Enhancing Interpretability in Autonomous Driving Models Through the Use of Concept Bottleneck Layers

---

A

Thesis

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

---

in partial fulfillment

of the requirements for the degree

Master of Science

by

Weiheng Peng

May 2025

APPROVAL SHEET

This

Thesis

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

Master of Science

Author: Weiheng Peng

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

Advisor: Yen-Ling Kuo

Advisor:

Committee Member: Matthew B. Dwyer

Committee Member: Miaomiao Zhang

Committee Member:

Committee Member:

Committee Member:

Committee Member:

Accepted for the School of Engineering and Applied Science:

Jennifer L. West, School of Engineering and Applied Science

May 2025

Abstract

Newer and more complex neural networks are being developed each day, these systems continue to grow more intelligent and better at doing specific tasks. Yet, ever since the introduction of deep neural networks (DNN), the black-box nature of such models makes it harder to understand or justify the actions predicted. One of the key limitations of autonomous driving systems is the absence of reasoning that users can easily interpret. This lack of clarity becomes especially problematic in high-stakes, safety-critical scenarios such as when the vehicle encounters difficult road conditions. In these scenarios, understanding how and why specific actions were taken could reassure the driver not to worry.

This thesis explores the application of concept bottleneck layers (CBLs) to pre-trained autonomous driving models, enabling the extraction of the reasoning behind each predicted control signal. In addition, it presents a method to generate training data for the CBL with little manual effort. In this framework, vision grounding and segmentation models are used to extract information from each frame. These extractions are integrated to generate concept labels for the dataset used in training the original model. This semi-automatic approach enables frequent concept adjustment without requiring manual data relabeling or recollection.

To illustrate the feasibility of such an approach, experiments are conducted on a transformer-based autonomous driving model with a set of extracted concepts for explanation. The results demonstrate that the proposed approach can maintain similar or better accuracy than the original model while providing a certain level of interpretability. The flexible structure of concept generation allows it to handle as many concepts as needed as well as incorporate more complex concepts. This scalability paves the way for future enhancements and refinements.

Acknowledgments

I would like to express my gratitude to everyone who has supported me throughout my Master's studies and contributed in various ways to complete this thesis. This journey has been both challenging and rewarding, and I deeply appreciate all the encouragement and help I received along the way.

First and foremost, I would like to thank my advisor, Professor Yen-Ling Kuo, for her continuous guidance, insightful feedback, and patience throughout the entire research process. Her expertise and thoughtful suggestions helped shape this thesis and kept me focused and motivated, even during difficult times. I am truly grateful for her mentorship and support.

I am also sincerely thankful to the University of Virginia and the Computer Science Department for providing me with a supportive academic environment and the opportunity to pursue my studies. Special thanks to Jai Maupin, who was incredibly helpful in coordinating all the administrative communications. I am also grateful to Professor Felix Lin for always being available to answer my program-related questions and for helping me navigate the academic process smoothly.

I would like to express my appreciation to the members of my thesis committee, Professor Matthew B. Dwyer and Professor Miaomiao Zhang, for their time, attention, and valuable feedback. Their perspectives and suggestions have helped improve the quality and clarity of my work, and I am thankful for their willingness to support my research.

To my friends, thank you for your constant encouragement, for being a source of positivity, and for sharing this academic journey with me. Your friendship, understanding, and the moments of laughter, games, and distraction helped me stay balanced and made the experience far more enjoyable and meaningful.

Most importantly, I would like to thank my family for their unwavering support, love, and belief in me. Your encouragement gave me strength when I needed it the most, and your sacrifices made it possible for me to pursue this degree. I owe this accomplishment to you.

Thank you all!

Table of Contents

1 Introduction

1.1 Background and motivation

Deep neural networks (DNNs) have shown significant improvement in recent years, leading to breakthroughs across various domains such as computer vision, natural language processing, etc. The key components behind this success include the advancement in hardware, training techniques, and most importantly, the dramatic increase in model size [44, 53]. It is now easy for a modern DNN to contain billions or even trillions of parameters, allowing them to learn complex patterns from an enormous amount of training data. As illustrated in Fig. 1, we can observe the exponential growth in trainable parameters as new models are released. Various scaling laws [22] have been proposed to describe the effect of larger models and larger data on their performance.

The rapid growth in model size brings both advantages and drawbacks. On the one hand, state-of-the-art models have demonstrated remarkable improvements in performance across a wide range of tasks, from natural language processing [63] to computer vision [60], providing more accurate predictions with robust performance-even in worst-case scenarios [35]. On the other hand, the increase in parameters introduces several challenges. Larger models demand significantly more data and computational resources, increasing training time and higher costs. More importantly, people are concerned about their lack of interpretability–these models are frequently referred to as ”black boxes”, making it increasingly difficult to understand the rationale behind their decisions. This lack of interpretability raises concerns, especially in high-risk, high-stakes domains such as finance, healthcare, autonomous vehicles, and military applications [29].

Autonomous vehicles have surged in popularity over the past decade as they have taken advantage of the rapid advancements in neural network architectures and designs. Models ranging from earlier convolutional neural networks (CNNs) to more recent adoption of deep neural networks such as transformers have significantly enhanced the perception, decision-making, and control capabilities of these systems. However, as these models grow in complexity, so do concerns about their safety, ethics, and trustworthiness. The lack of transparency makes it difficult to fully understand the reasoning behind critical decisions made by these systems–especially in life-or-death scenarios–leaving many hesitant to fully entrust their safety to such vehicles.

Control is one of the most important components in autonomous driving systems, as it directly affects the vehicle’s ability to navigate safely, smoothly, and effectively in dynamic environments. A

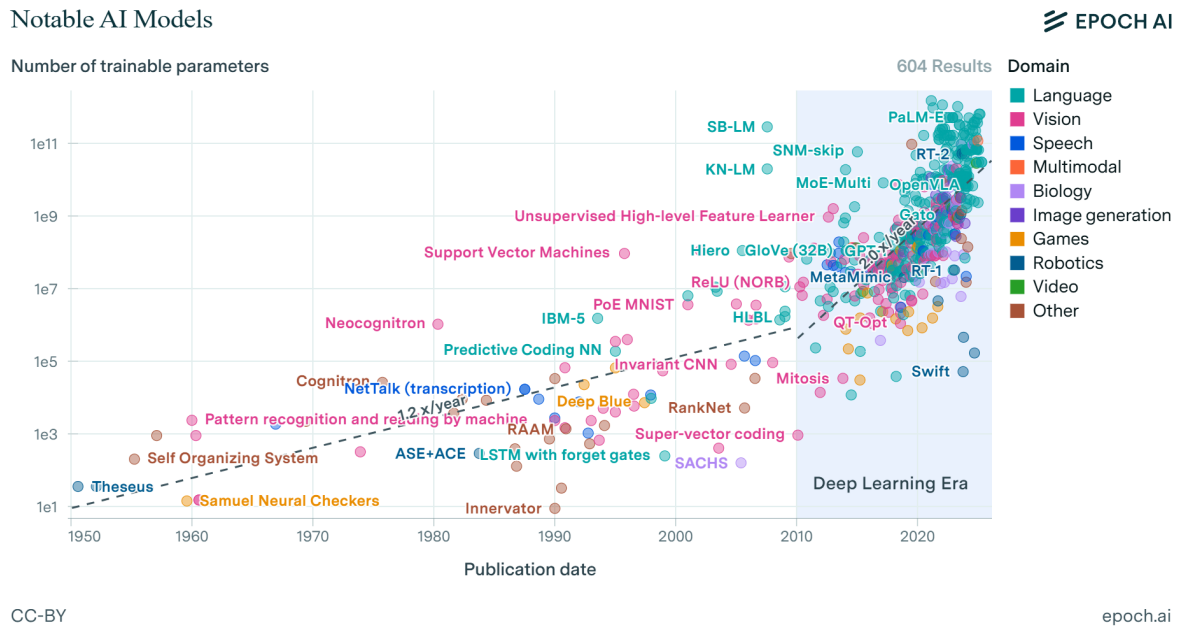


Fig. 1. Parameter size of popular models over time. (created by [14])

reliable control network enables decisions made by other modules, such as perception and planning, to be translated into accurate physical actions, including steering, braking, and accelerating. Given its central role, enhancing the interpretability of the control system can provide several key benefits. It offers clear justifications for the vehicle's driving actions; helps identify salient features that indicate what input features the model is prioritizing; and enables real-time debugging and monitoring to detect anomalies or failures as they occur. Together, these capabilities contribute to a more transparent and accountable system, improving driver-model coordination by clarifying when and why human intervention may be necessary. Ultimately, such transparency is at the heart of what society values most.

1.2 Aims and objectives

In this work, we will present a method to improve the interpretability of pre-trained autonomous driving models through the application of concept bottleneck layers (CBLs). In simplified terms, a specialized layer will be inserted into the model where each node of this layer corresponds to a human-understandable concept. The pre-trained model takes in a consecutive sequence of images and the previous control signal as input and predicts the next control signal. In this thesis, we refer control signal as a pair of values representing the velocity and turning angle of the vehicle. The model's performance on the test set is then used as a baseline to compare with our modified models to determine their ability to maintain accuracy while enhancing interpretability. The paper also explores three different setups of the CBL layer and compares their effectiveness.

In addition to the proposed model, we will introduce a novel way of concept generation using vision grounding and segmentation models for a fast and efficient way of providing training data for the CBL as opposed to manual collection or labeling. Together, we demonstrate a framework that would be able to apply CBL on any vision-based autonomous driving model and train it without manual concept labeling. We would also offer ways to interpret the activation of concepts and how they would affect the control signal prediction.

1.3 Thesis structure

The rest of the paper will be structured as follows. Chapter 2 introduces related works, discussing previous attempts in various ways to improve interpretability in various settings. In addition, it covers information about relevant models and works that will be used in this paper. The proposed model and training method will be presented in Chapter 3. It contains details about model design and concept generation. Chapter 4 will first provide a brief description of the dataset used to train the base model as well as any pre-processing and hyperparameters used. Then we will present various tests and metrics used to evaluate different variations of the CBL and discuss the findings from the experiments. Finally, Chapter 5 closes the thesis and suggests directions for future work.

2 Related Works

2.1 Explanation in Autonomous Driving

Some prior work in this area employs visualization techniques such as attention map [25], optical flow [54], or transformation [43] to interpret neural network decisions. However, these visualizations usually require the user to have extensive knowledge to understand their meaning. In addition, the time required to infer information from these visualizations may be too long to be useful in real-world driving scenarios.

Another approach to tackling such problems is through the use of language, particularly given the rapid advancements in language models following the introduction of transformers. These developments have significantly improved the ability of models to understand, generate, and respond to natural language with greater accuracy and efficiency. Modern large language models (LLMs) demonstrate a certain level of contextual understanding and reasoning, making them promising tools for generating explanations across various scenarios. Recent work [12] has extended these capabilities to the multimodal domain, where vision-enabled LLMs can interpret visual content [4], describe visual inputs [18], and provide spatially grounded explanations [8]. Studies have shown that such multimodal LLMs can support tasks like interpreting a vehicle's surroundings [56], explaining its actions [21], assessing potential risks [31], and even planning future trajectories [50].

However, many of these models rely on additional fine-tuning [19, 51, 62] or architectural modifications [45]–such as language adapters or residual pathways–to achieve task-specific reasoning. They often require large amounts of training data as well as extensive prompt engineering to elicit desirable responses and behavior [56]. Therefore, future research is needed to make these models more adaptable and efficient.

2.2 Interpretability in Neural Networks

Numerous prior studies [40] have explored interpretability in neural networks. A variety of approaches have been developed to offer unique insights into model behavior. Feature attribution methods, such as saliency maps [6] and integrated gradients [48], focus on identifying which input features most influence a model's predictions. Example-based approaches, like local interpretable model-agnostic explanations (LIME) [42], generate local approximations of a model's behavior around specific instances. In output analysis, methods including feature ablation [59], partial dependency plots [15], and counterfactual explanations [52] examine how changes in the input features affect the model's overall decisions. The neural activation approach focuses on interpreting the model at a lower level

through methods like activation maximization [57], network dissection [2], or testing With concept activation vectors (TCAV) [24], which aim to reveal which features or concepts a neural network is using during its decision-making process. Lastly, visualization tools such as Lucid [16] enable the direct visualization of neural network representations, providing insights into how the model operates internally.

Concept-based models represent one approach to map human-comprehensible concepts to unknown meanings extracted by the neural network [27]. In particular, a concept bottleneck layer (CBL) is inserted before the prediction head, where each node in the CBL is associated with a predefined concept. During training, only the weights associated with the layer are trained while the rest is frozen. This design produces a lightweight and efficient training process. Since the model does not need to learn complex feature representations from the beginning, it simply adjusts the mapping from the last hidden layer to the CBL. This significantly reduces the computational time and cost required and it is also easily adaptable to various base model architectures. In addition, the weights connecting the CBL to the prediction layer provide insights into how much each concept contributes to each predicted class. The magnitude of these weights indicates the strength of the influence of that concept, while the sign indicates whether the concept has a positive or negative correlation with the prediction. However, most models that employ this structure are based on CNN or LLM [33, 58], where the outputs are within a fixed set.

In the original concept bottleneck model, all information is forced to pass through the bottleneck layer to associate with each individual concept. However, the performance of such a model depends on the quality of the concepts generated. In fact, it is very difficult to summarize an extensive list of concepts that limit the performance of CBM [58]. Various strategies have been proposed to address this limitation, including the use of residual layer [46, 58]. In this paper, we also evaluate the effect of CBL with different levels of residual connections in our setup.

One issue that remains for such a model is the need for data to train the CBL. Since concepts are not decided initially (while training the original model) and can change frequently depending on the task, the effort required for data collection is still significant. Previous works have demonstrated the ability to generate relevant concepts dynamically using large language models and to utilize a vision language model like CLIP [39] to extract concept labels from training data [33]. Their framework eliminates the need for manually determining the list of concepts as well as the work required to label training data for the CBM. Hence, dramatically improves the efficiency of training such models. In their framework, the CBL weight is learned by transferring the activation of the concept when the input images are passed through CLIP, and a sparse final layer is used to predict the object. Their work demonstrates the viability of this approach for image classification tasks, which inspires our work

to do something similar.

A previous study has already tried to incorporate the concept bottleneck model and the automatic training process into an autonomous driving model. Echterhoff et al. demonstrates a similar way of generating concepts relevant for driving from GPT-3.5 [34], and utilizing CLIP to calculate the similarity between each individual concept and each frame. In their model, the concept activation is then combined with various visual backbones, including ResNet [17], ViT [11], etc, and passed through a Longformer [3] architecture before making the final prediction. Their work claims that the activated concept can be used to explain the actions behind their model. However, their work focuses solely on mapping activated concepts to predicted control variables by training a LongFormer model, which does not guarantee that these concepts are actually used in the final decision-making process. Moreover, since CLIP is trained for the image classification task, it is unable to extract many temporal concepts related to autonomous driving, which further hinders its performance. To address these issues, our work will try to not only find a better way to predict temporal concepts across frames but also explain the effect of the activated concept on the final control signal prediction.

2.3 End-to-End Autonomous Driving

End-to-end autonomous driving models are aimed at taking various sensor inputs (e.g., camera, LiDAR) and mapping them to control signals for the autonomous vehicle [7]. A typical architecture for such models consists of a vision backbone and a prediction head. The vision backbone refers to the core neural network used for feature extraction from input data (such as video) as well as making critical decisions. Earlier work in this area [36] utilized convolutional neural networks (CNN) to predict vehicle control from images and range sensor data. However, performance was limited due to the lack of temporal dependency in the model. With the advancement of new neural network architectures, recent studies have incorporated a variety of backbones ranging from recurrent neural networks [9, 10], transformers [13, 21], graph neural networks (GNN) [61], and multi-modal networks [37, 55].

Among these, Vision Transformers (ViT) have shown particular promise [28]. ViT divides the input image into fixed-size patches and processes them like a sequence of tokens using self-attention. This enables the model to effectively capture the temporal relationship between patches as well as global contextual information more effectively than previous CNNs[11]. Building on ViT, the Video Vision transformer (ViViT) extends the idea of totalizing patches of images across the entire input frames. This allows the model to not only determine the spatial relationship between objects in each frame but also the temporal relationship between them across multiple frames. As a result, it is able to understand the movements of objects across time which is crucial for tasks like driving that require drivers to foresee danger ahead[1].

3 Method

In this chapter, we will present our framework that implements the concept bottleneck layer (CBL) with a set of customizable concepts and minimal manual effort. Given a pre-trained autonomous driving model with a visual backbone, our framework is able to integrate and train the CBL applied to the model without the need to manually label concepts from the dataset in the following steps:

Step 1 Define the list of concepts that may affect driving behavior.

Step 2 Generate concept labels from training data.

Step 3 Insert a CBL into the base model.

Step 4 Learn the weights for CBL.

Step 5 Learn the weights for the prediction head (and residual layer).

Given the abundance of autonomous driving models capable of predicting multiple control commands and/or over several future time steps, we chose to simplify our approach for the purpose of this thesis. In particular, we will train a custom base model that focuses on the sole prediction of control signals (velocity and turning angle) for one-time steps only. Having a custom model not only provides for greater control over the architecture and training process but also allows more efficient experimentation and iteration by removing unnecessary components.

We will cover the model design of the base model in Chapter 3.1, explaining the architecture and implementation details as well as the training procedure. Chapter 3.2 will include the details of creating the list of concepts and generating them from training data for **Step 1** and **2**. Afterwards, we will discuss how CBL is applied to the base model (**Step 3**) in Chapter 3.3. Finally, we will introduce the procedure to train the CBL (**Step 4**) and final model (**Step 5**) in Chapter 3.4.

3.1 Base Model

3.1.1 Design

Our base model takes inspiration from other works [13, 21] that utilize a video transformer network [1, 32] as the backbone to encode the input video into a latent space. As illustrated in Fig. 2, we use the ViViT model [1] pre-trained on the Kinetic-400 dataset [23] as the visual backbone. To adapt this

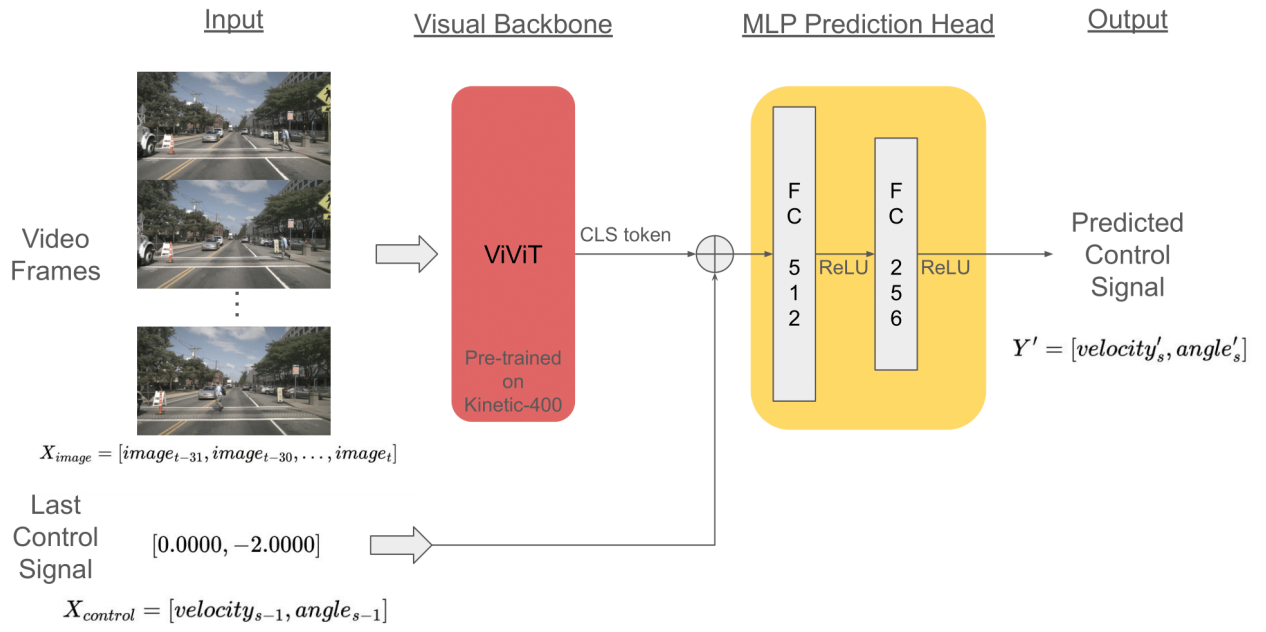


Fig. 2. Structure of the base model. The CLS token embedding is obtained by passing input frames through ViViT, which is then concatenated with the last control signal before passing an MLP where it predicts the final control signal.

model to our prediction task, we replace the last fully connected (FC) layer for human action prediction with a customized multi-layered perceptron (MLP) that predicts the control signal (velocity and turning angle). In addition, we also concatenate the last control signal with the [CLS] token embedding before they pass through the MLP, allowing the model to use previous control signals as extra features to aid its decision. Therefore, our model takes in 32 image frames with dimension $224 \times 224 \times 3$, passing them through the ViViT to obtain the CLS token embedding of dimension 768, concatenating with a 2-dimensional last control signal input, and finally predicts the final control signal after passing through the MLP.

We chose the ViViT as the backbone of our model for various reasons: First of all, as mentioned earlier, ViViT utilizes the self-attention mechanism of transformers to capture long-range dependencies across both spatial and temporal dimensions. This allows the model to not only understand the relationship between objects within each frame but also track the movement of objects and their interactions over time, which is essential for dynamic environments like autonomous driving. Moreover, ViViT has demonstrated relatively strong performance on various benchmarks, making it a reliable and compelling choice for our task. Furthermore, with ViViT's modular and well-documented architecture design and its popularity in the field, it is very easy for us to implement and integrate it into our own model.

3.1.2 Training

The base model is trained using supervised learning. For each input X with corresponding control signal $Y = (V, A)$, the model predicts a control signal $Y' = (V', A')$. To calculate the loss, we first transform both label and prediction into horizontal velocity V_h, V'_h, and vertical velocity V_v, V'_v corresponding to the following equations:

$$V_h = V \cdot \sin A$$
$$V_v = V \cdot \cos A$$
$$V'_h = V' \cdot \sin A'$$
$$V'_v = V' \cdot \cos A'$$

Then the total loss L is computed as $L = RMSE(V_h, V'_h) + RMSE(V_v, V'_v) + RMSE(A, A')$ where $RMSE$ is the root mean squared error calculated as:

$$RMSE(Z, Z') = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(z_i - z'_i)^2}$$

The n is the batch size. The transformation allows the turning angle and velocity to be converted to the same scale for better results. The ablation study (in Appendix C) shows that the inclusion of RMSE in the turning angle resulted in better performance.

3.2 Concepts Generation

Inspired by previous works that use pre-trained models such as CLIP to extract open-vocabulary concepts [49], we want to find a way to allow concept generation using a flexible set of concepts as well as an easy way to generate training data for the CBL without much manual effort.

3.2.1 Step 1: Define the list of concepts

In the original CBM paper [27], the set of concepts is determined by domain experts based on their perceived importance to the task. However, for the thesis, our focus is on evaluating the validity of the CBL structure in our model rather than optimizing for the best possible prediction. Therefore, we selected a few concepts we believe are relevant to driving. We first separate concepts into two types: boolean and numerical. The prior is simply a true or false question in the following format:

Is there [object] within the last [duration] of the input video?

[object] in this case refers to things that we can see from the input image, like a pedestrian, a car, etc. One example of such a concept used in our model is “Is there any human within the last 1 second of the input video?”. This concept is simply based on the intuition that drivers may pay more attention and drive slowly if they see people nearby. The latter type expects some numerical results in the following format:

Average/Total [measurement] of [object] within the last [duration] of the input video.

In this case, [measurement] can vary from ”number” - refer to the amount, “distance” - between pixels, “score” - some percentage, etc. Some concepts used in our model under this category include: “Average number of vehicles within the last 1 second of the input video”, and “Average vertical distance (measured in pixels) to the closest vehicle within the last 1 second of the input video”. We believe that the first concept may be a measure of congestion level and hence affect the velocity of the vehicle, while the second relates to the following distance, which also affects the speed of the vehicle.

Our list of concepts is structured as follows: 4 binary concepts relating to the existence of humans or cars/vehicles within the last 2 or 1 second of the video; 4 numerical concepts relating to the number of humans or cars/vehicles within the last 2 or 1 seconds; 2 numerical concepts relating to color of traffic light within the last 0.5 seconds; 1 numerical concept measures the horizontal movement of human within the last 2 seconds; 4 numerical concepts measuring the distance to closest human or car/vehicle within the last 1 second; 1 numerical concept measuring size of closest car/vehicle within the last 1 second. The full list of concepts we used and the reasons can be seen in the Appendix A.

3.2.2 Step 2: Generate Concept Label

Since our list of concepts associates with objects across different frames, a vision grounding and segmentation tool is needed. We used Grounded SAM 2’s [20] tracking with a continuous id feature. The model takes in a video input along with a grounding text prompt and generates bounding boxes of objects for each frame, as well as assigning an object ID for each object, which can be used to track them through multiple frames. Compared to other vision grounding tools [30, 41], Grounded SAM 2 not only allows the use of open vocabulary to track objects but also makes it easy to detect the movement of objects.

The following are our steps to generate concepts from the training data. For demonstration purposes, we include how we obtain the concept “Average score of green traffic light in last 0.5 seconds”:

1. Determine the list of concepts and the physical objects that are relevant to the concepts. In this case, we just need to focus on one object: the traffic light.

2. For each scene, pass the item to track into the Grounded SAM 2 model to obtain object IDs and their bounding boxes for each frame. In our case, we prompt Grounded SAM 2 with “traffic light” and run for all training data.

3. For each data point, calculate a score/measure for each concept according to measurements obtained in the previous step. In this example, since we are only interested in the last 0.5 seconds, we only look at the last 6 frames of the 32 consecutive frames. First, we focus on the important traffic lights located at the center of the frame, filtering out others. Then, for each traffic light, we calculate the percentage of green pixels over the entire bounding box. Lastly, we average the value across all frames.

4. Normalize the score/measure obtained across all training scenes. This step is only taken if the concept is numerical, which allows different concepts to be transformed into the same scale for balanced loss in training and easier evaluation. To do so, we calculate the mean μ_i and standard deviation σ_i for each concept i on the training set. Then we calculate the standard score (z-score) z_i from the original activation a_i as:

$$a_i = \frac{a_i - \mu_i}{\sigma_i}$$

The resulting distribution should have a mean of 0 and a standard deviation of 1.

In the actual concept generation, the [object] prompt provided to Grounded SAM 2 may affect the performance of the bounding box generation. For instance, our experiment found that prompting with “road” in addition to concepts like “human” leads to more accurate results than “human” alone. In addition, to mitigate the effect of wrong prediction from Grounded SAM 2, we only considered objects that appear for more than half of [duration]. This approach is based on the observation that misclassifications only happen in a few frames compared to correctly classified objects. Chapter 4 will contain an evaluation of our concepts on the actual dataset.

Fig. 3 contains an example containing a few frames from the output of Grounded SAM 2 when prompted with “road” and “traffic light”. In this example, we can clearly see that the Grounded SAM 2 is able to mark all the traffic lights it detected from each frame, even when the traffic light is not facing the ego vehicle. Therefore, we chose to filter only the top center region when generating traffic light-related concepts. Furthermore, we can see that across the 3 frames displayed, Grounded SAM 2 labels the same objects with the same ID and color which allows for tracking across frames. It is also worth noting that sometimes the object is lost and a new ID is assigned, hence it may have an impact on the final performance.

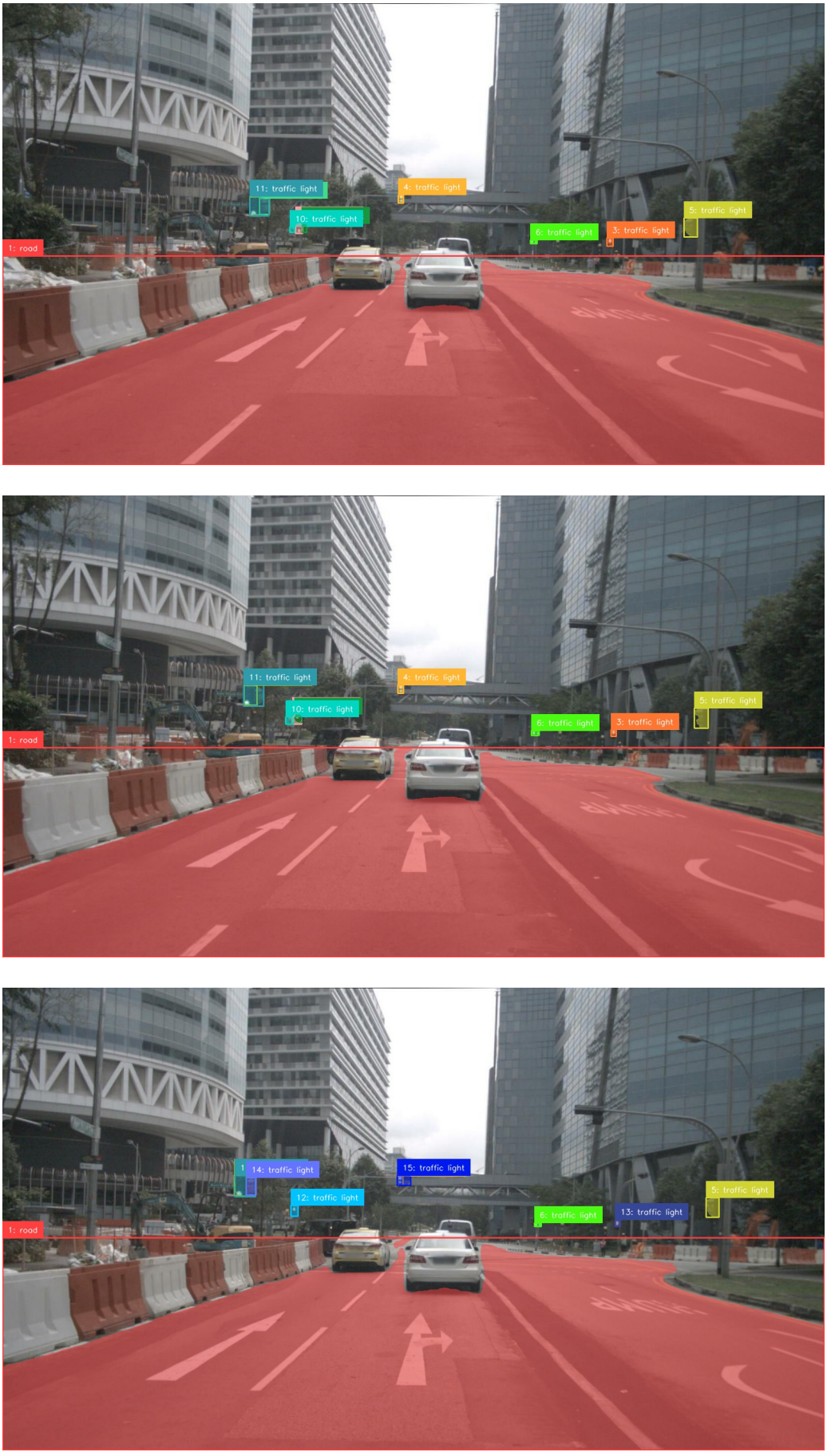


Fig. 3. Results of prompting ”road, traffic light”

3.3 Step 3: Concept Bottleneck Layer Design

In most CBM papers [27, 33, 46, 58], the concept bottleneck layer is inserted between the final hidden layer and the output layer. However, this may not be suitable for a regression task as a linear mapping between concepts and control signal is seemingly unlikely even with the addition of residual. Therefore, the CBL in our model is inserted between the final layer of ViViT and the beginning of the MLP. We believe that this will enable the non-linearity of MLP to better learn and predict the expected control signals.

To understand how residuals affect the behavior of our model, we have designed three variations of how CBL was applied to our model to determine their effects. Despite the variations, all the CBL layers have 16 nodes corresponding to the list of concepts we decided on earlier.

3.3.1 Variation 1: Traditional CBL

As shown in Fig. 4a, this is the implementation of a conventional CBL where all information is forced through the bottleneck. This design ensures that only concepts and the last control signal are the only factors affecting the final prediction. Therefore, each individual concept is more likely to have an effect on the final result. We will use this variation as a baseline to compare with the other 2 variations.

3.3.2 Variation 2: CBL with Residual

Displayed in Fig. 4b, this variation incorporates the CBL alongside a residual layer of the same dimension which is 16 in this case. The addition of residual provides a pathway for information to bypass the bottleneck, allowing latent features that cannot be described by the list of features to still have an effect on the final decision while retaining the effect of CBL. Papers [46, 58] have shown that the addition of residuals is able to result in better prediction than those without. In this thesis, we would also like to compare with the base model to determine if this will also apply to our task.

3.3.3 Variation 3: CBL with Original Mapping

The last variation of CBL can be seen as an extension to variation 2, where instead of a residual layer, we retain the original connection between ViViT to MLP in addition to the connection from CBL to MLP. Illustrated in Fig. 4c, the CLS token embedding is concatenated with CBL output as well as the last control signal and passed into the MLP. Therefore, the CBL functions as a specialized feature extractor, extracting interpretable concepts from the latent space and utilizing them to aid the

prediction. Compared to variation 2, we believe that this setup can result in better accuracy on the final prediction due to the increase in the information that bypasses the CBL. However, at the same time, each concept may have the least effect on the final outcome for the same reason among the 3 variations.

3.4 Training

Following the steps proposed by the original paper [27], training is separated into two parts: training CBL and training prediction head to map from CBL to the final prediction. This approach enables the model to first adjust the mapping from the last hidden layer to the CBL, then use it for prediction.

3.4.1 Step 4: Training CBL

To train the CBL, we first freeze the backbone(ViViT)'s weight. Then we calculate the loss between the predicted concept from the base model and the true concept generated in the previous steps. The loss calculation is separated into two categories depending on the type of concept. For boolean concepts like "Is there any human within the last 1 second of the input video?", we use binary cross-entropy error (BCE) with the label. Given the true label $Y = \{y_i | y_i \in \{0, 1\}\}$ and predicted probabilities $Y' = \{y'_i | 0 \leq y'_i \leq 1\}$, BCE is calculated as:

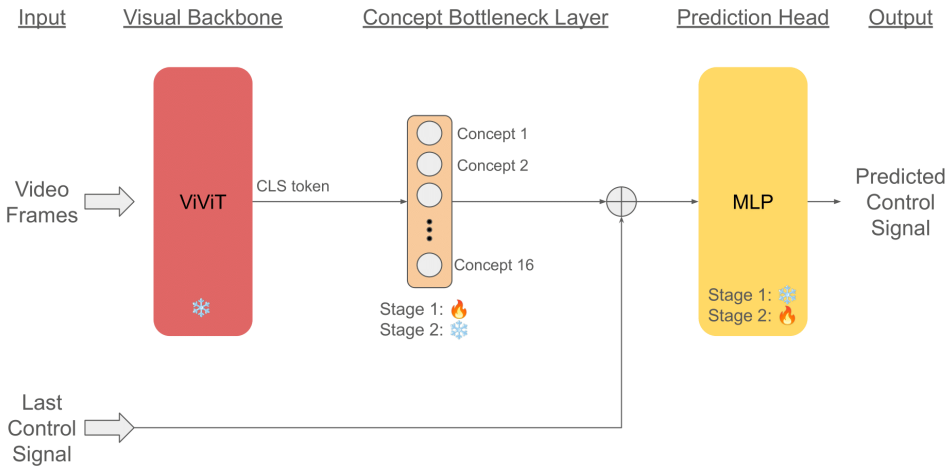
$$BCE(Y, Y') = -\frac{1}{n}\sum_{i=1}^{n}[y_i \cdot \log(y'_i) + (1 - y_i) \cdot \log(1 - y'_i)]$$

Where n is the number of samples within each batch. For numerical concepts such as "Average score of traffic light (in the top center region) being green within the last 0.5 seconds," we calculate the RMSE compared to the labels generated.

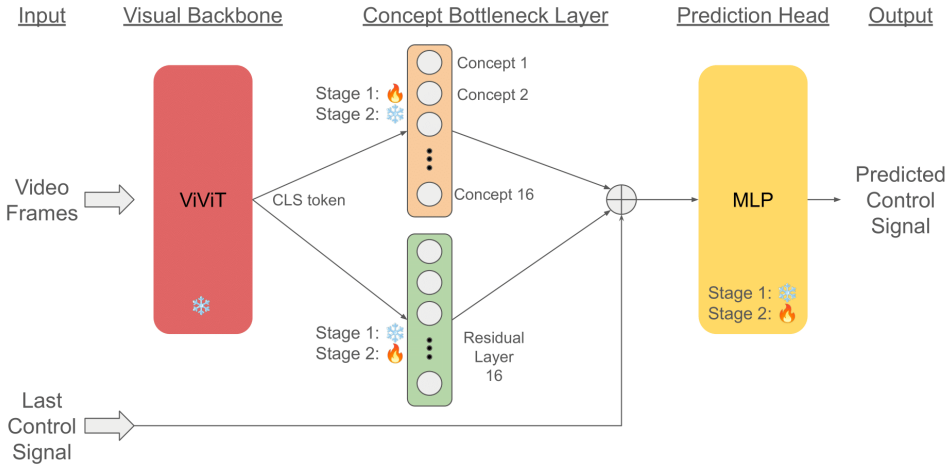
Since the CBL in all three variations refers to the same set of concepts and is all mapped from the CLS token produced by ViViT. We only need to train CBL for one variation and apply the same weights to the other variations.

3.4.2 Step 5: Training MLP Prediction Head (and Residual)

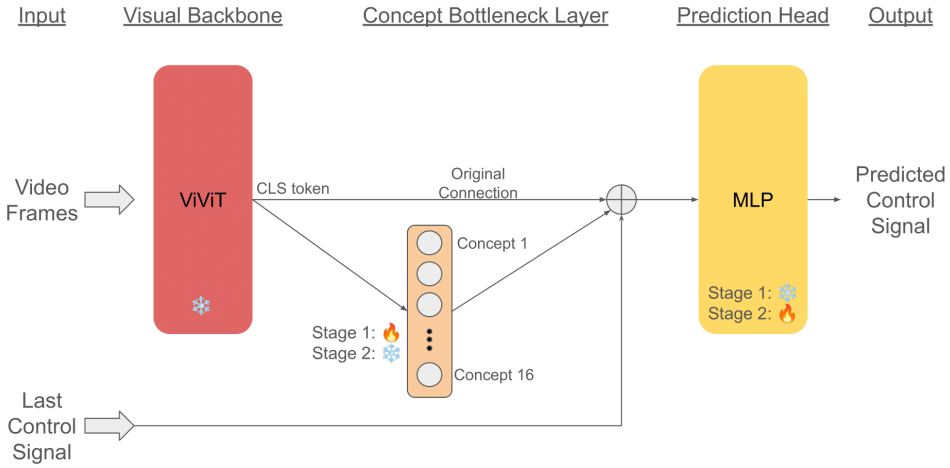
After the loss converges in the previous step, we will begin the second part of the training. In this step, in addition to freezing the backbone, we will also freeze the CBL weight that was just trained to prevent it from diverging from concept activations. Therefore, only the prediction head (MLP) and residual (if applicable) are trained. For the loss calculation, we use the same method as our base model training in subsubsection 3.1.2 as it leads to the best performance.



(a) Variation 1: Traditional CBL



(b) Variation 2: CBL with Residual



(c) Variation 3: CBL with Original Mapping

Fig. 4. Three CBL designs (a) Traditional CBL, (b) CBL with Residual, (c) CBL with Original Mapping

4 Experiment and Results

In this chapter, we first discuss the training process, hyperparameters used, as well as the dataset that we used to train our base model. Then we assess the generated concept labels from this dataset using our proposed method. Afterward, we evaluate the performance of our model after the application of CBL in terms of control signal prediction and its interpretability in terms of concept prediction. Last but not least, we compare the three variations of our model and determine each of their strengths and weaknesses.

4.1 Dataset

We use the NuScenes dataset [5] both for training the base model and to generate concept labels to train our model after the application of CBL. The NuScenes dataset is an open-source, large-scale dataset designed for autonomous driving research. It contains 15 hours of driving data which is separated into 1000 scenes, each lasting around 20 seconds. These scenes are carefully selected to cover a wide variety of realistic and challenging conditions, including various road conditions, traffic conditions, weather, time of day, etc. It was collected in both Boston and Singapore, which drive on opposite sides of the road. The dataset includes information collected from 6 cameras, 1 LiDAR, and 5 radars mounted on the ego vehicle, providing an all-around view of the surroundings during driving. At the same time, IMU and GPS information are also provided for better understanding and tracking of the ego vehicle.

For our models, we take the native 12Hz RGB video recorded by the front camera as the primary input. For the control signal, we use the velocity and turning angle in the vehicle motion message released from the CAM bus expansion, recorded at 2Hz. One remaining problem is that these two sensors are not synced, therefore, we match the timestamp of each control signal with the closest that of the frame. Then, for each matched frame, we will treat it as the last frame used for predicting the matching control signal. In other words, each matched frame is an end frame of the 32 consecutive frames we used as the input to our model, and the past control signal before the matched control signal is also used. In our base model as well as all CBMs based on it, only the last control signal is used - meaning that the model accepts the 32 consecutive images $\{image_{t-31}, frame_{t-30}, \ldots, frame_t\}$ along with 1 velocity and 1 turning angle (v_{t-1}, a_{t-1}) and predicts the next velocity and turning angle (v_t, a_t). Since our model is finetuned from the Kinetic-400 dataset, we pre-process every frame with the corresponding pre-processor to the desired shape for our model.

4.2 Training on NuScenes

To train the NuScenes dataset, we first take advantage of the predefined train, validation, and test splits. Then we filter out scenes that do not contain the control signal from the CAN bus expansion. The model is trained using Adam [26] with a learning rate of $1e^{-5}$ and weight decay of $1e^{-5}$ and with a batch size of 8. For each scene in training, we applied the alignment described above and drew one random segment (32 frames, around 2.66s) from all potential endframes, resulting in 8 segments from 8 different scenes for each batch. Then, for validation, we reduce the batch size to 1 but use all the potential endframes each scene has to obtain the loss on all possible validation data. The same is done for the test set to evaluate the performance of each model in the end.

For our base model, we trained it for around 2500 epochs on one A100-80GB GPU until the model mostly converged. Then we trained the CBL layer embedding for about 1700 epochs using the same GPU until convergence. Since all three models use the same mapping from ViViT to the CBL layer, the same learned weights are applied to all three models. Lastly, we trained each variation on the same GPU for around 700, 500, and 900 epochs respectively before converging.

4.3 Generated Concept

To evaluate the quality of the generated concepts from NuScenes, we randomly selected a few scenes from the training data and observed the activation of concepts across the frames. Fig. 5 is an example where we show matched frames from the training video and the generated concept label. We can see that as the ego vehicle approaches a human crossing the road, the activation of Concept 11: "Average horizontal movement of human within the last 2 seconds of the input video" increases. Then, when the human leaves the camera's view (shown in the last two frames), the concept activation drops quickly. We believe such a concept can be used to help the car slow down as it approaches the human and then speed up after the activation lowers.

The second example we selected is a scene involving a traffic light change. As illustrated in Fig. 6, for the first three frames, the traffic light is red, and for the last three frames, the traffic light is green. At the same time, we can observe a similar pattern for activation of Concept 9: "Average score of traffic light (in the top center region) being red + yellow within the last 0.5 second" and Concept 10 "Average score of traffic light (in the top center region) being green within the last 0.5 second". We can see clearly that concept 9 (red light)'s activation is high for the first three frames and goes down quickly afterward. On the other hand, the activation for concept 10 (green light) is low at a red light and high at a green light. This example further demonstrates the success of our concept generation process in terms of extracting the correct label from the dataset.

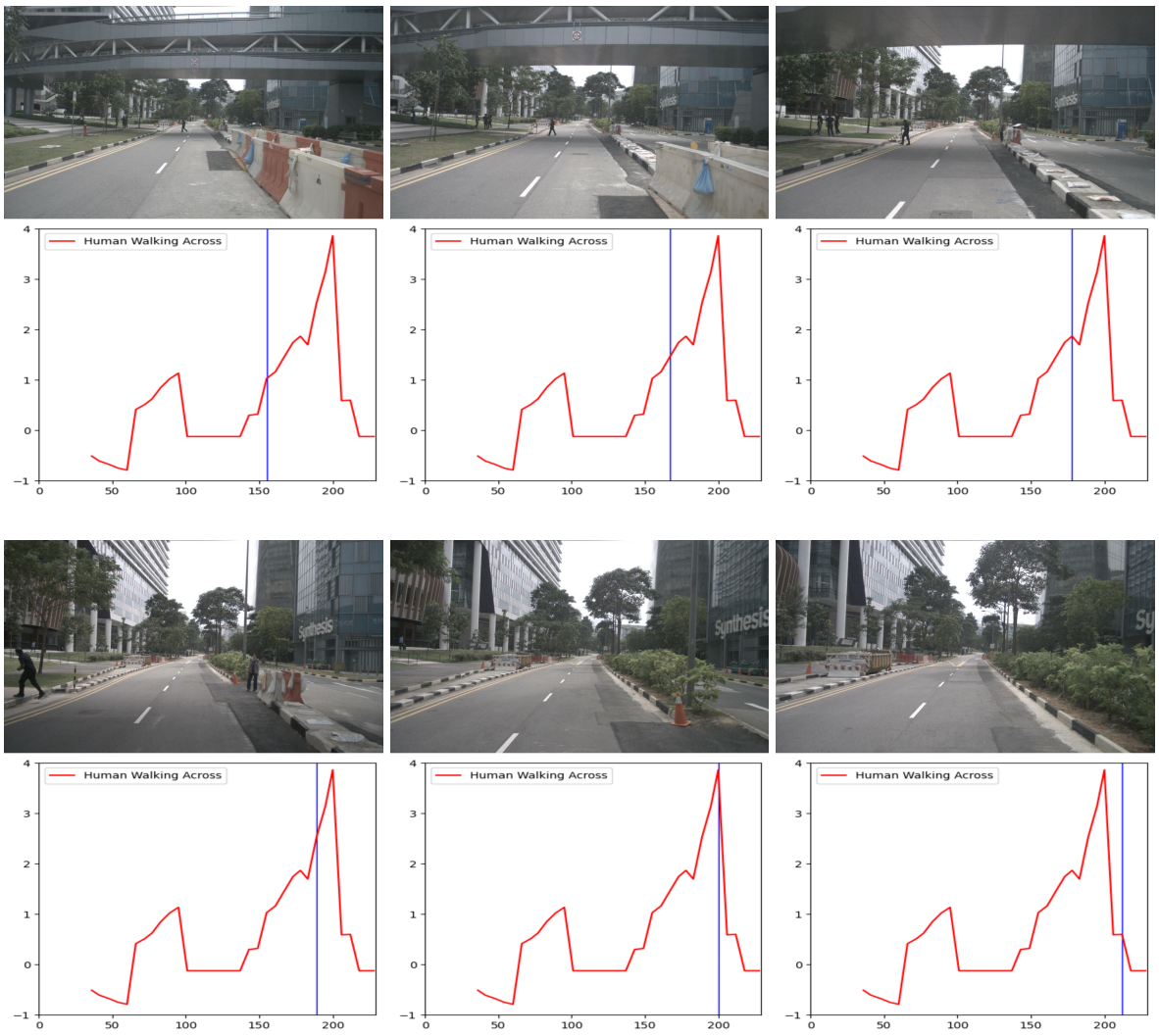


Fig. 5. Activation of Concept 11: "Average horizontal movement of human within the last 2 seconds of the input video" generated from Grounded SAM 2

To further verify the effectiveness of our concept generation process, we randomly selected 100 segments from random training scenes and manually evaluated each of the concepts' activation. As a result, we obtained 99% accuracy for binary concepts and 90% for numerical ones.

4.4 Control Signal Prediction

Table 1 displays the RMSE of predicted velocity, turning angle, and their sum for the base model and the three models after the application of CBL. For our base model, we are able to achieve an RMSE of 1.20 for velocity and 8.92 for turning angle. The unit of measurement used by NuScenes is km/h and $^\circ$, which means that our base model is able to predict the control signal quite well. It is surprising that both Variations 2 and 3 are able to outperform the base model with lower loss in velocity and turning angles. Variation 1, on the other hand, is not able to reach the same level of accuracy as the base model. This is likely due to the missing residual layer that makes it difficult for the model to

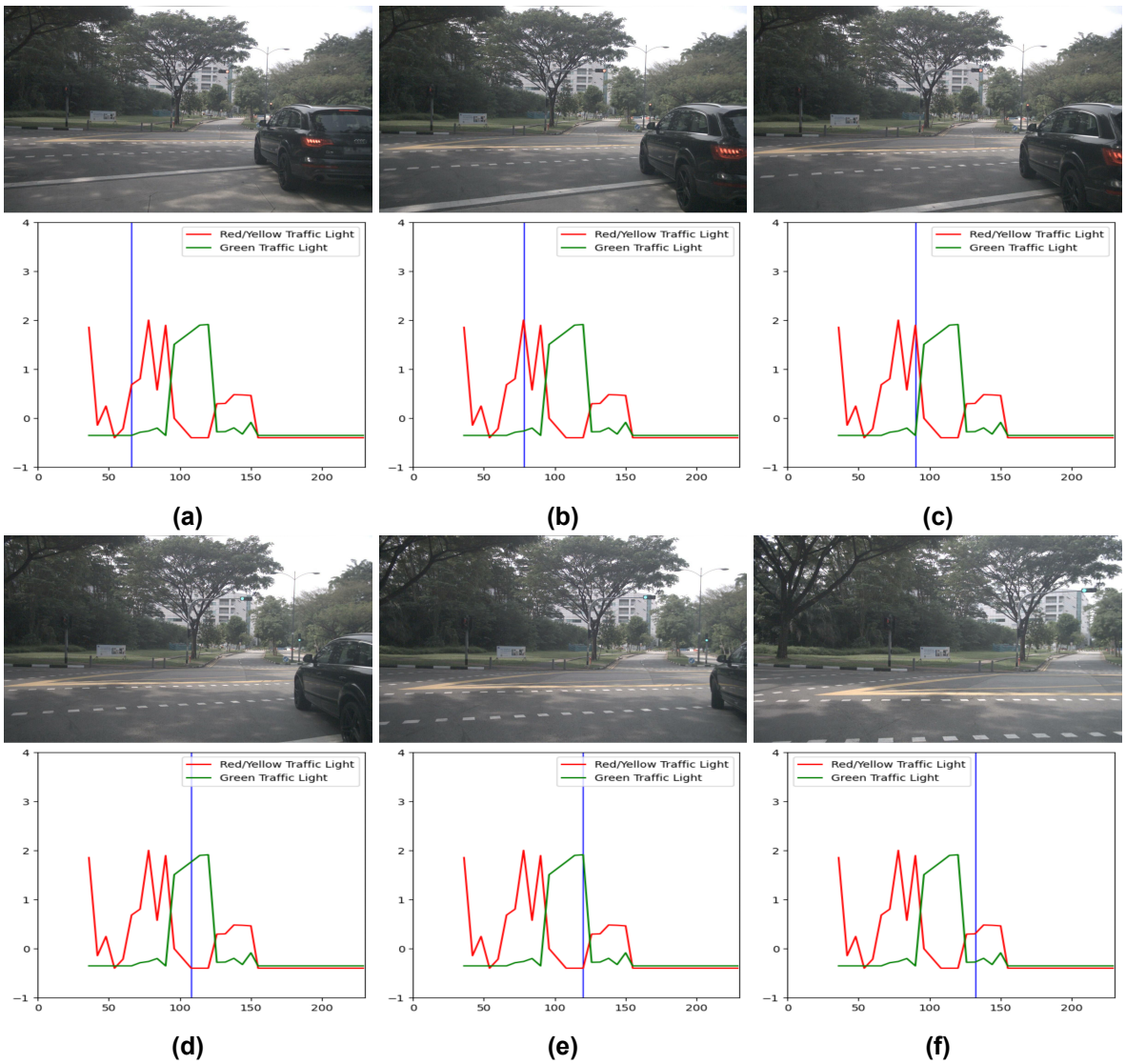


Fig. 6. Activation of Concept 9: “Average score of traffic light (in top center region) being red + yellow within the last 0.5 second” and Concept 10: ”Average score of traffic light (in top center region) being green within the last 0.5 second” generated from Grounded SAM 2

Table 1. Loss of each model. RMSE of predicted velocity, turning angle, and total of the base model and three models after insertion of CBLs.

Model	Loss (RMSE)		
	Velocity	Turning Angle	Total
Base	1.1959	8.9248	10.1207
CBL Variation 1	1.3369	9.7711	11.1081
CBL Variation 2	1.0471	8.7653	9.8125
CBL Variation 3	1.0689	8.6687	9.7376

better predict the control signal. Comparing Variation 2 and 3, it is difficult to decide which one has a better result, with Variation 2 having a lower RMSE in predicted velocity and Variation 3 having a lower RMSE in predicted turning angle.

4.5 Concept Prediction

In order to determine the effectiveness of the bottleneck layer, we run our model on the test set and compare the predicted concepts with the generated ones. For the binary concepts, we calculate the accuracy and RMSE for the numerical concepts. For binary concepts, we obtained an average accuracy of 91.05% with a minimum of 79.03% and a maximum of 99.63% and an average RMSE of 0.7533 with a minimum of 0.5070 and a maximum of 0.9625. It seems that our model is able to grasp simpler boolean concepts more effectively. One reason might be due to the inherent characteristics of boolean concepts being easier to predict. In addition, the CBL's single-layer design lacks the capacity for complex non-linear mapping from the latent space, further limiting its ability to predict numerical concepts. Another potential cause for such a result is within the designs of visual transformers. In particular, the mechanism within transformers simply associates or attends between input tokens rather than counting the occurrence of individual objects unless they are explicitly trained [38]. Yet since the training of CBL requires the backbone to be frozen, a better base model that incorporates various counting mechanisms may be needed. Furthermore, the normalization process could make the concepts harder to predict. In particular, the standard score (z-score) works best when the original data is normally distributed which may not be the case for all numerical concepts. Therefore, more refined techniques should be used to convert data into a more normalized shape before normalization or different techniques should be used depending on the actual distribution of the dataset.

A potential solution to such a problem is to convert numerical concepts to binary by assigning a threshold that can assign "on/off" to each of the concepts. For instance, if the value of Concept 10: "Average score of traffic light (in the top center region) being green within the last 0.5 second" is changed into "Is traffic light (in the top center region) green within the last 0.5 seconds?" and the original score can then be compared with the threshold to determine whether this concept is true or false. This could therefore be a direction for future work to improve the performance of our model.

Another issue contributing to the reduced concept prediction performance is the backbone's failure to capture certain concepts during its initial training. If such information is not encoded into the latent features by the backbone, it becomes impossible for the CBL to map them to the corresponding concepts. Since the CBL training process does not involve fine-tuning the backbone, recovering these missing features is particularly challenging. Furthermore, concepts that are poorly learned or inaccurately represented can introduce noise into the model, leading to meaningless activations that negatively impact the overall prediction performance. To mitigate this issue, future work could consider filtering out concepts with low prediction accuracy or high loss, and retaining those with reliable performance when training the second stage (the prediction head) of the model. This selective training approach could enhance the model's final prediction quality by focusing on more trustworthy

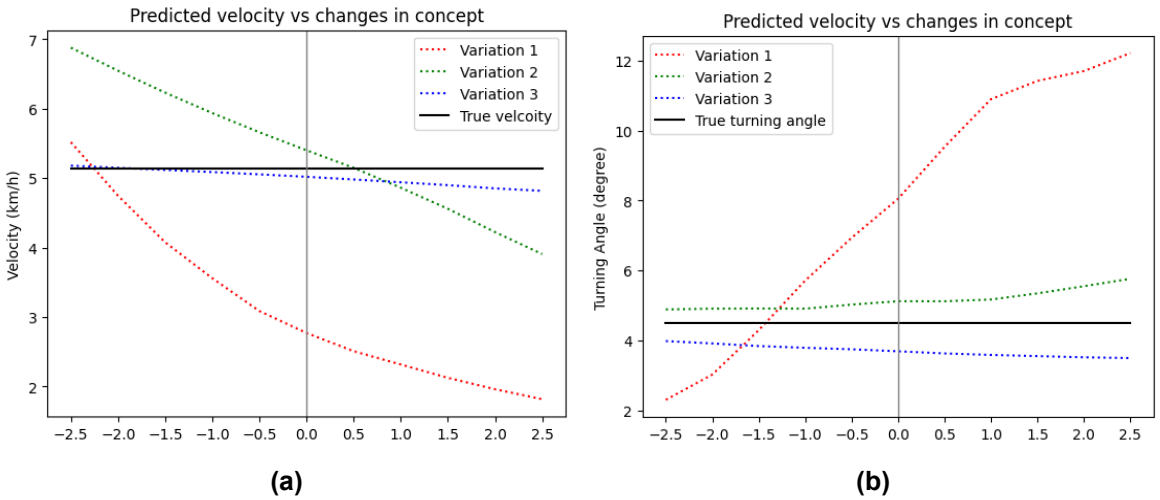


Fig. 7. Change in activation of concept 9: ”Average score of traffic light (in top center region) being red + yellow within the last 0.5 second”

and informative concept representations.

A detailed table showing the results obtained for each individual concept is presented in Appendix B. In our evaluation, we found that the RMSE for Concepts 9 and 10 is the lowest among other numerical concepts. These correspond to the activations of red+yellow or green traffic lights within the last 0.5 seconds of the video. Therefore, we focus more on analysis using this concept which is expected to yield better results than other concepts.

4.6 Effect of Concept

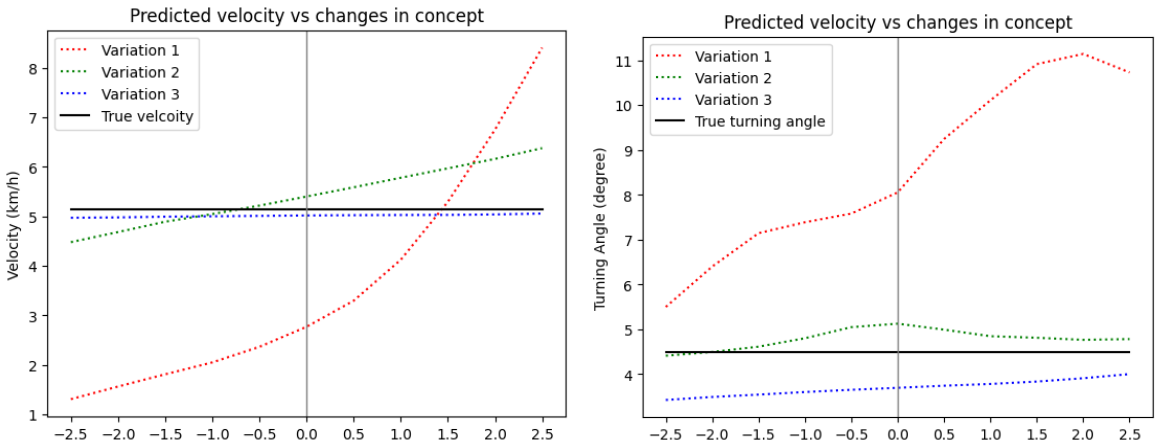
In traditional CBL design, the mapping from CBL to the final prediction can be used to explain the contribution of each concept. However, since our CBL is not placed directly before the final layer, we need another way to verify its contribution. To do so, we first provide an input to the model and obtain the concept prediction as well as the control signal prediction. Then we apply a small change to the activation of a certain concept and observe the relative behavior. In Fig. 7, we used Fig. 6c as the base input and plotted the change in the prediction of the three variations after small changes in the activation of Concept 9: ”Average score of traffic light being red + yellow within the last 0.5 second”. From the predicted velocity (Fig. 7a), we can see that all three models predict velocity to decrease when the activation increases. This corresponds to how the vehicle needs to slow down and stop before a red light. The slopes for variations 1 and 2 are steeper compared to variation 3 which indicates that the significance of the concept is greater for Variations 1 and 2 compared to 3 as we expected in the model design. On the other hand, the turning angle (Fig. 7b) should not be affected much by the existence of red light. Variations 2 and 3 are able to follow this behavior as their turning angle changes slightly across the change in activations. A similar result is shown when

we manipulate the activation of Concept 10: ”Average score of traffic light (in the top center region) being green within the last 0.5 second” instead. In this case (Fig. 8), we can see that the increase in activation of the green traffic light led to an increase in velocity for both Variations 1 and 2.

We also conducted an extensive study on the sensitivity of all concepts across the test set. In particular, we obtained the activation of each individual concept for each input to our model. Then we manually edit each concept's activation similarly to what we did above. For a binary concept, we flip between true and false; for a numerical concept, we record both adding to and subtracting from the activation value. Then we let our model predict the control signal using the entire set of manipulations we did (we only edit the activation of a single concept at a time). Finally, we compare the intervened results with the original results and average them across all test set data. For binary concepts (1 - 4) which correspond to the existence of a human or a car/vehicle within the last 2 or 1 seconds, we found that changing its activation from true to false is more likely to cause an increase in velocity among the three models. However, most results were deemed inconclusive due to their heavy dependency on the original prediction and concept activation. There are still valid findings from this experiment. For instance, we see that Variation 1 is able to show the highest sensitivity compared to the other two variations. Moreover, we found that the manipulation of concept 9 (red traffic light) and concept 10 (green traffic light) produces the most consistent change across test scenes. For Variation 1, an increase in activation of the red traffic light (concept 9) causes a decrease in velocity for 60.44% of test data, and a decrease in activation causes an increase in velocity for 62.40%. Similarly, an increase in activation of green traffic light (concept 10) causes an increase in velocity for 73.86% of test cases, and a decrease in activation causes a 72.8% decrease in velocity. Again, we want to emphasize that these are not conclusive results; for instance, a low activation of a green light can still happen when there is no traffic light detected and hence may not necessarily be associated with a decrease in velocity.

4.7 Comparison of CBL designs

It is clear from the various results that Variation 1: CBL with no residual led to the worst performance among all three variations. Not only does it perform poorly in terms of predicting control signals, but it produces unreasonable changes in prediction when concepts are manually intervened. In contrast, both Variations 2 and 3 with different levels of residual are able to maintain and even outperform the base model after the insertion of the bottleneck layer. Despite both models still suffering from the poor accuracy of the bottleneck layer, some level of interpretability can be achieved through the intervention study on the change of prediction caused by the activation of different concepts. The same study also demonstrates that Variation 2 may offer a clear relationship between the activation



(a) **(b)**

Fig. 8. Change in activation of concept 10: ”Average score of traffic light (in top center region) being green within the last 0.5 second”

of concepts and the final prediction due to the limited size of the residual.

5 Conclusion and future work

5.1 Conclusion

In this thesis, we presented a framework for applying a concept bottleneck layer with a flexible list of concepts to an autonomous driving model as an attempt to improve interpretability. Evaluation of the generated concepts has demonstrated our framework's ability to successfully extract meaningful concept labels from the training data with minimal manual effort. Despite the relatively low accuracy in numerical concepts learned by our CBL, the model is still capable of outperforming the base model in control signal prediction tasks with the help of residual connections. Furthermore, our intervention experiment also verified that changes in individual concept activations led to coherent changes in model predictions. Overall, our framework shows promising potential in improving interpretability in autonomous driving models and provides a viable direction for future research.

5.2 Future work

While our framework shows promising results, there are several areas open for future exploration:

5.2.1 Improving Concept Prediction Accuracy

As mentioned in Chapter 4.5, transforming numerical concepts into binary ones by setting a threshold could improve the accuracy of the concept. The nature of binary concepts makes it easier for the model to apply a linear mapping between the embedded latent space and the concept bottleneck layer. Moreover, in cases where the backbone either lacks the necessary information or fails to learn the relevant latent features that correspond to certain concepts, specialized architectures could be explored to better adapt the mapping from input to the concept.

5.2.2 Expand the List of Concepts

In our thesis, we used a simple set of 16 concepts to demonstrate the ability of our framework. In reality, there could be a large set of concepts that can be used for the interpretation of a model's behavior in autonomous driving. In particular, a domain expert may be invited to produce an extensive list of concepts that are relevant to driving. An alternate approach could be to automatically generate relevant concepts through the use of large language models, as demonstrated in other works [33].

5.2.3 Better Concept Label Generation

Our current approach uses Grounded SAM 2 to extract bounding boxes for relevant objects from input video frames and applies simple algorithms to generate concept scores based on the movement of these bounding boxes. However, there are several limitations associated with this method. For instance, Grounded SAM 2 may produce incorrect predictions—for example, it frequently misclassifies various signs as stop signs. To address this, more specialized detection algorithms, such as dedicated traffic sign detectors, could be employed to improve object classification accuracy. Moreover, an ensemble of multiple grounding algorithms may be used together to ensure the accuracy of the objects detected.

Additionally, in our current method for calculating movement-based scores, we treat the input image as a flat plane and use only the 2D x and y coordinates to estimate object motion. A more accurate approach would incorporate scene geometry and camera calibration to infer 3D movement, which could provide more precise and reliable concept scores.

5.2.4 Better training for the residual layer

Residual in CBM is important to allow information that is not categorized by concepts to aid in prediction. However, there are also concerns that the residual could contain all the necessary information, including those defined by the CBL layer. Hence, the model may choose to use all information from the residual instead of the activation of the concept. Some research [47] of concept bottleneck in large language models suggests a way to apply adversarial training for the residual layer. Their paper shows that this procedure would force the residual layer to forget information related to the concept, thereby forcing the model to use the output of the CBL.

Since our model did not control the residual layer, we believe that such a procedure could also be applied to our model, especially for variation 3, to strengthen the effect of CBL.

References

[1] Anurag Arnab et al. *ViViT: A Video Vision Transformer*. 2021. arXiv: 2103.15691 [cs.CV]. URL: https://arxiv.org/abs/2103.15691 (cited on pp. 10, 11).

[2] David Bau et al. *Network Dissection: Quantifying Interpretability of Deep Visual Representations*. 2017. arXiv: 1704.05796 [cs.CV]. URL: https://arxiv.org/abs/1704.05796 (cited on p. 9).

[3] Iz Beltagy, Matthew E. Peters, and Arman Cohan. *Longformer: The Long-Document Transformer*. 2020. arXiv: 2004.05150 [cs.CL]. URL: https://arxiv.org/abs/2004.05150 (cited on p. 10).

[4] Luca M. Schulze Buschoff et al. *Visual cognition in multimodal large language models*. 2024. arXiv: 2311.16093 [cs.LG]. URL: https://arxiv.org/abs/2311.16093 (cited on p. 8).

[5] Holger Caesar et al. *nuScenes: A multimodal dataset for autonomous driving*. 2020. arXiv: 1903.11027 [cs.LG]. URL: https://arxiv.org/abs/1903.11027 (cited on p. 20).

[6] Aditya Chattopadhay et al. "Grad-CAM++: Generalized Gradient-Based Visual Explanations for Deep Convolutional Networks". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, Mar. 2018. DOI: 10.1109/wacv.2018.00097. URL: http://dx.doi.org/10.1109/WACV.2018.00097 (cited on p. 8).

[7] Li Chen et al. *End-to-end Autonomous Driving: Challenges and Frontiers*. 2024. arXiv: 2306.16927 [cs.RO]. URL: https://arxiv.org/abs/2306.16927 (cited on p. 10).

[8] An-Chieh Cheng et al. *SpatialRGPT: Grounded Spatial Reasoning in Vision Language Models*. 2024. arXiv: 2406.01584 [cs.CV]. URL: https://arxiv.org/abs/2406.01584 (cited on p. 8).

[9] Lu Chi and Yadong Mu. *Deep Steering: Learning End-to-End Driving Model from Spatial and Temporal Visual Cues*. 2017. arXiv: 1708.03798 [cs.CV]. URL: https://arxiv.org/abs/1708.03798 (cited on p. 10).

[10] Pranav Singh Chib and Pravendra Singh. *Recent Advancements in End-to-End Autonomous Driving using Deep Learning: A Survey*. 2023. arXiv: 2307.04370 [cs.RO]. URL: https://arxiv.org/abs/2307.04370 (cited on p. 10).

[11] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929 (cited on p. 10).

[12] Danny Driess et al. *PaLM-E: An Embodied Multimodal Language Model*. 2023. arXiv: 2303.03378 [cs.LG]. URL: https://arxiv.org/abs/2303.03378 (cited on p. 8).

[13] Jessica Echterhoff et al. *Driving through the Concept Gridlock: Unraveling Explainability Bottlenecks in Automated Driving*. 2023. arXiv: 2310.16639 [cs.CV]. URL: https://arxiv.org/abs/2310.16639 (cited on pp. 10, 11).

[14] Epoch AI. *Data on Notable AI Models*. Accessed: 2025-04-07. June 2024. URL: https://epoch.ai/data/notable-ai-models (cited on p. 6).

[15] Jerome H. Friedman. "Greedy function approximation: a gradient boosting machine". In: *Annals of Statistics* 29.5 (2001), pp. 1189–1232 (cited on p. 8).

[16] Google. *Lucid: A collection of infrastructure and tools for research in neural network interpretability*. https://github.com/tensorflow/lucid. 2018 (cited on p. 9).

[17] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385 (cited on p. 10).

[18] Wenbo Hu et al. *BLIVA: A Simple Multimodal LLM for Better Handling of Text-Rich Visual Questions*. 2023. arXiv: 2308.09936 [cs.CV]. URL: https://arxiv.org/abs/2308.09936 (cited on p. 8).

[19] Jyh-Jing Hwang et al. *EMMA: End-to-End Multimodal Model for Autonomous Driving*. 2024. arXiv: 2410.23262 [cs.CV]. URL: https://arxiv.org/abs/2410.23262 (cited on p. 8).

[20] IDEA-Research. *Grounded-SAM-2*. https://github.com/IDEA-Research/Grounded-SAM-2. Accessed: 2025-04-03. 2024 (cited on p. 14).

[21] Bu Jin et al. *ADAPT: Action-aware Driving Caption Transformer*. 2023. arXiv: 2302.00673 [cs.CV]. URL: https://arxiv.org/abs/2302.00673 (cited on pp. 8, 10, 11).

[22] Jared Kaplan et al. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG]. URL: https://arxiv.org/abs/2001.08361 (cited on p. 5).

[23] Will Kay et al. *The Kinetics Human Action Video Dataset*. 2017. arXiv: 1705.06950 [cs.CV]. URL: https://arxiv.org/abs/1705.06950 (cited on p. 11).

[24] Been Kim et al. *Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)*. 2018. arXiv: 1711.11279 [stat.ML]. URL: https://arxiv.org/abs/1711.11279 (cited on p. 9).

[25] Jinkyu Kim and John Canny. "Interpretable Learning for Self-Driving Cars by Visualizing Causal Attention". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2961–2969. DOI: 10.1109/ICCV.2017.320 (cited on p. 8).

[26] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980 (cited on p. 21).

[27] Pang Wei Koh et al. *Concept Bottleneck Models*. 2020. arXiv: 2007.04612 [cs.LG]. URL: https://arxiv.org/abs/2007.04612 (cited on pp. 9, 13, 17, 18).

[28] Quoc-Vinh Lai-Dang. *A Survey of Vision Transformers in Autonomous Driving: Current Trends and Future Directions*. 2024. arXiv: 2403.07542 [cs.CV]. URL: https://arxiv.org/abs/2403.07542 (cited on p. 10).

[29] Zachary C. Lipton. *The Mythos of Model Interpretability*. 2017. arXiv: 1606.03490 [cs.LG]. URL: https://arxiv.org/abs/1606.03490 (cited on p. 5).

[30] Shilong Liu et al. *Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection*. 2024. arXiv: 2303.05499 [cs.CV]. URL: https://arxiv.org/abs/2303.05499 (cited on p. 14).

[31] Srikanth Malla et al. *DRAMA: Joint Risk Localization and Captioning in Driving*. 2022. arXiv: 2209.10767 [cs.CV]. URL: https://arxiv.org/abs/2209.10767 (cited on p. 8).

[32] Daniel Neimark et al. *Video Transformer Network*. 2021. arXiv: 2102.00719 [cs.CV]. URL: https://arxiv.org/abs/2102.00719 (cited on p. 11).

[33] Tuomas Oikarinen et al. *Label-Free Concept Bottleneck Models*. 2023. arXiv: 2304.06129 [cs.LG]. URL: https://arxiv.org/abs/2304.06129 (cited on pp. 9, 17, 28).

[34] Long Ouyang et al. *Training language models to follow instructions with human feedback*. 2022. arXiv: 2203.02155 [cs.CL]. URL: https://arxiv.org/abs/2203.02155 (cited on p. 10).

[35] Alan Pham et al. *The Effect of Model Size on Worst-Group Generalization*. 2021. arXiv: 2112.04094 [cs.LG]. URL: https://arxiv.org/abs/2112.04094 (cited on p. 5).

[36] Dean A. Pomerleau. "ALVINN: An Autonomous Land Vehicle in a Neural Network". In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky. Vol. 1. Morgan-Kaufmann, 1988. URL: https://proceedings.neurips.cc/paper_files/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf (cited on p. 10).

[37] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. *Multi-Modal Fusion Transformer for End-to-End Autonomous Driving*. 2021. arXiv: 2104.09224 [cs.CV]. URL: https://arxiv.org/abs/2104.09224 (cited on p. 10).

[38] Muhammad Fetrat Qharabagh, Mohammadreza Ghofrani, and Kimon Fountoulakis. *LVLM-COUNT: Enhancing the Counting Ability of Large Vision-Language Models*. 2025. arXiv: 2412.00686 [cs.CV]. URL: https://arxiv.org/abs/2412.00686 (cited on p. 24).

[39] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV]. URL: https://arxiv.org/abs/2103.00020 (cited on p. 9).

[40] Tilman Räuker et al. *Toward Transparent AI: A Survey on Interpreting the Inner Structures of Deep Neural Networks*. 2023. arXiv: 2207.13243 [cs.LG]. URL: https://arxiv.org/abs/2207.13243 (cited on p. 8).

[41] Joseph Redmon et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV]. URL: https://arxiv.org/abs/1506.02640 (cited on p. 14).

[42] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. *"Why Should I Trust You?": Explaining the Predictions of Any Classifier*. 2016. arXiv: 1602.04938 [cs.LG]. URL: https://arxiv.org/abs/1602.04938 (cited on p. 8).

[43] Avishkar Saha et al. *Translating Images into Maps*. 2022. arXiv: 2110.00966 [cs.CV]. URL: https://arxiv.org/abs/2110.00966 (cited on p. 8).

[44] Jaime Sevilla et al. "Compute Trends Across Three Eras of Machine Learning". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE, July 2022, pp. 1–8. DOI: 10.1109/ijcnn55064.2022.9891914. URL: http://dx.doi.org/10.1109/IJCNN55064.2022.9891914 (cited on p. 5).

[45] Hao Sha et al. *LanguageMPC: Large Language Models as Decision Makers for Autonomous Driving*. 2023. arXiv: 2310.03026 [cs.RO]. URL: https://arxiv.org/abs/2310.03026 (cited on p. 8).

[46] Chenming Shang et al. *Incremental Residual Concept Bottleneck Models*. 2024. arXiv: 2404.08978 [cs.LG]. URL: https://arxiv.org/abs/2404.08978 (cited on pp. 9, 17).

[47] Chung-En Sun et al. *Concept Bottleneck Large Language Models*. 2025. arXiv: 2412.07992 [cs.CL]. URL: https://arxiv.org/abs/2412.07992 (cited on p. 29).

[48] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. *Axiomatic Attribution for Deep Networks*. 2017. arXiv: 1703.01365 [cs.LG]. URL: https://arxiv.org/abs/1703.01365 (cited on p. 8).

[49] Andong Tan, Fengtao Zhou, and Hao Chen. *Explain via Any Concept: Concept Bottleneck Model with Open Vocabulary Concepts*. 2024. arXiv: 2408.02265 [cs.CV]. URL: https://arxiv.org/abs/2408.02265 (cited on p. 13).

[50] Xiaoyu Tian et al. *DriveVLM: The Convergence of Autonomous Driving and Large Vision-Language Models*. 2024. arXiv: 2402.12289 [cs.CV]. URL: https://arxiv.org/abs/2402.12289 (cited on p. 8).

[51] Jiahang Tu et al. *DriveDiTFit: Fine-tuning Diffusion Transformers for Autonomous Driving*. 2024. arXiv: 2407.15661 [cs.CV]. URL: https://arxiv.org/abs/2407.15661 (cited on p. 8).

[52] Sahil Verma et al. *Counterfactual Explanations and Algorithmic Recourses for Machine Learning: A Review*. 2022. arXiv: 2010.10596 [cs.LG]. URL: https://arxiv.org/abs/2010.10596 (cited on p. 8).

[53] Pablo Villalobos et al. *Machine Learning Model Sizes and the Parameter Gap*. 2022. arXiv: 2207.02852 [cs.LG]. URL: https://arxiv.org/abs/2207.02852 (cited on p. 5).

[54] Hengli Wang et al. *Learning Interpretable End-to-End Vision-Based Motion Planning for Autonomous Driving with Optical Flow Distillation*. 2021. arXiv: 2104.12861 [cs.CV]. URL: https://arxiv.org/abs/2104.12861 (cited on p. 8).

[55] Yi Xiao et al. "Multimodal End-to-End Autonomous Driving". In: *IEEE Transactions on Intelligent Transportation Systems* 23.1 (Jan. 2022), pp. 537–547. ISSN: 1558-0016. DOI: 10.1109/tits.2020.3013234. URL: http://dx.doi.org/10.1109/TITS.2020.3013234 (cited on p. 10).

[56] Zhenhua Xu et al. *DriveGPT4: Interpretable End-to-end Autonomous Driving via Large Language Model*. 2024. arXiv: 2310.01412 [cs.CV]. URL: https://arxiv.org/abs/2310.01412 (cited on p. 8).

[57] Jason Yosinski et al. *Understanding Neural Networks Through Deep Visualization*. 2015. arXiv: 1506.06579 [cs.CV]. URL: https://arxiv.org/abs/1506.06579 (cited on p. 9).

[58] Mert Yuksekgonul, Maggie Wang, and James Zou. *Post-hoc Concept Bottleneck Models*. 2023. arXiv: 2205.15480 [cs.LG]. URL: https://arxiv.org/abs/2205.15480 (cited on pp. 9, 17).

[59] Matthew D Zeiler and Rob Fergus. *Visualizing and Understanding Convolutional Networks*. 2013. arXiv: 1311.2901 [cs.CV]. URL: https://arxiv.org/abs/1311.2901 (cited on p. 8).

[60] Xiaohua Zhai et al. *Scaling Vision Transformers*. 2022. arXiv: 2106.04560 [cs.CV]. URL: https://arxiv.org/abs/2106.04560 (cited on p. 5).

[61] Yunpeng Zhang et al. *GraphAD: Interaction Scene Graph for End-to-end Autonomous Driving*. 2024. arXiv: 2403.19098 [cs.CV]. URL: https://arxiv.org/abs/2403.19098 (cited on p. 10).

[62] Ou Zheng et al. *TrafficSafetyGPT: Tuning a Pre-trained Large Language Model to a Domain-Specific Expert in Transportation Safety*. 2023. arXiv: 2307.15311 [cs.CL]. URL: https://arxiv.org/abs/2307.15311 (cited on p. 8).

[63] Ruiqi Zhong et al. *Are Larger Pretrained Language Models Uniformly Better? Comparing Performance at the Instance Level*. 2021. arXiv: 2105.06020 [cs.CL]. URL: https://arxiv.org/abs/2105.06020 (cited on p. 5).

Appendices

A Chosen Concepts

Here are the list of concepts we used for our model and reasons for choosing them

1. Is there vehicle/car within the last 2 seconds
2. Is there vehicle/car within the last 1 second
3. Is there people within the last 2 seconds
4. Is there people within the last 1 second
5. Average number of vehicle/car within the last 2 seconds
6. Average number of vehicle/car within the last 1 second
7. Average number of people within the last 2 seconds
8. Average number of people within the last 1 second
9. Average score of traffic light (in top center region) being red + yellow within the last 0.5 second
10. Average score of traffic light (in top center region) being green within the last 0.5 second
11. Average horizontal movement of human within the last 2 seconds
12. Average vertical distance (vertical pixels from bottom edge) to closest vehicle/car within the last 1 second
13. Average vertical distance (vertical pixels from bottom edge) to closest people within the last 1 second
14. Average distance (pixels from bottom center) to closest vehicle/car within the last 1 second
15. Average distance (pixels from bottom center) to closest people within the last 1 second
16. Average area (numbers of pixel) of closest vehicle/car within the last 1 second

The first 4 binary concepts are simple checks for the existence of a human or car/vehicle in front of the ego vehicle. We chose these concepts following human intuition that we may pay more attention and be slower if we see things in front of us than if we see a clear road. The 5-8 concepts are counting the average number of humans or cars/vehicles which is also based on the intuition for

Table 2. Accuracy of binary concept

Concept Number	1	2	3	4
Accuracy /%	99.63	98.97	86.55	79.03

Table 3. RMSE of numerical concept

Concept Number	5	6	7	8	9	10
RMSE	0.7810	0.7934	0.7015	0.7235	0.5070	0.5420
Concept Number	11	12	13	14	15	16
RMSE	0.8250	0.9625	0.8921	0.9234	0.7122	0.6763

driving. For instance, if there are a lot of car, then it probably means the road is congested hence we need to reduce speed. Similarly, for humans, if there are many humans around, drivers would typically drive slowly to prevent accidents from happening.

B Full loss on concept

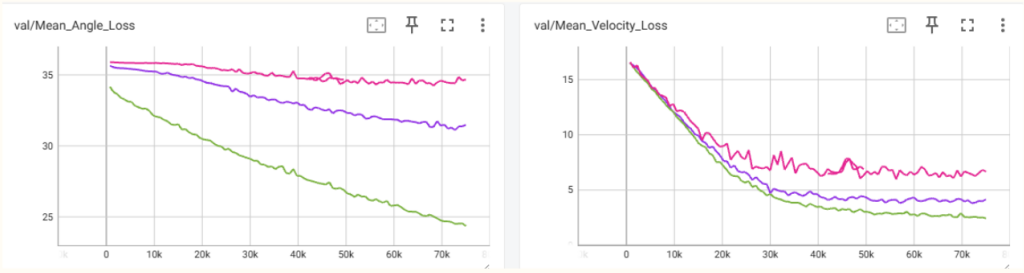
In order to determine how much our CBL learns the concept activation from the dataset, we evaluate its accuracy (for binary concept) and root mean squared error (RMSE) (for numerical concept).

The accuracy is calculated as the sum of instances where predicted activation (true or false) matches the label across the test set over the entire test set size. From Table 2, we can see that concept 1 about the existence of a vehicle has the highest accuracy, while concept 4 about the existence of a human shows the lowest accuracy. We think one explanation for this result is that vehicles/cars are probably the most common object that is present in the driving dataset. In addition, the majority of scenes contain cars which may result in high accuracy even if the model always predicts true.

The RMSE is calculated in the same ways during training. Since all of the numerical concepts were normalized to $\mu = 0$, $\sigma = 1$, an initial impression from Table 3 shows a poor result across all numerical concepts. Comparing among all concepts of this type, we can see that concept 9 and 10 have the least RMSE. They correspond to the score given to the traffic light being red+yellow and green. Therefore, we focus on the evaluation of these two concepts in our thesis.

C Ablation study on base model

During the development of our base model, we evaluated the effect of concatenation of previous control signals as well as the loss function used. Fig. 9 shows that the model with no control signal is given (pink) shows worse results during training compared to the addition of a previous control signal. This is not really a surprise for us, as it is common to think that previous velocity and turning



Color	Takes previous control signal	Loss (RMSE)
Pink	False	horizontal and vertical velocity
Purple	True	horizontal and vertical velocity
Green	True	horizontal and vertical velocity + turning angle

Fig. 9. Loss on validation set during training

angle should help with the next prediction. The purple and green then compare the results after the addition of the RMSE of the turning angle into the total loss calculation. It is clear from the graph that such an addition is able to massively decrease the RMSE on the turning angle and produce a better performance in the end. Therefore, for our base model, we decided to include the last control signal as well as using the RMSE of horizontal and vertical velocity, with the addition of the RMSE of turning angle.