1% by 6.2%, $\boldsymbol{C}_3$'s accuracy drops to 89.6% by 5%. Reverberation distorts the samples, and the drop of accuracy is expected. The fact that the drop in accuracy is insignificant illustrates that the hierarchical structure is robust to reverberation.

Compared to their performance when evaluated on samples of interested emotions that are distorted by de-amplification and background noise shown in Table 3.7, $\boldsymbol{C}_1$ is more resistant to reverberation (drop by only 0.3% on reverberated sample), while $\boldsymbol{C}_2$ and $\boldsymbol{C}_3$ are more resistant to de-amplification and background noise (drop by 0.5% and 2.1% on samples that are de-amplified and mixed with background noise), as indicated in Table3.7 and Table 3.8.

**The hierarchical structure evaluated on $(\mathcal{D}_{test} \setminus C)$**

Table 3.9 illustrates the performance of the classifiers evaluated on the samples of interested emotions in $\mathcal{D}_{train}$. $\boldsymbol{C}_1$ achieves an accuracy of 97.6 %. $\boldsymbol{C}_2$ achieves an accuracy of 91.0%. $\boldsymbol{C}_3$ achieves an accuracy of 90.3%. Since $\mathcal{D}_{test} \setminus C$ is the set of samples of interested emotions that are either not environmentally distorted

| Classifier | Happy | Angry | Neutral | Sad | Confounding | Overall |
|---|---|---|---|---|---|---|
| $\boldsymbol{C}_1$ | 95.9% | 98.1% | 98.4% | 98.1% | 0% | 83.3% |
| $\boldsymbol{C}_2$ | 92.3% | 89.8% | | | 0% | 68.4% |
| $\boldsymbol{C}_3$ | | | 91.9% | 90.3% | 0% | 66.6% |

Table 3.10: The hierarchical structure evaluated on $\mathcal{D}_{test}$, the entire testing set that contains both samples of interested and confounding emotions. On the entire testing set, the hierarchical structure achieves an accuracy of 56.2%. The metric is accuracy.

or environmentally distorted by the myriad combinations of the de-amplification amount measured in decibels, various segment of background noise collected from real home environments, and the three reverberation factors, this set is descriptive of speech samples of interested emotions that will take place in home environments. The high accuracy of the three classifiers demonstrate that the hierarchical structure on emotion detection of the interested emotions (happiness, anger, neutrality, and sadness) is robust to environmental distortions in home environments.

**The hierarchical structure evaluated on $\mathcal{D}_{test}$**

Table 3.10 shows the performance of the three classifiers of the hierarchy when tested on four of the emotions of interest and the confounding emotion. Table 3.10 shows the disadvantage of a common theme of emotion detection classifiers: emotions outside those of interest are assumed to not exist. However, many other emotions may exist in the real world and, thereby, reduce the overall accuracy of classifiers solely trained on the emotions of interest.

In Table 3.10, the three classifiers still achieve the exact same performance on the interested emotions, happiness, anger, neutrality, and sadness as Table 3.9. However, since confounding emotions are introduced and the hierarchical structure must classify the samples of confounding emotions (fear and disgust) as one of the interested emotions, it is bound to make mistakes. As a result, $\boldsymbol{C}_1$, $\boldsymbol{C}_2$, and $\boldsymbol{C}_3$ achieves an accuracy of 0% on confounding samples. This result in a significantly decrease of the their overall performance measured in accuracy: $\boldsymbol{C}_1$'s accuracy drops from 97.6% to 83.3%, $\boldsymbol{C}_2$'s accuracy from 91.0% to 68.4%, and $\boldsymbol{C}_3$'s accuracy from 90.3% to 66.6%. The deterioration of the performance of the three classifiers in the hierarchy suggests that classifiers that are trained to achieve very high accuracy on only interested emotions will not perform adequately when given samples of confounding emotions.

**Evaluation on the 5-class classifier Without Out-Of-Distribution Samples**

Next, we evaluate our solution, the 5-class CNN, to understand its performance on recognizing each emotion with and without environmental distortions.

Tables 3.11 and 3.12 show the evaluation of the 5-class classifier in four different scenarios that account for environmental distortions. The first column is the

| Class | $\mathcal{D}_{test,1}$ | $\mathcal{D}_{test,2}$ | $\mathcal{D}_{test,3}$ | $\mathcal{D}_{test}$ |
|---|---|---|---|---|
| Happy | 92.6% | 92.2% | 80.2% | 89.6% |
| Angry | 92.3% | 92.4% | 84.8% | 88.9% |
| Neutral | 95.9% | 91.3% | 90.7% | 88.7% |
| Sad | 94.9% | 88.6% | 93.8% | 89.1% |
| Confounding | 88.9% | 79.0% | 78.4% | 81.4% |
| Mean | 92.9% | 88.4% | 85.6% | 88.0% |

Table 3.11: Evaluation on the 5-class classifier that categorizes its input into categories: Happiness, Anger, Neutrality, Sadness, and Confounding emotions.The metrics for evaluation is accuracy. The average is weighted.

| | $\mathcal{D}_{test,1}$ | $\mathcal{D}_{test,2}$ | $\mathcal{D}_{test,3}$ | $\mathcal{D}_{test}$ |
|---|---|---|---|---|
| f1 score | 93.2% | 86.1% | 87.6% | 87.9% |

Table 3.12: Evaluation on the 5-class classifier on different subsets of the testing set. **The metric for evaluation is f1 score**. The average is weighted. Since our synthetic dataset is very well balanced, we can see that the f1 scores are very similar to the scores in Table 3.11. The metric is f1 score.

evaluation on this classifier on samples in the testing set that are padded, but no environmental distortion is introduced to the sound. The second column is the evaluation on this classifier on padded samples that are reverberated. The third column is the evaluation on this classifier only on padded samples that are de-amplified and mixed with noise. The last column is the evaluation on this classifier on the entirety of the testing set. The samples used for evaluation in Table 3.11 are denoted as $\mathcal{D}_{test,k}, k = 1, 2, 3$, defined as Equation 3.10.

$$\mathcal{D}_{test,k} = \mathcal{D}_{test} \cap \mathcal{D}_k, k = 1, 2, 3 \tag{3.10}$$

The average accuracy of the 5-class CNN drops from 92.9% when evaluated on $\mathcal{D}_{test,1}$, the set that resembles the idealistic scenario in which the speaker is close to the microphone and the room reverberation and background noise have minimal impact on the quality of the sound signal captured by the microphone, to 85.6% when reverberation is introduced, and to 88.4%, when de-amplification and background noise are introduced. This indicates that environmental distortions such as reverberation, background noise, and de-amplification still affect the performance of the 5-class CNN - which is as expected, since the signal is distorted, the classification result of any classifier is expected to deteriorate slightly, moderately, or severely depending on the robustness of the classifier. Evaluated on $\mathcal{D}_{test}$, the weighted average of accuracy of our classifier is 88.0%, which is only a 4.9% drop of accuracy from 92.9% obtained from the evaluation on the ideal $\mathcal{D}_{test,1}$ dataset. In other words, our 5-class CNN deteriorates only by 4.9%, despite the various combinations of en-

vironmental distortions that are collectively illustrated by the reverberation factors (the wet/dry ratio, decay factor, and diffusion), the number of value measured in decibels deduced from the amplitude, and the various background noise from home environments.

The previous paragraph discusses the overall performance of the classifier in $\mathcal{D}_{test,i}, i = \{1, 2, 3\}$ and $\mathcal{D}_{test}$, but we can also observe how the classifier performs in the four scenarios on each emotion. Its accuracy on happiness drops from 92.6% on happy samples in $\mathcal{D}_{test,1}$ to 80.2% on happy samples in $\mathcal{D}_{test,2}$ when reverberation is introduced; meanwhile, the accuracy on happy samples in $\mathcal{D}_{test,1}$ drops only 0.4% when compared to its accuracy on happy samples in $\mathcal{D}_{test,3}$. The difference between the two drops in accuracy suggests that reverberation is harder to deal with by our classifier than noise and de-amplification for happiness. Similarly, when classifying angry samples, the drop from the accuracy achieved on angry samples in $\mathcal{D}_{test,1}$ to the accuracy achieved on angry samples in $\mathcal{D}_{test,2}$ is 7.5%, while the accuracy on angry samples that are distorted and de-amplified actually increases 0.1% compared to the accuracy achieved on angry samples that are not distorted at all. The classifier's performance on happy and angry samples suggests that noise and de-amplification have minimal influence on our classifier's performance when compared to reverberation.

On neutral and sad samples, the observation that noise and de-amplification have less influence on the classifier's performance than reverberation no longer holds. Compared to the classifier's performance on neutral samples that are not altered, the accuracy drops from 95.9 % to 90.7% by 5.2% and 91.3% by 4.6%, respectfully on reverberated neutral samples and neutral samples that are de-amplified and contaminated with noise. Reverberation and amplification with noise result in similar deterioration of the classifier's accuracy on neutral samples. On sad samples, reverberation results in a drop in accuracy from 94.8% to 93.8% caused by reverberation, and to 88.6% caused by de-amplification and noise. The observation on the classifier's performance on distorted sad samples is that reverberation has minimal influence on the classifier's performance when compared to de-amplification and noise.

On confounding samples, the classifier achieves an accuracy of 88.9% on clean samples, but it drops to 78.4% by 10.5% and 79.0% by 9.9% on reverberated samples and samples that are de-amplified and contaminated with noise. For happy, angry, neutral, and sad samples, the drop of accuracy obtained from clean samples to distorted samples is always less than 5%, which is also expected, because the set of confounding emotions $\mathcal{C}$ is more complex than the sets $\mathcal{H}$, $\mathcal{A}$, $\mathcal{N}$, $\mathcal{S}$. $\mathcal{C}$ consists of more samples of more than one emotions, while the others consists of samples of only one emotion.

Note that the entirety of the testing set $\mathcal{D}_{test}$ is descriptive of the actual environment in which the classifier is envisioned to be deployed, because it encompasses a variety of different combinations of the environmental distortions illustrated by different factors that range from having no effect on the acoustic signals to significantly

distorting the acoustic signals.

Our solution reaches an accuracy of 92.9% on $\mathcal{D}_{test,1}$, the set of samples that are not environmentally distorted, for all emotions, and it achieves 88.0% high accuracy in $\mathcal{D}_{test}$, for all emotions. The environmental distortions only reduce the overall accuracy of our solution by 4.9%. Therefore, our solution is highly robust to environmental distortions.

The evaluation of our solution described in Table 3.11 illustrates the pattern that noise and de-amplification have less impact on the performance of our solution than reverberation for happy, angry, neutral, and confounding emotions. However, sad utterances do not following the same pattern, as noise and de-amplification decrease our solution's performance by 6.3% while reverberation decreases our solution's performance by only 1.1%. This is because the original sad utterances have lower volume (measured in decibels) compared to the other emotions. During the process of adding environmental distortions, we subject all utterances to the same standards (for example, the same range measured in decibels of possible de-amplification). As a result, the sad utterances, whose volumes are lower than the other emotions, are more susceptible to de-amplification.

**5-class classifiers with out-of-distribution detection technique versus 4-class classifiers with out-of-distribution detection technique**

We have stated that, out of the 5 classes of our solution, there is one class that we are not interested in. The aforementioned performance that the 5-class classifier achieved with out-of-distribution detection technique prompts us to ask the question - what if we train a classifier only on the four interested classes and use the out-of-distribution detection technique to intercept emotional utterances of other classes? In other words, we investigate the value of having a confounding class. In the following paragraphs, we answer the question:

**With the out-of-distribution technique and samples of classes that we are not interested in, which approach is better: (1) should we include those samples during training, or (2) should we exclude them from training and let the out-of-distribution algorithm intercept samples from this "confounding"/uninterested class during training?**

To do so, we train a 4-class classifier (the four classes being happiness, anger, neutrality, sadness). To make it comparable to the 5-class classifier, the 4-class classifier is required to achieve the similar level of performance on testing samples of those four emotions as the 5-class classifier on the testing samples of five emotions (happiness, anger, neutrality, sadness, and confounding). Via direct experiment, we have obtained a 4-class classifier that achieves an f1 score of 87.18% on all the samples of the 4 classes in the testing set. It achieves a similar level of performance the 5-class classifier achieves (with an f1 score of 87.9%).

Having obtained a 4-class classifier, we evaluate it the same way we evaluate the 5-class classifier - first, just itself without out-out-distribution detection technique;

| | without OOD | with OOD |
|---|---|---|
| f1 score | 65.86% | 76.66% |

Table 3.13: Evaluation on the 4-class classifier on the combined set of the testing set and the set of the Calm emotion. Samples of the calmness class and the confounding class are considered out-of-distribution, since the 4-class classifier is trained on and can only assign any given input to the four targeted classes. The metric is f1 score.

second, this 4-class classifier paired up with the out-of-distribution detection technique.

Without the out-of-distribution detection technique, the 4-class classifier's performance drops from an f1 score of 87.9% to 65.86%, by 22.04%, as every sample of the calmness class and the confounding class is predicted to be of the happy, anger, neutrality, and sadness class. As a result, the predictions on calm samples and samples of the confounding class are always wrong.

With the out-of-detection technique to intercept samples that are potentially of classes unknown by the 4-class classifier, the f1 score improves to 76.66%, by 10.8% compared to its performance without the out-of-distribution detection technique. However, it is still 11.24% lower than the 4-class classifier's performance on testing samples from its 4 targeted classes.

The fact that the out-of-distribution detection technique improves the performance significantly by 10.8% demonstrates that it is still advantageous to pair the classifier with the out-of-distribution detection technique.

However, in Section 6.6, we demonstrate that, with the out-of-distribution detection technique, the 5-class classifier's performance (an f1 score of 87.71%) on its targeted classes and a previously unseen class is almost identical to its performance (an f1 score of 87.9%) without the out-of-distribution detection technique. The same improvement is not observed when we pair the 4-class classifier with out-of-distribution detection technique. The less significant improvement (by 10.8%) suggests that the samples in the confounding class are very similar to one or more of the 4 targeted classes. As a result, the out-of-distribution detection technique is not effective at picking them out and allows them to be passed to the 4-class classifier.

As a result, we conclude that **With the out-of-distribution technique and samples of classes that we are not interested in, we should include those samples during training**: there is always a possibility that the samples of the uninterested classes resemble one or more of the samples of the interested classes. In this case, the out-of-distribution is not effectively at distinguishing samples of interested classes from samples from uninterested classes, as **the vector representations of samples of interested classes can be within the distribution of the vector representations of samples of interested classes**.

| | without OOD | with OOD |
|---|---|---|
| f1 score | 77.78% | 86.64% |

Table 3.14: Evaluation on the 5-class classifier on the combined set of the testing set and the set of the Calm emotion, which is not one of the 5 classes and therefore considered out-of-distribution (OOD). The metric is f1 score.

**Evaluation on the 5-class classifier With Out-Of-Distribution Samples**

We now evaluate our full solution by including the detection of samples that are out of the distribution of the training set. Tables 3.10 and 3.12 show that our classifier is able to achieve an accuracy score of 88.0% and an f1 score of 87.9% on the testing set, which contains not only clean samples but also samples that are environmentally distorted. We have achieved these scores using the testing set, which also has five classes. Despite that we have 4 interested emotion classes and 1 uninterested class, we must acknowledge the possibility that the classifier encounters samples that are not of the 5 classes. Since the classifier can only assign a class out of the five, its prediction is always wrong because the true label is not among the five.

We hypothesize that many samples not belonging to our recognized 5 classes are out of distribution of our training set. Therefore, if we intercept the out-of-distribution samples, we can significantly improve the performance of our classifier in a more realistic setting.

Table 3.14 shows the evaluation result of the 5-class solution on the combined set of the testing set and the set of the Calm emotion, which is not one of the 5 classes and therefore considered out-of-distribution(OOD) with and without the out-of-distribution detection technique. This combined set has 9995 samples. All the samples in the testing set are among those samples. The newly added samples are all of the Calm emotion. The Calm samples have been preprocessed in the same way as the samples in the synthetic dataset: we have clean Calm samples and environmentally reverberated Calm samples.

Without the out-of-distribution detection technique, we have achieved an f1 score of 77.78%. This is a significant drop (10.12%) from the f1 score achieved only on the testing set that has 5 targeted emotions (87.9%). The drop is to be expected, since the classifier can only assign the Calm samples to be happiness, anger, neutrality, sadness, and confounding emotions (fear and disgust), its prediction on a Calm samples is always wrong.

With the out-of-distribution detection technique, the f1 score improves to 86.64%, a significant increase (8.86%) compared to the experiment result in which no out-of-distribution detection technique is used and consequently all the out-of-distribution samples are wrongly assigned by the classifier. Note that the f1 score (86.64%) achieved on five targeted emotions and one previously unseen emotion class with the out of distribution technique is very close to the f1 score achieved on the five targeted emotions (87.9%) only. This indicates that, with the out-of-distribution

detection technique, the classifier can maintain its level of good performance on sets that have out-of-distribution samples, compared to its performance on sets that only have samples that are of one of the classifier's targeted classes.

**Real Time Computation**

Our solution is currently part of a home Patient-Caregiver monitoring, modeling, and interactive recommendation system for caregivers of dementia patients (Gao, Ma, et al., 2020), titled the Patient-Caregiver Relationship (PCR) system. One of the objectives of this system is to detect the onset of anger of the caregiver using only the acoustical modality (which our 5-class solution is a part of) and immediately notify the recommendation system which will recommend mindfulness/relaxation techniques intelligently. In the PCR system, we use an external microphone and an off-the-shelf laptop to detect emotions with our 5-class solution with OOD.

Real-time applications such as PCR on require an emotion detection algorithm to notify the caregiver promptly when their difficult emotions are expressed in their own speech. The notification would not be helpful to the caregiver if the notification is sent too late. Since the PCR problem is time-sensitive, it is imperative that the emotion classifier is capable of real-time computation.

The obvious computation architecture to choose is a standard architecture for smart home speakers such as Google Home speaker. However, smart home speakers do not perform the classification themselves; they only serve as acoustic sensors that stream data. The actual classification is handled using cloud computing: the acoustic signals are sent to a powerful cloud computing server from which the smart home speakers receive classification results. In other words, even in the case of smart home speakers, the acoustic signals are also processed by a powerful computing architecture. By using a standard PC architecture, we demonstrate that an off-the-shelf PC is able to run our CNN in real-time. There are also small form factor processors such as Toshiba's dynaEdge DE-100 (*dynaEdge DE-100* n.d.) that have the same capabilities as a laptop, but without a screen and our solution can execute on one of these when we want the deployment equipment to take up less space than a laptop.

The average running time of the components of our 5-class CNN on 184 samples of 5-second duration on a Intel(R) Core(TM) i5-9300H CPU at 2.40 GHz is 2.62 seconds. The average time consists of the time required to perform feature extraction, the time required to perform out-of-distribution detection, and the time to perform classification. If the caregiver's speech starts to become angry or sad, it will take 5-seconds for their speech to be recorded, and 2.62 second for our 5-class CNN to generate the classification result. Within the amount of time required to compute the emotion of the caregiver based on their speech, the caregiver will likely still be under the influence of the emotion; thus a notification to them is more likely to help than a notification that is sent to them long after they are no longer under the influence of the emotion. In other words, our 5-class CNN is capable of real-time computation, which is a necessary condition that an emotion detection algorithm

must meet in order to be used in applications like PCR.

## 3.6 Conclusion

Emotional health is a crucial part of one's well-being. The rapid development of machine learning in the field of acoustic signal processing has resulted in a surge of interest in detecting emotions from speech. We have created a combined convolutional neural network (CNN) and out of data distribution (OOD) solution that is robust to environmental distortions such as reverberation, noise, distance, and handles emotions that are not the targeted emotions through a class we call confounding emotions. To test our solution we created synthetic datasets that combined five standard datasets and enhanced them with de-amplification, home noises, reverberation, and a real-world padding scheme. Our solution outperforms a state of art baseline and achieves high accuracy in the presence of environmental distortions and confounding emotions.

*Chapter 4*

# DETECT CONFLICT IN REALISTIC HOME ENVIRONMENTS

Chapter 4 deals with realism that arises in a specific application: detecting verbal conflict between two people. The realism comes with the fact that the model we develop for conflict detection is deployed in people's homes. Multiple things stemming from realism can take place that hinders the model's performance. In particular, we deal with one type of realism - the first type, task-specific, that arises from the interaction between the deep learning model and the complex environment in which it is deployed: deammplification, reverberation, and noise-contamination of audio signals that occur in this environment.

In this Chapter, we develop a new (unsupervised) domain adaptation approach. Domain adaptation is a way to deal with the fact that the models, trained on the source domain with a certain distribution, are used on the target domain with a different distribution. Recall that we have listed four realisms, three of which have to do with the development stage of the DL models, and domain adaptation is effective at dealing with these three realisms, because these realisms are the result of the fact that the training samples (clean, undistorted) have a different distribution than the samples collected in the CPS where they are environmentally distorted.

## 4.1 Introduction

Domain Adaptation (DA) has drawn a lot of interest (S. Zhao, Qiu, and Y. He, 2021; Tzeng, Hoffman, N. Zhang, et al., 2014; L. Zhang et al., 2019) because it deals with the problem that arises within a core assumption of machine learning: machine learning assumes that the testing samples are from the domain of the training samples. This assumption often results in the fact that the machine learning model's testing performance is significantly worse than its validation performance when the training and testing samples are from different distributions or domains. Unsupervised Domain Adaptation (UDA) (Wei Wang et al., 2020; Tzeng, Hoffman, Saenko, et al., 2017) is a popular sub-field of DA because it allows the target domain to be unlabeled, which is more appropriate for real-life application as a lot of samples collected from real environments are not labeled.

Among the methods developed for UDA, adversarial-based methods are very popular. There are two types of adversarial-based methods: generative, and non-generative. Generative methods (M. Xu et al., 2020; Hoffman et al., 2018), inspired by GAN (Goodfellow, Pouget-Abadie, et al., 2020), aim at generating samples that aid in the task of (unsupervised) DA. For example, Hoffman et al. (Hoffman et al., 2018) try to adapt the source samples in the style of the target domain. The resulting adapted samples can be used to train a classifier using the labels of these adapted source samples to classify the samples in the target domain. Non-generative methods (Tzeng, Hoffman, Saenko, et al., 2017; Ganin, Ustinova, et al., 2016) aims at

domain confusion. It usually involves one or more generators/encoders, a domain discriminator, and a category classifier (Tzeng, Hoffman, Saenko, et al., 2017) that does the final classification of samples. The generator/encoder and the discriminator engage in a mini-max game in which the generator/encoder tries to deceive the domain discriminator, masking the true origin (which can be the source or target) of an incoming sample.

in this thesis, we study non-generative adversarial methods for UDA and uncover a challenge still existing in the field of UDA: We observe that, while these methods attempt to maximize domain confusion via adversarial training, their effort to achieve domain confusion is implicit rather than explicit. There still exists room for improvement on the task of domain confusion if domain confusion can be attempted to be achieved both implicitly via adversarial training and explicitly. To further enforce domain confusion (explicitly) and address the challenge, we introduce a novel variation of the Mahalanobis distance loss. The original Mahalanobis distance (K. Lee et al., 2018) measures how one sample deviates from a distribution. The Mahalanobis distance loss is a loss function used to train the encoder/generator, which aims at making the encoder/generator achieve the minimization of the distribution-wise distance between the source samples and the encoded target samples, or vice versa.

**There are two novelties in our Mahalanobis distance loss function.** First, although the idea of Mahalanobis distance loss has been defined (Wen et al., 2022), it is defined by taking the true values and predicted values as input. In other words, the previously defined Mahalanobis distance loss (Wen et al., 2022) minimizes the distance between a predicted value and the distribution of the set of true values. Our Mahalanobis distance loss, on the other hand, minimizes the distance between two distributions (the source domain distribution and the masked/encoded target domain distribution) instead of one value and one distribution, as our goal is to make the masked/encoded target domain, not an individual sample, closer to the distribution of the source domain, so that domain confusion is achieved. Second, to the best of our knowledge, we are the first to apply the Mahalanobis distance in the field of (unsupervised) non-generative adversarial domain adaption to achieve domain confusion.

We also investigate if it is possible to improve the performance of UDA tasks even further. We hypothesize that two, instead of one, category classifiers are needed. One is trained on the source samples and their (true) labels. The other is trained on the target samples and their (pseudo) labels. Then, we use an out-of-distribution (OOD) detection subroutine to determine if an encoded sample should be classified by the source category classifier or the target category classifier. The out-of-distribution is facilitated once again via the original Mahalanobis distance, as we have found studies that compare various OOD detection approaches' efficacy, and the Mahalanobis distance wins. For more details, please see Section 2.5.

In addition to the Mahalanobis distance loss and the OOD detection subroutine, we use the architecture of ADDA (Tzeng, Hoffman, Saenko, et al., 2017) in which

the source encoder (not a generator) and the source category classifier are trained end-to-end with source samples and their true labels. A target encoder (a generator) and a domain discriminator engage in a mini-max game whereas the target encoder tries to mask the incoming target samples as source-passing to fool the domain discriminator, thus achieving domain confusion. The encoded target samples are sent to the source category classifier for classification. Extensive evaluations show the superiority of the improved ADDA (we call it *E-ADDA* or the Enforced ADDA), Mahalanobis distance loss-enhanced, OOD detection subroutine-enhanced, over the vanilla ADDA and various state-of-the-art algorithms, achieving the new state-of-the-art performance on popular UDA benchmarks such as Office-31 and Office-Home.

The contributions of this thesis are:

- We identify the room for improvement of existing non-generative methods for (unsupervised) domain adaptation because they solely rely on the adversarial training to achieve domain confusion, which is implicitly achieved.
- We introduce a new loss function that minimizes the distribution-wise distance between the source distribution and the masked/encoded target distribution to further enforce domain confusion that is experimentally superior to adversarial training alone.
- Our solution, E-ADDA, outperforms various state-of-the-art domain adaptation/transfer learning algorithms on the acoustic modality in the field of domain-adapting/transfer-learning from angry voices to speeches of verbal conflict by up to 29.8% improvement in f1 scores.
- To further demonstrate the generalizability of E-ADDA, we evaluate it against various state-of-the-art domain adaptation algorithms in the modality of computer vision. E-ADDA outperforms the state-of-the-art algorithms by up to 17.9% improvement in accuracy scores on popular UDA benchmarks such as Office-31 and Office-Home.

## 4.2 Enforced Adversarial Discriminative Domain Adaptation (E-ADDA)

### Settings of Unsupervised Domain Adaptation

In UDA, the source samples and their labels are available. The source data is represented as $\boldsymbol{X}_s = \{(x_s^i, y_s^i)\}_{i=1}^{N_s}$. Only the samples of the target domain are available; their labels are not available. The target data is represented as $\boldsymbol{X}_t = \{(x_t^i)\}_{i=1}^{N_t}$. $N_s$ and $N_t$ represent the sizes of the sets of the source and target domains, respectively.

### Adversarial Training with the Mahalanobis Distance Loss

Because E-ADDA is based on ADDA (Tzeng, Hoffman, Saenko, et al., 2017), we briefly recap the architecture of ADDA. In ADDA, there exist four neural networks:

the source encoder/generator $E_s$, the source category classifier $F_s$, the target encoder/generator $E_t$, and the domain label discriminator $D$. $E_s$ and $F_s$ are trained end-to-end using the true labels of the source samples. Then, with $E_s$ as an input, $E_t$ and $D$ engage in a mini-max game in which $E_t$ tries to mask the target samples to appear as if they are (source samples that are) encoded by $E_s$, and $D$ tries to spot its trick and recover the true origin (source or target) of an encoded sample. Thus, domain confusion is achieved as $E_t$ is able to mask the target data to appear source-like, and $F_s$ is then able to classify them with satisfactory performance.

In E-ADDA, we keep the four neural networks and the framework of ADDA remains unchanged. The only thing that is added is the Mahalanobis distance loss function to train $E_t$ to further enhance/enforce domain confusion. In the following equations, we define the loss function for $E_s$, $E_t$, $F_s$, and $D$.

The source category classifier $F_s$'s loss is the standard supervised loss. It is noted that $E_s$ and $F_s$ are trained jointly, which is achieved by Equation 4.1.

$$\begin{aligned}\min_{E_s,F_s} \mathcal{L}_{F_s}(\boldsymbol{X}_s, Y_s) = \mathbb{E}_{(x_s,y_s)\sim(\boldsymbol{X}_s,Y_s)} \\ - \sum_{k=1}^{K} \log F_s(E_s(x_s))\mathbb{1}(k, y_s)\end{aligned} \tag{4.1}$$

The domain label discriminator $D$ is also trained using the standard supervised loss using $E_s$ and $E_t$ as well as the domain information of samples in the source and target domains, as in Equation 4.2.

$$\begin{aligned}\mathcal{L}_D(\boldsymbol{X}_s, \boldsymbol{X}_t, E_s, E_t) = &- \mathbb{E}_{x_s\sim\boldsymbol{X}_s}[\log D(E_s(x_s))] \\ &- \mathbb{E}_{x_t\sim\boldsymbol{X}_t}[\log(1 - D(E_t(x_t)))]\end{aligned} \tag{4.2}$$

In this paragraph, we describe the adversarial training loss and the Mahalanobis distance loss for $E_t$, as in Equation 4.3.

$$\begin{aligned}\mathcal{L}_{E_t}(\mathcal{X}_s, \mathcal{X}_t, D) = &- \sum_{d\in\{s,t\}} \mathbb{E}_{x_d\sim\boldsymbol{X}_d}[\frac{1}{2}\log D(E_d(x_d))] \\ &+ [\frac{1}{2}\log(1 - D(E_d(x_d)))] + \theta_M\mathcal{L}_M\end{aligned} \tag{4.3}$$

How do we define $\mathcal{L}_M$? To define it, we consider the domain confusion task to be achieved. We want to train $E_t$ and $D$ adversarially so that $E_t$ encodes the target samples such that $D$ thinks these encoded samples were source samples encoded by $E_s$. Therefore, to further maximize domain confusion, we define $\mathcal{L}_M$ as Equation 4.4. $\hat{\mu}_s$ is the empirical mean of all source samples encoded by $E_s$ defined as Equation 4.5, and $\hat{\Sigma}$ is the empirical covariance defined as Equation 4.6.

$$\mathcal{L}_M = \sum (E_t(x_t) - \hat{\mu}_s)^\top \hat{\Sigma}_s^{-1} (E_t(x_t) - \hat{\mu}_t) \tag{4.4}$$

$$\hat{\mu}_s = \frac{1}{N_s} \sum E_s(x_s), x_s \in \boldsymbol{X}_s \tag{4.5}$$

$$\Sigma_s = \frac{1}{N_s} \sum (E_s(x_s) - \hat{\mu}_s)(E_s(x_s) - \hat{\mu}_s)^\top \tag{4.6}$$

**Mahalanobis Distance Based Out-of-Distribution Detection Subroutines**
To safeguard the scenario in which domain confusion still somehow fails despite our best effort with the Mahalanobis loss, we add two OOD detection subroutines to catch a (target) sample if the adversarial training fails to allow $E_t$ to mask it as if it was a source sample encoded by $E_s$. If this happens, we send this target sample to the target category classifier, instead of the source category classifier, for final classification. The target classifier is trained using the target training samples and their pseudo-labels.

To determine if a sample $x$ is still within the distribution of the target domain or if it is successfully encoded to look like its origin is the source domain, we measure the Mahalanobis distance between $E_s(x)$ and the set of $E_s(x_s)$, $\forall x_s \in \boldsymbol{X}_s$, as well as the Mahalanobis distance between $E_t(x)$ and the set of $E_t(x_t)$, $\forall x_t \in \boldsymbol{X}_t$. To get the parameters that the calculation of the Mahalanobis distance requires, we need the empirical mean and the empirical covariance of the distribution. For the source distribution, we have already defined the source empirical mean $\hat{\mu}_s$ in Equation 4.5 and the source empirical covariance $\hat{\Sigma}_s$ in Equation 4.6. Similarly, we define the target empirical mean $\hat{\mu}_t$ and the target empirical covariance $\hat{\Sigma}_t$ in Equations 4.7 and 4.8.

$$\hat{\mu}_t = \frac{1}{N_t} \sum E_t(x_t), x_t \in \boldsymbol{X}_t \tag{4.7}$$

$$\Sigma_t = \frac{1}{N_t} \sum (E_t(x_t) - \hat{\mu}_t)(E_t(x_t) - \hat{\mu}_t)^\top \tag{4.8}$$

The Mahalanobis distance between $x$ and a distribution is defined using an empirical mean $\hat{\mu}$ and empirical covariance $\hat{\Sigma}$ that describe the distribution, as defined in Equation 4.9, in which $E$ can be either $E_s$ or $E_t$, depending on if this is the source or the target distribution that we are talking about.

$$\hat{M}(x) = (E(x) - \hat{\mu})^\top \hat{\Sigma}^{-1} (E(x) - \hat{\mu}) \tag{4.9}$$

With the aforementioned information, we create two OOD subroutines. The first one checks the Mahalanobis distance between $E_s(x)$ and the distribution of $E_s(x_s)$, $\forall x_s \in \boldsymbol{X}_s$, The second one checks the Mahalanobis distance between $E_t(x)$ and the distribution of $E_t(x_t)$, $\forall x_t \in \boldsymbol{X}_t$. If the Mahalanobis distance score between $E_s(x)$ and the distribution of $E_s(x_s)$ is smaller than an empirically determined $\lambda_s$, $\forall x_s \in \boldsymbol{X}_s$

then it is considered within the distribution of the source domain. If the Mahalanobis distance score between $E_t(x)$ and the distribution of $E_t(x_t), \forall x_t \in \boldsymbol{X}_t$ is smaller than an empirically determined $\lambda_t$, then it is considered within the distribution of the target domain.

## 4.3 Evaluation

### Overview

We first evaluate E-ADDA on a domain adaptation task on an acoustic modality: we domain-adapt from a dataset consisting of emotional utterances to a dataset that contains audio samples of speech in which, sometimes, the speakers are in a verbal conflict (we map the anger emotion to conflict and other emotions to non-conflict). Then, to demonstrate that E-ADDA not only works on domain-adapting from the domain of emotions to the domain of conflict speech, but also in other fields such as computer vision, we compare E-ADDA against various other state-of-the-art deep domain adaptation algorithms on standard datasets and tasks of UDA in the field of computer vision such as Office-31 and Office-Home.

### The Domain Adaptation Task on the Acoustic Modality

### The Source Dataset

In the following paragraphs we describe our source dataset in the domain adaptation task on the acoustic modality. The EMOTION dataset **contains the all samples from the following 5 public datasets**: RAVDESS (Livingstone and Russo, 2018), CREMA-D (Cao et al., 2014), EMA (S. Lee et al., 2005), TESS (Dupuis and Pichora-Fuller, 2010), and SAVEE (Haq and Jackson, 2010). In addition we extend these 5 datatsets with samples that are distorted to account for environmental conditions by artificially adding environmental distortions into the clean samples from the original five datasets. The reverberation effect is described by the combination of the decay factor, diffusion, and wet/dry ratio. EMOTION consists of training and testing sets. In the training set, there are 8,816 samples in the anger class, 8,786 samples in the happiness class, 7,742 samples in the neutral class, 8,811 samples in the sadness class, and 5,761 samples in the disgust/fear class. In the testing set, there are 1,942 samples in the anger class, 1,966 samples in the happiness class, 1,696 samples in the neutral class, 1,947 samples in the sadness class, and 1,292 samples in the fear/disgust class.

### The Target Dataset

Our target dataset, CONFLICT, is the dataset where we want to apply E-ADDA solution so that the source classifier trained on EMOTION can be re-purposed. It is collected from real home environments in which 19 couples talk (collected with approved IRB) about topics that they previously disagree on and have their conversation recorded. In total, there are 3027 training samples and 1009 testing samples.

**Comparison with State-of-the-Art Baselines**

In this experiment, shown in Table 4.1, we compare E-ADDA with the scenario in which no domain adaptation or transfer learning is used (No DA/TL) and three baselines: direct training (directly training the model on the data from the target dataset), two selected state-of-the-art approaches, ADDA and ADDA with CORAL loss. Our solution, E-ADDA, outperforms various state-of-the-art domain adaptation/transfer learning algorithms on the acoustic modality in the field of domain-adapting/transfer-learning from angry voices to speeches of verbal conflict by up to 29.8% improvement in f1 scores.

Each of the audio samples on which we test the situation in which no DA/TL is used, the three baselines, and E-ADDA contains environmental distortions and/or overlapped speech. The usage of CORAL loss (in Deep CORAL) and ADDA has garnered a lot of interest in the field of DA/TL; in this paragraph, we briefly describe these two approaches. CORAL loss proposes that the domain shift can be mitigated by using linear transformations to align the second-order statistics of the two domains. ADDA proposes to encode the target samples to the feature space of the source and have a domain discriminator that tries to distinguish encoded target samples from source samples. ADDA and ADDA with CORAL loss achieve in f1 scores of 38.29% and 63.28% respectively.

| | Env. Distortion | F1 |
|---|---|---|
| ADDA | ✓ | 38.29% |
| ADDA + CORAL | ✓ | 63.28% |
| No TL/DA | ✓ | 77.25% |
| Trained on target | ✓ | 85.82% |
| E-ADDA | ✓ | **93.10%** |

Table 4.1: The performance of four baselines against E-ADDA on data that has overlapped speech and environmental distortions.

As shown in Table 4.1, No TL/DA's performance is 77.25%, a value that is higher than the state-of-the-art solutions ADDA and ADDA with CORAL loss. No TL/DA stands for that we directly apply the source classifier on the target samples. In the case of domain-adapting from a classifier of emotions to conflict detection, no TL/DA suggests that we directly apply the mood classifier on the conflict samples and the performance is calculated based on that anger denotes conflict while other emotions denote no conflict. ADDA performance was 38.29%, which is lower than the no TL/DA by 38.96%. ADDA with CORAL loss achieved significantly higher performance, 63.28%. Since with have more than 7000 samples in the target dataset, we also directly train a classifier using only the target sample and yield an f1 score of 85.82%, which is higher than ADDA by 47.53% and ADDA combined with CORAL loss by 22.54%. Still, it is 7.28% lower than E-ADDA's performance. Our E-ADDA results in an improvement over ADDA with CORAL loss by 29.82%.

It is worth noting that the source and target domains in this setting are distribution-wise distant because they are not even from the same class (the source domain is about people's emotions and the target domain is about verbal conflict). Therefore, the task should be more appropriately called unsupervised transfer learning instead of unsupervised domain adaptation. We present this task as part of our evaluation to test if E-ADDA can really enforce domain confusion, safeguard catching samples on which domain confusion fails, and send these samples to their respective category classifiers. The ADDA architecture only yields an f1 score of 38.2%, suggesting that ADDA's basic mechanism of domain confusion fails. However, with the Mahalanobis distance loss and the OOD detection subroutine on top of the same architecture, E-ADDA is able to achieve an f1 score of 93.1%. This indicates that the Mahalanobis distance loss is very effective at enforcing, on top of the adversarial training, domain confusion. In addition, it suggests the necessity of the OOD detection subroutine to send samples on which domain confusion fails to their respective category classifiers.

| Algorithm | A→W | A→D | D→W | D→A | W→A | W→D | Avg |
|---|---|---|---|---|---|---|---|
| ResNet-50 (K. He et al., 2016) | 68.4% | 68.9% | 96.7% | 62.5% | 60.7% | 99.3% | 76.1% |
| DANN (Ganin and Lempitsky, 2015) | 82.0% | 79.7% | 96.9% | 68.2% | 67.4% | 99.1% | 82.2% |
| MSTN (Xie et al., 2018) | 91.3% | 90.4% | 98.9% | 72.7% | 65.6% | **100%** | 86.5% |
| CDAN+E (M. Long et al., 2018) | 94.1% | 92.9% | 98.6% | 71.0% | 69.3% | **100%** | 87.7% |
| DMRL (Yuan Wu, Inkpen, and El-Roby, 2020) | 90.8% | 93.4% | 99.0% | 73.0% | 71.2% | **100%** | 87.9% |
| SymNets (Y. Zhang et al., 2019) | 90.8% | 93.9% | 98.8% | 74.6% | 72.5% | **100%** | 88.4% |
| GSDA (L. Hu et al., 2020) | 95.7% | 94.8% | 99.1% | 73.5% | 74.9% | **100%** | 89.7% |
| CAN (Kang et al., 2019) | 94.5% | 95.0% | 99.1% | 78.0% | 77.0% | 99.8% | 90.6% |
| SRDC (Tang, K. Chen, and K. Jia, 2020) | 95.7% | 95.8% | 99.2% | 76.7% | 77.1% | **100%** | 90.8% |
| RSDA-MSTN (Gu, J. Sun, and Z. Xu, 2020) | **96.1%** | 95.8% | 99.3% | 77.4% | 78.9% | **100%** | 91.1% |
| E-ADDA | 95.4% | **96.2%** | **100%** | **95.3%** | **90.9%** | **100%** | **95.3%** |

Table 4.2: The results on the domain adaptation tasks among the three domains in the dataset Office-31. The metric is accuracy.

| Algorithm | Pr→Ar | Ar→Pr | Cl→Ar | Ar→Cl | Rw→Ar | Ar→Rw | Pr→Cl | Cl→Pr | Rw → Pr | Pr→Rw | Rw→Cl | Cl→Rw | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet-50 (K. He et al., 2016) | 38.5% | 50% | 37.4% | 34.9% | 53.9% | 58% | 31.2% | 41.9% | 59.9% | 60.4% | 41.2% | 46.2% | 46.1% |
| DANN (Ganin and Lempitsky, 2015) | 41.6% | 59.3% | 47.0% | 45.6% | 63.2% | 70.1% | 43.7% | 58.5% | 76.8% | 68.5% | 51.8% | 60.9% | 57.6% |
| CDAN (M. Long et al., 2018) | 55.6% | 69.3% | 54.4% | 49.0% | 68.4% | 74.5% | 48.3% | 66.0% | 80.5% | 75.9% | 55.4% | 68.4% | 63.8% |
| MSTN (Xie et al., 2018) | 61.4% | 70.3% | 60.4% | 49.8% | 70.9% | 76.3% | 48.9% | 68.5% | 81.1% | 75.7% | 55.0% | 69.6% | 65.7% |
| SymNets (Y. Zhang et al., 2019) | 63.6% | 72.9% | 64.2% | 47.7% | 73.8% | 78.5% | 47.6% | 71.3% | 82.6% | 79.4% | 50.8% | 74.2% | 67.2% |
| GSDA (L. Hu et al., 2020) | 65.0% | 76.1% | 65.4% | 61.3% | 72.2% | 79.4% | 53.2% | 73.3% | 83.1% | 80.0% | 60.6% | 74.3% | 70.3% |
| GVB-GD (Cui et al., 2020) | 65.2% | 74.7% | 64.6% | 57.0% | 74.6% | 79.8% | 55.1% | 74.1% | 84.3% | 81.0% | 59.7% | 74.6% | 70.4% |
| RSDA-MSTN (Gu, J. Sun, and Z. Xu, 2020) | 67.9% | 77.7% | 66.4% | 53.2% | 75.8% | **81.3%** | 53.0% | 74.0% | 85.4% | **82.0%** | 57.8% | 76.5% | 70.9% |
| SRDC (Tang, K. Chen, and K. Jia, 2020) | **68.7%** | 76.3% | **69.5%** | 52.3% | **76.3%** | 81.0% | 53.8% | 76.2% | 85.0% | 81.7% | 57.1% | **78.0%** | 71.3% |
| E-ADDA | 66.8% | **78.6%** | 59.6% | **61.0%** | 67.7% | 79.7% | **64.9%** | **79.8%** | **85.8%** | 79.2% | **64.9%** | 70.4% | **71.5%** |

Table 4.3: The results on the domain adaptation tasks among the four domains in the dataset Office-Home. The metric is accuracy.

**Domain Adaptation Tasks on Images**

In this section, we discuss the performance of E-ADDA against state-of-the-art baselines on popular benchmarks for UDA such as Office-31 and Office-Home. Then, to show that E-ADDA also achieves state-of-the-art performance on simpler domain adaptation tasks such as MNIST → USPS, SVHN → MNIST, as well as

| Algorithm | MNIST → USPS | SVHN → MNIST |
|---|---|---|
| Source only | 75.2% | 60.1% |
| Gradient Reversal (Ganin and Lempitsky, 2015) | 77.1% | 73.9% |
| Domain Confusion (Tzeng, Hoffman, Darrell, et al., 2015) | 79.1% | 68.1% |
| CoDAN (M.-Y. Liu and Tuzel, 2016) | 91.2% | did not converge |
| ADDA (Tzeng, Hoffman, Saenko, et al., 2017) | 89.4% | 76.0% |
| Associative (Haeusser et al., 2017) | 94.1% | 93.6% |
| DANN (Ganin, Ustinova, et al., 2016) | 60.8% | 76.3% |
| Deep Coral (B. Sun and Saenko, 2016) | 69.5% | 76.3% |
| VADA (Shu et al., 2018) | 90.6% | 92.6% |
| E-ADDA | **95.4%** | **95.4%** |

Table 4.4: We compare our technique, E-ADDA, with nine other state-of-the-art deep domain adaptation techniques on two tasks (the performance is measured in accuracy, per the evaluation standard of the computer vision community).

| Algorithm | STL-10 → CIFAR-10 |
|---|---|
| DRCN (Ghifary et al., 2016) | 58.6% |
| SE (French, Mackiewicz, and Fisher, 2017) | 64.2% |
| Source only | 63.6% |
| VADA (Shu et al., 2018) | 75.3% |
| Co-DA (Kumar et al., 2018) | 76.4% |
| DTA (Kumar et al., 2018) | 72.8% |
| ET | **86.1%** |

Table 4.5: We compare our technique, E-ADDA, with five other state-of-the-art deep domain adaptation techniques on the domain adaptation task to domain-adapt from STL-10 to CIFAR-10 (the performance is measured in accuracy, per the evaluation standard of the computer vision community).

CIFAR-10 → STL-10, we also compare E-ADDA's performance against state-of-the-art baselines on these UDA tasks.

**Office-31**

In Table 6.2, we compare our E-ADDA against ResNet-50 (K. He et al., 2016) and nine other state-of-the-art domain adaptation algorithms using the dataset Office-31. Office-31 contains three subdomains: Amazon (A), Webcam (W), and Dslr (D). Each domain contains 31 classes of everyday office objects such as rulers or projectors. There are 4,110 images in total in Office-31. Across the three domains, six domain adaptation tasks can be formed, as shown in Table 6.2. The performance of each algorithm is measured in the accuracy that is the percentage of samples that

are correctly classified by the algorithm out of all the samples in the testing set.

On the six domain adaptation tasks, we have achieved state-of-the-art performance on five of them, except for the task of A → W, where RSDA-MSTN (Gu, J. Sun, and Z. Xu, 2020) outperforms E-ADDA by 0.7%. RSDA-MSTN (Gu, J. Sun, and Z. Xu, 2020) proposes to redefine the feature space as a spherical feature space and create a spherical classifier and discriminator, creating a pseudo-label loss in this spherical feature space. However, it fails to deal with the situation in which the pseudo-labels are not very accurate and the pseudo-label loss is very large. E-ADDA does not have that problem.

It is worth noting that RSDA-MSTN is a non-generative adversarial algorithm whose superiority comes from the fact that the adversarial training is defined in the spherical feature space. As a non-generative adversarial method, RSDA-MSTN is a perfect candidate to compare E-ADDA against. On tasks with large domain shifts, such as W → A and D → A, we outperform RSDA-MSTN by 17.9% and 12%, a very large improvement. This suggests E-ADDA is better at achieving domain confusion on UDA tasks whose source and target domains are more distributionally distant while other algorithms that aim at domain confusion fail to achieve a performance that is as high.

**Office-Home**

In Table 6.3, we compare E-ADDA against ResNet-50 and eight other state-of-the-art domain adaptation algorithms on Office-Home. Office-Home has four subdomains: Product (Pr), Art (Ar), Clipart (Cl), and Real World (Rw). There are 15,500 images in Office-Home, each of which is of a typical object that can be found in an office or home, such as flowers. Twelve domain adaptation algorithms can be formed based on the four subdomains. The performance of each algorithm is measured in the accuracy that is the percentage of samples that are correctly classified by the algorithm out of all the samples in the testing set.

Out of the twelve domain adaptation tasks, we outperform the next best-performing algorithm on six of them. On the task of Ar → Rw, the state-of-the-art, RSDA-MSTN, outperforms us by 1.6%. Again, RSDA fails to deal with the situation in which the pseudo-labels are not very accurate and the pseudo-label loss is very large. On the task Pr → Ar, the state-of-the-art, SRDC, outperforms us by 1.9%. SRDC proposes to alleviate the risk of damaging the intrinsic domain discrimination resulting from finding domain-aligned features. However, the proposition to minimize the KL divergence between the distribution of predictive labels and the distribution of auxiliary labels is a rather naive approach, as the authors fail to compare their algorithm with other measurements to minimize the Jensen–Shannon divergence.

Once again, we have observed that E-ADDA is better at achieving domain confusion than the other non-generative adversarial method, RSDA-MSTN, when domain shifts are large. For example, on the task Ar → Cl, we outperform the second-best-performing algorithm RSDA-MSTN by 7.7%. This suggests that, when domain

shifts are large, or when domain confusion is harder, E-ADDA can still achieve good domain confusion results, while the state-of-the-art non-generative methods cannot.

#### MNIST → USPS and SVHN → MNIST

MNIST, USPS, and SVHN datasets all have ten classes of hand-written digits. The first task, which is to domain-adapt from MNIST to USPS, is considered easier while the second task, which is to domain-adapt from SVHN to MNIST, is considered more challenging. This claim is supported by the observation that, in Table 4.4, five out of the state-of-the-art domain adaptation algorithms and the baseline of directly using source classifier on the target dataset results in lower performance of the second task compared to the first task. E-ADDA achieve an accuracy of 95.41% on the first task and 95.43% on the second. On the first task, it outperforms the best-performing state-of-the-art baseline, Associative, by 1.31%. On the second task, it outperforms the best-performing state-of-the-art baseline by 2.83%. We present our evaluation results on MNIST → USPS and SVHN → MNIST to demonstrate that E-ADDA can also achieve state-of-the-art performance on simpler UDA tasks.

#### STL-10 → CIFAR-10

In this section, we further investigate if the E-ADDA can outperform state-of-the-art baselines on a more complicated vision task that is not digits. Therefore, we transfer learn from STL-10 to CIFAR-10. Both CIFAR-10 and STL-10 are image datasets that contain 10 classes. We outperform all the other five state-of-the-art deep domain adaptation baselines and outperform the second best-performing algorithm, Co-DA, by 9.7%. The source classifier that we use to train is ResNet-50 (K. He et al., 2016). We yield the highest performance of an accuracy score of 86.1% after we inject the E-ADDA Cell after the fifth layer.

Note that, compared to the previous section, we choose a different set of baselines to fully evaluate our solution, the E-ADDA, against as many baselines as possible. The task to domain-adapt from STL-10 to CIFAR-10, which is more complex than domain-adapt among MNIST, SVHN, and USPS, as these three datasets contain only digits. On the contrary, CIFAR-10 and STL-10 contains images such as the automobile and dog classes. Again, we present our evaluation results on STL-10 → CIFAR-10 to demonstrate that E-ADDA can also achieve state-of-the-art performance on simpler UDA tasks.

### 4.4 Conclusion

We have discovered that there exists room for improvement on the existing non-generative adversarial UDA algorithms that attempt to achieve domain confusion. The challenge lies in the observation that these algorithms do not explicitly minimize the distance between the distribution of the masked/encoded target samples and the source samples; instead, they let the adversarial training achieve domain confusion rather implicitly. To address this challenge, we propose E-ADDA that

uses a novel variation of the Mahalanobis distance loss to minimize the distribution-wise distance between the masked/encoded target domain samples and the source domain samples. Then, the OOD subroutine further eliminates samples on which the domain confusion is unsuccessful. We have performed extensive evaluations on E-ADDA on two modalities: the acoustic modality and the computer vision modality. On both modalities, we outperform the state-of-the-art algorithms and achieve new state-of-the-art performance.

*Chapter 5*

# INCORPORATE PROPERTIES INTO MODELS THAT ARE MODELING CYBER PHYSICAL SYSTEMS

State-of-the-art deep learning models (Tzeng, Hoffman, Saenko, et al., 2017; Tzeng, Hoffman, N. Zhang, et al., 2014; Ganin, Ustinova, et al., 2016; Gretton et al., 2012) are not robust because they just rely on data. The problem of under-specification leads to the brittleness of the neural networks such that they are sensitive to changes imperceptible to humans. Physics-guided neural networks (J. Wang et al., 2020; W. Li, Bazant, and J. Zhu, 2021; X. Hu et al., 2020; Daw, Thomas, et al., 2020) is one approach to deal with under-specification - by incorporating physics rules into the loss function of a neural network during training. The trained neural network is less likely to violate the incorporated physics rules and therefore the robustness of the model is guaranteed to a certain degree. More precisely, this approach add a separate term to the original loss function of the neural network, as stated in Equation 5.1. In Equation 5.1, $\mathcal{L}_{NN}$ is the loss that traditional neural networks use, such as MSE loss, and $\mathcal{L}_{physics}$ is the loss regarding the physics rule. $\alpha$ and $\beta$ are scalar coefficients. The second term $\beta\mathcal{L}_{physics}(y, \hat{y})$ indicates the loss regarding the predicted value $\hat{y}$ by the neural network and the value that it supposed to be based on the physics rule. An open challenge regarding this method is to find the the appropriate values for $\alpha$ and $\beta$ (Von Rueden et al., 2019). However, some constraints, which we call properties, are too complex and there is no single physical equation for them. We define a property as a characteristic or constraint that is cannot be incorporated into deep learning models during the training time without specifying them explicitly.

$$\mathcal{L} = \alpha\mathcal{L}_{NN}(y, \hat{y}) + \beta\mathcal{L}_{physics}(y, \hat{y}) \tag{5.1}$$

In Chapter 5, we tackle the problem of traffic speed prediction in smart cities. Models on traffic speed prediction tend to be brittle and not robust since they are not trained with properties being enforced into the training. A straightforward idea to incorporate properties in the training of such models is through loss functions, but traffic flow is too complex for physics-informed loss functions to describe. Existing works (Yaguang Li et al., 2017; Z. Wu, Pan, G. Long, Jiang, and C. Zhang, 2019; Z. Wu, Pan, G. Long, Jiang, Chang, et al., 2020) propose to exploit the graph-like structure of a city, with sensors that read traffic speed being the nodes and the distances between the nodes as the weights of the edges, which is a way to incorporate properties into the training of a model by using the Graph Neural Network (GNN). However, we discover that their results could be further improved if we add another kind of property, on top of the properties being enforced by GNN. This kind of property is called the points of interest (POIs), and we shall elaborate on

what POIs are and how to incorporate them during the training of the deep learning model for traffic speed prediction in the Sections in this Chapter.

## 5.1 Introduction

Spatiotemporal forecasting has seen a surge of interest in dynamic learning systems. A wide range of applications benefits from spatiotemporal forecasting, such as smart transportation systems and climate modeling. in this thesis, we study the task of forecasting the traffic speed on road networks, which is an important part of smart transportation systems, with information on the historic traffic speeds detected by the sensors and the layout of the road networks.

Traffic speed forecasting is a challenging task because of the spatial dependencies of the road networks and the temporal dependencies of the traffic conditions that are ever-changing. Temporal dynamic is observed in the context of traffic speed forecasting, such as the repeated occurrences of rush hours that affect the traffic speed. Spatial dynamic is observed in the context of traffic forecasting since traffic speed detected by a sensor is going to have an effect on the traffic speed at nearby places where other (nearby) sensors are placed. It is worth noting that the spatiotemporal dependencies of traffic speeds are not Euclidean, because it is not necessarily true that two roads that are spatially close to each other have similar traffic speeds (Yaguang Li et al., 2017).

In this paragraph, we briefly discuss the existing works on traffic (speed) prediction, a field that has existed for decades. The works can be divided into two categories: world-knowledge-driven solutions and data-driven solutions. The former integrate world knowledge such as queuing theory (Cascetta, 2013) in modeling the dynamics of traffic flow. However, a problem with these approaches is that they depend on stationary hypotheses while the traffic is ever-changing and non-stationary. Data-driven solutions (W. Liu et al., 2011; Lippi, Bertini, and Frasconi, 2013; Bai et al., 2021) fail to take advantage of theories on modeling dynamic systems such as the diffusion/infusion theory. Consequently, a natural outcome of taking the best of both worlds is a data-driven solution that uses world knowledge, such as DCRNN (Yaguang Li et al., 2017), in which the authors train a diffusion convolution recurrent neural network for the task of traffic prediction. DCRNN yields significant improvement over the state-of-the-art baselines, but we hypothesize that its results can be further improved if we incorporate the attention mechanism to capture the global spatiotemporal dynamics. In addition, we also propose to add more explicit world knowledge that affects traffic to guide the attention mechanism, and the explicit world knowledge is represented by Points of Interest (POIs). in this thesis, we propose two types of POIs, one is binary and one is numerical. The binary POIs indicate if a node is located at a place that can significantly affect the traffic, such as a school. The numerical POIs are the averages of the traffic information at a particular node.

By building upon the architecture of DCRNN, which establishes that the road networks can be interpreted as a graph with nodes being sensors and edges being the

connections among sensors (with weights represented as the proximity between the sensors/nodes), we propose our algorithm, D-A3T-GCN. In addition to interpreting the dynamic of traffic flow as a process of infusion/diffusion and applying the diffusion convolution operation for traffic speed prediction, we add the attention mechanism to each step of the infusion/diffusion process to capture the global spatiotemporal dynamics, and, in addition to the attention mechanism, we add the POIs to guide the attention mechanism. Our solution is evaluated on two large-scale real-world datasets, METRA-LA and PEMS-BAY. We observe an improvement of up to 33% over the performances of the state-of-the-art baselines.

The contributions of the proposed D-A3T-GCN are:

- We propose the attention-enhanced diffusion convolution built on top of GCNs.
- We incorporate the Points of Interest (POIs) as an effective way to incorporate world knowledge to guide the attention mechanism (to pay more attention to the more “important" nodes during the message passing procedure of DCRNN).
- We evaluate our proposed solution on two large-scale real-world datasets, METRA-LA and PEMS-BAY, and obtains significant improvements over the state-of-the-art baselines.

## 5.2 D-A3T-GCN

In this Section, we discuss the setting of spatiotemporal traffic forecasting using attention-enhanced diffusion convolutional recurrent neural network, which is then enhanced by adding the world knowledge represented as the points of interest.

### The Problem of Traffic Forecasting

In our problem setting, the sensors, placed at different locations, detect the speed of vehicles. The entire sensor network can be seen as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, W)$. Each node is seen as a sensor, and $\mathcal{V}$ is the set of sensors/nodes; $\mathcal{E}$ is the set of edges whose weights are determined by the distance between the two nodes at its two endpoints; and $W = \mathbb{R}^{|V|\times|V|}$ is the adjacency matrix that includes the weights of the edges of nodes that are adjacent to each other. Here, following the tradition and threshold established by previous works such as Li et al. (Yaguang Li et al., 2017), we claim that two nodes are adjacent if the distance between them is smaller than a threshold which has been chosen by previous works (Yaguang Li et al., 2017). Let $N$ be the number of features that at time $t$ a node $\in \mathcal{V}$ reads from the traffic flow. At time $t$, a graph signal $\boldsymbol{X}^t \in \mathbb{R}^{|\mathcal{V}|\times N}$ can be extracted or retrieved. The traffic forecasting problem attempts to uncover a function $f(\cdot)$ that maps $T$ historical graph signals to $T'$ graph signals in the future whereas $\mathcal{G}$ is given. In other words, the problem of traffic forecasting can be formulated as Equation 5.2.

$$[\mathcal{G}; \boldsymbol{X}^{t-T+1}, ..., \boldsymbol{X}^t] \overset{f(\cdot)}{\rightarrow} [\boldsymbol{X}^{t+1}, ..., \boldsymbol{X}^{t+T'}] \tag{5.2}$$

**Diffusion Convolution**

Li et al. (Yaguang Li et al., 2017) have argued that the traffic flow at a particular node can be represented as the end result of traffic diffusing from it to its $K$-hop neighbors as well as the end result of traffic infusing from its $K$-hop neighbors to it. As a result, they invent the diffusion convolution operation to model traffic flow defined in Equation 5.3. The diffusion convolution process is repeated for all $n \in \{1, ..., N\}$ or all features. $*$ is the diffusion convolution symbol and $\theta$ is the parameters for the filter. $D_O^{-1}W$ is the transition matrix for the diffusion process whereas $D_I^{-1}W^\top$ is the transition matrix for the infusion (reverse of diffusion) process.

$$\boldsymbol{X}_{:,n*f_\theta} = \sum_{k=0}^{K-1} (\theta_{k,1}(D_O^{-1}W)^k + \theta_{k,2}(D_I^{-1}W^\top)^k)\boldsymbol{X}_{:,n} \tag{5.3}$$

**Soft Attention-enhanced Diffusion Convolution**

In order to talk about the attention-enhanced diffusion convolution, we need to talk about the soft attention mechanism itself. Suppose there is a time series represented as graph signals, $[\boldsymbol{X}^{t-T+1}, ..., \boldsymbol{X}^t]$, we now introduce the soft attention mechanism. In the beginning, the hidden states at times $t-T+1, ..., t$ are calculated using the convolutional neural networks (CNNs) or their variations, or recurrent neural network (RNNs) or their valuations, and the hidden states $h_{t-T+1}, ..., h_t$ are produced. After this, a scoring function is introduced to calculate the weights at the hidden states. Then, an attention function is introduced to calculate the context vectors capable of describing global information. The context vectors are also used to obtain the final outputs. In our design of the soft attention-enhanced diffusion convolution, we use a multilayer perceptron as the scoring function, per the standard established by Bai et al. (Bai et al., 2021).

In particular, the hidden state $h_i \in H = \{h_{t-T+1}, ..., h_t\}$ at the moment $i$ is treated as the input to calculate the weight of the particular hidden state, and the output is obtained via two layers (the 2-layered perceptron as the scoring function). The weight or attention at each hidden state is calculated using a softmax normalized index function. Per the definitions of Bai et al. (Bai et al., 2021), $w_1$ and $b_1$ represent the weight and bias of the first perceptron layer and $w_2$ and $b_2$ represent the weight and bias of the second perceptron layer.

$$e_i = w_2(w_1 H + b_1) + b_2 \tag{5.4}$$

$$\alpha_i = \frac{\exp(e_i)}{\sum_{j=1} \exp(e_j)} \tag{5.5}$$

And the context vector is calculated based on Equation 5.6.

$$C_t = \sum_{i=1} \alpha_i \times h_i \tag{5.6}$$

We have finished talking about calculating the soft attention. However, how do we incorporate it with the diffusion convolution operation? Recall that in the diffusion convolution in Equation 5.3, we diffuse $K$ steps. When $K = 0$, no diffusion or infusion happens. When $K = 1$, a node's information diffuses to and infuses from its 1-hop neighbors, etc. We would like to add the soft attention mechanism to the $K$-hop diffusion/infusion process to figure out *which diffusion/infusion step or steps should the model pays more attention to, in terms of each node and their K-hop neighbors*.

$$\boldsymbol{X}_{:,n*f_\theta} = \sum_{k=0}^{K-1} \alpha_k (\theta_{k,1}(D_O^{-1}W)^k + \theta_{k,2}(D_I^{-1}W^\top)^k)\boldsymbol{X}_{:,n} \tag{5.7}$$

Why do we want to put a soft attention mechanism to find out which diffusion/infusion step(s) is/are most important and therefore should pay attention to? Our intuition comes from realistic scenarios. Suppose a sensor is placed at a train station and $k$-hops away from it, another sensor is placed at a school. We would like to know the traffic speed at the train station detected by its sensor. It is well known that many places mandate that vehicles should slow down around schools. We hypothesize that the slowing down of traffic at the school will diffuse/infuse to the train station. As a result, the model should pay more attention to this particular node (the school) that is $k$-hop away from the node (the train station) whose value we want to predict. In general, we are letting the model learn, using the attention mechanism, which process (i.e. how many hops away) among the infusion/diffusion processes (total $K$-hops) should the model pays more attention to.

**Use Points of Interests to Further Enhance the Soft Attention-enhanced Diffusion Convolution**

In Section 5.2, we have already given the intuition that the "types" (i.e. school, train station, hospital, etc.) play an important role in helping the attention mechanism to determine which process (i.e. how many hops away) among the infusion/diffusion processes (total $K$-hops) should the model pays more attention to. In this Section, we propose to introduce the points of interest as a feature for each node so that we can denote its type. Specifically, in previous work such as Li et al. (Yaguang Li et al., 2017), if the goal is to predict traffic speed after a time-lapse, then a time sequence that describes the prior observed traffic speed readings is passed to a predictive model. In D-A3T-GCN, we propose adding another feature, called the point of interest (POI), at each timestamp. The POI is determined by the sensor - for example, the POI of a sensor will be "school" if the sensor is close to a school (its proximity to the school is smaller than an empirically determined threshold). In other words, we add another dimension to the time sequence passed to the predictive model. In the case of this thesis, our goal is to find out if POIs can play an important role in helping improve the forecasting of traffic speed, so we only use a binary type of POI - in other words, all nodes/sensors are assigned a POI that is either 1 (close

to a school) or 0 (not close to a school). We choose “school" because it is widely known that vehicles reduce their speed near schools.

After adding the values of POIs to the time sequence, note that the value of the POI is affected by the message passing of the GNN architecture in D-A3T-GCN, and this is the expected/desired outcome: suppose that at node $v_i$, the POI is 0, meaning that it is not adjacent to a school that affects traffic speed. However, 1 hop away, one of its neighbors, $v_j$, has a POI that is 1, meaning that this node is adjacent to a school. The reduced traffic speed at $v_j$ is going to affect the traffic speed at $v_i$, even though $v_i$ itself does not have a POI that affects traffic speed. This observation is facilitated by the fact that the POI, being a feature, can be diffused to the nodes $K$-hops away. After one round of message passing (between $v_i$ and its neighbors, one of which is $v_j$), the POI at $v_i$ is updated to reflect that there is a school nearby (or that there is a node with a POI that can affect traffic speed nearby).

### 5.3 Evaluation

| Type | Dataset | # of Sensors | # of Edges | # of Instances |
|---|---|---|---|---|
| Speed | METR-LA | 207 | 1722 | 34272 |
| | PEMS-BAY | 325 | 2694 | 52116 |

Table 5.1: Descriptions on the datasets METR-LA and PEMS-BAY.

In this section, we conduct experiments on two often used datasets, METR-LA (Yaguang Li et al., 2017) and PEMS-BAY (Yaguang Li et al., 2017), that are large-scale and collected from the real world. METR-LA (Jagadish et al., 2014) is the collection of traffic information (volume, speed) detected by loop detectors in the Los Angeles County. There are in total 207 sensors and the data is collected from March 1st, 2012 to June 30th, 2012; in other words, there are data in METR-LA on the traffic information for four months. PEMS-BAY is created by California Transportation Agencies (CalTrans) Performance Measurement System, or PeMS for short. There are in total 325 sensors and the data is collected from January 1st, 2017 to May 31st, 2017; in other words, there are data in PEMS-BAY on the traffic information (volume, speed) for six months.

For both METR-LA and PEMS-BAY, we focus on the traffic speed. We aggregate traffic speed information into 5-minute intervals, after which we apply the Z-score normalization, as per the standard established by Li et al. (Yaguang Li et al., 2017). For both datasets, 80% of the data is in the training set, and the remaining 20% is in the testing set. Note that there do not exist “edges" in the datasets, so we construct the edges. We use the calculated pair-wise road network distances between the nodes/sensors and the adjacency matrix with edge weights is calculated by applying a threshold Gaussian kernel (Shuman et al., 2013). In particular, the weight of the edge between the $i$th and $j$th nodes, $W_{i,j}$, is described as Equation 5.8. If $\text{dist}(v_i, v_j)^2$ is smaller than a certain threshold $\lambda$, we proceed to calculate the edge weight between $v_i$ and $v_j$ based on Equation 5.8. Otherwise, we assign

the edge weight between $v_i$ and $v_j$ to be 0. In Equation 5.8, $\text{dist}(v_i, v_j)$ is the Euclidean distance between the nodes $v_i$ and $v_j$, and $\sigma$ is the standard deviations of the Euclidean distances between nodes.

$$W_{i,j} = \exp(-\frac{\text{dist}(v_i, v_j)^2}{\sigma^2}) \tag{5.8}$$

**Baselines**

In this Section, we describe the baselines that we have selected to compare against our D-A3T-GCN. Note that the baselines include both traditional methods such as the historical average (HA) method and deep learning methods such as DCRNN (Yaguang Li et al., 2017).

- **HA**: the Historical Average Model. HA slices the traffic flow into periods and uses the weighted averages from before as the predicted results for the periods in the future.
- **VAR**: the Vector Auto Regression Model (X. Wang et al., 2020). It treats the previous time series as stationary and calculates the correlation between that series and its lag value.
- **SVR**: the Support Vector Regression model. It employs the linear support vector machine to perform time series regression.
- **FC-LSTM**: the LSTM with fully connected hidden units (Sutskever, Vinyals, and Le, 2014). It has been used as a baseline in various works and is well-equipped in teasing out the spatial dependency of data.
- **DCRNN**: the Diffusion Convolutional Recurrent Neural Network Model (Yaguang Li et al., 2017), which treats the traffic flow as a diffusion and infusion process at each node/sensor. We have taken inspiration from the DCRNN to create the D-A3T-GCN model.
- **Graph WaveNet**: the Graph WavNet model (Z. Wu, Pan, G. Long, Jiang, and C. Zhang, 2019) employs gated TCN layers and GCN layers. The gated TCN layers capture the temporal dependencies and the GCN layers capture the spatial dependencies.
- **MTGNN**: the MTGNN model (Z. Wu, Pan, G. Long, Jiang, Chang, et al., 2020) is an extension of the Graph WavNet model by introducing a new mix-hop propagation layer in the temporal mechanism.
- **ASTGCN**: the ASTGCN model (Guo et al., 2019) uses the attention mechanism to jointly capture the dynamics of the spatial-temporal features of the traffic flow.

- **STSGCN**: the STSGCN (Song et al., 2020) takes into consideration the heterogeneity of the spatiotemporality in traffic flow data and localizes the spatiotemporal relationships to be captured.
- **GMAN**: the GMAN (Zheng et al., 2020) model uses stacked spatial attention and temporal attention to perform prediction on traffic flow data.
- **DGCRN**: The DGCRN model (F. Li et al., 2021) treats the graph as a dynamic entity and introduces the dynamic graph convolutional recurrent model, inspired by the seq2seq architecture.

**Experiment Design**

The experiments conducted to evaluate the proposed D-A3T-GCN are implemented in Pytorch on an NVIDIA A100 GPU. The optimizer is an Adam optimizer with a learning rate of 0.001 without decay. The time steps or horizons are set to be 3, 6, and 12 to account for $3 \times 5 = 15$ minutes, $6 \times 5 = 30$ minutes, and $12 \times 5 = 60$ minutes into the past (for reference of the past spatiotemporal dependencies) and future (for prediction). We use the three commonly used metrics for traffic forecasting - Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Mean Absolute Percentage Error (MAPE). We have included the Equations 5.9, 5.10, 5.11 for the calculation of RMSE, MAE, and MAPE. In Equations 5.9, 5.10, 5.11, $\Phi$ is the total number of instances, $y_i$ is the true value, and $\hat{y}_i$ is the predicted value.

$$\text{RMSE}(y_i, \hat{y}_i) = \sqrt{\frac{\sum_{i=1}^{\Phi}(y_i - \hat{y}_i)^2}{\Phi}} \tag{5.9}$$

$$\text{MAE}(y_i, \hat{y}_i) = \frac{\sum_{i=1}^{\Phi} |y_i - \hat{y}_i|}{\Phi} \tag{5.10}$$

$$\text{MAPE}(y_i, \hat{y}_i) = \frac{1}{\Phi} \frac{\sum_{i=1}^{\Phi} |y_i - \hat{y}_i|}{y_i} \tag{5.11}$$

**The Performance of D-A3T-GCN**

In Table 5.2 and Table 5.3, we evaluate the performance of D-A3T-GCN on the datasets METR-LA and PEMS-BAY respectively. As we can see from the Tables 5.2 and 5.3, D-A3T-GCN achieves the new state-of-the-art, significantly improving the state-of-the-art in the three metrics: RMSE, MAE, and MAPE.

In Tables 5.2 and 5.3, we observe the performance of D-A3T-GCN on METR-LA and PEMS-BAY against 12 baselines that include statistical, classical machine learning, and deep learning approaches. We have briefly discussed what these baselines are in Section 5.3. Observing Table 5.2, we conclude that methods such as HA perform the worst across the three horizons, because they have incorrect assumptions of the data. For example, HA assumes that the data is stationary. FC-LSTM performs better than traditional methods such as HA because it does not have this incorrect

| | Horizon = 3 | | | Horizon = 6 | | | Horizon = 12 | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | RMSE | MAE | MAPE | RMSE | MAE | MAPE | RMSE | MAE | MAPE |
| HA | 10.00 | 4.79 | 11.70% | 11.45 | 5.47 | 13.50% | 13.89 | 6.99 | 17.5% |
| VAR | 7.80 | 4.42 | 13.00% | 9.13 | 5.41 | 12.70% | 10.11 | 6.52 | 15.80% |
| SVR | 8.45 | 3.39 | 9.30% | 10.87 | 5.05 | 12.10% | 13.76 | 6.72 | 16.70% |
| FC-LSTM | 6.30 | 3.44 | 9.60% | 7.23 | 3.77 | 10.09% | 8.69 | 4.37 | 14.00% |
| DCRNN | 5.38 | 2.77 | 7.30% | 6.45 | 3.15 | 8.80% | 7.60 | 3.60 | 10.50% |
| STGCN | 5.74 | 2.88 | 7.62% | 7.24 | 3.47 | 9.57% | 9.40 | 4.59 | 12.70% |
| Graph WaveNet | 5.15 | 2.69 | 6.90% | 6.22 | 3.07 | 8.37% | 7.37 | 3.53 | 10.01% |
| ASTGCN | 9.27 | 4.86 | 9.21% | 10.61 | 5.43 | 10.13% | 12.52 | 6.51 | 11.64% |
| STSGCN | 7.62 | 3.31 | 8.06% | 9.77 | 4.13 | 10.29% | 11.66 | 5.06 | 12.91% |
| MTGNN | 5.18 | 2.69 | 6.88% | 6.17 | 3.05 | 8.19% | 7.23 | 3.49 | 9.87% |
| GMAN | 5.55 | 2.80 | 7.41% | 6.49 | 3.12 | 8.73% | 7.35 | 3.44 | 10.07% |
| DGCRN | 5.01 | 2.62 | 6.63% | 6.05 | 2.99 | 8.02% | 7.19 | 3.44 | 9.73% |
| D-A3T-GCN | 0.33 | 0.21 | 1.41% | 0.39 | 0.25 | 1.58% | 0.62 | 0.44 | 1.84% |

Table 5.2: The performance of D-A3T-GCN against the baselines on the METR-LA dataset. Horizon = 3 indicates 15 minutes into the future; horizon = 6 indicates 30 minutes into the future, and Horizon = 12 indicates 60 minutes into the future.

| | Horizon = 3 | | | Horizon = 6 | | | Horizon = 12 | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | RMSE | MAE | MAPE | RMSE | MAE | MAPE | RMSE | MAE | MAPE |
| HA | 4.30 | 1.89 | 4.16% | 5.82 | 2.50 | 5.62% | 7.54 | 3.31 | 7.65% |
| VAR | 3.16 | 1.74 | 3.60% | 4.25 | 2.32 | 5.00% | 5.44 | 2.93 | 6.50% |
| SVR | 3.59 | 1.85 | 3.80% | 5.18 | 2.48 | 5.50% | 7.08 | 3.28 | 8.00% |
| FC-LSTM | 4.19 | 2.05 | 4.80% | 4.55 | 2.20 | 5.20% | 4.96 | 2.37 | 5.70% |
| DCRNN | 2.95 | 1.38 | 2.90% | 3.97 | 1.74 | 3.90% | 4.74 | 2.07 | 4.90% |
| STGCN | 2.96 | 1.36 | 2.90% | 4.27 | 1.81 | 4.17% | 5.69 | 2.49 | 5.79% |
| Graph WaveNet | 2.74 | 1.30 | 2.73% | 3.70 | 1.63 | 3.67% | 4.52 | 1.95 | 4.63% |
| ASTGCN | 3.13 | 1.52 | 3.22% | 4.27 | 2.01 | 4.48% | 5.42 | 2.61 | 6.00% |
| STSGCN | 3.01 | 1.44 | 3.04% | 4.18 | 1.83 | 4.17% | 5.21 | 2.26 | 5.40% |
| MTGNN | 2.79 | 1.32 | 2.77% | 3.74 | 1.65 | 3.69% | 4.49 | 1.94 | 4.53% |
| GMAN | 2.91 | 1.34 | 2.86% | 3.76 | 1.63 | 3.68% | 4.32 | 1.86 | 4.37% |
| DGCRN | 2.69 | 1.28 | 2.66% | 3.63 | 1.59 | 3.55% | 4.42 | 1.89 | 4.43% |
| D-A3T-GCN | 0.27 | 0.15 | 0.87% | 0.36 | 0.21 | 1.24% | 0.57 | 0.38 | 0.57% |

Table 5.3: The performance of D-A3T-GCN against the baselines on the PEMS-BAY dataset. Horizon = 3 indicates 15 minutes into the future; horizon = 6 indicates 30 minutes into the future, and Horizon = 12 indicates 60 minutes into the future.

| | Horizon = 3 | | | Horizon = 6 | | | Horizon = 12 | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | RMSE | MAE | MAPE | RMSE | MAE | MAPE | RMSE | MAE | MAPE |
| D-A3T-GCN | 0.33 | 0.21 | 1.41% | 0.39 | 0.25 | 1.58% | 0.62 | 0.44 | 1.84% |
| D-A3T-GCN (Complex) | 0.32 | 0.20 | 1.36% | 0.41 | 1.52 | 1.52% | 0.62 | 0.44 | 1.81% |

Table 5.4: The performance of D-A3T-GCN (with binary POIs) against D-A3T-GCN with numerical/more complex POIs on the METR-LA dataset.

| | Horizon = 3 | | | Horizon = 6 | | | Horizon = 12 | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | RMSE | MAE | MAPE | RMSE | MAE | MAPE | RMSE | MAE | MAPE |
| D-A3T-GCN | 0.27 | 0.15 | 0.87% | 0.36 | 0.21 | 1.24% | 0.57 | 0.38 | 0.57% |
| D-A3T-GCN (Complex) | 0.26 | 0.15 | 0.84% | 0.35 | 0.21 | 1.22% | 0.56 | 0.37 | 2.09% |

Table 5.5: The performance of D-A3T-GCN (with binary POIs) against D-A3T-GCN with numerical/more complex POIs on the PEMS-BAY dataset.

| | Horizon = 3 | | | Horizon = 6 | | | Horizon = 12 | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | RMSE | MAE | MAPE | RMSE | MAE | MAPE | RMSE | MAE | MAPE |
| D-A3T-GCN (no POIs) | 0.97 | 0.84 | 1.03% | 0.98 | 0.85 | 1.11% | 0.98 | 0.84 | 1.06% |
| D-A3T-GCN | 0.33 | 0.21 | 1.41% | 0.39 | 0.25 | 1.58% | 0.62 | 0.44 | 1.84% |

Table 5.6: The performance of D-A3T-GCN (without POIs) against D-A3T-GCN with numerical/more complex POIs on the METR-LA dataset. As we can observe from this Table, even without POIs, D-A3T-GCN still achieves state-of-the-art performance on the dataset of METR-LA, which indicates the superiority of the D-A3T-GCN architecture itself.

| | Horizon = 3 | | | Horizon = 6 | | | Horizon = 12 | | |
|---|---|---|---|---|---|---|---|---|---|
| Methods | RMSE | MAE | MAPE | RMSE | MAE | MAPE | RMSE | MAE | MAPE |
| D-A3T-GCN (no POIs) | 0.89 | 0.65 | 1.05% | 0.90 | 0.66 | 1.04% | 0.90 | 0.65 | 1.05% |
| D-A3T-GCN | 0.27 | 0.15 | 0.87% | 0.36 | 0.21 | 1.24% | 0.57 | 0.38 | 0.57% |

Table 5.7: The performance of D-A3T-GCN (without POIs) against D-A3T-GCN with numerical/more complex POIs on the PEMS-BAY dataset. Once again, As we can observe from this Table, even without POIs, D-A3T-GCN still achieves state-of-the-art performance on the dataset of PEMS-BAY, which indicates the superiority of the D-A3T-GCN architecture over the state-of-the-arts.

assumption, but it does not perform as well as the newer methods that consider both the spatial and temporal dependencies of the data. DCRNN achieves improvements across the three horizons over the three metrics because it considers modeling both the spatial and temporal correlations of the data; so does Graph WaveNet, which achieves even better performance than DCRNN. GMAN performs very well because it also includes the attention mechanism, which is effective at modeling long-term dependency. Based upon the DCRNN architecture, DGCRN is the best-performing state-of-the-art solution.

However, we still outperform DGCRN by a large margin across the horizons in terms of RMSE, MAE, and MAPE. Specifically, in Table 5.2, across all three horizons, in terms of RMSE, D-A3T-GCN outperforms the DGCRN, by 4.86 when the horizon is set to 3, 5.66 when the horizon is 6, and 6.57 when the horizon is 12. This is a significant improvement over the DGCRN's performance in terms of RMSE at the three horizons when they are 5.01, 6.05, and 7.19 in comparison. Similar trends of

significant improvement can be observed on the metrics of MAE and MAPE across the three horizons as well. Similarly, in Table 5.3, we observe the same significant improvement: in terms of RMSE, D-A3T-GCN results in a 0.27 (compared to DGCRN's 2.69) when the horizon is set to 3, a 0.36 (compared to DGCRN's 3.63) when the horizon is set to 6, and a 0.57 (compared to DGCRN's 4.42). The evaluation results on the METR-LA and PEMS-BAY datasets as indicated in Tables 5.2 and 5.3 confirms that the proposed D-A3T-GCN achieves the new state-of-the-art, outperforming the second best-performing methods by a huge margin and confirming the proposed method's superiority.

There is one limitation of D-A3T-GCN: the rate of its deterioration when the horizon is increased is steeper than the baselines that we are comparing against. In Table 5.2, the best-performing baseline achieves an RMSE of 5.01 when the horizon is set to 3, which deteriorates by 20.7% when the horizon is increased to 6, which then deteriorates by 18.8% when the horizon is increased to 12. D-A3T-GCN, on the other hand, starts at an RMSE of 0.33, which deteriorates by 18.1% when the horizon is increased to 6, which then deteriorates by 58.9%. Similarly, we see the same trend in the scores of MAE and MAPE when D-A3T-GCN is evaluated on METR-LA: the deterioration rates for the other two metrics, MAE and MAPE are steep as well. In terms of MAE, when the horizon is set to 3, the MAE score starts at 0.21, which then deteriorates to 0.25 by 19%, which then deteriorates to 0.44 by 0.76%. In Table 5.3, the same phenomenon of steep metric scores deterioration is also observed.

However, one must also take into consideration that the metrics scored achieved by D-A3T-GCN are much smaller than the baselines, to begin with. If we simply measure the deterioration by the increase in the metric scores, D-A3T-GCN is considered to have a better deterioration rate across the metric scores than the baselines. Once again we compare D-A3T-GCN against the second best-performing baseline, DGCRN. Since in the previous paragraph we used 5.2 as the concrete example, in this paragraph we use Table 5.3. In terms of RMSE, DGCRN's performance deteriorates by 0.94 when the horizon is increased from 3 to 6, and by 0.79 when it is further increased from 6 to 12. In terms of RMSE, D-A3T-GCN's performance decreases by only 0.09 when the horizon is increased from 3 to 6, and 0.21 when it is further increased from 6 to 12. A similar trend can be observed in the deterioration of the MAE and MAPE scores. It is worth noting that when the horizon is 12, the MAPE score is 0.57%, which is better than the MAPE scores when the horizon is set to 3 and 6. This can take place when the percentage errors for each prediction become more proportional to the ground truth values of the target variable.

In Tables 5.4 and 5.5, we observe the discrepancy that the difference in the type of the POIs made. Table 5.5 demonstrates the discrepancy on the PEMS-BAY dataset: across all three horizons, the difference between the case where the POIs are numerical and the case where the POIs are binary is not pronounced. For example, when the horizon is set to 3, the difference in terms of RMSE scores between the two cases is that the numerical POIs are slightly better than the binary POIs because

its RMSE score is 0.01% better. The same phenomenon can be observed when the horizon is set to 6 and 12. This hardly demonstrates the superiority of the numerical POIs. In Table 5.4 that demonstrates the discrepancy between the two types of POIs on the METR-LA dataset, we observe when horizion is set to 3, the RMSE score achieved by the numerical POIs is 0.01 better than the RMSE score achieved by binary POIs. However, the numerical POIs' performance is worse than the binary POIs's performance when the horizon is set to 6: its RMSE is 0.02 worse. When the horizon is set to 12, the both types of POIs achieve the same RMSE scores. Once again, this hardly demonstrates that one type of POIs is superior than the other. However, notice that both types of POIs achieve way better performance than the best-performing state-of-the-art baseline - DGCRN. Therefore, we conclude that both types of POIs are effective, but there is no clear superiority.

**Ablation Study**

So far, we have talked about incorporating POIs in the architecture of D-A3T-GCN, which begs the question: how good is the D-A3T-GCN architecture, without the POIs? Will it still achieve state-of-the-art performance? We answer this question in Tables 5.6 and 5.7. As we can observe from Tables 5.6 and 5.7, in the cases of both datasets, applying POIs does improve the performance of the D-A3T-GCN architecture, which demonstrates the importance of the POIs. However, even without the POIs, meaning that when the input vectors to D-A3T-GCN are the same as the input vectors fed to the other models in Tables 5.2 and 5.3, D-A3T-GCN still achieves the state-of-the-art performance. This indicates that the architecture of D-A3T-GCN is superior to the state-of-the-art architectures: Specifically when we do not apply POIs at all and we use the RMSE as an example, the D-A3T-GCN architecture achieves an RMSE score of 0.89 (compared against DGCRN's 2.69) when horizon is 3, an RMSE score of 0.90 (compared against DGCRN's 3.63), an RMSE score of 0.90 (compared against DGCRN's 4.42) when compared on the PEMS-BAY dataset.

## 5.4 Limitation

We have already mentioned one limitation of the work is that it could have a worse deterioration rate compared to the baselines. We use the word "could" because it depends on how we determine the deterioration rate. If the deterioration rate is defined by percentage, then D-A3T-GCN is achieving a worse deterioration rate than the baselines. However, we the deterioration is defined by the absolute value, then D-A3T-GCN achieves a better deterioration rate than the baselines.

## 5.5 Conclusion

Spatiotemporal forecasting has been crucial in various application domains such as traffic prediction and climate prediction. in this thesis, we focus on the task of traffic prediction, which remains challenging due to the spatial dependencies of the road networks and the temporal dependencies of changing traffic conditions. To address the challenges, we treat the road map as a graph with the nodes being

sensors and edges being the connections among the sensors, and model the traffic flow as a diffusion/infusion model built upon a graph convolutional net (GCN). To further capture the global spatiotemporal dynamics, we propose to use the attention mechanism to enhance the diffusion/infusion theory-enhanced GCN modeling for traffic prediction. Titled D-A3T-GCN, our solution is evaluated on two large-scale real-world datasets, METRA-LA and PEMS-BAY. And the performance result on the two datasets suggest that we achieve the state-of-the-art performance: for example, on METR-LA, in terms of RMSE, D-A3T-GCN shows an improvement over the second best performing algorithm, DGCRN, by 1400% when the horizon is set to 3.

*Chapter 6*

# USE GAN TO GENERATE DOMAIN AGNOSTIC SAMPLES

In Chapter 4, we have already mentioned that domain adaptation is effective at dealing with the distribution shift in the data. Precisely, it deals with the scenario in which a DL model is trained on its training samples with one distribution but applied to testing samples with a different distribution. In this Chapter, we propose another (unsupervised) domain adaptation algorithm that improves the state-of-the-art on unsupervised domain adaptation. It is called the MiddleGAN, which utilizes the generative adversarial network (GAN) architecture to generate a large number of samples that are similar to both distributions. Then, we train a state-of-the-art classifier on the synthetic samples so that the classifier implicitly learns the features that are domain-agnostic.

## 6.1 Introduction

In recent years, deep learning has achieved impressive results across different application domains (Esteva et al., 2021; K. He et al., 2016; Szegedy, Ioffe, et al., 2017; Y. Zhu and Newsam, 2017; Purwins et al., 2019; Noda et al., 2015; L. Wu et al., 2021; Wahab et al., 2021). However, a deep neural net does not necessarily perform well on a new domain with different distribution than its training set. This problem is called domain shift, and domain adaptation (DA) have been invented to tackle the issue of domain shift. One approach of DA is to find domain-invariant features (H. Zhao et al., 2019).

Instead of explicitly selecting domain-invariant features, we propose to let a classifier that will perform the classification task on the target domain implicitly learn to use domain-invariant features (to perform classification). In this way, we do not have to hand-engineer the features which may not be inclusive enough to include all the features that are domain-invariant. Our intuition is based on the observation that deep neural networks such as the ResNet-50 (K. He et al., 2016) or Inception (Szegedy, Vanhoucke, et al., 2016) generalize well when trained on a large amount of data. If we want the classifier to learn the domain invariant features (implicitly), we need a large amount of samples that is similar to both the source domain samples and the target domain samples. We call those samples domain agnostic samples. If we train a neural network such as the ResNet-50 with a large quantity of domain agnostic samples, the neural network will implicitly learn to use the domain-invariant features to perform classification.

How do we generate those domain agnostic samples? We propose a variation of GAN, called the MiddleGAN, which has two discriminators and a generator. One discriminator is for the source domain; it tries to distinguish a generated sample from real samples from the source domain. Another discriminator is for the target domain; it tries to distinguish a generated sample from the real samples from the

target domain. The generator is trying to generate samples that can deceive both discriminators at the same time. The three neural networks engage in a minimax game in which the generator is trying to generate samples to confuse both the source discriminator and the target discriminator. Ideally, after training, the generated samples will be indistinguishable from both the real source domain samples and the real target domain samples, thus achieving the similarity to samples of both domains. We have also extended the theory of GAN to theoretically prove that there exist optimal values for the source and target discriminators and the generator.

**The contributions of this thesis are**:

- We create a novel variation of GAN, the MiddleGAN, that generates samples that are similar to samples from both the source and target domains. In other words, these generated samples are domain invariant.
- We extend the theory of GAN to show that there exist optimal solutions for the parameters of the two discriminators and one generator in MiddleGAN.
- We empirically show that the samples generated by the MiddleGAN are similar to both samples from the source domain and samples from the target domain.
- We have observed in our Evaluation Section (Section 6.3) that the MiddleGAN is significantly better than the state-of-the-art algorithms on domain adaptation tasks in which the domain shifts are large, indicating that the MiddleGAN can be applied to a wider range of domain adaptation tasks than the other state-of-the-art domain adaptation algorithms.
- We conduct extensive evaluations on 24 benchmarks; on the 24 benchmarks, we compare MiddleGAN against various state-of-the-art algorithms and outperform the state-of-the-art by up to 20.1% on certain benchmarks.

## 6.2 MiddleGAN

Before we discuss our MiddleGAN, we need to discuss the original GAN on which MiddleGAN is based. In the original GAN (Goodfellow, Pouget-Abadie, et al., 2014), the generator $G$ and the discriminator $D$ engage in a minimax game in which $G$ tries to minimize a value objective $V(G, D)$ whereas $D$ tries to maximize it. $V(G, D)$ is defined in Equation 6.1, in which $p$ is the distribution of the real samples and $q$ is the distribution of the noise. A key observation obtained from Equation 6.1 is that $G$'s effort is to generate $G(z)$ whereas $z$ is an input noise such that $G(z)$ will be in-distribution with the distribution of the real samples $p$.

$$\begin{aligned}\min_G \max_D V(G, D) = \mathbb{E}_{x\sim p(x)}[\log(D(x)) \\ + \mathbb{E}_{z\sim q(z)}[\log(1 - D(G(z)))]\end{aligned} \tag{6.1}$$

Based on the key observation that we obtain from Equation 6.1, in MiddleGAN we propose to employ two discriminators, $D_s$ and $D_t$. $D_s$ tries to distinguish a

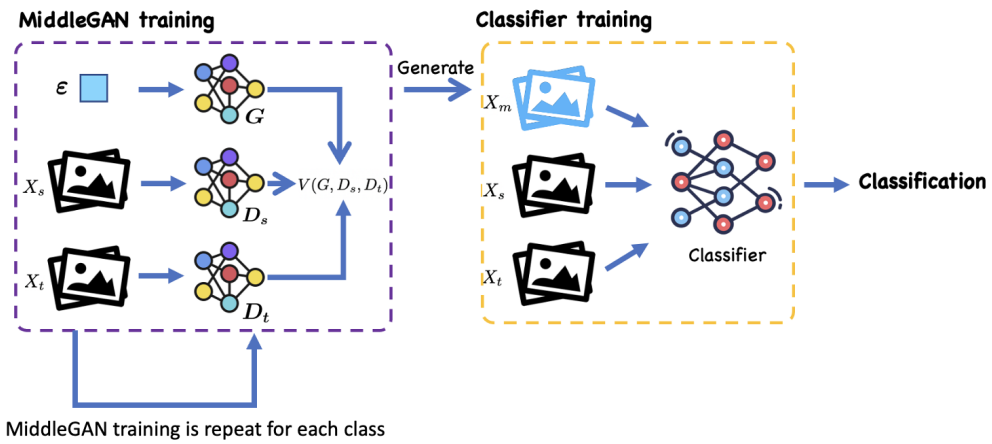



Figure 6.1: In this Figure we describe how to use the MiddleGAN to generate fake, domain agnostic samples and how to use those domain agnostic samples to train the classifier that eventually performs the classification task on the target domain.

generated sample from real source domain samples, and $D_t$ tries to distinguish a generated sample from real target domain samples. The generator $G$ engages in a two-way minimax game with the two discriminators. The samples it generates will be in the middle of the feature space of the source and the target domains. Below, we empirically prove that the generated samples in $p_m$ are represented by the features that are invariant across the source and target domains.

Formally, the objective function of $D_t$, $D_s$, and $G$ is described by Equation 6.2.

$$\begin{aligned} &\min_{G} \max_{D_s, D_t} V(G, D_s, D_t) \\ &= \mathbb{E}_{x_s \sim p_s(x_s)}[\log(D(x_s))] + \mathbb{E}_{z \sim q(z)}[\log(1 - D_s(G(z)))] \\ &+ \mathbb{E}_{x_t \sim p_t(x_t)}[\log(D(x_t))] + \mathbb{E}_{z \sim q(z)}[\log(1 - D_t(G(z)))] \end{aligned} \tag{6.2}$$

In the previous paragraphs we have described how to generate samples that are similar to both the source samples and the target samples. Recall that we have argued that we will feed these samples during training to the classifier that performs the final classification task on the target domain (in a supervised fashion). In this case, how do we obtain the labels of the generated samples? The labels of the generated data is the same as the labels of the source and target samples that are used to generate them. In other words, only source samples of a particular class and target samples of that particular class get to be used to generate fake samples of this class. We repeat the generation process for all classes in the source and target domains to generate fake samples.

Note that in the setting of unsupervised domain adaptation, the labels of the target domain are not available. How do we obtain those labels to train the generator and classifier? We propose to use an existing unsupervised domain adaptation algorithm, Fixbi, to obtain pseudo-labels for the target domains. Then, we use the

pseudo-labels of the target domain, together with the labels of the source domain, to train the generator and the classifier.

Figure 6.1 shows the flowchart of how to use the generated domain agnostic samples to train the classifier that eventually performs classification on the target domain. In Figure 6.1, $\epsilon$ is noise, $X_s$ and $X_t$ are the source samples and target samples respectively, and $X_m$ is the generated samples by $G$. During the training of MiddleGAN, the three neural networks $G$, $D_s$, and $D_t$ engaged in a minimax game. The process of MiddleGAN training (in the purple box) is repeated for all classes in the source domain (and the target domain). During the training of the classifier, both $X_s$, $X_t$, and $X_m$ and their labels are used.

**Theoretical Results**

We first discuss the two discriminators $D_s$ and $D_t$ given a fixed $G$. We propose Theorem 6.2 regarding the optimal values for $D_s$ and $D_t$, represented as $D_s^*$ and $D_t^*$

Given $p_m$, the distribution of samples generated by a fixed generator $G$, the optimal values for the parameters of $D_s$ and $D_t$ are $D_s^* = \frac{p_s}{p_s+p_m}$ and $D_t^* = \frac{p_t}{p_t+p_m}$.

*Proof.* The value objective $V(G, D_s, D_t)$ can be expanded.

$$
\begin{aligned}
V(G, D_s, D_t) = & \int_{x_s} p_s(x_s) \log(D_s(x_s)) dx_s \\
& + \int_{x_t} p_t(x_t) \log(D_t(x_t)) dx_t \\
& + \int_z q(z) \log(1 - D_s(G(z))) dz \\
& + \int_z q(z) \log(1 - D_t(G(z))) dz \\
& = \int_{x_s} p_s(x_s) \log(D_s(x_s)) dx_s \\
& + p_m(x_s) \log(1 - D_s(x_s)) dx_s \\
& + \int_{x_t} p_t(x_t) \log(D_t(x_t)) \\
& + p_m(x_t) \log(1 - D_s(x_t)) dx_t
\end{aligned}
\tag{6.3}
$$

We observe that $p_s$, $p_t$ and $p_m$ belong in $\mathbb{R}$. For the source discriminator $D_s$, any pair of $p_s$ and $p_m$ in the form of $p_s \log(y) + p_m (1 - \log(y))$, $p_s \log(y) + p_m (1 - \log(y))$ achieves its maximum value at $\frac{p_s}{p_s+p_m}$ (Goodfellow, Pouget-Abadie, et al., 2014). Similarly, for the target discriminator $D_t$, any pair of $p_t$ and $p_m$ in the form of $p_t \log(y) + p_m (1 - \log(y))$, $p_t \log(y) + p_m (1 - \log(y))$ achieves its maximum value at $\frac{p_t}{p_t+p_m}$. □

Now we bring forth Theorem 6.2 which proposes that there exists an optimal solution for the parameters of not only $D_s$ and $D_t$, but also $G$. There exists a global minimum

for the virtual training criterion $C(G)$ defined as

$$C(G) = \max_{D_s, D_t} V(G, D_s, D_t). \tag{6.4}$$

In other words, there exists an optimal solution for the parameters of the generator $G$.

*Proof.* Goodfellow et al. (Goodfellow, Pouget-Abadie, et al., 2014) have proved that, in the original GAN where there is only one discriminator $D$ and one generator $G$, the virtual training criterion can be written as the following:

$$\begin{aligned} C_{original}(G) &= \max_{D} V(G, D) \\ &= -log(4) + KL(p \parallel \frac{p + p_m}{2}) \\ &+ KL(p_m \parallel \frac{p + p_m}{2}) \end{aligned} \tag{6.5}$$

in which $p$ is the distribution of the real samples and $p_m$ is the distribution of generated fake samples, and KL is the Kullback–Leibler divergence. With two discriminators, our virtual training criterion $C(G)$ can be rewritten as:

$$\begin{aligned} C(G) =& - log(4) + KL(p_s \parallel \frac{p_s + p_m}{2}) \\ &+ KL(p_m \parallel \frac{p_s + p_m}{2}) \\ &- log(4) + KL(p_t \parallel \frac{p_t + p_m}{2}) \\ &+ KL(p_m \parallel \frac{p_t + p_m}{2}) \\ =& - 2log(4) + 2JSD(p_s \parallel p_m) + 2JSD(p_t \parallel p_m) \end{aligned} \tag{6.6}$$

In Equation 6.6, JSD is the Jensen–Shannon divergence. To find the global minimum, $M(G)$, we want to obtain

$$\begin{aligned} M(G) = \operatorname*{argmin}_{p_m} & - 2log(4) \\ & + 2JSD(p_s \parallel p_m) + 2JSD(p_t \parallel p_m) \end{aligned} \tag{6.7}$$

We observe in Equation 6.7 that we are looking for the optimal value of the JSD centroid defined as $Centroid^* = arg \min_{Q} \sum_{i=1}^{n} JSD(P_i \parallel Q)$ in which $P_i$ and $Q$ are distributions. We can see that the generator is essentially looking for the JSD centroid of the source domain distribution $p_s$ and the target domain distribution $p_t$. The convexity of the problem has been proved in (Nielsen, 2020). □

**Guaranteed Domain Agnosticism of Generated Samples**

Are the samples generated by the MiddleGAN similar to both the source and the target domains? In this section we use two examples to show that fake samples in the distribution $p_m$ are similar to both the samples in the source and target distributions $p_s$ and $p_t$.

To test if the generated samples are indeed similar to both the source and target samples (domain agnostic), we propose a simple, but effective way to attest it. We treat the original, unaltered MNIST (L. Deng, 2012) as the source domain. MNIST is a dataset that contains pictures of 10 classes of handwritten digits. For the target domain, we alter the MNIST dataset by rotating each sample 180 degrees. Then, we use the MiddleGAN to generate the intermediate samples. After obtaining the fake samples, we perform the first round of an experiment by training a classifier (Inception v3 with the last layer replaced to have 10 neurons to correspond to ten classes of handwritten digits) on the combination of the training sets of both the source and the target domains as well as the fake samples. We achieve an accuracy of 99.4% on the source domain's testing set and an accuracy of 99.4% on the target domain's testing set (Accuracy is calculated in terms of whether the classifier correctly classifies a sample that is of one of the ten classes of handwritten digits). We use a learning rate of 0.0002 and the Adam optimizer and train 5 epochs. Note that the difference between the source and target domains in the first round of the experiments is only caused by the direction of the MNIST samples. To demonstrate if the fake samples are robust to the difference (i.e. domain agnostic), we rotate those fake samples by 180 degree as well. Then, we train a classifier with the same structure using the same hyperparameters including the learning rate, the optimizer, and the training epochs. Then, we train the classifier on the combination of the training sets of both the source and the target domains as well as the **rotated** fake samples. We have achieved an accuracy of 99.4% on the source testing set and 99.3% on the target testing set.

| | Source Acc. | Target Acc. |
|---|---|---|
| Upright fake samples | 99.4% | 99.4% |
| Rotated fake samples | 98.9% | 99.3% |

Table 6.1: The results from the two rounds of experiments. There is no significant change to the performance measured in accuracy on both the source and target's testing sets.

From Table 6.1 we conclude that there is no significant change to the performance of the classifier, despite that one's training samples contain only upright fake samples and the other's training samples contain only rotated fake samples. **This indicates that the fake samples are domain agnostic because whether or not we rotate them makes no difference.**

We have included another example to demonstrate that the fake samples generated

Figure 6.2: The left figure is from the source, real samples of cis gender women. The middle figure is from the target, real samples of cis gender men. The right samples are fake samples generated by MiddleGAN. As we can observe, the fake samples contain both feminine and masculine facial features.

by the MiddleGAN are similar to both the source and the target domains. Figure 6.2 contains three subfigures. The first subfigure (from CelebA Dataset (Z. Liu et al., 2015)) contains 64 real samples of cisgender women (the source domain). The second subfigure (from CelebA Dataset (Z. Liu et al., 2015)) contains 64 samples of cisgender men (the target domain). The third subfigure contains 64 samples of fake samples (generated by the MiddleGAN) that visually have both the characteristics of femininity and masculinity. Therefore, it is attested visually that the fake samples generated by MiddleGAN have the similarity of both the source and the target domains. Upon inspection by two human inspectors, both agree that the generated samples have both feminine and masculine features. In other words, the generated samples are similar to both the source and target domains.

## 6.3 Evaluation

| Algorithm | A→W | A→D | D→W | D→A | W→A | W→D | Avg |
|---|---|---|---|---|---|---|---|
| ResNet-50 (K. He et al., 2016) | 68.4% | 68.9% | 96.7% | 62.5% | 60.7% | 99.3% | 76.1% |
| DANN (Ganin and Lempitsky, 2015) | 82.0% | 79.7% | 96.9% | 68.2% | 67.4% | 99.1% | 82.2% |
| MSTN (Xie et al., 2018) | 91.3% | 90.4% | 98.9% | 72.7% | 65.6% | **100%** | 86.5% |
| CDAN+E (M. Long et al., 2018) | 94.1% | 92.9% | 98.6% | 71.0% | 69.3% | **100%** | 87.7% |
| DMRL (Yuan Wu, Inkpen, and El-Roby, 2020) | 90.8% | 93.4% | 99.0% | 73.0% | 71.2% | **100%** | 87.9% |
| SymNets (Y. Zhang et al., 2019) | 90.8% | 93.9% | 98.8% | 74.6% | 72.5% | **100%** | 88.4% |
| GSDA (L. Hu et al., 2020) | 95.7% | 94.8% | 99.1% | 73.5% | 74.9% | **100%** | 89.7% |
| CAN (Kang et al., 2019) | 94.5% | 95.0% | 99.1% | 78.0% | 77.0% | 99.8% | 90.6% |
| SRDC (Tang, K. Chen, and K. Jia, 2020) | 95.7% | **95.8%** | 99.2% | 76.7% | 77.1% | **100%** | 90.8% |
| RSDA-MSTN (Gu, J. Sun, and Z. Xu, 2020) | **96.1%** | **95.8%** | 99.3% | 77.4% | 78.9% | **100%** | 91.1% |
| FixBi (Na et al., 2021) | **96.1%** | 95.0% | 99.3% | 78.7% | 79.4% | **100%** | 91.4% |
| MiddleGAN | 92.4% | 94.1% | **100%** | **84.9%** | **83.5%** | **100%** | **92.4%** |

Table 6.2: The results on the domain adaptation tasks among the three domains in the dataset Office-31. The metric is accuracy. A stands for the Amazon domain in Office-31. Similarly, W stands for Webcam, and D stands for DSLR.

In this section, we evaluate the MiddleGAN on the following tasks: CIFAR-10 ↔ STL-10 (two tasks), MNIST ↔ USPS (two tasks), MNIST ↔ SVHH (two

| Algorithm | Pr→Ar | Ar→Pr | Cl→Ar | Ar→Cl | Rw→Ar | Ar→Rw | Pr→Cl | Cl→Pr | Rw → Pr | Pr→Rw | Rw→Cl | Cl→Rw | Avg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ResNet-50 (K. He et al., 2016) | 38.5% | 50% | 37.4% | 34.9% | 53.9% | 58% | 31.2% | 41.9% | 59.9% | 60.4% | 41.2% | 46.2% | 46.1% |
| DANN (Ganin and Lempitsky, 2015) | 41.6% | 59.3% | 47.0% | 45.6% | 63.2% | 70.1% | 43.7% | 58.5% | 76.8% | 68.5% | 51.8% | 60.9% | 57.6% |
| CDAN (M. Long et al., 2018) | 55.6% | 69.3% | 54.4% | 49.0% | 68.4% | 74.5% | 48.3% | 66.0% | 80.5% | 75.9% | 55.4% | 68.4% | 63.8% |
| MSTN (Xie et al., 2018) | 61.4% | 70.3% | 60.4% | 49.8% | 70.9% | 76.3% | 48.9% | 68.5% | 81.1% | 75.7% | 55.0% | 69.6% | 65.7% |
| SymNets (Y. Zhang et al., 2019) | 63.6% | 72.9% | 64.2% | 47.7% | 73.8% | 78.5% | 47.6% | 71.3% | 82.6% | 79.4% | 50.8% | 74.2% | 67.2% |
| GSDA (L. Hu et al., 2020) | 65.0% | 76.1% | 65.4% | 61.3% | 72.2% | 79.4% | 53.2% | 73.3% | 83.1% | 80.0% | 60.6% | 74.3% | 70.3% |
| GVB-GD (Cui et al., 2020) | 65.2% | 74.7% | 64.6% | 57.0% | 74.6% | 79.8% | 55.1% | 74.1% | 84.3% | 81.0% | 59.7% | 74.6% | 70.4% |
| RSDA-MSTN (Gu, J. Sun, and Z. Xu, 2020) | 67.9% | 77.7% | 66.4% | 53.2% | 75.8% | **81.3%** | 53.0% | 74.0% | 85.4% | **82.0%** | 57.8% | 76.5% | 70.9% |
| SRDC (Tang, K. Chen, and K. Jia, 2020) | **68.7%** | 76.3% | **69.5%** | 52.3% | 76.3% | 81.0% | 53.8% | 76.2% | 85.0% | 81.7% | 57.1% | 78.0% | 71.3% |
| Fixbi (Na et al., 2021) | 65.8% | 77.3% | 67.7% | 58.1% | **76.4%** | 80.4% | 57.9% | 79.5% | **86.7%** | 81.7% | 62.9% | **78.1%** | 72.7% |
| MiddleGAN | 65.0% | **86.9%** | 63.3% | **78.2%** | 66.2% | 76.8% | **73.8%** | **86.4%** | 86.1% | 71.5% | **75.2%** | 73.7% | **75.3%** |

Table 6.3: The results on the domain adaptation tasks among the four domains in the dataset Office-Home. The metric is accuracy. Office-Home has four domains: Art (Ar), Clipart (Cl), Real World (Rw), and Product (Pr).

| Algorithm | CIFAR-10 → STL-10 | STL-10 → CIFAR-10 |
|---|---|---|
| Source Only (Yan et al., 2020) | 75.9% | 61.8% |
| VADA (Shu et al., 2018) | 80.0% | 73.5% |
| IIMT (Yan et al., 2020) | 83.1% | 81.6% |
| Enforced Transfer (Gao, Baucom, et al., 2022) | 86.1% | - |
| SE (French, Mackiewicz, and Fisher, 2017) | 76.3% | 83.9% |
| SEMA (Zuo et al., 2021) | 78.7% | 86.6% |
| MiddleGAN | **89.5%** | **98.7%** |

Table 6.4: The results on the domain adaptation task of CIFAR-10 → STL-10 and STL-10 → CIFAR-10 of 5 state-of-the-art domain adaptation algorithms and the MiddleGAN. On both tasks, we outperform the second best-performing algorithms by a large margin (3.4% on CIFAR-10 → STL-10 and 12.1% on STL-10 → CIFAR-10), which demonstrates the superiority of the MiddleGAN. The metric is accuracy.

tasks), and on two domain adaptation benchmarks Office-31 and Office-Home which contain three domains and four domains, respectively. Therefore, there are 6 domain adaptation tasks derived from Office-31 and 12 domain adaptation tasks derived from Office-Home. On all 24 tasks that we evaluate, MiddleGAN outperforms the state-of-the-art by up to 20.1% on certain benchmarks.

## Setups

### Datasets

The following datasets are used to evaluate the MiddleGAN.

**CIFAR-10** (Krizhevsky, Hinton, et al., 2009) contains 10 classes of images that are $32 \times 32$ pixels in size. It is a fairly large dataset; each of its classes has 6000 images. Its training set contains 50,000 images and its testing set contains 10,000 images. Accuracy on its testing set is calculated in terms of whether the classifier correctly classifies a sample that is of one of the ten classes of images.

**STL-10** (Coates, Ng, and H. Lee, 2011) contains 10 classes of images that are $96 \times 96$ pixels in size. It is different from CIFAR-10 by one class. For each of its classes, there are 500 training samples and 800 testing samples. Accuracy on its testing set is calculated in terms of if the classifier correctly classifies a sample that is one of

the ten classes of images.

**MNIST** (LeCun, Bottou, et al., 1998) contains 10 classes of handwritten digits. They are 28 × 28 pixels in size. There are 60,000 training samples and 10,000 testing samples. Accuracy on its testing set is calculated in terms of if the classifier correctly classifies a sample that is of one of the ten handwritten digits.

**USPS** (Hull, 1994) contains 10 classes of handwritten digits obtained via scanning the envelopes from the USPS. There are 9298 images in total of the 10 classes, and each of them is of size 16 × 16 pixels. The samples are in grayscale. We have converted it to RGB. Accuracy on its testing set is calculated in terms of if the classifier correctly classifies a sample that is one of the ten handwritten digits.

**SVHN** (Netzer et al., 2011) stands for Street View House Numbers. It contains 10 classes of digits obtained by street view cameras. There are 600,000 samples of printed images of size 32 × 32 pixels. Accuracy on its testing set is calculated in terms of whether the classifier correctly classifies a sample that is of one of the ten street view digits.

**Office-31** (Saenko et al., 2010) has three domains: Amazon (A), Dslr (D), and Webcam (W). Each domain contains 31 classes of office objects such as projectors and rulers. In total, there are 4,110 images. Six domain adaptation tasks can be formed from the Office-31 dataset and we evaluate the MiddleGAN against state-of-the-art solutions on all of the domain adaptation tasks. Accuracy on its testing set is calculated in terms of if the classifier correctly classifies a sample that is one of the 31 object classes.

**Office-Home** (Venkateswara et al., 2017) has four domains: Art (Ar), Clipart (Cl), Real World (Rw), and Product (Pr). Each domain contains 65 classes of objects that can be found in an office or a home, such as flowers and bikes. In total, there are 15,500 images. Twelve domain adaptation tasks can be formed from the Office-Home dataset and we evaluate the MiddleGAN against state-of-the-art solutions on all of the domain adaptation tasks. Accuracy on its testing set is calculated in terms of if the classifier correctly classifies a sample that is one of the 65 object classes.

### Implementation details

On all 12 domain adaptation tasks, our source discriminator $D_s$, target discriminator $D_t$, and generator $G$ share the same structures. For $D_s$ and $D_t$, we have 5 2D convolutional layers followed by Leaky ReLu layers with a negative slope of 0.2. After each of the 2nd, 3rd, and 4th 2D convolutional layers, a 2D batch norm layer is added. The activation function is Sigmoid. The learning rate of the Adam optimizers for both $D_s$ and $D_t$ are 0.0002. For the generator $G$, its structure contains 5 transposed 2D convolutional layers. After each of the 1st, 2nd, 3rd, and 4th layers, a 2D batch norm layer is added. The activation function is tanh. The learning rate of the Adam optimizer for $G$ is 0.0002. The structures of the discriminators and generator are based on the DCGAN (Radford, Metz, and Chintala, 2015) (Note that there is only one discriminator in DCGAN and we use the DCGAN's discriminator's


Figure 6.3: Examples from the dataset office-31 which has three domains: Amazon, DSLR, and Webcam. It is visually attested that the Amazon domain is more different and therefore distributionally distant to the other two domains, while DSLR and Webcam are visually attested to be similar and therefore distributionally close to each other.

structure for both our discriminators). There is no weight decay for any of the three neural nets. Regarding the final classifier trained on the combination of the source training set, the target training set, and the fake images, its architecture is Inception v3 (Szegedy, Vanhoucke, et al., 2016; Szegedy, Ioffe, et al., 2017). We train on a NVIDIA A100 GPU. For each domain adaptation task, the number of generated images is empirically determined that resulting in the final classifier giving the best performance measured in accuracy scores.

**CIFAR-10 ↔ STL-10**

Table 6.4 demonstrates the comparison of the MiddleGAN against 5 state-of-the-art baselines in terms of accuracy: VADA (Shu et al., 2018), IIMT (Yan et al., 2020), Enforced Transfer (Gao, Baucom, et al., 2022), SE (French, Mackiewicz, and Fisher, 2017) and SEMA (Zuo et al., 2021). The Source Only algorithm indicates the performance of training a classifier on the source domain and directly applies it to the target domain without mitigating the domain shift. On the task of CIFAR-10 → STL-10, it outperforms the second best-performing algorithm, the Enforced Transfer, by 3.4%; on the task of STL-10 → CIFAR-10, it outperforms the second best-performing algorithm, SEMA, by 12.1%. The superiority of the MiddleGAN suggests that the fake samples are invariant to domain shift. We conduct the experiment on CIFAR-10 ↔ STL-10 to get a basic sense of the capacity of domain adaptation by the MiddleGAN when compared to the state-of-the-art algorithms. In later Sections such as 6.3, we delve into deeper reasons on why the MiddleGAN outperforms the state-of-the-art algorithms.

**Office-31**

In Table 6.2, we compare the MiddleGAN against ResNet-50 and 10 other state-of-the-art domain adaptation algorithms. Again, the metric is accuracy. Out of the six domain adaptation tasks, we achieve state-of-the-art performance on three of them. However, note that our improvement over the state-of-the-art algorithms is very significant: On the task D → A, we improve over the second best-performing algorithm Fixbi by 6.2%. On tasks that we do not outperform the state-of-the-art, the difference between the MiddleGAN's performance and the state-of-the-art's performance is usually small. For example, on the task of A → D, the state-of-the-art performance is 95.8% and we are only 1.7% off. Overall, the MiddleGAN achieves state-of-the-art performance on average, outperforming all the other baselines in comparison.

It is noted that the state-of-the-art algorithms do not achieve satisfactory results on tasks with a larger domain shift, such as D → A and W → A, as the best SOTA performance on D → A is an accuracy score of 78.7% and the best SOTA on W → A is an accuracy score of 79.4%. On the other hand, the MiddleGAN achieves an accuracy score of 84.9% on D → A and an accuracy score of 83.5% on D → A. The fact that the MiddleGAN succeeds where the others fail when the domain shifts are large indicates that the MiddleGAN has the potential to allow for domain adaptation (DA) on two domains that are more distributionally distant. In turn, this indicates that the MiddleGAN can be used for more DA tasks.

**Office-Home**

In Table 6.3, we compare the MiddleGAN against Resnet-50 and 9 state-of-the-art domain adaptation algorithms on the Office-Home dataset. Since there are four sub-domains in the Office-Home dataset, there are in total 12 domain adaptation tasks to be done. Out of the 12 domain adaptation algorithms, the MiddleGAN achieves state-of-the-art performance on 5 of them. When the MiddleGAN achieves state-of-the-art performance on a domain adaptation task, it usually outperforms the second best-performing algorithm by a large margin. For example, on the task of Pr → Cl, we outperform the second best-performing algorithm Fixbi by 20.1%. Overall, the MiddleGAN achieves state-of-the-art performance on average, which is an accuracy score of 75.3%, 2.6% higher than the second best-performing algorithm, Fixbi. In Table 6.3, we have found a similar observation as noted in Section 6.3: the MiddleGAN is able to outperform the state-of-the-art algorithms significantly on domain adaptation tasks whereas the domain shifts are large. For example, on the task Ar → Pr, we outperform the second best-performing algorithm by 9.6%. Similarly, on the task Ar → Cl, we outperform the second best-performing algorithm by 20.1%. Again, because of this observation that the MiddleGAN is able to outperform the state-of-the-art algorithms when the domain shifts are large, the MiddleGAN is more suited to perform domain adaptation tasks whereas the two domains are more distributionally dissimilar to each other. This, consequently, suggests that the MiddleGAN can be helpful when other state-of-the-art domain adaptation tasks cannot - the MiddleGAN can be helpful in performing domain

adaptation tasks from domains that are more dissimilar. As a result, the MiddleGAN can be applied to more domain adaptation tasks.

**MNIST ↔ SVHN**

| Algorithm | SVHN → MNIST | MNIST → SVHN |
|---|---|---|
| Source Only (French, Mackiewicz, and Fisher, 2017) | 66.5% | 25.4% |
| Reverse Grad (Bousmalis, Silberman, et al., 2017) | 73.9% | 35.6% |
| DCRN (Bousmalis, Trigeorgis, et al., 2016) | 81.9% | 40.0% |
| ADDA (Tzeng, Hoffman, Saenko, et al., 2017) | 76.0% | - |
| ATT (Ganin and Lempitsky, 2015) | 86.2% | 52.8% |
| SBADA-GAN (Ghifary et al., 2016) | 76.1% | 61.0% |
| Mean Teacher (French, Mackiewicz, and Fisher, 2017) | 99.2% | 97.0% |
| MiddleGAN | **99.5%** | **99.9%** |

Table 6.5: The results on the domain adaptation task of SVHN → MNIST and MNIST → SVHN.

In Table 6.5 we compare the MiddleGAN against six state-of-the-art algorithms and we observe that the MiddleGAN achieves the new state-of-the-art performance on both SVHN → MNIST and MNIST → SVHN: on the first task, it achieves an accuracy score of 99.5% and on the second task an accuracy score of 99.9%. Both scores are nearly 100%. Compared to the second best-performing algorithm, the Mean Teacher, the MiddleGAN only achieves an improvement of 0.3% on the first task. This is because there is not enough room for improvement, considering that the Mean Teacher already achieves an accuracy score of 99.2% on the first task. On the second task, the second best-performing algorithm the Mean Teacher achieves an accuracy score of 97.0%, and there is more room for improvement since it is not nearly 100%. As a result, on the second task, we outperform the Mean Teacher by 2.9%, a more significant improvement compared to our improvement on the first task. Note that, although our improvements on MNIST ↔ SVHN is not large, which is due to the fact that there is not enough room for improvement, we report the evaluation results on MNIST ↔ SVHN to show that we do not lose to the state-of-the-art algorithms.

## 6.4 Conclusion

We propose the MiddleGAN, a variation of GAN that generates these domain-agnostic samples. We have extended the theory of GAN to prove that there exist optimal solutions for the weights of the two discriminators and one generator in MiddleGAN. We have empirically shown that the generated samples are similar to both the source and target domain samples (domain agnostic). We have conducted extensive evaluations using 24 benchmarks; on the 24 benchmarks, we compare MiddleGAN against various state-of-the-art algorithms and outperform the state-of-the-art by up to 20.1% on certain benchmarks. One novelty of this thesis is that we have observed in the Evaluation Section (Section 6.3) that the MiddleGAN significantly outperforms the state-of-the-art domain adaptation algorithms on tasks whereas the domain shifts are larger. This indicates that the MiddleGAN can be

applied to tasks where other state-of-the-art algorithms fail at successful domain adaptation. In other words, the MiddleGAN can achieve good performance on domain adaptation tasks whereas the two domains are more distributionally dissimilar, meaning that there is a wider range of applications/tasks that the MiddleGAN can be applied.

*Chapter 7*

# REAL DEPLOYMENTS IN WHICH REALISMS ARE PRESENT

To demonstrate the value of machine learning-based smart health technologies, researchers have to deploy their solutions into complex real-world environments with real participants. This gives rise to many, oftentimes unexpected, challenges for creating technology in a lab environment that will work when deployed in real home environments. In other words, like more mature disciplines, we need solutions for what can be done at development time to increase success at deployment time. To illustrate an approach and solutions, we use an example of a project that is a pipeline of voice-based machine learning solutions that detects verbal conflicts of the participants. We call it the Patient Caregiver System (PCR). PCR is a smart health technology because, by notifying the participants of their conflict, it encourages the participants to better manage their emotions. This is important because being able to recognize one's emotions is the first step to better managing one's anger. PCR was deployed in 6 homes for 4 months each and monitors the verbal conflict of the caregiver of a dementia patient. In this chapter, we re-confirm the claims demonstrated in Chapter 3 and 4: if you are aware of the realism (in this case, the acoustical environmental distortion) beforehand, integrating the realism (environmental distortion) into the training samples can ensure deployment time success.

## 7.1 Introduction

Smart health research teams often develop novel machine learning technologies. To prove the value of the technologies, the research teams have to deploy the solutions into a complex environment such as a smart home (Bedón-Molina, Lopez, and Derpich, 2020; Costin et al., 2009; Khan and Chattopadhyay, 2017; Gao, Ma, et al., 2020). However, the aforementioned works do not describe the problems related to the transition from the development stage to the deployment stage. In other words, during the development stage, the research teams try to develop their solution in an environment, usually a lab environment and/or a controlled home environment, which is less complex than the environment in which the technology is going to be deployed. It is well known that when new technology is actually deployed in real complex environments, issues previously unseen in less complex environments are going to occur, especially for long-term deployments. this Chapter describes the conflict detection classifier developed in 4 when it is integrated into the PCR System, a pipeline of voice-based machine learning solutions for in-home monitoring of verbal conflict experienced by a caregiver of a dementia patient. The conflict detection model, as well as the PCR System, is considered a smart home technology because it helps caregivers of dementia patients better manage their anger by notifying them when verbal conflict is detected.

The main challenge of integrating novel voice-based machine learning technology into complex environments is that the physical environment is complicated. For a smart health technology that uses voice for emotion detection (GAO et al., 2021), the environment is complicated because the solution needs to address the problems of acoustic signal's deterioration due to background noise such as birds chirping, room reverberation due to the signal bouncing off the surface of furniture, and deamplification as a result of the distance between the human speaker and the microphone. We call these environmental distortions acoustical realisms.

The gist of our solutions at pre-deployment time to ensure maximizing the post-deployment success of the machine learning algorithms in the PCR System is that we integrate previously known realism (acoustical environmental distortions) into the deep learning model (the conflict detection model). Using the PCR System as an example, the (acoustical) samples that the algorithms in the PCR System are about to encounter in its designated environment (a participant's home) are voice samples that are environmentally distorted. Note that in the PCR System, in addition to the conflict detection model, we also have the voice activity detection (VAD) model that filters out non-speech clips for the conflict detection model and the speaker identification model that filters out speech not produced by the registered participants for the conflict detection model.

It is worth noting that some works, such as Chen et al. (Zeya Chen, Mohsin Y Ahmed, et al., 2019a), do consider the designated environment in which their algorithm is going to be deployed during the pre-deployment stage. However, they do not evaluate if their hypothesis that the algorithm *indeed performs well during post-deployment time* holds true. Unlike works such as Chen et al. (Zeya Chen, Mohsin Y Ahmed, et al., 2019a), we perform evaluations to show that after our rigorous pre-deployment time assessment of the algorithms to be deployed in the designated environment, our chosen algorithms indeed perform well during the post-deployment stage when they are being deployed in the designated environment (the home of a patient-caregiver dyad), which is one novelty of this thesis.

The contributions of this Chapter are based on the in-home deployed PCR system from **six completed 4-month deployments of real caregiver-Alzheimer's patient interactions**. This work was performed under an approved IRB. The main contribution is:

- We confirm the pre-deployment approach for resulting in novel deep learning models deployment time success: to achieve this, we need to incorporate known realism into the training samples of these models.

In this section, we describe the PCR System, which includes the voice activity detection (VAD) algorithm, the speaker identification (SID) algorithm, and the conflict detection algorithm. Two out of the three algorithms are off-the-shelf, and note that the purpose of these two are to filter out unwanted/ineligible audio clips for the conflict detection model. The VAD algorithm is Google's transcription services,

which filtered out audio clips that are not human speech. The SID algorithm is the WavLM algorithm developed by Microsoft (S. Chen et al., 2022), which makes sure that the conflict detection model only looks at audio clips by registered speakers. We developed the conflict detection by ourselves and a detailed description of the algorithm can be found in Chapter 4, and we demonstrate using the conflict detection model that if we know the realism (acoustic environmental distortion) beforehand, we will be able to deal with them during the development stage. We also demonstrate in this Chapter that because we have dealt with the acoustic environmental distortion beforehand, our conflict detection model performs well on the six dyads that we evaluate the conflict detection model on (see Section 7.2.

In later sections of this thesis, we thoroughly tested the off-the-shelf algorithms to make sure that they meet our needs in the PCR system (i.e. they are good candidates for filtering out unwanted/ineligible clips for the conflict detection model).

Now we briefly describe the PCR system that the conflict detection model is a part of. The microphone placed in a central place in a room constantly listens to the ambient environment. The audio stream is sliced into 5-second audio clips and send to the voice activity detection (VAD) model to decide if a given clip contains human voice. The choice of each audio clip being 5-second is based on the observation from previous works that 5-second is long enough to be indicative of the speakers' emotions (GAO et al., 2021). The PCR System discards those audio clips that are invalid; i.e., they contain no human voice. The valid audio clips are sent to the speaker identification (SID) model to detect if the audio clips contain the voice of registered speakers. If yes, they are sent to the conflict detection models.

## 7.2 Evaluations

Many speech processing works have collected or augmented datasets with real world sounds. Good solutions are usually then developed. But, in many cases the resultant datasets have limited in-the-wild sounds so where they actually work is limited, and many times the solutions are not validated in-the-wild, but only on the datasets.

In this Section, we evaluate the conflict detection model's performance at the pre- and post-deployment stages. However, recall that we need the VAD model to filter out non-speech audio samples, and the SID model to filter out samples that do not belong to the registered speakers; therefore we must ensure their effectiveness as well. As a result, before we evaluate the conflict detection model's performance, we evaluate the VAD and SID models' performance at the pre- and post-deployment stages.

### Voice Activity Detection

The VAD model filters silence and other sounds that are not produced by the human vocal tract. Since the acoustic system is constantly listening to the environment, we do not want to activate the conflict classifiers when an input sound window contains no human speech. As a result, the VAD model is an important and necessary component in the PCR System. In this Section, we describe testing on the VAD

| Existing SOTA VAD | Accuracy |
|---|---|
| VQVAD | 45.66% |
| Sohn et al. | 31.21% |
| Kaldi Energy VAD | 11.72% |
| DSR AFE | 40.07% |
| rVAD | 66.23% |

Table 7.1: Evaluation of existing VAD algorithms on the Aurora-2 database. This Table is reported by Tan et al. (Tan, Dehak, et al., 2020).

model for the question: Is it possible to perform comprehensive and realistic pre-deployment testing to improve post-deployment success? Note that the acoustic realisms that the VAD model faces are deamplification, reverberation, and (non-speech) background noise, so in our pre-deployment stage assessment we seek to find a VAD solution that is robust against these three types of acoustical realisms.

**Pre-deployment Stage Assessment**

During pre-deployment time, we first looked into several state-of-the-art VAD algorithms. In particular, we studied the performance of a set of SOTA VAD algorithms on the Aurora-2 database (Hirsch and Pearce, 2000). The performance of the algorithms on the Aurora-2 database is a good indicator of how they might perform in the real world because Aurora-2's speech samples are mixed with noise collected from realistic settings such as streets, airports, and cars. Table 7.1 is a list of existing SOTA VAD algorithms' accuracy scores on the testing set of Aurora-2. Unfortunately, as we can see, the highest-performing one is rVAD, which only achieves an accuracy score of 66.23%, which is far from being usable in the real world. In other words, there exists solutions in the literature that do not work on datasets with deamplification, reverberation, and background noise. It is good to discover that these solutions are not likely to work at post-deployment time, because this helps us filter out existing solutions so that we won't use those solutions.

However, there was another algorithm, the Google Speech Recognition (GSR) algorithm, that had not been evaluated on a dataset that contained the three environmental distortions: reverberation, deamplification, and background noise. As a result, we next evaluated the Google Speech Recognition solution. we aimed to evaluate it in a comprehensive way to demonstrate that it would be robust against environmental distortions such as reverberation, deamplification, and background noise. Again, this is to set up the necessary condition to prove our hypothesis that a solution, during the pre-deployment stage, must be able to deal with the unique challenges given the real, designated environment in which it will be deployed.

To do so, we collected a dataset that contains diverse environmental distortions: first, we collected the clean samples - samples that are not environmentally distorted, by

| Event | Instances |
|---|---|
| (object) rustling | 60 |
| (object) snapping | 57 |
| cupboard | 40 |
| cutlery | 76 |
| dishes | 151 |
| drawers | 51 |
| glass jingling | 36 |
| object impact | 250 |
| people walking | 54 |
| washing dishes | 84 |
| water tap running | 47 |

Table 7.2: Events that are present in the background noise collected from real homes from the dataset (Mesaros, Heittola, and Virtanen, 2016). All of them are covered in the process of contaminating audio samples with background noise. Note that this list do not include sounds from the tv, which are very important to make sure the robustness of the emotion detection model and conflict detection model.

having an individual talk next to the microphone for 5 minutes. The 5-minute clip was then sliced into 60 5-second segments, each of which is individually labeled as positive if it contained a human voice, or negative if it did not contain human voices. Note that the individual took long pauses intentionally to make sure that there were negative samples. Second, we collected the audio clips that were deamplified and contained background noise. To do so, we took copies of the clean samples. For each of the copies, we randomly deamplify them by m decibels ($0 < m < 12$) as per the practice of a previous work on emotion detection (GAO et al., 2021). Then, we randomly picked household sounds from the household ambience dataset (Mesaros, Heittola, and Virtanen, 2016). Table 7.2 lists the events that occur in the dataset. Note that each of the ambience sounds is greater than 5 seconds, so we randomly picked a segment from it that was 5-seconds long, and overlaid it with a deamplified clip. We repeated this process for all 60 deamplified clips. Third, we created the data for reverberated speech. To do so, we took another set of copies of the clean samples, and overlaid each of them with reverberation that was described by the combination of the three parameters: the wet/dry ratio $r$, diffusion $d$, and decay factor $f$. Finally, we created samples that are deamplified, noise-contaminated, and reverberated. To do so, we took a set of copies of the 60 deamplified and noise-contaminated samples, and overlaid them with the same reverberation effect as the samples that only contained reverberation effect and nothing more. In the end, we have 60 clean samples, 60 deamplified and noise-contaminated samples, 60 reverberated samples, and 60 samples that had all three environmental distortions. As a result, we claim that we created a dataset that was comprehensive enough to account for all three kinds of environmental distortions.

| | Dyad 1 | Dyad 2 | Dyad 3 | Dyad 4 | Dyad 5 | Dyad 6 |
|---|---|---|---|---|---|---|
| GSR | 100% | 100% | 94.0% | 95.0% | 100% | 98.0% |

Table 7.3: The evaluation results for the voice activity detection model on the dyads. The high accuracy scores achieved from the dyads indicate that the VAD algorithm (Google Speech Recognition) is highly effective at differentiating non-speech from human speech audio samples.

We evaluated GSR on the dataset that we just created. **GSR achieved an accuracy score of 95.83%**, correctly classifying 230 out of the 240 samples each of which accounted for the environmental distortions to a certain degree. The high performance led us to decide to deploy GSR as our VAD model since, during the pre-deployment stage assessment, it is shown to be robust against the challenges that it is about to encounter in the real, designated environment: reverberation, background noise, and deamplification.

**Post-deployment Stage Assessment**

Using post-deployment data on six completed dyads, we validate how well the chosen solution worked in practice. Table 7.3 shows the evaluation results of the VAD model on the dyads. We randomly select samples generated by each dyad during their deployment, and have human labelers label them if they are of human speech or not. We obtained 100 samples for all the dyads. The high performance of the VAD model indicates that this part of our system is highly effective at filtering out non-human speech samples such as background music (without lyrics) and footsteps. It is noted that the VAD does not filter out TV sounds if there is human speech in the sounds, such the voices of actors or news anchors. These unwanted sounds are filtered by the next model, SID.

The VAD model achieves an accuracy score of 94.0% to 100% on the six dyads. The high performance on post-deployment data validates our choice of the Google Speech Recognition in the pre-deployment phase. This implies that this VAD algorithm was originally made very robust to real world complexities. The high performance also indicates that, in order for the deployment to be successful, smart health groups using audio should perform pre-deployment tests with comprehensive real-world distortions. In addition, the high performance suggests that our hypothesis holds true - recall that our hypothesis is that, during the pre-deployment stage assessment, an about-to-be-deployed algorithm must be proven to overcome the challenges that are perceived to be present in the real, designated environment in order for it to perform well in said environment. The high performance on post-deployment data also indicates that it is sometimes possible to perform comprehensive and realistic pre-deployment testing to improve VAD post-deployment success.

**Speaker Identification (SID)**

The SID model determines the identity of a speaker. The SID is a crucial part of the PCR System because we only want the voices of the registered speakers to be sent to conflict detection models. However, in real deployments, voices from the TV and visitors must be filtered out. In this Section, we test SID model to answer this question: is it possible to perform comprehensive and realistic pre-deployment testing to improve post-deployment success? Note that the acoustical realisms that the SID faces, in addition to (non-speech) background noise, reverberation, and deamplification, also include sounds from the tv such as the dialogues from tv characters, for the presence of another person's voice in an audio sample could confuse the speaker identification model.

**Pre-deployment Stage Assessment**

During pre-deployment time, we investigated a state-of-the-art SID algorithm, the Google Speaker Identification API. However, the API asks us to input the maximum number of speakers there can be in a clip. This is impractical because a dyad can have the TV on and there could be many people's voices from the TV, or there may be multiple visitors. It is good to discover that this solution is not likely to work at post-deployment time, because this helps us filter out existing solution(s).

Now we describe how we test to make sure the about-to-be-deployed SID algorithm developed by Microsoft (S. Chen et al., 2022) is robust to environmental distortions such as reverberation, background noise, and deamplification. Again, this is to verify our hypothesis that for an algorithm to be successful in the real, designated environment, it must be able to overcome the challenges present in the real, designated environment during the pre-deployment stage. In our case, the challenges are reverberation, deamplification, non-speech background noise and TV sounds. Specifically, we have two persons, P1 and P2, each of whom spoke next to the microphone for 2.5 minutes. Then, for each of their voice files, we sliced it into 28 audio samples. Because these 56 (28×2) samples were collected when the speakers were right next to the microphone, they were considered clean speech, free of the three types of environmental distortions. We needed to craft environmentally distorted samples out of the 56 clean samples to ensure that the testing samples accounted for both clean and environmentally distorted samples. To do so, we copy each of the 56 clips and deamplify them by randomly choosing a real number between 0 and 12 decibels. Then, we randomly chose a noise clip from Table 7.2 as well as TV sounds we recorded using a microphone, out of which we randomly chose a consecutive 5-second segment to be overlaid with one of the copies. This guaranteed samples that were deamplified and contaminated with noise, and the last step was to reverberate it. Again, the reverberation effect is described by the three parameters: the wet/dry ration $r$, diffusion $d$, and decay factor $f$, as per the practice of a previous work (Salekin et al., 2017). When we reverberated a (noise-contaminated and deamplified) copy, the values of $r$, $d$, and $f$ are randomly chosen. In total, we had 112 samples, 56 of which belonged to P1 and the other 56 belonged to P2. We fed

the 112 samples to our SID model. **The SID model achieves an f1 score of 85.7% on P1 and 92.8% on P2**. The high performance of the SID model led us to believe that it was reasonably robust to reverberation, deamplification, background noise, and TV sound.

### Post-deployment Stage Assessment

To validate post-deployment success, from all audio samples that our speaker identification algorithm identifies to contain the voice of the caregiver or the patient, or both, we randomly chose 28 from the first dyad, 28 from the second dyad, and 28 from the third dyad, 100 from the fourth dyad, 80 from the fifth dyad, and 100 from the sixth dyad. The results are reported in Table 7.4. In the following sentences we describe how we obtain the f1 scores in Table 7.4. For a sample, if it only contains the voice of the caregiver, then it is labeled as belonging to the caregiver; it if only contains the voice of the patient, then it is labeled as belonging to the patient. If it contains voices from both the caregiver and patient, then it is labeled as belonging to both. Otherwise, it labeled as belonging to neither. With this labeling scheme, we obtain the positives and negatives of the caregiver's voice and the positives and negatives of the patient's voice. The SID model can label a sample as belonging to the caregiver, belonging to the patient, or neither. As a result, we obtain the results in Table 7.4 in which we report the f1 scores to measure the performance of our SID model for both the caregiver and patient of each dyad. The SID model achieves an f1 score in the range of 93.1% to 97.4% for the caregivers and 91.6% to 98.3% on the patients in the six dyads. The high performance in Table 7.4 indicates that our SID algorithm is effective at picking out the voices by the caregiver and the patient in each home in their real home environment. Given that the SID algorithm was specifically assessed to see if it could overcome the challenges (reverberation, deamplification, and background noise) present in the real, designated environment (homes), we have shown that for an algorithm to be successful in the real, designated environment, it must be able to overcome the challenges present in the real, designated environment during the pre-deployment stage. The high performance of the SID during the post-deployment time suggests that our way to perform comprehensive and realistic pre-deployment is effective at improving post-deployment SID success. Note that we only validate the SID solution on the voices of the caregiver and patient of each dyad, because at post-deployment time, the SID solution filtered out voice samples that belonged to neither. As a result, we only have samples that are labelled by the SID solution as either the caregiver or the patient. For samples that made through the SID solution, we have the performance reported in Table 7.4.

In Table 7.5 we report the f1 score of a model (LeCun, Bengio, et al., 1995) that we did not use because at pre-deployment time it achieves bad performance (an f1 score of 79.3% on P1 and an f1 score of 71.2% on P2). As we can see, this model also achieves bad performance on the post-deployment data. This indicates that at pre-deployment time, the model that performs badly also performs badly at post-deployment time.

| | Dyad 1 | Dyad 2 | Dyad 3 | Dyad 4 | Dyad 5 | Dyad 6 |
|---|---|---|---|---|---|---|
| Caregiver | 94.5% | 97.4% | 95.7% | 93.1% | 92.0% | 89.2% |
| Patient | 94.6% | 95.9% | 96.0% | 94.6% | 91.6% | 98.3% |

Table 7.4: The evaluation results for the speaker identification model on the dyads. The results are the f1 scores.

| | Dyad 1 | Dyad 2 | Dyad 3 | Dyad 4 | Dyad 5 | Dyad 6 |
|---|---|---|---|---|---|---|
| Caregiver | 57.1% | 71.4% | 83.6% | 52.6% | 44.3% | - |
| Patient | 74.8% | 75.9% | 79.5% | 77.7% | 74.2% | 97.0% |

Table 7.5: The post-deployment evaluation results for a speaker identification model that **we did not use** because it achieved bad performance pre-deployment time. As we can observe, its performance on all dyads is bad at post-deployment time.

### The Conflict Detection Model

For conflict detection, currently, there is no available conflict detection algorithm that is acoustics-based. Therefore, we developed our own conflict detection algorithm. In this Section, we aim to test on the conflict detection model: is it possible to perform comprehensive and realistic pre-deployment testing to improve post-deployment success?

### Pre-deployment Stage Assessment

Here we briefly describe how the new algorithm we developed is trained and why the training process makes it specifically account for the (three) challenges that arise in the real, designated environment in which the algorithm is going to be deployed. The training and testing samples are from 19 couples and each sample is labeled conflict if the content of the sample indicates that the couple are in a verbal conflict. It is labeled non-conflict if the couple are not in a verbal conflict. Since the samples are already collected from home-environments, de-amplification and reverberation are accounted for, but the samples are free of background noise. As a result, we mix each of the samples with background noise by randomly selecting a segment from a randomly chosen indoor background noise sample in Table 7.2 and overlaying that segment with each sample. Out of the samples, there are 3,072 in the training set and 1,009 in the testing set. As a result, both the training and the testing set accounts for a variety range of indoor environmental distortions. Since the training samples are touched by deamplification, reverberation, and background noise, our conflict detection algorithm trained on them is designed to be able to handle the three challenges (deamplification, reverberation, and background noise).

The conflict detection model's performance on the testing set achieves an f1 score of 93.1%. The high performance suggests that the conflict detection is robust against environmental distortions such as reverberation, background noise, and deamplifi-

| | Dyad 1 | Dyad 2 | Dyad 3 | Dyad 4 | Dyad 5 | Dyad 6 |
|---|---|---|---|---|---|---|
| Ours | 63.4% | 65.9% | 70.7% | 86.2% | 82.7% | 90.1% |

Table 7.6: The evaluation results for the conflict detection model. The measurement is f1 score.

cation. This help us set the stage to prove our hypothesis that, for an algorithm to work sufficiently in the real, designated environment in which challenges are perceived to be present, the algorithm must show that, during pre-deployment stage, it is able to handle the challenges. Our conflict detection algorithm has indicated that during pre-deployment stage, it is able to handle the three challenges: reverberation, deamplification, and background noise. Note that we do not include TV sounds as one of the acoustic realisms that the conflict detection model needs to address. In the future, we plan to develop a conflict detection algorithm that takes TV sounds into consideration.

**Post-deployment Stage Assessment**

In the post-deployment time, we seek to prove our hypothesis that, for an algorithm to work well post-deployment time in the real, designated environment in which it is going to be deployed, during pre-deployment stage it must show that it is capable of overcoming the challenges that are present in the real, designated environment. Our conflict detection algorithm has showed that it is capable of overcoming the challenges (it achieves an f1 score of 93.1% pre-deployment time). However, is it going to work well in the post-deployment time?

Table 7.6 shows our conflict detection algorithm's performance during the post-deployment time at the six homes. Now we explain how we obtain the f1 score results in Table 7.6. If a clip is labeled by the labelers such that it contains verbal conflict and the classifier also thinks this clip contains verbal conflict, then it is a hit. If the clip is labeled by the labelers as not containing verbal conflict and the classifier also thinks that it does not contain verbal conflict, then it is a hit. All other cases are misses (for example, the labelers think that a sample contain verbal conflict but the classifier fails to classify it as so). By looping through all samples produced by a dyad, we produce an f1 score on that dyad. From Table 7.6, we observe that the sixth dyad achieves the best performance with an f1 score of 90.1% while the first dyad achieves the lowest performance with an f1 score of 63.4%. For each of the dyads, we observe a drop in performance compared to 93.1% obtained when the same model is evaluated on the dataset containing speech samples from the 19 couples. This indicates that despite our effort in mitigating environmental distortions, the effects of the environmental distortions such as room reverberation, background noise, and the deamplification effect are not fully mitigated. But the relatively satisfactory performance of the conflict detection model on dyads 4, 5 and 6 indicates that our way to perform comprehensive and realistic pre-deployment

testing to improve post-deployment success is effective for exapected conditions.

We also investigate why the performance of the conflict detection model is lower in dyads 1-3 (f1 score of 63.4% to 70.7%). Upon communicating with the dyads, we learned that dyad 1 moved the system (which included the microphone) to the hallway which is very far away from the usual places that the participants were speaking. Dyad 2 had a construction team rennovating their home, so there was a lot of construction noise to confuse the conflict detection model. When we developed the conflict detection model, we did not take construction noises into consideration. In dyad 3, the caregiver's voice was always very low, almost inaudible, and our conflict detection model was not designed to handle such low-to-inaudible voice samples.

**Summary**

In this paragraph we briefly summarize our findings: To ensure post-deployment success of an algorithm, during pre-deployment time it must be rigorously tested on samples that are touched by the challenges that are perceived to be present in the real, designated environment during post-deployment time. In other words, the deep learning models, in order to do well during the deployment stage, need to be evaluated on samples that have realism in them. The success of the conflict detection model post-deployment time suggests that it can successfully handle the realism (acoustic environmental distortion), because during its training we have incorporated the realism within the training samples.

## 7.3 Conclusion

In this Chapter, we confirm a previous claim in this thesis: if the realisms are known beforehand before the development of the deep learning models, we can incorporate the known realisms into their training samples, which results in deep learning models robust against these realisms. We use the conflict detection classifier we developed in Chapter 4, whose training samples contained the known realism of acoustic environmental distortion, to prove the claim. It is integrated into an acoustic system called the PCR System, which contains two more off-the-shelf acoustic processing components to make sure that only wanted/eligible clips are sent to the conflict detection model. We deployed the PCR system in six homes, each deployment lasting for four months. The conflict detection model's post-deployment success across the six dyads indicates that we have successfully anticipated the realism to be encountered in smart homes and our model, developed under our philosophy, is good at handling the realism.

*Chapter 8*

# DEPLOY DEEP LEARNING MODELS IN THE ONSET OF COVID

In this Chapter, we describe one problem of deploying deep learning models, which are the components in the acoustic pipeline in the Patient-Caregiver Recommendation System. Note that the Patient-Caregiver Recommendation System is short for PCR as well, **which is not to be confused with the Patient-Caregiver Relationship System** described in Chapter 7. The problem being that amidst the onset and first years of COVID, in-person contact was not allowed. As a result, we had to rely on the participants who agreed to have the DL models deployed in their home environments (CPS systems) to set up the system that contains the deep learning models.

To overcome the *challenge* of continuing in-home studies without in-person contact, we developed a collection of techniques for out-of-the-box deployments usable by the general public. In this article, we examine the feasibility and practicality of developing out-of-the-box deployments as applied to a study of Alzheimer's patient–caregiver dynamics in home settings. We describe the obstacles that we solved and the lessons learned from the effort. We believe that our out-of-the-box deployment solution will also help other research studies that require in-person contact.

Typically, for these types of in-home research systems, technical experts deploy the system. This limits the geographical location of deployments to be near the experts. A major value of the solutions described in this chapter is that there is no geographical limitation for finding participants since the system is simply mailed to them. All the deployments described in this thesis were conducted in this out-of-the=box manner.

## 8.1 Introduction

Research work in many fields was affected significantly due to the COVID-19 pandemic (Saberi, 2020), especially the research tasks needed to be deployed in households, such as these three works (Brush et al., 2011; Gao, Ma, et al., 2020; Spruijt-Metz et al., 2016). Equipment deployments are usually performed by research teams and the participants do not have to set up or fix equipment problems themselves. However, COVID-19 curtailed new deployments and ongoing deployments with in-person contacts. A major challenge is how can such in-home studies continue without in-person contact.

To date, approaches adopted by researchers to overcome the challenge that no contact is allowed during traditionally contact-permitted activities, include, but not limited to teaching in classrooms and collecting data. Martin et al. (Martin et al., 2020)

described the challenges faced by dermatology students participating in a class along with possible remedies such as using online tools. However, conventional online like video conferencing cannot replace some practical aspects of education such as laboratory experiments. To mitigate this shortcoming, smartphones and augmented reality techniques have been suggested to monitor students' virtual experiments. An additional advantage of the suggested virtual laboratory can collect physiological data for educational purposes (Lellis-Santos and Abdulkader, 2020). Data collection for experimental purposes has to change due to the pandemic. Hensen et al. (Hensen et al., 2021) suggest using mobile phones and online platforms to collect data instead of using traditional in-person methods during the pandemic.

To overcome the challenge of continuing in-home studies without in-person contact, we developed a collection of techniques for out-of-the-box deployments usable by the general public. In this Chapter, we examine the feasibility and practicality of developing out-of-the-box deployments as applied to a study of Alzheimer's patient–caregiver dynamics in home settings. We describe the obstacles that we solved and the lessons learned from the effort. We believe that our out-of-the-box deployment solution will also help other research studies that require in-person contact.

In more detail, the overall challenges that we have encountered are:

- COVID-19 permits no contact. The research team is no longer able to physically meet the participants. Also, face-to-face meetings among the entire research team are also limited, because it is not advisable for research team members to meet (frequently) either, which potentially slows down the research study. The quarantine demands that participant training must be virtual and online.
- Participants are not knowledgeable in the technology of the deployed system. Our participants are dyads of persons with dementia and their informal family caregivers. Dementia mostly affects older adults and 30% of caregivers are also over the age of 65, hence study participants may have limited familiarity with technology (Association, Thies, and Bleiler, 2013).
- The system is complex and setting it up requires training. Participants must complete the system setup, including the installation of a laptop, smartphone, microphone, and router, all without in-person contact. Participants must also provide voice samples so we can perform speaker identification. Additionally, study procedures include the use of mindfulness-based stress management techniques, and participants must be trained in these techniques prior to initiation of study procedures.
- Logistics and Budget. New EMA surveys and updates to IRBs, documentation, budgets, and logistics are required.

The main contributions of this work are as follows:

- We provide a set of solutions for successful no-contact out-of-the-box deployments as a useful experience to other research teams facing similar challenges.
- Our evaluation demonstrates that our out-of-the-box solutions are effective in enabling research study in-home deployments without any in-person contact. In other words, our techniques can allow ongoing studies even when no contact is permitted.
- We present key lessons learned from our experiences: First, the deployment techniques provide an added degree of robustness, which improves the initial deployment process of a complicated in-home system (the participants were successful and less frustrated with setting up the system). Second, the Zoom and TeamViewer combination is able to overcome the more technical and difficult aspects of having dyads deploy a system by themselves.
- It is now possible to recruit participants irrespective of geographical location thereby increasing the potential of finding participants.

Our evaluation consists of seven out-of-the-box deployments performed in three stages. There were two participants from Stage 1, two participants from Stage 2, and three participants from Stage 3. In stage 1, the first two deployments were performed by skilled technical people as a first trial to identify needed improvements. Based on their feedback, we made changes to the out-of-the-box deployment solution. Stage 2 had three deployments with nontechnical individuals, one elderly and two middle-aged. We made changes based on the feedback from stage 2. Finally, stage 3 had three elderly people perform the deployments.

## 8.2 The Patient Caregiver Recommendation (PCR) System

In this section, we briefly describe the PCR System in order to provide the context for the updates we made for an out-of-the-box deployment. We stress that the goal of this study is to assess the out-of-the-box deployment protocol, not the PCR system itself. This section explains the major components (the acoustic pipeline, the recommendation system, the EMA, and M2G) so that the deployment approach can be understood in context. The details of the out-of-the-box solution are in the following section.

### Overview

The PCR system is deployed in homes with a family caregiver and an Alzheimer's patient. The hardware consists of a laptop, an external microphone, a router, and a smartphone. Using a microphone, the system detects affective states of the caregiver and reduces their stress by presenting learned, personalized stress reduction recommendations. To serve as the backdrop to the changes needed to handle COVID-19 restrictions, we briefly describe the system. The PCR system consists of four major components, as shown in 8.1. In addition to the major components, we also upload real-time data and logs to the cloud.

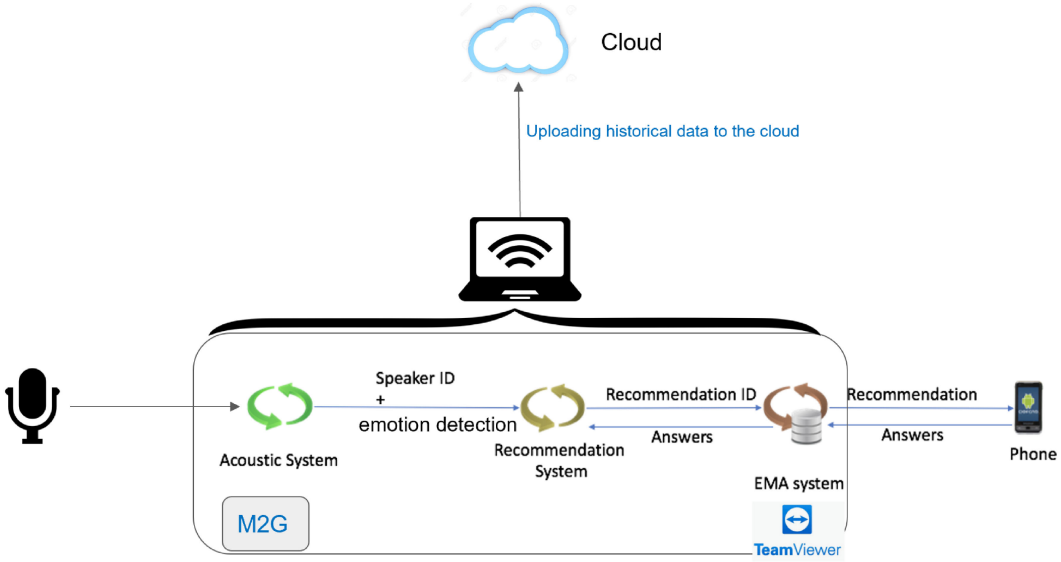



Figure 8.1: The Overview of the PCR system with its four major components: the Acoustic Pipeline, the Recommendation System, and EMA system, and M2G.

**Acoustic Pipeline**

The acoustic pipeline monitors the vocal interaction between the caregiver and patient and recognizes the caregiver's mood. When our system is on during awake hours, the microphone constantly listens to the ambient environment. The incoming data stream is sliced into non-overlapping five-second audio windows. For each window, we drop the segment if it is silent.

If there is a sound, we apply a robust voice activity detection (VAD) module to determine if there exist discernible segments of speech. After the VAD module classifies that an audio window contains speech, the audio window is passed to the speaker identification (SID) model to identify if the speech is from the caregiver, the patient, or another speaker (including speakers on TV). The SID model is pretrained using the voice of the caregiver and patient. If the SID model decides that a particular audio window contains speech by the caregiver or the patient, this audio window is sent to a CNN-based emotion detection model. The model has five output classes: happiness, anger, neutrality, sadness, and fear/disgust. If the model classifies a sample as angry speech, it notifies the recommendation system.

**Recommendation System and the EMA**

The goal of our recommendation system is to increase the mindfulness skills of caregivers. Randomized control trials indicate that brief psychoeducation on mindfulness and self-guided practice using online exercises significantly reduce depression and anxiety, and a brief intervention involves training in mindfulness and ecological momentary assessment strategies. The PCR system crafts four stress management techniques: 1) emotion regulation and 2) time-out techniques, as well as 3) brief mindfulness training, and 4) environment modification techniques to

increase emotional acceptance, as our recommendation candidates. PCR learns to adapt recommendations based on the monitored acoustic events and caregiver's feedback on previous recommendations via federated learning based on a contextual bandit algorithm. We consider time of the day, category of the recommendations, and detected acoustic events as context for recommendation generation. To deliver recommendations to the caregiver, we utilize an Ecological Momentary Assessment (EMA) system (the software of EMA is Nubis developed by USC9). The EMA is installed on a workstation deployed in the dyads' homes, which connects the acoustic monitoring system, the recommendation system, and an EMA app on a smartphone to send recommendation messages to caregivers. This feedback is used to update the estimation of recommendation effectiveness for future improvement. To ensure the execution of these stress management techniques by the caregivers, we provide them with an instructional handout and brief training before the deployment of the system.

The recommendation system is backed by a contextual bandit algorithm, which is designed to handle cold start in recommendations. Specifically, the algorithm adapts its recommendation policy based on users' feedback over time: from nearly random recommendations (i.e., exploration) to precisely calculated ones (i.e., exploitation). The algorithm is able to quickly find the most effective recommendation under each given context (e.g., detected emotional state).

Our EMA has two periods: the baseline period and the recommendation period. The baseline period lasts for four weeks, while the recommendation period lasts until the end of the deployment (4 months). During the baseline period, the EMA is triggered by acoustic events, such as angry voices from the participants, and asks the participants to confirm if they are angry. This is to provide ground truth for us to evaluate the emotion detection model. In the baseline period, we also randomly recommend recommendations, such as mindfulness techniques to the participants and later ask for the effectiveness of the techniques at the calming effect. The participants' response to such questions help us estimate recommendation effectiveness, so that in the recommendation period, we recommend items that are most effective at helping the participants calm down. The detailed technique backing up our recommendation module is an upper confidence bound based contextual bandit algorithm. As the purpose of this Chapter is not to explain this specific module, and due to space limit, we decided to withhold the technical details. Also, due to the purpose of the article being evaluating the deployment protocol instead of the PCR system, we do not provide how the participants reacted to the randomized recommendations during the baseline period. The participants are aware that the EMA has two periods (the cold-start, baseline period in which the contextual bandit algorithm learns and the recommendation period).

**Monitoring Support**

M2G (Ma et al., 2017) is a real-time and automated system for operation monitoring and system ground truth validation of research-oriented residential applications. PCR installs M2G to monitor the operation of devices and subsystems, including the

processes, files, device battery levels, disk memory, connectivity of the microphone and smartphone, and the cloud server. It sends notifications to remote administrators and other personnel to report any dysfunction or inaccuracy of the system in real-time.

**Other Components**

The system also includes deployment time support software that tests the operation of each of the system components to ensure the initial correct operation of the system. This testing would be controlled by technical members of our team when, prior to COVID-19, we were allowed to go into people's homes. The TeamViewer software (*The Remote Connectivity Software* 2023) is also installed on the laptop and smartphone to allow remote monitoring and updates or corrections as needed.

## 8.3 Out-of-the-box Deployment Solution

**Deployment Preparation**

To create a user-friendly out-of-the-box deployment experience, the study team premarked study equipment at all connection points (for example, the charging port of the phone and the wire used to charge the phone were labeled with tags of the same color). To minimize setup requirements at the time of out-of-the-box deployment, the team preconnected portions of the system that would not create equipment destruction during shipment to participants' homes. For example, the ethernet cable and router power cord were preconnected to the study router prior to shipment. The system included connection points for power supply to the laptop, router, and smartphone. Additional connection points included, "microphone to USB cable to study laptop" and "study router to ethernet cable to participant home router." The study team also marked power buttons on the study laptop and smartphone to increase ease of use. Each connection point was given a label (microphone, power port, etc.) and designated with a different color. The labels and color scheme were incorporated into the step-by-step out-of-the-box deployment instructions given to the participants

**Deployment Instructions**

Prior to the pandemic, research team members on the system development side created deployment instructions to be used internally by team members on the patient-caregiver relation side during system deployment in participant homes. To facilitate out-of-the-box deployment by study participants, the team revised deployment instructions to target older adult audiences. More technical descriptions were rewritten using layman's terms, pictures of study equipment were added, and, as already mentioned, a color scheme was incorporated. Figure 8.2 shows three examples of the major changes.

**At Deployment Time Itself**

The new contactless deployment includes two distinct processes, the out-of-the-box participant deployment, and Zoom-assisted research team deployment. During

| Staged Changes of Written Instructions | | | |
|---|---|---|---|
| Equipment Type | Stage 1 | Stage 2 | Stage 3 |
| Ports | No description provided | *What is a port?*<br>*You will use three different types of "ports" to complete the set-up process.*<br>Provided descriptions and images of USB, laptop power, router power, and ethernet ports | *What is a port?*<br>*A port is an opening where you can connect and plug in electrical wires.*<br>*You will use three different types of "ports" to complete the set-up process. The ports on our study equipment and the corresponding wires will be color-coded.* |
| Laptop | *Turn on the laptop and plug the power cord into the wall and logon.* | *2) Turn on the laptop we provided.*<br>*3) Connect the power cord to the laptop and plug the other end in to an outlet or power source in your home.*<br>*4) Press the power button to turn on the laptop.* | *1) Connect the power cord to the laptop (YELLOW Port) and plug the other end in to an outlet or power source in your home.*<br>*2) Turn on the laptop we provided by pressing the button on the left side of the computer. It is a small button marked with ORANGE tape.* |
| Router | *Connect the router to the modem or router using an Ethernet cable.*<br>*a. If connecting to user's modem, make sure connecting the Ethernet port of the modem and the Ethernet port of our router (blue one).*<br>*b. If connecting to user's router, connect one port from their router to our Ethernet port (blue one).* | *5) Connect the Ethernet port on the router we provided (blue port) to the Ethernet port on your home modem or router using the Ethernet cable we provided.*<br>*6) Plug the router we provided into an outlet or power source in your home and press the power button on the router we provided.* | *1) If it is not already connected, connect the BLUE ethernet cable we provided to the Internet port on the router we provided (it will be a different color than all the other ports).*<br>*2) Next connect the other end of the BLUE ethernet cable to the Ethernet port on your home modem or router.*<br>*3) Plug the router we provided into an outlet or power source in your home.*<br>*4) If green lights do not appear on the router, press the power button on the back of the router we provided* |
| Note: Italicized font represents text as it appears in the participant-facing written instructions | | | |

Figure 8.2: Examples of the changes that we made in our three main stages/phases of developing the out-of-the-box solution.

telephone screening and consenting, participants provide their home address for shipment delivery and are instructed to contact the study team upon receipt. Upon receipt of study equipment via UPS delivery, the team instruct the participants to complete initial deployment using the out-of-the-box deployment instructions. Instructions provided to participants offer step-by-step instructions up to initiation of a video call between the research team and participants. The last step of the study instructions directs participants to await initiation of this call by the study team. Once study equipment is "online," the research team proceeds with the Zoom-assisted portion of deployment, wherein team members use TeamViewer11 to access the study laptop, launch the Zoom application, and complete the remaining deployment via video call with participants. During Zoom-assisted deployment, the research team launches the appropriate computer programs to initiate audio monitoring and the recommendation system, completes speaker identification training by recording participants' uninterrupted speech for five minutes, connects the study smartphone to the server, tests all study equipment and programs to verify successful deployment, orients participants to study smartphone and EMA messaging application functionality, and provides instructions for completing study activities.

**EMA**

Prior to COVID-19, we designed the EMA questions to focus on the caregiver's mood, anger, stress, and conflict, and the effectiveness of the recommendations. We also had morning and evening messages with positive encouragements. With the extra delay in preparing the out-of-the-box deployments, we received more time to rethink the EMA questions. We made the following changes.

We added additional messages that inquire about the mental and physical health of the caregiver. These messages ask the users about their physical health, emotional health, stress level, loneliness, and unpleasant interactions.

We added a recommendation request button. As quarantine has increased tensions inside homes, we decided to give caregivers the ability to request recommendations at any time during the day to help relieve stress. With the added stress of COVID-19, we expect more missed EMA messages by the caregiver. Consequently, we added the functionality of giving caregivers a "secondand third chance" to answer questions in case they are occupied by other responsibilities. We also decided to keep messages on the screen (available to be answered) for longer periods of time to give caregivers a greater degree of flexibility. We also tried to become more accommodating with our recommendations by providing in-app meditation sessions for the caregiver. We have included an example of an in-app meditation and the survey questions associated with it in Figure 8.3.

**Logistics**

Contactless delivery procedures were added to the study IRB application and protocol in addition to previously planned in-person procedures. Procedures for in-person deployment were purposefully retained in the event that in-person research activities are deemed necessary or reinstituted. Additionally, changes to the study budget

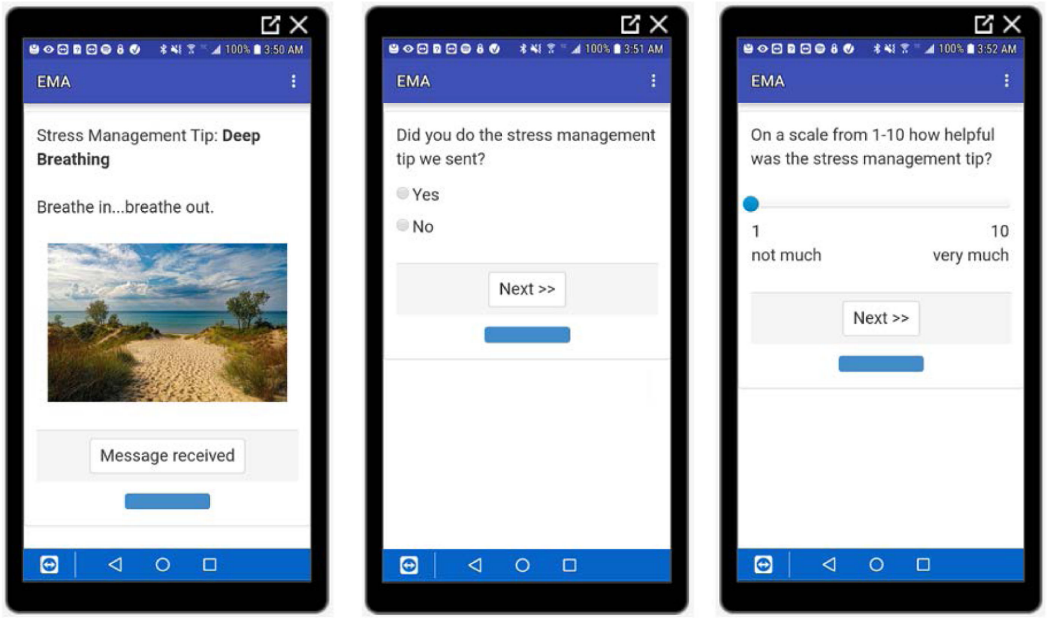



Figure 8.3: An example on how the EMA is conducted: In-app meditation and the survey questions associated with it are pushed to the participant's phone. The first image is the in-app meditation. After a preset time of recommending this meditation, we ask the participants if they did follow through the meditation (middle image) and how effective the meditation was (last image) if they did follow through. The responses to the two survey questions help us figure out what mindfulness techniques work better for the particular participants and recommend those effective ones more often to those participants.

were requested from the grant-holding institution. Previously, budgeted mileage for travel to and from participant homes was replaced with anticipated shipping costs. Logistically, shipping and receiving coordination occurred with front office staff at the recruiting institution. Graduate research associates performed equipment processing prior to and between deployments. Processing included disinfecting all equipment, performing previously described deployment preparation, and repackaging equipment with new study documents (the deployment instructions, training manual, and consent document).

## 8.4 Evaluation

Our evaluation consists of seven out-of-the-box deployments performed in three stages. There were two participants from Stage 1, two participants from Stage 2, and three participants from Stage 3. In stage 1, the first two deployments were performed by skilled technical people as a first trial to identify needed improvements. Based on their feedback, we made changes to the out-of-the-box deployment solution. Stage 2 had two deployments with nontechnical individuals, one elderly and one middle-aged. We made changes based on the feedback from stage 2. Finally, stage 3 had three elderly people perform the deployment. Below is a list of the main

changes made between stages. The instructions remained the same except for what we described below on the in-stage changes. During the first 24–48 hours, no critical failure of any of the participants' systems was seen and all processes of all systems were functioning as expected until the deployments were manually terminated.

We would like to emphasize that the purpose of the results in this Chapter is to evaluate the out-of-the-box deployment protocol, not the components of the PCR system. Therefore, the evaluation is not about the performance measurements such as the accuracy of the acoustic system or how well the participants responded to the recommendations.

During the three stages of the experiments, all participants, including the experts from the first stage and the non-experts from the other stages, used the same manual.
› Major Changes from Stage 1 to Stage 2:

- Technical terms in the written instructions were replaced with names that a layperson understands.
- A progress bar was added to the interface of Speaker ID training software to inform the participants when the model training is expected to finish.
- We assigned members from the research team to use Zoom as a video call to the participants to walk them through the set-up process.

Major Changes from Stage 2 to Stage 3:

- Text size was increased on the smartphone.
- Other visible apps on the smartphone were removed except the EMA app.
- The lock screen on the smartphone was turned OFF.
- Keypad and button sensitivity levels were adjusted.
- Name stickers were tagged near the ports on the equipment.

Major Changes made after feedback from stage 3:

- Page numbers to written instructions were added.
- Participants were instructed to read port descriptions prior to initiating the setup.
- Participants were familiarized with color coded scheme in the beginning of written instructions.
- Pictures of completed system setup were added.

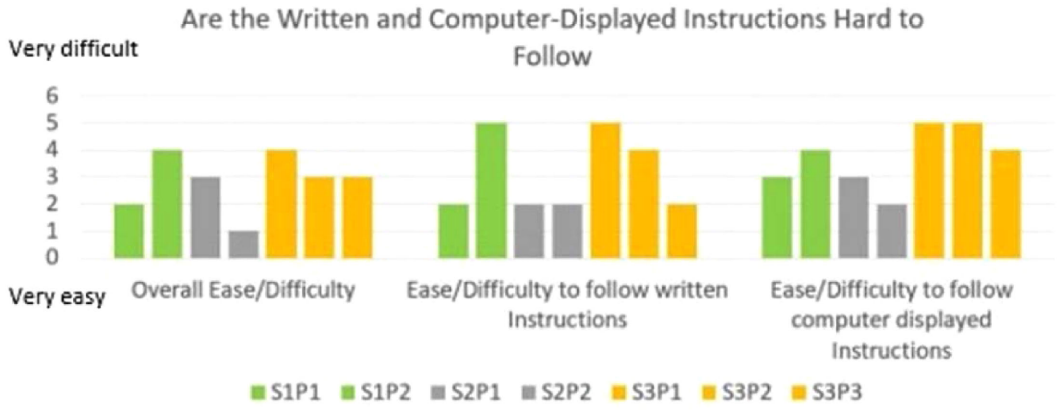
- Tightness of binding around system wires was adjusted.
- Equipment was labeled with number corresponding to the specific instruction step.

After each of the seven out-of-the-box deployments, we collected data on their deployment experience using a survey questionnaire. Appropriate questions below used a Likert scale from 1 to 5 where 1 was very difficult and 5 was very easy. We have listed the questions that we asked

- Overall, how easy was the system setup process?
- Overall, how easy were the written instructions to follow?
- How easy were the computer display and instructions to follow?
- Were you eventually successful in setting up the system?
- How long did it take to get the system setup?
- If you had trouble in the setup process, which part(s) of the setup process confused you?
- Which principle(s) that we adopted do you find helpful during the setup process?
- How comfortable/familiar are you with computers and smartphones?

Figure 8.4's three subplots show the participants' responses to our survey questions. The two participants from stage 1 are denoted in green. The two participants from stage 2 are denoted in gray. The three participants from stage 3 are denoted in yellow. To interpret the legends of the first subplot: Each of the 7 participants are denoted by the stage that they are in and the number used to represent them in that stage. For example, S1P1 means the first participant from the first stage, and S3P1 means the first participant from the third stage. We made changes to stage 2 based on the responses of participants of stage 1, and to stage 3 based on the responses of participants of stage 2.

From Figure 8.4, we observe that the two participants from stage 1 experienced less difficulties than the participants in stage 2, as the first stage participants rated that the average difficulty being 3, the difficulty of following written instructions being 3.5, and the difficulty of following computer-displayed instructions being 3.5. Meanwhile, the averages of the scores that the participants of stage 2 gave are 2, 2, 2.5. The difference between the scores given by the two stages is expected, as the stage 1 participants were skilled technical people while the stage 2 participants had no technical background.




(a)

| | Stage 1 | Stage 2 | Stage 3 |
|---|---|---|---|
| No significant trouble | 0 | 0 | 1 |
| Smartphone hard to navigate | 0 | 0 | 0 |
| Equipment hard to assemble | 0 | 1 | 1 |
| Confusing technical terms | 1 | 2 | 1 |
| Connect laptop to Wi-Fi | 0 | 1 | 1 |
| Unclear Computer-displayed instructions | 0 | 0 | 0 |
| Unclear Written instructions | 1 | 1 | 1 |

(b)

| | Stage 1 | Stage 2 | Stage 3 |
|---|---|---|---|
| Enlarged font size | 1 | 1 | 2 |
| Image display in addition to text | 1 | 1 | 2 |
| Labels and color-coding scheme | 0 | 1 | 3 |

(c)

Figure 8.4: Participants' responses to Questions 1-3, from which we evaluate the clarity of the written and computer-displayed instructions. For all questions in the three subplots, we asked the participants to answer them after they finished following the manual to deploy the system out-of-the-box.

After making the improvement based on the responses from stage 2, we see a significant increase of ratings. The three participants of stage 3 rated a difficulty score of 3.33 on average on the overall easiness/difficulty to follow the instructions, and an average of 3.66 and an average of 4.66 on written and computer-displayed instructions, respectively. Comparing the ratings obtained from stage 1 and stage 3, our improvements enable the third stage elderly participants to follow the instructions as easily as the technically skilled people from stage 1. We conclude that our improvements made on the written and computer displayed instructions are effective.

In Figure 8.4(b), we identify what caused confusion to the participants. The technical terms in our instructions were the leading cause of confusion. This is to be expected, as the general population are not familiar with terms that skilled technical people are.

Figure 8.4(c) describes the assessment result of the effectiveness of the principles that we adopted to help the participants set up the process. Each of our three principles received four votes; all three participants of stage 3 reported that our labeling and color coding scheme were helpful. Five out of our seven participants finished the setup process within an hour, and the other two of them spent between 1 to 2 hours. Also, all participants of stage 3 were able to successfully set up the deployment within an hour, suggesting that the improvements we made between stages 2 and 3 were effective. The average time to complete the deployment for all our participants was 1.28 hours with a standard deviation of 0.49 hour.

**Evaluate the Final Stage (Stage 3) Out-of-the-Box Instructions on Six More new Participants**

In the following paragraphs, we evaluate our stage 3 out-of-the-box instructions on six more participants (so a total of 9 real caregivers) to further attest if these instructions are sufficient enough to ensure successful deployments. In addition to the three participants in Section 8.4, we present Tables 8.1, 8.2, 8.3, 8.4, 8.5, 8.6, 8.7, and 8.8 to describe the out-of-the-box questionnaire and their responses.

In Table 8.1, 2 out of 9 participants reported that setting up the system was difficult, while 7 out of 9 thought that the setup process was easy. The majority vote on the easiness of the system indicates that in general, our stage 3 out-of-the-box instructions are effective. However, there are reports that our system was difficult to set up, and this brings us to Table 8.7, which indicates the parts where the participants found confusing or hindered their setup process. Among the six potential candidates for confusing factors, technical terms caused 3 out of the 9 participants to have trouble with the setup process, followed by computer-displayed instructions and phone navigation. However, Table 8.7 indicates that about half of the 9 participants did not encounter any trouble, which suggests the effectiveness of the out-of-the-box deployment strategy, although we must also take into consideration that, as indicated in table 8.6, 8 out of the 9 participants are at least somewhat familiar with computers and smartphones.

In Tables 8.2 and 8.3, we observe the participants report on the ease to follow

written and computer-displayed instructions. In Table 8.2, 8 out of 9 participants believed that the written instructions were at least somewhat easy to follow, while 1 participant reported that it was very hard to follow. Similarly, in Table 8.3, 8 out of 9 participants reported that the computer-displayed instructions were at least somewhat easy to follow, while the same participant who reported that the written instructions were very difficult to follow reported that the computer-displayed instructions were very hard to follow as well. This person was elderly and held a bachelor's degree, and out of the 9 participants, this person was the only one who reported that he was somewhat unfamiliar with computers and smartphones (see Table 8.6). Therefore, we conclude that our computer-displayed and written instructions at stage 3 are at least somewhat easy to follow, for people who are at least somewhat familiar with computers and smartphone technology.

In Table 8.4, we demonstrate that all participants, despite that they might have had trouble setting up the system, were all successful in setting up the system. This indicates that the changes we made from stage 1 to stage 2 and from stage 2 to stage 3 are effective strategies to help ensure the out-of-the-box deployment successful.

In Table 8.5, we observe that 8 out of the 9 participants spent 0-2 hours setting up the system, with the majority (5 of 9) spending less than 1 hour. Interestingly, the participant who was most unfamiliar with smartphones and computers spent only 1-2 hours, while the person who spent the most time setting up the system reported that they were somewhat familiar with computers and smartphones. This person was also elderly and held a bachelor's degree.

In Table 8.7, we list the parts which might have confused the participants when they set up the system. We observe that 5 out of the 9 reported that they did not encounter any trouble. This indicates that our out-of-the-box deployment strategies at stage 3 are mostly successful. However, 3 out of 9 of them still reported that the technical terms were confusing. In our effort, we tried to tone down the technical terms, but in a deployment such as ours, technical terms are not completely avoidable.

In Table 8.8, we list the parts which might have helped the participants when they set up the system. 100% of the participants reported that the labels and color-coding schemes were helpful, which re-confirms the observation in Figure 8.4(c). 4 out of 9 of them also reported that the image display in addition to the text and the enlarged font size was helpful. This again re-confirms the observation in Figure 8.4(c) that these two were still helpful, but less so than the labels and color-coding scheme.

## 8.5 Lessons Learned and Generalization

In the previous sections we have described the out-of-the-box techniques we developed, provided observations about those techniques, and described an evaluation. In this section, we summarize the lessons learned and discuss the generalization of these techniques to other deployments. One main result was that there were few technical changes required to the core system. Rather, most changes were to auxiliary aspects of the deployed system.

| Ease | Number | Percentage |
|---|---|---|
| Very difficult | 1 | 11.1% |
| Somewhat difficult | 1 | 11.1% |
| Neutral | 0 | 0% |
| Somewhat easy | 4 | 44.4% |
| Very easy | 3 | 33.3% |

Table 8.1: Overall, how easy was the system set-up process?

| Ease | Number | Percentage |
|---|---|---|
| Very difficult | 1 | 11.1% |
| Somewhat difficult | 0 | 0% |
| Neutral | 0 | 0% |
| Somewhat easy | 2 | 22.2% |
| Very easy | 6 | 66.7% |

Table 8.2: Overall, how easy were the written instructions to follow?

| Ease | Number | Percentage |
|---|---|---|
| Very difficult | 1 | 11.1% |
| Somewhat difficult | 0 | 0% |
| Neutral | 0 | 0% |
| Somewhat easy | 2 | 22.2% |
| Very easy | 6 | 66.7% |

Table 8.3: How easy were the computer-displayed instructions to follow?

| Yes/No | Number | Percentage |
|---|---|---|
| Yes | 9 | 11.1% |
| No difficult | 0 | 0% |

Table 8.4: Were you eventually successful in setting up the system?

| Hours spent | Number | Percentage |
|---|---|---|
| 3-4 | 0 | 0% |
| 2-3 | 1 | 11.1% |
| 1-2 | 3 | 33.3% |
| 0-1 | 5 | 55.6% |

Table 8.5: How long did it take to get the system set up?

Many changes were required to the documentation for the caregiver who now has to set up the system. We made heavy use of pictures, videos, and large lettering labels on equipment, even for ON–OFF buttons. We found that budget flexibility was needed (e.g., transferring money from travel to mailing costs). Significant IRB

| Familiarity | Number | Percentage |
|---|---|---|
| Very unfamiliar | 0 | 0% |
| Somewhat unfamiliar | 1 | 11.1% |
| Neutral | 0 | 0% |
| Somewhat familiar | 4 | 44.4% |
| Very familiar | 4 | 44.4% |

Table 8.6: How comfortable/familiar are you with computers and smartphones?

| Confusing part | Number | Percentage |
|---|---|---|
| Written instructions | 1 | 11.1% |
| Computer displayed instructions | 2 | 22.2% |
| Wi-Fi connection | 1 | 11.1% |
| Technical terms | 3 | 33.3% |
| Router and laptop connection | 1 | 11.1% |
| Phone navigation | 2 | 22.2% |
| No trouble | 5 | 55.5% |

Table 8.7: If you had trouble in the set up process, which part(s) of the set-up process confused you?

| Helpful part | Number | Percentage |
|---|---|---|
| Labels and color-coding scheme | 9 | 100.0% |
| Image display in addition to text | 4 | 44.4% |
| Enlarged font size | 4 | 44.4% |

Table 8.8: Which principle(s) that we adopted do you find helpful during the set-up process?

changes were also required.

Changes made to study procedures increased the geographic reach of recruitment from clinic patients living in surrounding counties to out-of-state patients receiving services from the recruiting clinic. This can be considered a positive outcome for physical contactless deployments. Additionally, strategies employed for physical contactless deployment may also augment future in-person deployments.

While the techniques we developed were for a single research deployment, most of the techniques are general and can be applied to many home deployments. For example, Zoom, M2G, and TeamViewer are basic products that can be used by any deployment. The documentation we created can be used as a template for what is required for users, suitably changed based on the hardware used for a given deployment. More specifically, the essential takeaways are as follows:

- The deployment adjustments provide an added degree of robustness, which improves the initial deployment process of complicated in-home systems (the

participants are successful and less frustrated with setting up the system).

- The Zoom and Teamviewer combination is able to overcome the more technical and difficult aspects of having dyads deploy a system by themselves.

Our takeaways have the following implications for our discipline:

- Our techniques can allow non-disruption of studies even when no contact is permitted. This prevents the advancement of our discipline from being slowed down by the no-contact mandate.
- Even when contact is allowed, the techniques make it easier for project members such as behavioral scientist graduate students to deploy the system even though they are often not as aware of the technology as the computer scientists. This opens the door for more potential interdisciplinary collaboration between the computer science department and other departments, such as the behavioral science department.
- In-home deployments require the core system being developed and significant additional software and tools. The techniques, software, and tools such as M2G, Zoom, Teamviewer, etc., presented are suggested as key and enable technical researchers to focus on the core.
- Our techniques allows for increased geographical reach when researchers recruit participants, which allows for more data to be produced for the research projects. The additional data can yield more evaluation results for the studies.

## 8.6 Conclusion

Deploying technology in homes to study and improve healthcare can be a complex endeavor even for technically savvy people. COVID-19 delayed or stopped many studies. This Chapter describes a set of solutions and lessons learned that support participants in setting up the system by themselves without any personal contact. An evaluation demonstrates its effectiveness in an Alzheimer's study. It is hypothesized that the techniques and lessons are also useful to be applied to deployments even after personal contact returns.

*Chapter 9*

# CONCLUSIONS

## 9.1 Summary

In this thesis, we define realism as the reality as a result of DL models' interaction with CPS. The direct consequence of realisms is that the data are distorted because of the environment, and, as a result, their distribution is shifted from the training samples that the DL models are trained on. The shift of distribution results in the consequence that the DL models, not trained on the data with the new distribution, fail at performing adequately on the samples captured by the actual CPS. In other words, the models are not robust enough to be deployed with reasonable performance in the CPS. In this thesis, we have proposed various ways to deal with the realisms. In Chapter 3, we propose to just incorporate the known realisms into the training samples. We also propose to use OOD detection techniques to filter out samples that are out of the distribution of the training samples of a DL model, so that the DL model does not misclassify it (because the sample is not sent to it for classification). This is because of the unique perspective that samples whose classes are not previously seen during the training stage of the deep learning classifiers can be seen as OOD samples. In Chapter 4 and Chapter 6, we propose to use unsupervised domain adaptation to domain adapt from clean samples to samples that are environmentally distorted, as (unsupervised) domain adaptation is a way to deal with the fact that the models, trained on the source domain with a certain distribution, are used on the target domain with a different distribution. This is because of our unique perspective that dealing with (unknown) realisms during the development time of the deep learning models is essentially the premise of (unsupervised) domain adaptation. In Chapter 5, we look at further improving the robustness of the models using attention-based GNNs and world knowledge to guide the attention/transformer architecture. In addition to looking at the realisms that we can address during the deployment stage of the DL models, we also look at a case study illustrated in Chapter 7 where we attest that our previous way of dealing with known realisms is effective. In Chapter 8, the realism comes during the process when the deep learning model is deployed - it is a direct result from the outside world, such as COVID which forbids in-person contact so the developers can't go to the smart homes where the participants live to deploy the DL models. In summary, this thesis closely examines the most prominent realisms that take place as a result of the deep learning models' interaction with the cyber physical systems in which they are deployed, as well as offering solutions to the realisms.

## 9.2 Future Work

In this thesis, we have emphasized the fact that (unsupervised) domain adaptation is useful at dealing with three out of four realisms we address. Recall that the last

realism has nothing to do with the training of the DL models. Instead, it is the realism that comes during the process when the deep learning model is deployed, such as human behavior. However, unsupervised domain adaptation requires that we have access to the data in the target domain, which might not always be available if we are training a DL model to be deployed in a CPS, because we don't have access to the data to be generated in the CPS. This is when domain generalization comes in. There is a key difference: Domain adaptation modifies a model trained on the data from the source domain using the samples in the training set of the target domain. In other words, it has access to some of the target samples because these samples are used in training. Domain generalization, on the other hand, trains a model that that it performs well on multiple domains, but during the training process it does not necessarily expose the model to all of the domains. Therefore, the goal of domain generalization is to make the model to be able to generalize on domains that it has not previously seen at all. One direction of future work is to develop novel domain generalization solutions to address the three realisms, as domain generalization's application range is larger than domain adaptation - domain generalization does not necessarily require even the training samples from the previously unseen domain.

# BIBLIOGRAPHY


Alex, Starlet Ben, Ben P Babu, and Leena Mary (2018). "Utterance and Syllable Level Prosodic Features for Automatic Emotion Recognition". In: *2018 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*. IEEE, pp. 31–35.

Altun, Halis and Gökhan Polat (2007). "New frameworks to boost feature selection algorithms in emotion detection for improved human-computer interaction". In: *International Symposium on Brain, Vision, and Artificial Intelligence*. Springer, pp. 533–541.

Association, Alzheimer's, William Thies, and Laura Bleiler (2013). "2013 Alzheimer's disease facts and figures". In: *Alzheimer's & dementia* 9.2, pp. 208–245.

Bahreini, Kiavash, Rob Nadolski, and Wim Westera (2016). "Towards real-time speech emotion recognition for affective e-learning". In: *Education and information technologies* 21.5, pp. 1367–1386.

Bai, Jiandong et al. (2021). "A3t-gcn: Attention temporal graph convolutional network for traffic forecasting". In: *ISPRS International Journal of Geo-Information* 10.7, p. 485.

Barros, Pablo and Stefan Wermter (2016). "Developing crossmodal expression recognition based on a deep neural model". In: *Adaptive behavior* 24.5, pp. 373–396.

Beard, Rory et al. (2018). "Multi-Modal Sequence Fusion via Recursive Attention for Emotion Recognition". In: *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pp. 251–259.

Bedón-Molina, John, Mario J Lopez, and Ivan S Derpich (2020). "A home-based smart health model". In: *Advances in Mechanical Engineering* 12.6, p. 1687814020935282.

Bennett, Paul N and Nam Nguyen (2009). "Refined experts: improving classification in large taxonomies". In: *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, pp. 11–18.

Bousmalis, Konstantinos, Nathan Silberman, et al. (2017). "Unsupervised pixel-level domain adaptation with generative adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3722–3731.

Bousmalis, Konstantinos, George Trigeorgis, et al. (2016). "Domain separation networks". In: *Advances in neural information processing systems* 29.

Braun, Sebastian and Ivan Tashev (2021). "On training targets for noise-robust voice activity detection". In: *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE, pp. 421–425.

Brush, AJ Bernheim et al. (2011). "Home automation in the wild: challenges and opportunities". In: *proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2115–2124.

Burkhardt, Felix et al. (2005). "A database of German emotional speech". In: *Ninth European Conference on Speech Communication and Technology*.

Cai, Ru Ying et al. (2018). "Brief report: Inter-relationship between emotion regulation, intolerance of uncertainty, anxiety, and depression in youth with autism spectrum disorder". In: *Journal of autism and developmental disorders* 48.1, pp. 316–325.

Cao, Houwei et al. (2014). "CREMA-D: Crowd-sourced emotional multimodal actors dataset". In: *IEEE transactions on affective computing* 5.4, pp. 377–390.

Caraty, Marie-José and Claude Montacié (2015). "Detecting speech interruptions for automatic conflict detection". In: *Conflict and Multimodal Communication*. Springer, pp. 377–401.

Cascetta, Ennio (2013). *Transportation systems engineering: theory and methods*. Vol. 49. Springer Science & Business Media.

Castillo, José Carlos et al. (2018). "Emotion detection and regulation from personal assistant robot in smart environment". In: *Personal assistants: Emerging computational technologies*. Springer, pp. 179–195.

Cen, Ling et al. (2016). "A real-time speech emotion recognition system and its application in online learning". In: *Emotions, technology, design, and learning*. Elsevier, pp. 27–46.

Chatterjee, Rajdeep et al. (2021). "Real-Time Speech Emotion Analysis for Smart Home Assistants." In: *IEEE Trans. Consumer Electron.* 67.1, pp. 68–76.

Chen, Long, Venkatesh Ravichandran, and Andreas Stolcke (2021). "Graph-based label propagation for semi-supervised speaker identification". In: *arXiv preprint arXiv:2106.08207*.

Chen, Sanyuan et al. (2022). "Wavlm: Large-scale self-supervised pre-training for full stack speech processing". In: *IEEE Journal of Selected Topics in Signal Processing*.

Chen, Zeya, Mohsin Y Ahmed, et al. (2019a). "ARASID: Artificial Reverberation-Adjusted Indoor Speaker Identification Dealing with Variable Distances." In: *EWSN*, pp. 154–165.

– (2019b). "ARASID: Artificial Reverberation-Adjusted Indoor Speaker Identification Dealing with Variable Distances". In: *Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks*. EWSN '19. Beijing, China: Junction Publishing, pp. 154–165. ISBN: 978-0-9949886-3-8. URL: http://dl.acm.org/citation.cfm?id=3324320.3324339.

Cheng, Ming, Andrew Friesen, and Olalekan Adekola (2019). "Using emotion regulation to cope with challenges: a study of Chinese students in the United Kingdom". In: *Cambridge Journal of Education* 49.2, pp. 133–145.

Choi, Yunjey et al. (2018). "Stargan: Unified generative adversarial networks for multi-domain image-to-image translation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8789–8797.

Choudhury, Akash Roy et al. (2018). "Emotion Recognition from Speech Signals using Excitation Source and Spectral Features". In: *2018 IEEE Applied Signal Processing Conference (ASPCON)*. IEEE, pp. 257–261.

Coates, Adam, Andrew Ng, and Honglak Lee (2011). "An analysis of single-layer networks in unsupervised feature learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, pp. 215–223.

Costin, Hariton et al. (2009). "TELEMON-A complex system for real time medical telemonitoring". In: *World Congress on Medical Physics and Biomedical Engineering, September 7-12, 2009, Munich, Germany*. Springer, pp. 92–95.

Cui, Shuhao et al. (2020). "Gradually vanishing bridge for adversarial domain adaptation". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12455–12464.

Danisman, Taner and Adil Alpkocak (2008). "Emotion classification of audio signals using ensemble of support vector machines". In: *International Tutorial and Research Workshop on Perception and Interactive Technologies for Speech-Based Systems*. Springer, pp. 205–216.

Datcu, Dragos and Léon JM Rothkrantz (2005). "Facial expression recognition with relevance vector machines". In: *2005 IEEE International Conference on Multimedia and Expo*. IEEE, pp. 193–196.

Daw, Arka, Anuj Karpatne, et al. (2017). "Physics-guided neural networks (pgnn): An application in lake temperature modeling". In: *arXiv preprint arXiv:1710.11431*.

Daw, Arka, R Quinn Thomas, et al. (2020). "Physics-guided architecture (pga) of neural networks for quantifying uncertainty in lake temperature modeling". In: *Proceedings of the 2020 siam international conference on data mining*. SIAM, pp. 532–540.

Deng, Jun et al. (2017). "Universum autoencoder-based domain adaptation for speech emotion recognition". In: *IEEE Signal Processing Letters* 24.4, pp. 500–504.

Deng, Li (2012). "The mnist database of handwritten digit images for machine learning research". In: *IEEE Signal Processing Magazine* 29.6, pp. 141–142.

Dickerson, Robert F et al. (2014). "Resonate: reverberation environment simulation for improved classification of speech models". In: *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*. IEEE, pp. 107–117.

Dinkel, Heinrich et al. (2021). "Voice activity detection in the wild: A data-driven approach using teacher-student training". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29, pp. 1542–1555.

Dupuis, Kate and M Kathleen Pichora-Fuller (2010). *Toronto Emotional Speech Set (TESS)*. University of Toronto, Psychology Department.

*dynaEdge DE-100* (n.d.). URL: `https://asia.dynabook.com/laptop/dynaedge-de100/overview.php`.

Elhamod, Mohannad et al. (2020). "CoPhy-PGNN: learning physics-guided neural networks with competing loss functions for solving eigenvalue problems". In: *arXiv preprint arXiv:2007.01420*.

Esteva, Andre et al. (2021). "Deep learning-enabled medical computer vision". In: *NPJ digital medicine* 4.1, pp. 1–9.

Eyben, Florian et al. (2015). "The Geneva minimalistic acoustic parameter set (GeMAPS) for voice research and affective computing". In: *IEEE Transactions on Affective Computing* 7.2, pp. 190–202.

Fayek, Haytham M, Margaret Lech, and Lawrence Cavedon (2015). "Towards real-time speech emotion recognition using deep neural networks". In: *2015 9th international conference on signal processing and communication systems (ICSPCS)*. IEEE, pp. 1–5.

Fernandes, V et al. (2018). "Speech Emotion Recognition using Mel Frequency Cepstral Coefficient and SVM Classifier". In: *2018 International Conference on System Modeling & Advancement in Research Trends (SMART)*. IEEE, pp. 200–204.

Fernández-Caballero, Antonio et al. (2016). "Smart environment architecture for emotion detection and regulation". In: *Journal of biomedical informatics* 64, pp. 55–73.

French, Geoffrey, Michal Mackiewicz, and Mark Fisher (2017). "Self-ensembling for visual domain adaptation". In: *arXiv preprint arXiv:1706.05208*.

Ganin, Yaroslav and Victor Lempitsky (2015). "Unsupervised domain adaptation by backpropagation". In: *International conference on machine learning*. PMLR, pp. 1180–1189.

Ganin, Yaroslav, Evgeniya Ustinova, et al. (2016). "Domain-adversarial training of neural networks". In: *The journal of machine learning research* 17.1, pp. 2096–2030.

Gao, Ye, Brian Baucom, et al. (2022). "The Enforced Transfer: A Novel Domain Adaptation Algorithm". In: *arXiv preprint arXiv:2201.10001*.

Gao, Ye, Meiyi Ma, et al. (2020). "A monitoring, modeling, and interactive recommendation system for in-home caregivers: Demo abstract". In: *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, pp. 587–588.

GAO, YE et al. (2021). "Emotion Recognition Robust to Indoor Environmental Distortions and Non-targeted Emotions Using Out-Of-Distribution Detection". In.

Gaugler, Joseph et al. (2019). "2019 Alzheimer's disease facts and figures". In: *Alzheimers & Dementia* 15.3, pp. 321–387.

Ghaleb, Esam, Mirela Popa, and Stylianos Asteriadis (2019). "Multimodal and Temporal Perception of Audio-visual Cues for Emotion Recognition". In: *8th International Conference on Affective Computing & Intelligent Interaction (ACII 2019), Cambridge, United Kingdom*.

Ghifary, Muhammad et al. (2016). "Deep reconstruction-classification networks for unsupervised domain adaptation". In: *European conference on computer vision*. Springer, pp. 597–613.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. The MIT Press. ISBN: 0262035618, 9780262035613.

Goodfellow, Ian, Jean Pouget-Abadie, et al. (2014). "Generative adversarial nets". In: *Advances in neural information processing systems* 27.

– (2020). "Generative adversarial networks". In: *Communications of the ACM* 63.11, pp. 139–144.

Grekow, Jacek (2018). "Music Emotion Maps in the Arousal-Valence Space". In: *From Content-based Music Emotion Recognition to Emotion Maps of Musical Pieces*. Springer, pp. 95–106.

Gretton, Arthur et al. (2012). "A kernel two-sample test". In: *The Journal of Machine Learning Research* 13.1, pp. 723–773.

Grezes, Félix, Justin Richards, and Andrew Rosenberg (2013). "Let me finish: automatic conflict detection using speaker overlap." In: *Interspeech*, pp. 200–204.

Gross, James J and Ricardo F Muñoz (1995). "Emotion regulation and mental health". In: *Clinical psychology: Science and practice* 2.2, pp. 151–164.

Gross, James J, Helen Uusberg, and Andero Uusberg (2019). "Mental illness and well-being: an affect regulation perspective". In: *World Psychiatry* 18.2, pp. 130–139.

Gu, Xiang, Jian Sun, and Zongben Xu (2020). "Spherical space domain adaptation with robust pseudo-label loss". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9101–9110.

Guo, Shengnan et al. (2019). "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01, pp. 922–929.

Haeusser, Philip et al. (2017). "Associative domain adaptation". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2765–2773.

Haq, S. and P.J.B. Jackson (Aug. 2010). "Machine Audition: Principles, Algorithms and Systems". In: ed. by W. Wang. Hershey PA: IGI Global. Chap. Multimodal Emotion Recognition, pp. 398–423.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hendrycks, Dan and Kevin Gimpel (2016). "A baseline for detecting misclassified and out-of-distribution examples in neural networks". In: *arXiv preprint arXiv:1610.02136*.

Hensen, B et al. (2021). "Remote data collection for public health research in a COVID-19 era: ethical implications, challenges and opportunities". In: *Health Policy and Planning* 36.3, pp. 360–368.

Hirsch, Hans-Günter and David Pearce (2000). "The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions". In: *ASR2000-Automatic speech recognition: challenges for the new Millenium ISCA tutorial and research workshop (ITRW)*.

Hoffman, Judy et al. (2017). "Cycada: Cycle-consistent adversarial domain adaptation. arXiv". In: *arXiv preprint arXiv:1711.03213*.

– (2018). "Cycada: Cycle-consistent adversarial domain adaptation". In: *International conference on machine learning*. Pmlr, pp. 1989–1998.

Hu, Lanqing et al. (2020). "Unsupervised domain adaptation with hierarchical gradient synchronization". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4043–4052.

Hu, Xinyue et al. (2020). "Physics-guided deep neural networks for power flow analysis". In: *IEEE Transactions on Power Systems* 36.3, pp. 2082–2092.

Huang, Che-Wei and Shrikanth Narayanan (2018). "Stochastic Shake-Shake Regularization for Affective Learning from Speech." In: *Interspeech*, pp. 3658–3662.

Hull, J. J. (1994). "A database for handwritten text recognition research". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.5, pp. 550–554. DOI: 10.1109/34.291440.

Jagadish, Hosagrahar V et al. (2014). "Big data and its technical challenges". In: *Communications of the ACM* 57.7, pp. 86–94.

Jalili, Amin et al. (2018). “Speech Emotion Recognition Using Cyclostationary Spectral Analysis”. In: *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, pp. 1–6.

Jia, Fei, Somshubra Majumdar, and Boris Ginsburg (2021). “Marblenet: Deep 1d time-channel separable convolutional neural network for voice activity detection”. In: *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 6818–6822.

Kang, Guoliang et al. (2019). “Contrastive adaptation network for unsupervised domain adaptation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4893–4902.

Khan, Tarannum and Manju K Chattopadhyay (2017). “Smart health monitoring system”. In: *2017 International Conference on Information, Communication, Instrumentation and Control (ICICIC)*. IEEE, pp. 1–6.

Krizhevsky, Alex, Geoffrey Hinton, et al. (2009). “Learning multiple layers of features from tiny images”. In.

Kumar, Abhishek et al. (2018). “Co-regularized alignment for unsupervised domain adaptation”. In: *arXiv preprint arXiv:1811.05443*.

Lech, Margaret et al. (2020). “Real-time speech emotion recognition using a pre-trained image classification network: Effects of bandwidth reduction and companding”. In: *Frontiers in Computer Science* 2, p. 14.

LeCun, Yann, Yoshua Bengio, et al. (1995). “Convolutional networks for images, speech, and time series”. In: *The handbook of brain theory and neural networks* 3361.10, p. 1995.

LeCun, Yann, Léon Bottou, et al. (1998). “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Lee, Kimin et al. (2018). “A simple unified framework for detecting out-of-distribution samples and adversarial attacks”. In: *Advances in neural information processing systems* 31.

Lee, Sungbok et al. (2005). “An articulatory study of emotional speech production”. In: *Ninth European Conference on Speech Communication and Technology*.

Lefter, Iulia and Catholijn M Jonker (2017). “Aggression recognition using overlapping speech”. In: *2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII)*. IEEE, pp. 299–304.

Lellis-Santos, Camilo and Fernando Abdulkader (2020). “Smartphone-assisted experimentation as a didactic strategy to maintain practical lessons in remote education: alternatives for physiology education during the COVID-19 pandemic”. In: *Advances in physiology education* 44.4, pp. 579–586.

Letcher, Alistair et al. (2018). "Automatic conflict detection in police body-worn audio". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 2636–2640.

Li, Fuxian et al. (2021). "Dynamic graph convolutional recurrent network for traffic prediction: Benchmark and solution". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)*.

Li, Wei, Martin Z Bazant, and Juner Zhu (2021). "A physics-guided neural network framework for elastic plates: Comparison of governing equations-based and energy-based approaches". In: *Computer Methods in Applied Mechanics and Engineering* 383, p. 113933.

Li, Yaguang et al. (2017). "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting". In: *arXiv preprint arXiv:1707.01926*.

Liang, Shiyu, Yixuan Li, and Rayadurgam Srikant (2017). "Enhancing the reliability of out-of-distribution image detection in neural networks". In: *arXiv preprint arXiv:1706.02690*.

Lippi, Marco, Matteo Bertini, and Paolo Frasconi (2013). "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning". In: *IEEE Transactions on Intelligent Transportation Systems* 14.2, pp. 871–882.

Liu, Ming-Yu and Oncel Tuzel (2016). "Coupled generative adversarial networks". In: *Advances in neural information processing systems* 29, pp. 469–477.

Liu, Wei et al. (2011). "Discovering spatio-temporal causal interactions in traffic data streams". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1010–1018.

Liu, Ziwei et al. (Dec. 2015). "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*.

Livingstone, Steven R and Frank A Russo (2018). "The Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS): A dynamic, multimodal set of facial and vocal expressions in North American English". In: *PloS one* 13.5, e0196391.

Long, Mingsheng et al. (2018). "Conditional adversarial domain adaptation". In: *Advances in neural information processing systems* 31.

Lugger, Marko and Bin Yang (2007). "An incremental analysis of different feature groups in speaker independent emotion recognition". In: *16th Int. congress of phonetic sciences*.

Ma, Meiyi et al. (2017). "Mˆ 2G: a monitor of monitoring systems with ground truth validation features for research-oriented residential applications". In: *2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, pp. 10–18.

Mahalanobis, Prasanta Chandra (1936). “On the generalized distance in statistics”. In: National Institute of Science of India.

Mano, Leandro Y (2018). “Emotional condition in the Health Smart Homes environment: emotion recognition using ensemble of classifiers”. In: *2018 Innovations in Intelligent Systems and Applications (INISTA)*. IEEE, pp. 1–8.

Martin, Alice et al. (2020). “Continuing medical and student education in dermatology during the coronavirus pandemic–a major challenge”. In: *JDDG: Journal der Deutschen Dermatologischen Gesellschaft* 18.8, pp. 835–840.

Mesaros, Annamaria, Toni Heittola, and Tuomas Virtanen (2016). “TUT database for acoustic scene classification and sound event detection”. In: *2016 24th European Signal Processing Conference (EUSIPCO)*. IEEE, pp. 1128–1132.

Mirza, Mehdi and Simon Osindero (2014). “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784*.

Na, Jaemin et al. (2021). “Fixbi: Bridging domain spaces for unsupervised domain adaptation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1094–1103.

Nagrani, Arsha, Joon Son Chung, and Andrew Zisserman (2017). “Voxceleb: a large-scale speaker identification dataset”. In: *arXiv preprint arXiv:1706.08612*.

Netzer, Yuval et al. (2011). “Reading digits in natural images with unsupervised feature learning”. In.

Ngaruiya, Njeri, Daniel Orwa, and Peter Waiganjo (2017). “Towards a deployment model for emonitoring of geriatric persons in rural developing countries: Case of Kenya”. In: *2017 IST-Africa Week Conference (IST-Africa)*. IEEE, pp. 1–9.

Nielsen, Frank (2020). “On a generalization of the Jensen–Shannon divergence and the Jensen–Shannon centroid”. In: *Entropy* 22.2, p. 221.

Noda, Kuniaki et al. (2015). “Audio-visual speech recognition using deep learning”. In: *Applied Intelligence* 42.4, pp. 722–737.

Noroozi, Fatemeh et al. (2017). “Audio-visual emotion recognition in video clips”. In: *IEEE Transactions on Affective Computing* 10.1, pp. 60–75.

Odena, Augustus, Christopher Olah, and Jonathon Shlens (2017). “Conditional image synthesis with auxiliary classifier gans”. In: *International conference on machine learning*. PMLR, pp. 2642–2651.

Perera, Pramuditha and Vishal M. Patel (June 2019). “Deep Transfer Learning for Multiple Class Novelty Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Purwins, Hendrik et al. (2019). “Deep learning for audio signal processing”. In: *IEEE Journal of Selected Topics in Signal Processing* 13.2, pp. 206–219.

Radford, Alec, Luke Metz, and Soumith Chintala (2015). "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434*.

Saberi, Parya (2020). "Research in the time of coronavirus: continuing ongoing studies in the midst of the COVID-19 pandemic". In: *AIDS and Behavior* 24.8, pp. 2232–2235.

Saenko, Kate et al. (2010). "Adapting visual category models to new domains". In: *European conference on computer vision*. Springer, pp. 213–226.

Salekin, Asif et al. (2017). "Distant emotion recognition". In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.3, pp. 1–25.

Shchetinin, Eugene Yu et al. (2020). "Deep Neural Networks for Emotion Recognition". In: *International Conference on Distributed Computer and Communication Networks*. Springer, pp. 365–379.

Shu, Rui et al. (2018). "A dirt-t approach to unsupervised domain adaptation". In: *arXiv preprint arXiv:1802.08735*.

Shuman, David I et al. (2013). "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains". In: *IEEE signal processing magazine* 30.3, pp. 83–98.

Snyder, David et al. (2018). "X-vectors: Robust dnn embeddings for speaker recognition". In: *2018 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 5329–5333.

Song, Chao et al. (2020). "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 01, pp. 914–921.

Sonntag, Daniel et al. (2017). "Overview of the CPS for smart factories project: Deep learning, knowledge acquisition, anomaly detection and intelligent user interfaces". In: *Industrial internet of things*. Springer, pp. 487–504.

Spruijt-Metz, Donna et al. (2016). "M2FED: monitoring and modeling family eating dynamics". In: *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*, pp. 352–353.

Stolar, Melissa N et al. (2017). "Real time speech emotion recognition using RGB image classification and transfer learning". In: *2017 11th International Conference on Signal Processing and Communication Systems (ICSPCS)*. IEEE, pp. 1–8.

Sun, Baochen and Kate Saenko (2016). "Deep coral: Correlation alignment for deep domain adaptation". In: *European conference on computer vision*. Springer, pp. 443–450.

Sutskever, Ilya, Oriol Vinyals, and Quoc V Le (2014). "Sequence to sequence learning with neural networks". In: *Advances in neural information processing systems* 27.

Szegedy, Christian, Sergey Ioffe, et al. (2017). "Inception-v4, inception-resnet and the impact of residual connections on learning". In: *Thirty-first AAAI conference on artificial intelligence*.

Szegedy, Christian, Vincent Vanhoucke, et al. (2016). "Rethinking the inception architecture for computer vision". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826.

Tan, Zheng-Hua, Najim Dehak, et al. (2020). "rVAD: An unsupervised segment-based robust voice activity detection method". In: *Computer speech & language* 59, pp. 1–21.

Tang, Hui, Ke Chen, and Kui Jia (2020). "Unsupervised domain adaptation via structurally regularized deep clustering". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8725–8735.

Tang, Hui and Kui Jia (2020). "Discriminative adversarial domain adaptation". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04, pp. 5940–5947.

*The Remote Connectivity Software* (Mar. 2023). URL: `https://www.teamviewer.com`.

Triantafyllopoulos, Andreas et al. (2019). "Towards Robust Speech Emotion Recognition Using Deep Residual Networks for Speech Enhancement". In: *Proc. Interspeech 2019*, pp. 1691–1695.

Trigeorgis, George et al. (2016). "Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network". In: *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, pp. 5200–5204.

Tzeng, Eric, Judy Hoffman, Trevor Darrell, et al. (2015). "Simultaneous deep transfer across domains and tasks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 4068–4076.

Tzeng, Eric, Judy Hoffman, Kate Saenko, et al. (2017). "Adversarial discriminative domain adaptation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7167–7176.

Tzeng, Eric, Judy Hoffman, Ning Zhang, et al. (2014). "Deep domain confusion: Maximizing for domain invariance". In: *arXiv preprint arXiv:1412.3474*.

Venkateswara, Hemanth et al. (2017). "Deep hashing network for unsupervised domain adaptation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5018–5027.

Von Rueden, Laura et al. (2019). “Informed Machine Learning–A Taxonomy and Survey of Integrating Knowledge into Learning Systems”. In: *arXiv preprint arXiv:1903.12394*.

Vrebčević, N, I Mijić, and D Petrinović (2019). “Emotion Classification Based on Convolutional Neural Network Using Speech Data”. In: *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, pp. 1007–1012.

Wahab, Abdul et al. (2021). “DNA sequences performs as natural language processing by exploiting deep learning algorithm for the identification of N4-methylcytosine”. In: *Scientific reports* 11.1, pp. 1–9.

Wang, Jinjiang et al. (2020). “Physics guided neural network for machining tool wear prediction”. In: *Journal of Manufacturing Systems* 57, pp. 298–310.

Wang, Kunxia et al. (2015). “Speech emotion recognition using Fourier parameters”. In: *IEEE Transactions on Affective Computing* 6.1, pp. 69–75.

Wang, Wei et al. (2020). “Rethink Maximum Mean Discrepancy for Domain Adaptation”. In: *arXiv preprint arXiv:2007.00689*.

Wang, Weiqing, Xiaoyi Qin, and Ming Li (2022). “Cross-Channel Attention-Based Target Speaker Voice Activity Detection: Experimental Results for the M2met Challenge”. In: *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 9171–9175.

Wang, Xiaoyang et al. (2020). “Traffic flow prediction via spatial temporal graph neural network”. In: *Proceedings of the web conference 2020*, pp. 1082–1092.

Wen, Siyang et al. (2022). “Rotated Object Detection via Scale-invariant Mahalanobis Distance in Aerial Images”. In: *arXiv preprint arXiv:2204.00840*.

Wijayasingha, Lahiru and John A Stankovic (2021). “Robustness to noise for speech emotion classification using CNNs and attention mechanisms”. In: *Smart Health* 19, p. 100165.

Wu, Lingfei et al. (2021). “Deep learning on graphs for natural language processing”. In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 2651–2653.

Wu, Yuan, Diana Inkpen, and Ahmed El-Roby (2020). “Dual mixup regularized learning for adversarial domain adaptation”. In: *European Conference on Computer Vision*. Springer, pp. 540–555.

Wu, Zonghan, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, et al. (2020). “Connecting the dots: Multivariate time series forecasting with graph neural networks”. In: *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 753–763.

Wu, Zonghan, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang (2019). "Graph wavenet for deep spatial-temporal graph modeling". In: *arXiv preprint arXiv:1906.00121*.

Xie, Shaoan et al. (2018). "Learning semantic representations for unsupervised domain adaptation". In: *International conference on machine learning*. PMLR, pp. 5423–5432.

Xu, Minghao et al. (2020). "Adversarial domain adaptation with domain mixup". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04, pp. 6502–6509.

Yan, Shen et al. (2020). "Improve unsupervised domain adaptation with mixup training". In: *arXiv preprint arXiv:2001.00677*.

Zamil, Adib Ashfaq A et al. (2019). "Emotion Detection from Speech Signals using Voting Mechanism on Classified Frames". In: *2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*. IEEE, pp. 281–285.

Zhang, Lei et al. (2019). "Manifold criterion guided transfer learning via intermediate domain generation". In: *IEEE transactions on neural networks and learning systems* 30.12, pp. 3759–3773.

Zhang, Yabin et al. (2019). "Domain-symmetric networks for adversarial domain adaptation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5031–5040.

Zhao, Han et al. (2019). "On learning invariant representations for domain adaptation". In: *International Conference on Machine Learning*. PMLR, pp. 7523–7532.

Zhao, Mingmin, Fadel Adib, and Dina Katabi (2016). "Emotion recognition using wireless signals". In: *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, pp. 95–108.

Zhao, Shutao, Zhengjun Qiu, and Yong He (2021). "Transfer learning strategy for plastic pollution detection in soil: Calibration transfer from high-throughput HSI system to NIR sensor". In: *Chemosphere* 272, p. 129908.

Zheng, Chuanpan et al. (2020). "Gman: A graph multi-attention network for traffic prediction". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 34. 01, pp. 1234–1241.

Zhu, Jun-Yan et al. (2017). "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2223–2232.

Zhu, Yi and Shawn Newsam (2017). "Densenet for dense flow". In: *2017 IEEE international conference on image processing (ICIP)*. IEEE, pp. 790–794.

Zuo, Yukun et al. (2021). “Margin-based adversarial joint alignment domain adaptation”. In: *IEEE Transactions on Circuits and Systems for Video Technology*.