

Master's Thesis in Informatics

Knowledge Distilled Traffic Environment Understanding

Wissensdestillierte Verkehrs Umgebungserkennung

Supervisor Prof. Dr.-Ing. habil. Alois C. Knoll

Advisor Xingcheng Zhou, M.Sc.

Author Baran Ekin Özdemir

Date July 24, 2024 in Munich

Disclaimer

I confirm that this Master's Thesis is my own work and I have documented all sources and material used.

Munich, July 24, 2024



(Baran Ekin Özdemir)

Abstract

Vision-Language Models (VLMs) have recently demonstrated impressive capabilities in challenging tasks that require comprehensive understanding and reasoning across both modalities, such as image captioning, text-guided image generation, and visual question-answering leading to the widespread application of VLMs within computer vision. This work explores the utilization of VLMs for traffic environment understanding, where integrating linguistic and visual features can enhance generalization to unseen scenarios and improve the interpretability of autonomous driving systems. To achieve these goals, BEVBlip, an efficient, lightweight VLM employing spatio-temporal Bird's Eye View (BEV) maps as visual features is developed, and various methods to exploit refined vision knowledge are studied. BEVBlip undertakes a comprehensive Visual Question Answering task tailored for autonomous driving, achieving promising results on the DriveLM dataset with 54.9% accuracy on multiple-choice questions, 57.1% in GPT evaluation, 34.5% in matching, 61.7% in BLEU, and 68.9% in ROUGE-L. This study demonstrates the ability to answer questions by effectively utilizing vision and language features within a lightweight framework and showcases the potential of VLMs for traffic environment understanding.

Zusammenfassung

Visuelle Sprachmodelle (VLMs) haben kürzlich beeindruckende Fähigkeiten bei herausfordern den Aufgaben gezeigt, die ein umfassendes Verständnis und reasoning-Fähigkeiten in beiden Modalitäten erfordern, wie z. B. Bildbeschriftung, textgesteuerte Bilderzeugung und visuelles Frage-Antwort-System. Dies hat zur weitverbreiteten Anwendung von VLMs in der Computer Vision geführt. Diese Arbeit untersucht die Nutzung von VLMs zur Erkennung von Verkehrsumgebungen, wobei die Integration linguistischer und visueller Merkmale die Fähigkeit zur Generalisierung auf unbekannte Szenarien verbessern und die Interpretierbarkeit autonomer Fahrsysteme erhöhen kann. Um diese Ziele zu erreichen, BEVBlip, wird ein effizientes, leichtgewichtiges VLM entwickelt, das räumlich-zeitliche Karten aus der Vogelperspektive (BEV) als visuelle Merkmale verwendet, und verschiedene Methoden zur Nutzung verfeinerter visueller Kenntnisse werden untersucht. BEVBlip führt eine umfassende visuelle Frage-Antwort-Aufgabe durch, die auf das autonome Fahren zugeschnitten ist, und erzielt vielversprechende Ergebnisse auf dem DriveLM-Datensatz mit einer Genauigkeit von 54,9 % bei Multiple-Choice-Fragen, 57,1 % in der GPT-Bewertung, 34,5 % beim Matching, 61,7 % bei BLEU und 68,9 % bei ROUGE-L. Diese Studie demonstriert die Fähigkeit, Fragen zu beantworten, indem sie effektiv Merkmale aus Vision und Sprache in einem leichtgewichtigen Rahmen nutzt und das Potenzial von VLMs für das Verständnis von Verkehrsumgebungen aufzeigt.

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Task	2
1.3	Proposed Method	3
1.4	Outline	3
2	Background	5
2.1	2D Object Detection	5
2.1.1	CNN-based Methods	5
2.1.2	Transformer-based Methods	7
2.2	3D BEV Perception in Autonomous Driving	11
2.2.1	Bird’s Eye View (BEV) Perception	11
2.2.2	Camera-based Methods	12
2.2.3	BEVFormer	13
2.3	Language Models	14
2.3.1	Large Language Models	14
2.3.2	Vision-Language Models	15
3	Related Work	19
3.1	VLMs in Autonomous Driving	19
3.2	Traffic Scene Understanding	20
4	Implementation	23
4.1	Pre-training	23
4.1.1	Data Preparation	23
4.1.2	Model Architecture	25
4.1.3	Losses	27
4.2	Fine-tuning	28
4.2.1	Model Architecture	28
4.2.2	Visual Feature Extraction Strategies	29
5	Experiments	31
5.1	Experimental Setup	31
5.2	Metrics	32
5.2.1	Language Scores	32
5.2.2	DriveLM Metrics	34
5.3	Results	36
5.3.1	Quantitative Results	36
5.3.2	Qualitative Results	37

6 Discussion	41
6.1 Object Grounding	41
6.2 Visual Description of Objects	41
6.3 Language Generation	42
6.4 Multiple-choice Questions	42
6.5 Visual Feature Extraction	42
6.6 CVPR 2024 Autonomous Driving Challenge	43
7 Future Work	45
7.1 Increasing Model Size	45
7.2 Parameter-Efficient Fine-Tuning	46
7.3 Improving Data Generation	47
7.4 Utilizing Improved Datasets	47
8 Conclusion	49
Bibliography	57

Chapter 1

Introduction

1.1 Motivation

Autonomous driving technology refers to the use of advanced systems and algorithms that enable vehicles to navigate without human driver, holding the promise of revolutionizing transportation by enhancing safety, efficiency and convenience. As safety is the paramount concern while designing autonomous driving systems, self-driving vehicles can improve traffic safety by constantly attending to their surroundings. Furthermore, these vehicles can optimize routes to reduce traffic leading to improved fuel efficiency and lower emissions, provide mobility freedom for the elderly, disabled and others who are unable to drive. Achieving fully-autonomous driving is expected to have a tremendous positive societal impact in the near future. Thus, it has been one of the most active areas of research and has seen a rapid development over the past two decades.

To be able to drive without human intervention, autonomous driving systems rely on a combination of sensors such as cameras and radar and software systems. These systems often leverage artificial intelligence techniques in various aspects of self-driving tasks such as perceiving the environment, making driving decisions or planning trajectories. Especially concerning visual perception and prediction capabilities, advancements in computer vision field have propelled autonomous driving systems to a great extent in recent years, by enabling them to robustly and accurately detect objects in their surrounding environment. However, current autonomous driving technologies still face significant challenges that impede their full potential.

A crucial requirement for autonomous driving systems is the ability to generalize to unseen scenarios and unfamiliar objects. This objective necessitates traffic scene understanding, which involves comprehending contextual information and reasoning to better adapt to novel situations and dynamic traffic environment. Furthermore, such understanding and reasoning capabilities are essential for interpretability and explainability of these systems whereas traditional end-to-end approaches often view the systems as a black box, obscuring the logic of decisions and complicating failure diagnosis.

The emergence of Large Language Models (LLMs) offer promising solutions to these inherent limitations by incorporating textual world understanding and human-like reasoning and logic processes into autonomous driving systems to build a more comprehensive driving artificial intelligence system. These models are informed by extensive large scale data from the internet, benefiting from a foundational linguistic understanding of the world. The inclusion of natural language plays a key role in the interactivity and trustworthiness of the system, as it allows a more transparent human-machine interaction. An even more complete approach is provided by the very recent Vision-Language Models (VLMs) which synthesize

internet-scale linguistic and visual data. These models offer a promising approach for end-to-end autonomous driving intelligence that is robust against unprecedented situations while fostering greater interpretability and interactivity by using natural language.

However, the application of VLMs in the autonomous driving domain is still in its nascent stages, and the effective utilization of VLMs for tasks such as enhancing perception, reasoning, and understanding capabilities, remains a largely unexplored territory. At the onset of this thesis, methods leveraging VLMs for traffic environment understanding were just beginning to emerge, with available datasets limited primarily to NuScenes-QA [Qia+23]. The immense, yet unrealized potential of VLMs to enhance autonomous driving systems, supported with their impressive capabilities demonstrated in other domains, provided strong motivation for this research. This gap also prompted an exploration of feasible dataset generation possibilities using LLMs. As this thesis progressed, the adoption of VLMs in the autonomous driving domain gained increasing popularity, and larger datasets introducing more comprehensive tasks began to emerge.

1.2 Task

In order to train VLMs and evaluate their performance on traffic scene understanding, the task of Visual Question Answering (VQA) is commonly employed. VQA is a multidisciplinary task in computer vision and natural language processing that involves answering questions about visual content. The objective is to generate accurate answers to questions posed in natural language based on corresponding visuals. This task requires sophisticated perception capabilities as well as a high level understanding of context. Given that the goal of VQA task is to develop systems capable of understanding and reasoning about visual content in conjunction with textual information, it is a particularly suitable task for traffic scene understanding.

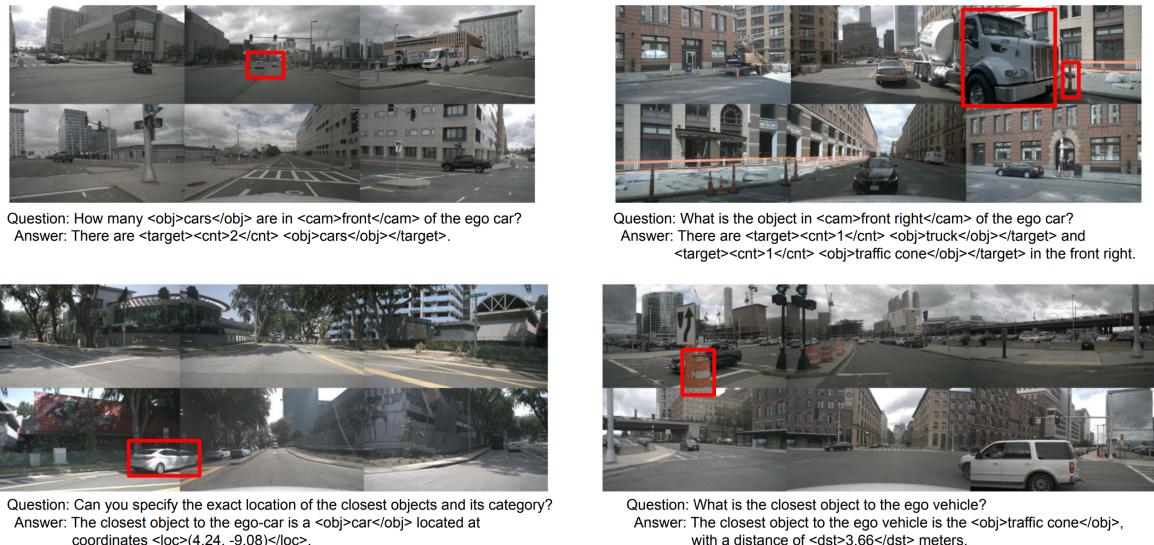


Figure 1.1: A set of examples from NuScenes-MQA [Ino+24] dataset, showing question-answer pairs under corresponding multi-view camera images from various traffic scenes. Questions include object detection, localization and counting. Key information such as cameras, objects, numbers or locations are enclosed within markups.

The application of VQA in autonomous driving is exemplified by recent research efforts and datasets, such as NuScenes-QA [Qia+23], NuScenes-MQA [Ino+24] and DriveLM [Sim+23]. Based on the large-scale autonomous driving dataset nuScenes, they offer diverse set of questions on various aspects of driving such as perception, prediction, planning, counting and reasoning, along with corresponding answers related to real-world driving scenarios. Figure 1.1 illustrates example questions from NuScenes-MQA dataset.

1.3 Proposed Method

In this study, the BLIP [Li+22a] framework is leveraged to enhance traffic scene understanding by integrating language features with vision. The BLIP framework as a multimodal vision-language approach with its efficient, lightweight and generalizable design, is adapted by the proposed BEVBlip to the autonomous driving domain by replacing traditional image inputs with Bird’s Eye View (BEV) feature maps. These BEV feature maps are generated using BEVFormer [Li+22c], which incorporates spatio-temporal information optimized for traffic scene perception. This adaptation allows for a richer representation of the driving environment, facilitating more effective integration of visual and textual data.

The implementation process is divided into two main stages: pre-training and fine-tuning. During pre-training stage, BEV feature maps are aligned with the unified multimodal encoder-decoder from the BLIP framework, which was originally trained on image features. This stage involves adapting the image captioning task to a traffic scene description task using BEV feature maps. To prepare the training data, BEV feature maps are extracted using a pre-trained BEVFormer model and textual descriptions of the scenes from nuScenes dataset are generated using GPT-3.5. The model architecture employs a compact Vision Transformer to process BEV feature maps, which are then passed through the multimodal encoder-decoder transformer for feature alignment, optimized on three pre-training objectives.

For the fine-tuning stage, the pre-trained BEVBlip is adapted for the VQA task using the DriveLM-nuScenes dataset. The model architecture is reconfigured to function as a question encoder and an answer decoder. The question encoder integrates visual and textual embeddings using cross-attention mechanisms, while the answer decoder generates answers auto-regressively. Various visual feature extraction strategies are explored, including the use of average pooled BEV features and object detections from the BEVFormer model, to enhance the model’s perception and understanding of the traffic scene. These adaptations and strategies aim to improve the model’s performance in comprehending and responding to traffic-related queries effectively.

1.4 Outline

The subsequent chapters of this thesis are organized as follows: Chapter 2 provides an overview of the fundamental elements, methodologies, and core components essential to this work and associated research, providing a concise overview of the background. Chapter 3 covers the related work, reviewing vision-language models in autonomous driving. Chapter 4 elaborates on the implementation details of the BEVBlip method, including the architecture, data preparation, and training processes. Chapter 5 describes the experiments conducted to evaluate the performance of the proposed method on the visual question answering task, outlining the experimental settings, evaluation metrics, and both quantitative and qualitative

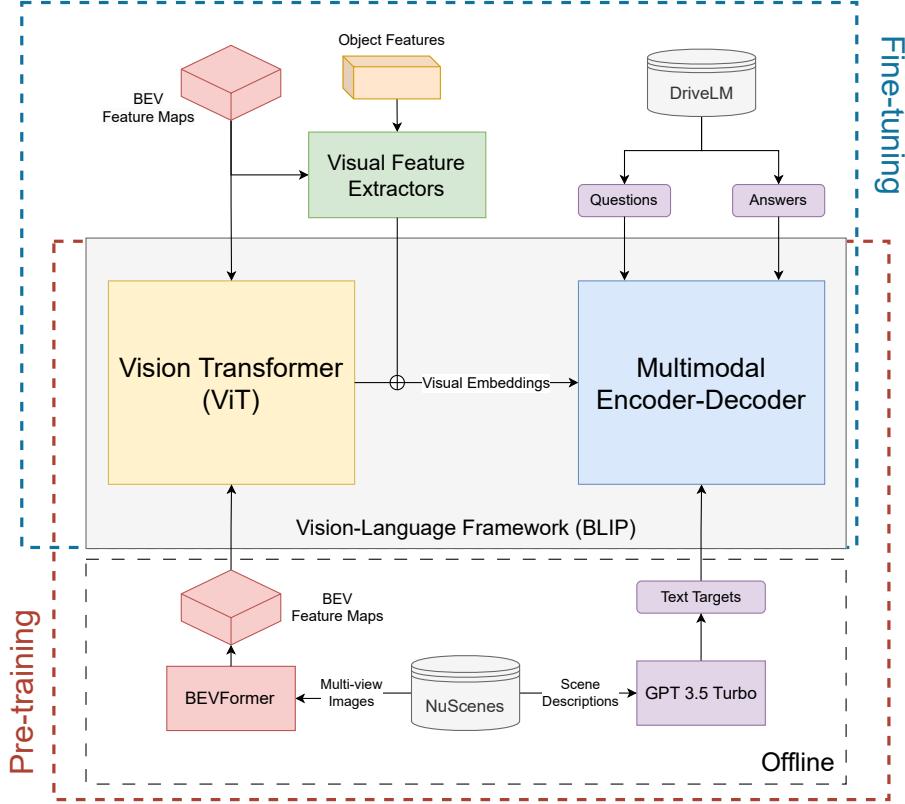


Figure 1.2: High-level outline of the proposed approach, utilizing BLIP [Li+22a] as a vision-language framework to address the DriveLM [Sim+23] VQA task for traffic scene understanding. The proposed approach replaces images with feature maps in the Bird's Eye View (BEV) perspective, acquired via BEVFormer [Li+22c] from multi-view camera images surrounding the vehicle. The bottom part illustrates the pre-training step for feature alignment, which includes offline BEV feature map acquisition and text target generation using GPT-3.5 [Bro+20] from nuScenes scene descriptions. The top part depicts the fine-tuning step for VQA using the DriveLM dataset and various visual feature extraction strategies. For more detailed architecture illustrations, refer to Figure 4.1, Figure 4.2, and Figure 4.3 in Chapter 4

results. Chapter 6 interprets and analyzes the results obtained in Chapter 5. Chapter 7 discusses potential future research directions and offers recommendations. Finally, Chapter 8 concludes the work with a brief summary.

Chapter 2

Background

This chapter reviews the essential elements, methodologies, and core components of this work and related research. It provides a concise overview of key topics, the historical development of significant methods, and the foundational mechanisms that support current advancements in the field. This overview aims to establish a solid foundation for understanding both this work and the related research. The review begins with an examination of 2D object detection, which serves as a basis for 3D perception in autonomous driving, and covers the progress of significant methods and essential architectures in computer vision. The subsequent section focuses on 3D BEV perception in autonomous driving, particularly camera-based methods. The final section introduces large language models and vision-language models, which are crucial for traffic scene understanding.

2.1 2D Object Detection

2D object detection is a fundamental task in computer vision that involves identifying and localizing objects within an image. The goal is to predict the bounding boxes indicating the location and the shape of the objects, together with the class labels of objects defining the types of localized objects present in a 2D image plane. This task is essential in understanding of the world from visual input which is crucial for various applications including autonomous driving. However, this task possesses significant challenges such as dealing with variations in object scale, occlusions, complex backgrounds, and often, real-time processing requirements.

Early attempts to detecting objects in images date far back, such as Viola-Jones Detector [VJ01] in 2001 and the Histogram of Gradients [DT05] in 2005. Even though such methods laid the groundwork in the history of object detection, only the later and far more advanced CNN-based and Transformer-based methods that are relevant to this work are covered in this section.

2.1.1 CNN-based Methods

Convolutional Neural Networks (CNNs) are a class of neural networks that specialize in processing data with a grid-like topology, such as images. Inspired by the human visual cortex, CNNs leverage spatial hierarchies and local connectivity to extract features at various levels of abstraction through their convolutional layers. Convolutional layers apply sliding filters to input data, capturing local patterns while pooling layers reduce the dimensionality while

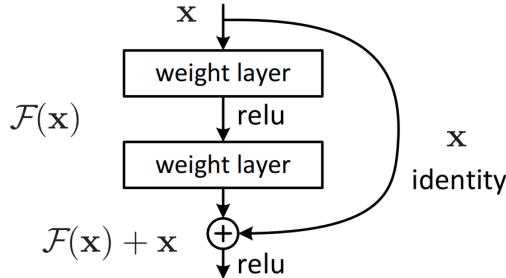


Figure 2.1: Implementation of residual block with a residual connection in ResNet [He+15].

maintaining essential information and introduce a degree of translation invariance by summarizing local features. This structure allows CNNs to learn complex representations in the grid-like data much more effectively compared to the earlier deep learning methods utilizing fully connected layers.

Starting from around the early 2010s, deep CNNs have revolutionized the field of computer vision. One of the pioneering works was the introduction of AlexNet [KSH12], which demonstrated the superiority of CNN over traditional methods on ImageNet image classification task. Since then, architectures improving the performance of CNNs have seen a rapid development and these CNN-based architectures has been employed by many methods as an image feature extraction backbone to tackle tasks including image classification, segmentation and object detection.

Following AlexNet, VGGNet [SZ14] was introduced in 2014, focusing on simplicity and depth using small 3×3 convolutional filters in very deep networks. In the same year, GoogleNet [Sze+14] innovated with inception modules which allowed for the computation of multiple convolution operations with different filter sizes in parallel.

ResNet [He+15], introduced in 2015, became a landmark in CNN architecture. It addressed the vanishing gradient problem that hindered the training of very deep networks by introducing residual connections. Residual connections allow gradients to flow through the network more effectively during the back-propagation, enabling training of very deep networks with hundreds of layers. Such deep networks enabled by improved CNN architectures led to significant performance gains in various computer vision tasks including object detection.

The popular family of two-stage object detectors, Region-based Convolutional Neural Networks (R-CNNs), marked significant advancements in the 2D object detection. The original R-CNN [Gir+14] applied CNNs to region proposals generated by selective search, providing a robust approach for object detection.

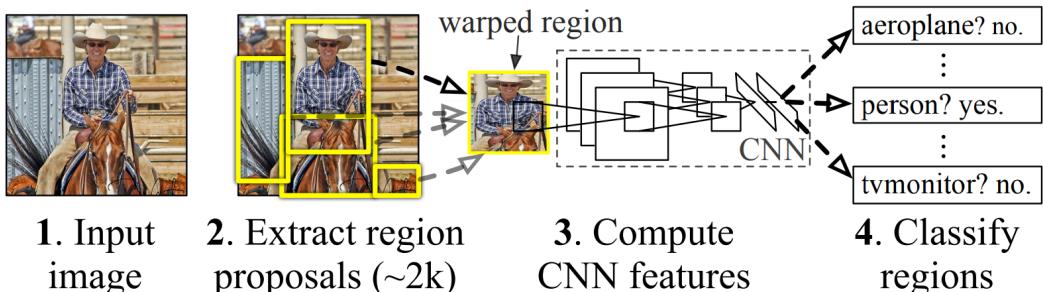


Figure 2.2: A high level overview of object detection pipeline of R-CNN [Gir+14].

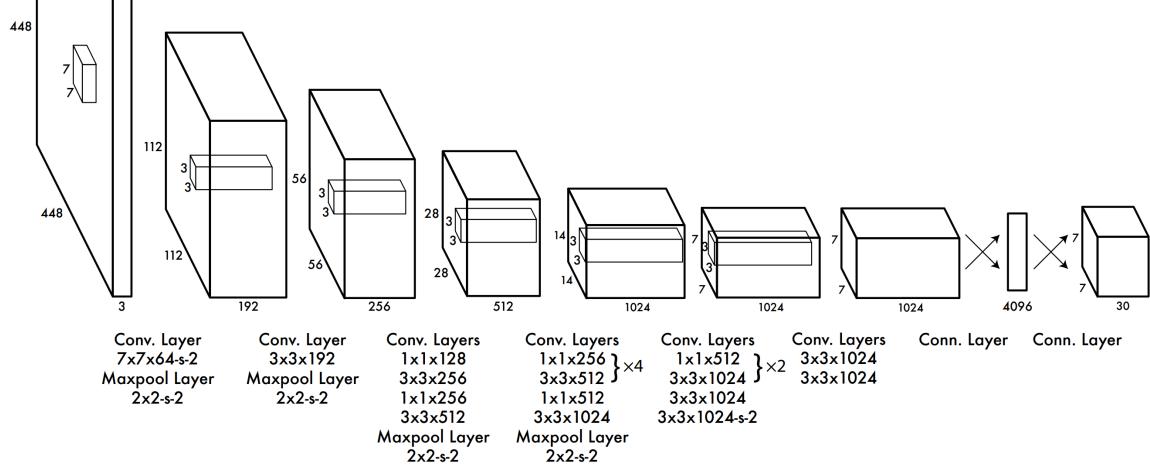


Figure 2.3: The architecture of YOLOv1 [Red+16].

Later, Fast R-CNN [Gir15] improved upon this by integrating region proposal generation and classification into a single network, reducing the computation time. Most recent member of the family Faster R-CNN [Ren+16] further optimized the process by introducing the Region Proposal Network (RPN) to generate region proposals directly from the input, leading to much faster detections compared to earlier R-CNNs.

Another very popular family of object detectors, You Look Only Once (YOLO) [Red+16], approached to object detection problem in a radically different way, re-framing it as a single regression problem. Rather than the two stages of region proposals and further localization and classification of objects in proposed regions, YOLO predicts bounding boxes and class probabilities directly from the images in a single stage. Sacrificing little accuracy, this end-to-end approach significantly improved inference speeds, enabling real-time applications.

2.1.2 Transformer-based Methods

Lately, transformers have completely changed the game in deep learning. Introduction of transformers by the 2017 paper "Attention is All You Need" [Vas+17] have caused a significant shift in the whole deep learning field. Initially, the transformer architecture was aimed at sequence-to-sequence machine translation task in Natural Language Processing (NLP), addressing the limitations of previous models by using a multi-head self-attention mechanism and positional encoding to capture the relationships between words in a sentence, regardless of their distance from each other. This architecture facilitated the development of very capable language models, such as Bidirectional Encoder Representations from Transformers (BERT) [Dev+19] and Generative Pre-trained Transformer (GPT) [Rad+18], which have set new benchmarks in all NLP tasks including text classification, translation, summarization and question-answering.

Transformers' success in NLP has inspired their application in other domains, most notably in computer vision. Vision Transformer (ViT) [Dos+20] was introduced in 2020 and brought a new way of handling image data without any convolution operation. Instead of using filters on tiny local receptive fields to capture spatial hierarchies like CNNs do, ViT treats an image as a sequence of large patches, similar to words in a sentence. This allows self-attention mechanisms to capture global context more effectively and often outperform convolutional methods in image classification tasks on larger datasets. On the other, some methods focused

on bringing the strengths of CNNs and transformers together rather than eliminating convolutions entirely. Introduced in the same year, DETR [Car+20] used both CNN and transformer encoder-decoder architecture to create an end-to-end approach for object detection. Transformer's ability to attend to global context more effectively combined with CNN's ability to focus on local details needed for object detection results in improved performance on complex detection tasks while providing a more streamlined and scalable approach. Rest of this section will cover the fundamental transformer architecture and these two methods which are relevant to the work, in greater detail.

Transformer

The Transformer architecture employs an encode-decoder structure, where both the encoder and the decoder are composed of a stack of repeated layers. Encoder is fed an input sequence and the decoder is fed the corresponding target output sequence shifted right. From these sequences, encoder-decoder structure aims to generate an output sequence to be compared with the given output. Each encoder layer consists of two main sub-layers: multi-head self-attention mechanism and position-wise fully connected feed-forward network. Similarly, each decoder layer includes three sub-layers: a self-attention mechanism, an encoder-decoder attention mechanism, and a position-wise feed-forward network. Each of these sub-layers are wrapped by residual connections and followed by layer normalization to stabilize training.

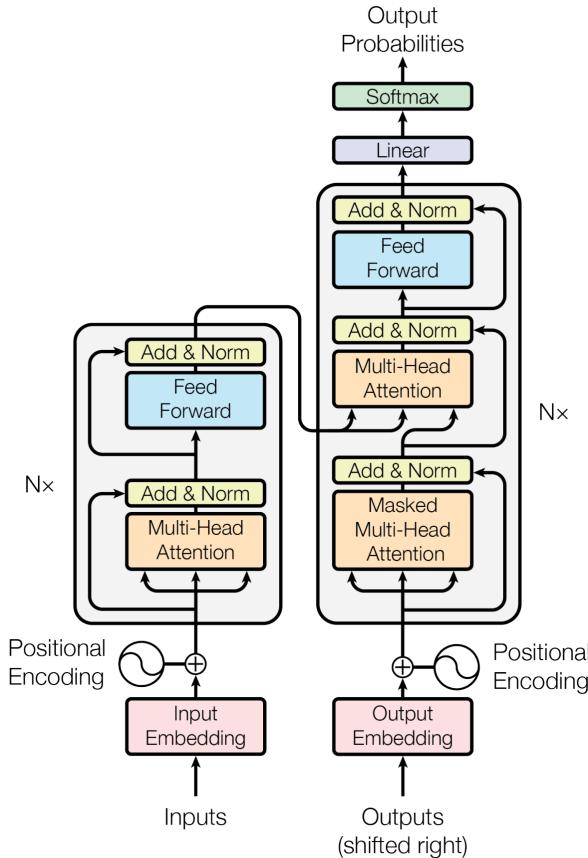


Figure 2.4: The original encoder-decoder transformer architecture from [Vas+17]. On the left is the encoder with multi-head attention, processing input embeddings. On the right is the decoder, with a masked multi-head attention, processing shifted output embeddings. In the decoder's multi-head attention, the key and value matrices are derived from the encoder, while the queries originate from within the decoder.

The self-attention mechanism is the most important part within the transformer. It allows the model to relate different positions of a sequence to compute its representation. The self-attention calculates three vectors from the input: the query (Q), the key (K), and the value (V). These vectors are derived using learned weight matrices:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V$$

where X is the input sequence, and W_Q , W_K , and W_V are the weight matrices. The aim is to compute a weighted sum of the values (V), where the weights are determined by the compatibility of the queries (Q) with the corresponding keys (K). Mathematically, this is expressed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Here, d_k is the dimension of the key vectors, and the division by $\sqrt{d_k}$ is a scaling factor to maintain stable gradients. This self-attention mechanism enables the model to capture dependencies regardless of their distance in the input sequence. Multi-head self-attention that is in the transformer is an extension of self-attention concept which allows model to jointly attend to information from different representation subspaces at different positions. Instead of performing a single attention function, the model splits the queries, keys, and values into multiple heads. Each head independently performs the self-attention operation, yielding different attention outputs. These outputs are then concatenated and linearly transformed into the expected dimension. This is formally defined as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, \dots, \text{head}_h)W_O$$

In the encoder, the self-attention mechanism only attends to the input sequence, ensuring that each position considers all other positions in the sequence. In contrast, the decoder uses masked self-attention, where each position can only attend to earlier positions to prevent information leakage from future tokens. Additionally, the decoder employs a cross-attention mechanism, where K and V are propagated from encoder, allowing it to attend to the encoder's output sequence. This enables the decoder to incorporate information from the entire input sequence while generating the output.

Positional encodings are crucial in the Transformer since it lacks the recurrence that naturally handles sequential information. They are defined as:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

Here, pos is the position, and i is the dimension. These encodings are added to the input embeddings to provide the model with information about the positions of tokens in the sequence.

The architecture combining multi-head self-attention and position-wise feed-forward networks with positional encodings, effectively models complex and long-range dependencies in sequence data without the need for recurrence and convolutions while also enabling more efficient parallelization.

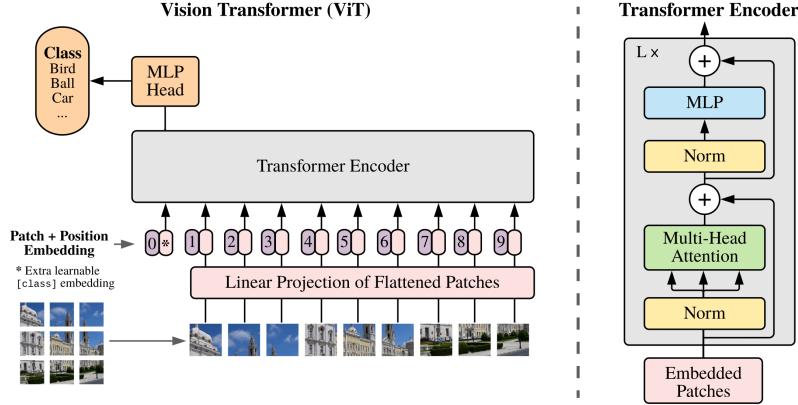


Figure 2.5: The architecture of ViT [Dos+20]. Left side illustrates the image classification pipeline. Right side details the architecture of the transformer encoder.

Vision Transformer

The main idea behind ViT is to interpret input image as a sequence. In order to achieve this, ViT splits the input image into 16×16 patches and applies flattening. Flattened patches are then embedded by a linear layer, which applies the same projection to each patch. Learnable 1D positional embeddings are then added to the patch embeddings to retain positional information. Final embeddings are used as input tokens to the transformer encoder, resembling the processing of a sentence in NLP. In contrast to original NLP transformer, ViT does not have a decoder. Instead, visual tokens produced by the encoder is fed to a simple MLP head for classification.

Detection Transformer

DETR is an innovative framework that combines CNN and transformer encoder-decoder architecture to create an end-to-end approach for object detection. The main idea behind DETR is to re-frame the object detection task as a direct set prediction problem. This approach eliminates the need for traditional components such as anchor generation or non-maximum suppression to handle duplicates, which are typically used to encode prior knowledge about the task. DETR's architecture consists of a transformer encoder-decoder and a set-based global loss function, which together enable unique and direct predictions through bipartite matching between predicted and ground truth boxes.

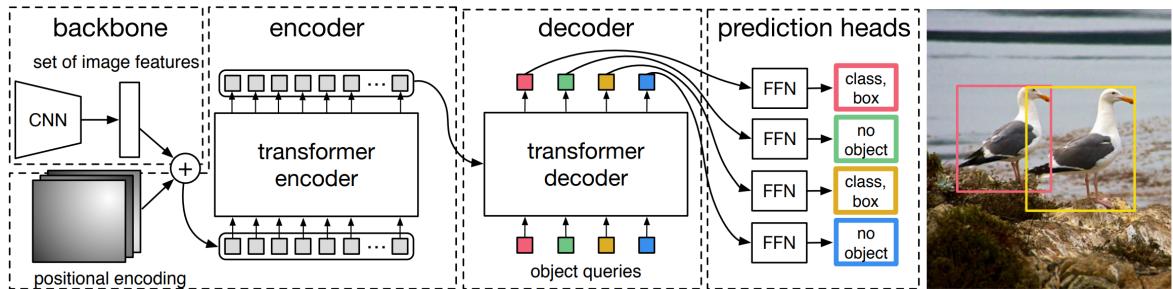


Figure 2.6: The object detection pipeline of DETR [Car+20]. Set of image features extracted via a CNN backbone are embedded by a transformer encoder. A transformer decoder, using object queries, then attends to the encoder output to produce output embeddings, which are subsequently used by prediction heads to perform object detection.

In DETR's pipeline, the CNN backbone extracts a compact feature representation from the input image, which is then flattened and integrated with positional encoding before being passed to the transformer encoder. Then, the transformer decoder processes object queries, which are a small fixed number of learned positional embeddings, attending to the encoder output to produce object predictions. Each output embedding of the decoder is processed by a feed-forward network, which predicts whether the embedding is an object or not, outputting a class label and a bounding box in case it is predicted as an object. This design leveraging the transformer attention mechanism, allows DETR to globally reason about all objects using pair-wise relations and the entire image context, thereby simplifying the detection pipeline and enhancing model performance.

2.2 3D BEV Perception in Autonomous Driving

In autonomous driving, accurate and comprehensive perception of the surrounding environment in 3D is a must for safe and efficient driving. The ego vehicle must be capable of identifying, localizing, and tracking other objects in the scene with precise scales and distances. This task essentially involves reconstructing the physical world around the vehicle to some degree, encompassing all elements relevant to driving decisions. To capture a 360-degree view around the vehicle, autonomous vehicles are equipped with multiple cameras and sensors facing different directions. This setup requires the integration of features from various perspectives into a unified representation of the vehicle's surroundings.

2.2.1 Bird's Eye View (BEV) Perception

A well-known and commonly used method for achieving this unified representation is the Bird's Eye View (BEV). BEV provides a top-down view of the vehicle's surroundings, with the ego vehicle at the center. BEV representations have multiple advantages over other types of representations such as front view or perspective view, which contribute to its popularity. Firstly, BEV allows for rich semantic information, precise localization, and absolute scales, all to be represented in world coordinates. This enables the seamless integration of information from multiple views and modalities without requiring additional stitching operations. Secondly, BEV does not suffer from occlusion, distortion, or scaling issues commonly found in 2D tasks. Moreover, representing objects or road elements in BEV representation makes it convenient for subsequent autonomous driving modules such as motion planning and control. Thus, the task of 3D perception of surroundings from different views to create a unified BEV representation, known as "BEV Perception", has become a prominent research area recently.

BEV perception methods can generally be categorized into three main areas: camera-based methods, LiDAR-based methods, and fusion methods. Camera-based methods use multi-view images acquired from cameras positioned around the vehicle to construct the BEV representation. Light Detection and Ranging (LiDAR) based methods use 3D point clouds obtained from LiDAR sensors to perform object detection and BEV construction directly in 3D space. While camera data contains dense color and texture information, it fails to capture depth information. Conversely, LiDAR provides precise depth and structural information but suffers from limited range and sparsity. Fusion methods combine both camera and LiDAR data to construct BEV representations by fusing information from both modalities. Ideally, fusion methods offer the best overall performance; however, the challenge of effectively fusing data from different modalities remains a significant research problem.

This work focuses on camera-based BEV perception. While LiDAR provides precise 3D depth information, a suite of LiDAR equipment is typically ten times more expensive than a camera setup. Therefore, enhancing camera-based methods to achieve LiDAR-level 3D perception holds significant value in the industry, offering cost-effectiveness and simplified sensor integration and maintenance, which reduces the overall system complexity. Additionally, LiDAR lacks the high-resolution visual data, including rich color and texture information, that cameras offer. This information is crucial for identifying and interpreting various objects and features in the driving environment, particularly color-based elements such as traffic lights and road signs. Thus, camera-based approaches have attracted extensive attention in recent years [Ma+23] and the gap between camera-based methods to others is narrowing, thanks to the recent advancements in the computer vision.

2.2.2 Camera-based Methods

As mentioned in the previous section, a single camera pointing in a certain direction cannot directly capture depth information. To calculate depth directly, one needs to use triangulation from two calibrated cameras directed at slightly different angles, a process known as stereo vision. However, stereo vision is less commonly used in autonomous driving due to several technical challenges and practical shortcomings. First, triangulation in stereo vision requires a computationally intensive process of matching corresponding points between the two cameras, known as stereo matching. This process, especially when dealing with high-resolution images and large disparities, incurs significant computational overhead, making it less suitable for applications where processing power and response time are critical. Another issue is that the accuracy of stereo depth measurements decreases with increasing distance from the cameras, limiting their effectiveness for long-range perception. As the distance to the target object increases, the disparity between the corresponding points in the images decreases, making it harder to measure depth accurately. To compensate for this, the separation between the two cameras must be increased to maintain sufficient disparity for accurate depth calculation at longer distances, which is impractical in an autonomous driving scenario. Thus, a single-camera setup, also called monocular vision, is more common in autonomous driving.

The task of estimating depth from a single image is known as Monocular Depth Prediction (MDP). MDP is considered an ill-posed problem because infinitely many 3D scenes can produce the same 2D projection, leading to ambiguity in depth estimation. Therefore, priors and assumptions learned from data are necessary to predict depth accurately from a single image. One approach to constructing a 3D scene is to start from 2D images and use such depth estimations to build the 3D space. The pioneering work in this approach, Lift-Splat-Shoot (LSS) [PF20], predicts a categorical depth distribution for each pixel on a 2D feature map, and "lifts" 2D features into 3D voxel space by the corresponding depth estimation, essentially projecting them along a ray in 3D. Subsequently, 3D points on top of each other are pooled together and further processed to construct the top-down BEV. Many LSS-based works [Hua+22; Li+22b; Hu+23] have since improved upon this idea. These models incorporate various enhancements, including better depth prediction accuracy, more efficient processing pipelines, and integration of temporal information to refine the BEV representations.

Another group of methods [Pan+20; Yan+21; GV22] utilize neural networks for the implicit representation of camera projection relationships rather than explicit depth estimation followed by a geometric projection. Implicit projection methods focus directly mapping 2D image features to BEV representations through end-to-end neural networks .Instead of generating explicit depth maps, the network learns to interpret the spatial relationships and

project these features into the BEV space, encoding the geometric and semantic information from the images into a latent space that can be decoded into a BEV representation. By bypassing the explicit depth estimation step, these methods can reduce computational complexity and potentially improve robustness against errors in depth prediction. They are particularly advantageous for tasks such as segmentation, where the precise geometric accuracy of depth estimation may be less critical thus, they are more commonly used for map segmentation.

Recently, another branch of works following DETR3D [Wan+21] has gained popularity. These methods start from 3D and adopt a transformer attention mechanism to project queries in the BEV space back to 2D images to construct BEV features from image features. Building upon the idea of DETR, which revolutionized object detection by framing object detection task as a direct set prediction problem, DETR3D incorporates a 3D object query framework. DETR3D starts with extracting 2D features from multi-view images using a backbone CNN, similar to the original DETR. It utilizes a transformer encoder-decoder architecture where the encoder processes multi-view image features, and the decoder generates 3D object queries that attend to these features to predict object locations and classes in 3D space. These queries attend to the encoded 2D features through multi-head self-attention and cross-attention mechanisms, allowing the model to reason about the spatial relationships between different views and accurately localize objects in 3D space.

One of the most popular and high-performing methods of this kind is BEVFormer [Li+22c]. This work employs BEVFormer as a BEV encoder to obtain BEV features extracted from multi-view camera images, and hence, it is discussed in detail in the following section.

2.2.3 BEVFormer

BEVFormer is a cutting-edge framework designed to generate unified BEV representations from multi-camera images using spatio-temporal transformers. The framework's architecture consists of 2D image feature extraction backbone and multiple transformer encoder layers that utilize specialized attention mechanisms to aggregate spatial and temporal features. The core mechanisms of BEVFormer include BEV queries, spatial cross-attention, and temporal self-attention.

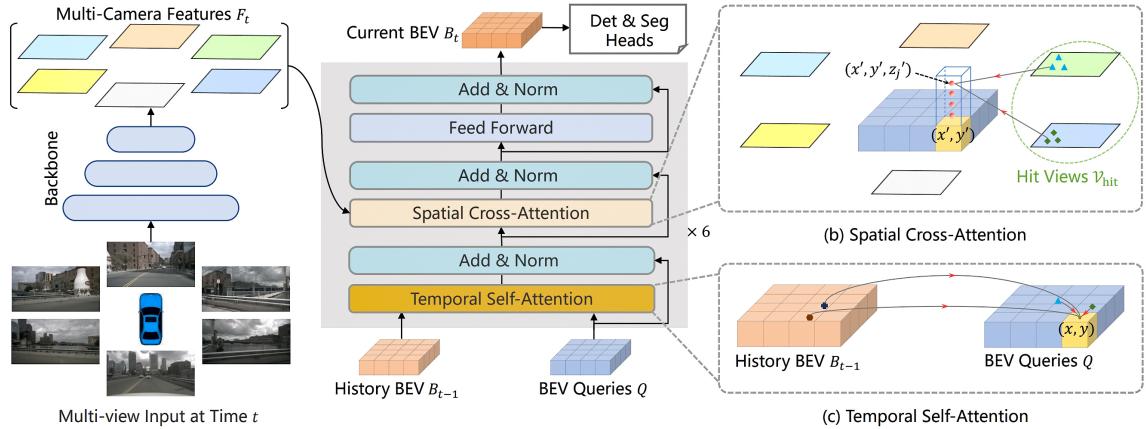


Figure 2.7: The architecture of BEVFormer [Li+22c]. The left section illustrates 2D multi-camera feature extraction using a convolutional backbone. The middle section details the transformer layers, which utilize BEV from the previous timestamp and BEV queries in 3D space, attending to multi-camera features through spatial cross-attention to generate the BEV feature representation. The right section demonstrates the operation of spatial cross-attention and temporal self-attention mechanisms.

BEVFormer’s pipeline begins with BEV queries. These are grid-shaped learnable parameters that query features in the BEV space. BEV queries and BEV features from the previous time-step are fed into temporal self-attention block in the transformer encoder. Temporal self-attention mechanism integrates temporal information by aligning current BEV queries with previous BEV features, accounting for ego-motion and ensuring temporal continuity. The intermediate BEV features are then processed together with image features extracted by the ResNet backbone from multi-view camera images, through the spatial cross-attention layer. The deformable attention mechanism in spatial cross-attention each BEV query only interacts with the regions of interest in the corresponding image features, thereby enhancing computational efficiency. Resulting features are processed further by a feed-forward network as in original transformer architecture. Transformer blocks are repeated multiple times refining the BEV features iteratively.

Resulting BEV feature map is a versatile map that can be used for various autonomous driving perception tasks by different task heads such as 3D object detection or map segmentation. In this work, BEV feature maps generated by BEVFormer are utilized as comprehensive visual features of the scene to achieve traffic scene understanding.

2.3 Language Models

A language model is a system designed to process, understand and generate human language. Fundamentally, it involves a mathematical representation of language that enables the prediction of next word or sequence of words based on the preceding context. In NLP, traditional models have long relied on statistical and rule-based approaches. Early models, such as n-grams, were based on the Markov assumption, which states that the probability of a word depends only on a fixed number of previous words. While this approach was effective to a degree, it struggled to capture long-range dependencies and contextual nuances in language.

To address these limitations, sequence models such as Recurrent Neural Networks (RNNs) and Long-Short Term Memory (LSTM) networks gained prominence, particularly for machine translation tasks. These models have significantly improved over earlier approaches by better handling sequential data. However, their effectiveness was still limited by issues such as vanishing gradients and the difficulty of managing long-range dependencies.

Later, the emergence of transformer-based architectures marked a pivotal turning point in NLP. Addressing many limitations of the previous models, transformers have become the foundation of nearly all top-performing language models today. Beyond their ability to handle long-range dependencies, transformer architectures offer an even more impactful attribute: their ability to scale both in terms of model size and the amount of data they can process. This scalability has enabled the development of enormous language models trained on vast, internet-scale datasets.

2.3.1 Large Language Models

Language models with a massive number of parameters, often in the order of billions, are typically referred as Large Language Models (LLMs). In 2018, the year following the introduction of the transformer architecture, BERT [Dev+19] was introduced. BERT only adopts the encoder part from the original transformer architecture and performs deep bi-directional training, which allows it to consider both left and right contexts simultaneously in all layers,

contrasts with traditional left-to-right or unidirectional models. This training strategy with a large corpus of text allows it to capture rich contextual information and nuanced patterns from natural language, which immediately made it a very popular method for both language and vision-language applications.

Many subsequent works improved upon the BERT model, such as A Lite BERT (ALBERT) [Lan+19], which reduced BERT’s model size while maintaining performance by sharing parameters across layers and decomposing the embedding matrix, and the Robustly optimized BERT approach (RoBERTa) [Liu+19], which optimized training procedures and scaled up the model. Language models began to rapidly increase in both model size and training data size, proving that larger scales boost performance. Scaling laws of LLMs [Kap+20] shed light on how a model’s quality evolves with its size increases, encouraging researchers in scaling their methods. NVIDIA’s Megatron-LM [Sho+19] demonstrated the benefits of model parallelism and large-scale training, achieving competitive performance on NLP tasks through efficient utilization of computational resources. The GPT series also scaled up significantly with GPT-2 [Rad+19], which, with 1.5 billion parameters, showcased impressive text generation capabilities. Building on this, the Turing-NLG [Mic20] model by Microsoft Research pushed the boundaries of large-scale language modeling further, with 17 billion parameters.

The GPT series, oriented towards language generation, demonstrated unprecedented advancements and capabilities in LLMs, especially starting in 2020 with GPT-3 [Bro+20]. Scaling up to 175 billion parameters, GPT-3 became one of the largest language models at the time. GPT-3, and its subsequent iteration GPT-3.5, which powers ChatGPT, has drawn immense attention from researchers, industry, and the general public due to its remarkable performance in understanding, reasoning, question answering, and text generation.

Following the success of GPT-3, GPT-4 [Ope+24] and many other LLMs such as LLaMA [Tou+23], and Gemini [Gem23] have been developed, continuing the trend of innovation and expansion in large language models. Today, an ever-increasing number of researchers and institutions are exploring ways to harness the linguistic understanding and reasoning capabilities of LLMs to tackle previously deemed extremely difficult.

LLMs leverage the transformer architecture but are scaled up significantly, both in terms of model size and training data. This scaling endows them with the capacity to perform a wide range of tasks with minimal fine-tuning, demonstrating impressive generalization capabilities. Their ability to perform diverse tasks with little or no task-specific training makes them incredibly valuable across various domains, with the zero-shot or few-shot transfer learning possibilities across various downstream tasks.

2.3.2 Vision-Language Models

Vision-Language Models (VLMs) are designed to handle tasks that require understanding and generating language in the context of visual inputs, such as image captioning, visual question answering, and multimodal translation by integrating visual and textual information. Before the introduction of transformers and BERT, various methods had explored the integration of vision and language, laying the groundwork for subsequent advancements. With the advent of transformers, significant progress was made in this field.

By leveraging the strengths of transformer architectures as LLMs do, VLMs process images and text through joint representations, allowing them to understand and describe visual content accurately. VisualBERT [Lu+19], introduced in 2019, was one of the first models to integrate visual features into the BERT architecture, aligning image regions with corresponding text. This model demonstrated effective performance on tasks such as visual question

answering and image captioning by leveraging joint representations of visual and textual data. In the same year, ViLBERT [Lu+19] advanced this approach by using a co-attention mechanism, which allowed for more sophisticated processing and integration of image and text data.

UNITER [Che+19], also introduced in 2019, leveraged large-scale pre-training on image-text pairs, achieving state-of-the-art results by learning joint representations for visual and textual data. ALIGN [Jia+21] extended this concept by scaling up pre-training to billions of image-text pairs, demonstrating impressive zero-shot performance across numerous vision-language tasks. This model highlighted the importance of large scale data and contrastive learning for improving the generalization capabilities of VLMs. The trend of scaling up continued with more recent models like CLIP [Rad+21], which can generate detailed image descriptions or multimodal-to-vision models such as DALL-E [Ram+21], that can create images based on text descriptions, showcasing their potential in bridging the gap between visual and linguistic cognition.

The integration of visual and language capabilities opens up new possibilities for AI agents to achieve deeper and contextual understanding of both human language and visual world. It also enables more intuitive and explainable approaches for enhanced human-computer interaction. Following the rise of LLMs in recent years, large VLMs have been increasingly attracting attention in various domains to incorporate visual understanding to language models.

This work uses the VLM architecture from BLIP [Li+22a] as a simple, lightweight yet highly generalizable and performant multimodal vision-language understanding framework. BLIP utilizes a transformer based unified multimodal encoder-decoder module that is pre-trained with image captioning task on large language-image data, offering great flexibility and transferability for various downstream tasks.

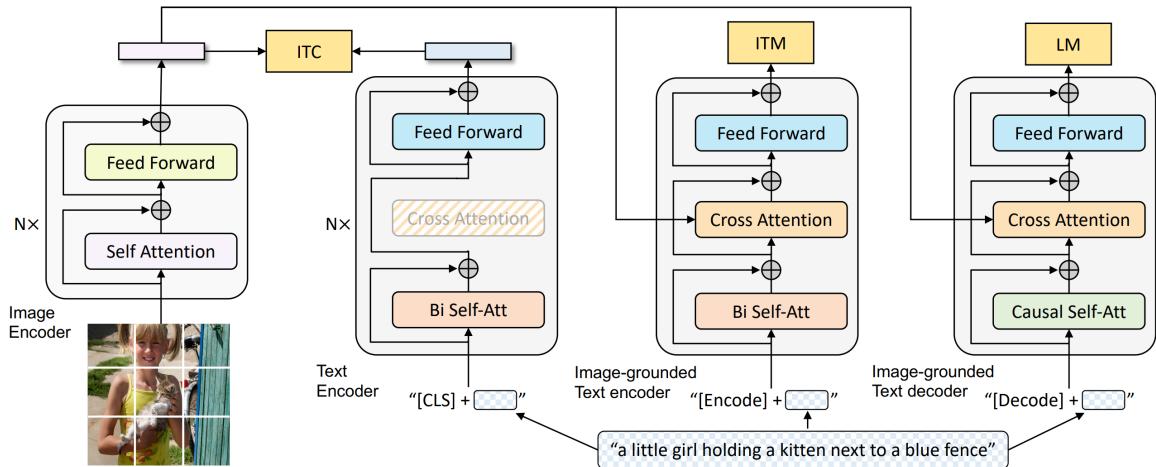


Figure 2.8: Pre-training model architecture and objectives of BLIP [Li+22a]. The leftmost transformer block illustrates a ViT as the image encoder. The remaining three transformer blocks represent the three configurations of the unified multimodal encoder-decoder, with same-colored layers indicating shared weights. The text encoder employs bidirectional self-attention layers. The image-grounded text encoder activates additional cross-attention layers to attend to image features from the image encoder. The image-grounded text decoder replaces the bidirectional self-attention layers of the encoder with causal self-attention layers.

The pre-training pipeline of BLIP is designed to align visual and textual features effectively. During this stage, the multimodal encoder-decoder optimizes for three pre-training objectives, operating in three configurations: text encoder, image-grounded text encoder and

image-grounded text decoder. The text encoder configuration optimizes an image-text contrastive loss, aligning their feature representations. Simultaneously, image-grounded text decoder configuration optimizes an image-text matching loss, further refining this understanding by training the model to identify whether a given image and text pair is semantically aligned. Finally, the image-grounded text decoder configuration optimizes a language modelling loss, focusing on generating coherent textual descriptions based on visual inputs.

The combination of these pre-training objectives ensures that BLIP develops a comprehensive understanding of both visual and textual modalities and their interconnections, leading to high performance on downstream tasks.

More recently, BLIP-2 [Li+23] builds on the original BLIP with a focus on the computational cost. Instead of the end-to-end approach of BLIP, BLIP-2 integrates a frozen image encoder and an LLM which are not trained hence significantly reducing the number of trainable parameters. As an intermediary between the frozen image encoder and the frozen LLM, BLIP-2 introduces a lightweight Querying Transformer (Q-Former), employing learnable query vectors to extract related visual features to bridge the modality gap between vision and language.

Using very similar objectives to the BLIP’s pre-training, Q-former undergoes a two-stage pre-training process. Initially, the Q-Former is trained for vision-language representation learning, focusing on extracting visual features relevant to textual descriptions. Subsequently, the Q-Former’s output is connected to the frozen LLM for vision-to-language generative learning, enabling the model to produce text based on visual inputs.

Chapter 3

Related Work

3.1 VLMs in Autonomous Driving

The implementation of VLMs across various tasks in autonomous driving has recently gained popularity. Numerous studies are investigating how to harness the generalization and few-shot learning capabilities of VLMs and LLMs to enhance various aspects of autonomous driving systems, such as accuracy, adaptability, explainability, and reasoning.

Method	Year	Focus	Contribution
Reason2Drive [Nie+23]	2023	Perception/Understanding	Introduces a dataset with video-text pairs and an approach empowering VLMs for enhanced reasoning.
NuPrompt [Wu+23]	2023	Perception/Understanding	Introduces an object-centric language prompt set for driving scenes, enabling trajectory prediction
Talk2BEV [Cho+23]	2023	Perception/Understanding	Enhances BEV maps with language, enabling improved general-purpose visual and linguistic reasoning.
NuScenes-QA [Qia+23]	2023	Perception/Understanding	Introduces a novel VQA task and dataset based on nuScenes along with a multimodal baseline.
NuScenes-MQA [Ino+24]	2023	Perception/Understanding	Introduces markup annotation strategy in a QA task and dataset based on nuScenes with a camera-only baseline.
DriveLM [Sim+23]	2023	Perception/Understanding	Introduces a comprehensive VQA task and dataset based on nuScenes, with a benchmark using multi-step reasoning.
EM-VLM4AD [GGT24]	2024	Perception/Understanding	Proposes an efficient, lightweight VLM for AD VQA using quantization and low-rank adaptation.
MAPLM [Cao+24]	2024	Perception/Understanding	Introduces a new out-of-domain AD dataset with comprehensive annotations to address limitations in public datasets.
DriveGPT4 [Xu+24]	2023	End-to-End Driving	Proposes a multimodal model for interpretable autonomous driving.
ADAPT [Jin+23]	2023	End-to-End Driving	Proposes an end-to-end framework for action narration and reasoning in autonomous driving.
GtN [Jai+22]	2022	Navigation/Planning	Proposes vision language navigation tool that controls vehicles by text-driven commands.
CoverNet-T [Key+23]	2023	Navigation/Planning	Proposes a text and image representation for traffic scenes, enhancing trajectory prediction.
DLaH [Fu+23]	2023	Decision/Control	Proposes an LLM for human-like reasoning, interpretation, and memorization in autonomous driving.
DriveGAN [Kim+21]	2021	Data Generation	Introduces a neural simulator offering controllable re-simulations of traffic scenes.
DriveDreamer [Wan+23b]	2023	Data Generation	Introduces a real-world-derived model for precise video generation and realistic driving policies.

Table 3.1: A non-exhaustive overview of recent works integrating VLMs in autonomous driving.

Projects like DriveGPT4 [Xu+24] and ADAPT [Jin+23] are incorporating VLMs to achieve end-to-end autonomous driving, thereby simplifying complex autonomous driving architectures that traditionally rely on multiple independent modules. DriveGPT4 processes raw sensor data and human inquiries, generating predicted control signals and corresponding responses. It retains the robust zero-shot generation capability of VLMs, enabling it to manage previously unseen scenarios effectively. ADAPT proposes an end-to-end action narration and reasoning framework for self-driving vehicles. Utilizing video input, ADAPT continuously generates control signals along with narration and reasoning descriptions for proposed actions, aiming for an interpretable driving system.

Other than end-to-end driving, many other work [Jai+22; Key+23; Fu+23; Kim+21; Wan+23b] utilize VLMs for various modules and sub-tasks within autonomous driving such as language guided navigation and planning, human-like decision-making and control, generative models driven realistic synthetic data creation. [Zho+23]

3.2 Traffic Scene Understanding

The majority of works utilize VLMs for traffic scene perception and understanding by leveraging prior world knowledge of pre-trained large VLMs to enhance perception and introduce many novel tasks for understanding the environment. The remainder of this section reviews recent related works published just prior to or during the development of this thesis.

Talk2BEV. Talk2BEV [Cho+23] proposes a method on directly augmenting BEV maps with language, alongside a QA benchmark. The integration of large vision-language models with BEV maps result in contextually enhanced maps that can be used to improve general-purpose visual and linguistic reasoning capabilities in autonomous driving scenarios.

NuScenes-QA. NuScenes-QA [Qia+23] introduces a novel VQA task, providing a large-scale multimodal QA dataset on top of nuScenes [Cae+20] and baseline model as a benchmark. VQA task is tailored to evaluate existing models’ ability to understand and reason with complex visual data in multi-modal and multi-frame traffic scenes. The proposed baseline uses LSS to construct BEV feature from multi-view images and fuses it with BEV features extracted from LiDAR point clouds. Subsequently, object features obtained from fused BEV maps are used by a transformer-based VQA model MCAN [Yu+19] to accomplish the VQA task. During training phase, only the VQA model is trained as the visual embeddings are pre-computed offline.

NuScenes-QA Dataset. Based on the popular autonomous driving dataset nuScenes, NuScenes-QA consists of 460K question-answer pairs on object existence, recognition, counting, status and comparison, making it one of the largest 3D-related QA datasets. In order to generate question-answer pairs, NuScenes-QA uses an automated method utilizing structured data from scene graphs and manually crafted question templates. Scene graphs, representing objects and their relationships, are created from 3D ground truth annotations, with object bounding-boxes and categories annotated at a frequency of 2Hz. Inter-object relations between traffic elements are defined based on spatial positions such as front, back, front left, front right, back left, and back right. Based on 66 manually designed question templates, diverse question formations are crafted by assigning parameters and deducing answers from the scene graphs. Ill-posed questions and inappropriate expressions are filtered out during post-processing. Additionally, questions are categorized into zero-hop, indicating no reasoning between objects, and one-hop, indicating one-step spatial reasoning, to evaluate reasoning performance.

NuScenes-MQA. Similarly, NuScenes-MQA [Ino+24] introduces another multimodal VQA framework and dataset based on nuScenes, annotated by a novel technique that encloses target information such as objects, counts, answers, locations and distances within markups. As their camera-only baseline VLM, NuScenes-MQA architecture employs six ViTs pre-trained on CLIP, each dedicated to extracting features from its corresponding camera’s image. Features from each camera are fed to a decoder-only language model after positional embedding and adaptation by dedicated adapter modules to produce answers enveloped in markups. They train all parameters spanning the ViTs, adapters and the language model.

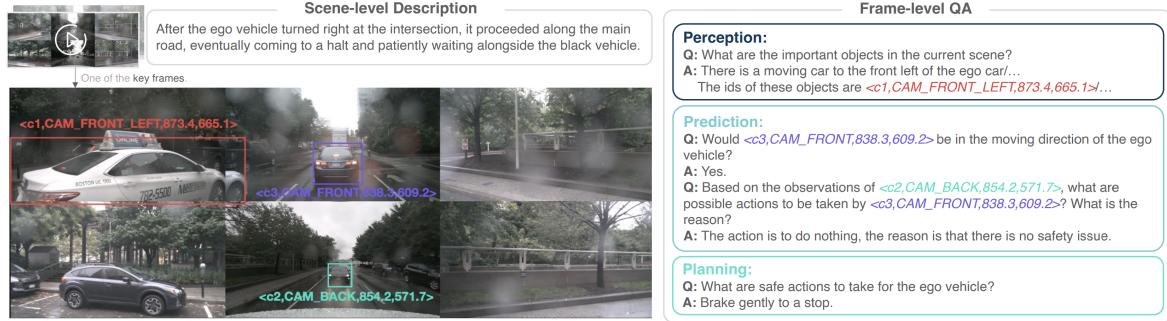


Figure 3.1: An overall Composition of DriveLM-nuScenes [Sim+23], illustrating example perception, prediction and planning questions from the dataset. Key objects are encoded using c tags that contain camera affiliations, and the center coordinates of their 2D bounding boxes within the corresponding camera frame.

DriveLM. Expanding on the question-answering paradigm, recent DriveLM [Sim+23] introduces a more comprehensive VQA task and dataset on top of nuScenes with questions covering perception, prediction, planning and behavior, which require holistic situational understanding and zero-shot generalization capabilities. DriveLM-Agent, the proposed baseline approach built upon a general pre-trained VLM, formulates the task as a graph VQA problem. DriveLM-Agent addresses questions from a logical dependency viewpoint, using predicted answers from previous step as the context for the following questions, inspired by the multi-step human reasoning process. They employ BLIP-2 as their base VLM, which trains an Q-Former adapter in between the ViT and the encoder-decoder. The DriveLM-Agent along with the DriveLM-Data containing significantly more text annotation per frame compared to the other similar datasets, provide a challenging benchmark for future research in autonomous driving scene understanding.

DriveLM-nuScenes Dataset. The dataset includes 4072 training frames and 799 validation frames, featuring scene-level descriptions and frame-level question-answer pairs with 2D bounding boxes within multi-view images from nuScenes. Each frame with six camera images are associated with around 90 questions on average, providing significantly more annotations per frame compared to the other existing benchmarks. Questions are categorized into perception, prediction, planning and behavior. Perception questions involve examining the entire frame and identifying, localizing and describing visual features of key objects in the frame. Prediction questions deal with projecting future states of key objects by estimating their possible actions and inter-actions with each other. Planning questions focus on the safety and outcomes of the possible future actions of the ego vehicle. Behavior questions involve estimating the current state and actions of the ego vehicle such as steering direction and driving speed. To generate question-answer pairs, DriveLM-nuScenes utilizes ground truth data from nuScenes supplemented by extensive manual annotations. Due to the complexity of prediction, planning and behavior tasks, answers to these questions are manually annotated. Annotators with driving experience, review entire video clips to identify frames rich in scene information, labeling the key objects and their behavior. Key objects are encoded with specific tags in the format $< c, CAM, x, y >$, where c is the identifier, CAM indicates the camera where the object's center point is located, and x, y represent the coordinates of the 2D bounding box in the respective camera's coordinate system. Each key frame includes a dictionary recording more basic information about the key objects, such as the size of the bounding box, the category, the moving state, and a visual description. DriveLM-nuScenes stands out for its larger scale, greater comprehensiveness, and more complex structure compared to previous datasets.

EM-VLM4AD. EM-VLM4AD [GGT24] approaches the task introduced by DriveLM from

the efficient model deployment perspective. The method emphasizes the importance of energy-efficient model designs that can operate in real-time while maintaining high accuracy. EM-VLM4AD employs smaller VLMs and uses techniques such as quantization and low-rank adaptation (LoRA) [Hu+21] to further boost efficiency.

MAPLM. Contrasting to studies leveraging commonly available public autonomous driving datasets such as nuScenes, newer MAPLM [Cao+24] argues that these datasets do not fully reflect real-world scenes primarily due to lack of exhaustive edge cases and their extensive use in training the latest LLMs, including GPT-4V. To address this, MAPLM introduces a new out-of-domain, large-scale dataset featuring panoramic 2D images, 3D LiDAR point clouds, and HD maps, all accompanied by human-verified annotations. In the baseline for MAPLM-QA benchmark, pre-trained CLIP-based visual encoders are utilized to extract features from the images and point clouds respectively. These extracted features are mapped into word embedding space by adaptors before integrating with HD map information tokens. Subsequently, instruction-tuned LLMs process all features to generate coherent responses.

Chapter 4

Implementation

With the goal of integrating language features with vision for enhanced traffic scene understanding, this work employs the BLIP framework, renowned for its simplicity, lightweight nature, and high generalizability as a multimodal vision-language approach. To better adapt this framework to the domain of autonomous driving, images in the original method are replaced with BEV feature maps, aiming to leverage their refined spatio-temporal information optimized for traffic scene perception.

This chapter presents the implementation of BEVBlip in two sections, Pre-training and Fine-tuning. The Pre-training section elaborates on the alignment of BEV features with the framework, encompassing data preparation and generation, pre-training model architecture, and pre-training objectives. The Fine-tuning section focuses on the downstream task of visual question-answering (VQA) for traffic scene understanding, detailing the use of the DriveLM-nuScenes dataset for training, modifications to the model architecture for the VQA task, post-processing steps, and experimental setups for various visual feature extraction strategies.

4.1 Pre-training

Replacing images with BEV feature maps necessitates the alignment of the unified multimodal encoder-decoder, originally trained with image features, to this new modality. Consequently, an initial pre-training step is conducted following the BLIP pre-training model architecture. The image captioning task from BLIP's pre-training is adapted to a traffic scene description task using BEV feature maps, facilitating the alignment of visual and text feature spaces. In the next section, preparation of the data needed for the traffic scene description task is detailed. Subsequent sections cover the model architecture and pre-training objectives.

4.1.1 Data Preparation

BEV Feature Map Extraction

In this study, BEVFormer is utilized to generate BEV feature maps that encompass extensive spatio-temporal information about the vehicle's surroundings. For this work, BEVFormer is not trained; instead, BEV feature maps from nuScenes frames are pre-computed and stored offline to mitigate computational overhead during training. The BEV feature maps are generated using the smallest BEVFormer model, BEVFormer-tiny, due to resource limitations concerning both the memory requirements for inference and the storage capacity needed to store computed BEV feature maps. BEVFormer-tiny, compared to the base model, employs a

smaller ResNet-50 rather than ResNet-101 as the image backbone, a reduced encoder with three transformer layers instead of six, a smaller input image size reduced from 1600x900 to 800x450, and single-scale features instead of multi-scale features. For each nuScenes frame containing multi-view images from six cameras, BEVFormer-tiny uses two previous frames from the within past two seconds to construct a history BEV. This integration allows temporal information, such as the motions of surrounding objects, to be included in the BEV feature map for the current frame. The resulting BEV feature map is a 50x50 grid consisting of 256-dimensional feature vectors at each grid location. The map covers a perceptive range of [-51.2m, 51.2m] around the ego vehicle in both the X and Y axes, resulting in a grid resolution of 2.048 meters. The computed BEV feature maps from 34,149 frames are linked with corresponding nuScenes frames and stored as tensors of shape (50, 50, 256), to be utilized later by data loaders for training and inference.

Data Generation with GPT-3.5

```
model="gpt-3.5-turbo",
messages=[
  {
    "role": "system",
    "content": "You will be given comma separated information on a traffic scene. Your task is to generate simple statement sentences using the information. Write from the perspective of the ego vehicle and always refer to yourself as 'the ego vehicle'. Use simple present tense. Only use the provided information. Do not add new information.",
  },
  {
    "role": "user",
    "content": "'Rain, parked cars, wait at intersection, ped crossing crosswalk, turn right, truck'",
  },
  {
    "role": "assistant",
    "content": "The ego vehicle is driving in the rain. There are parked cars around. The ego vehicle waits at an intersection. A pedestrian is crossing the crosswalk. The ego vehicle turns right and notices a truck.'",
  },
  {
    "role": "user",
    "content": f'{scene_description}'",
  },
]
```

Table 4.1: Prompt given to GPT-3.5 to generate statements used as text targets. Writing from the perspective of the ego vehicle and explicitly referring to it as 'the ego vehicle' helps stitching the given information together. Using simple present tense regularizes the verbs. The model is also given the instruction not to add new information to prevent hallucination. Alongside the instruction, a single example is also provided as a chat history to enhance responses.

The nuScenes dataset provides textual descriptions for each scene, containing key information such as road conditions, locations, weather, and significant objects and actions. To

align visual and textual spaces during the model’s pre-training, the scene descriptions from nuScenes serve as an excellent candidate, offering a global overview of the scene. However, these descriptions are presented as a list of unordered keywords and phrases, sometimes with spelling mistakes and often containing abbreviations like "veh" for vehicle or "peds" for pedestrians. In this format, they are not optimal for use as textual targets in pre-training objectives.

To regularize these words and generate coherent statements about the scenes in natural, full-sentence language that logically connects the keywords, the widely used general-purpose LLM GPT-3.5 is employed. Given the relative simplicity of the task for a modern LLM, GPT-3.5 Turbo version with 20 billion parameters is preferred over the much larger GPT-4 due to its speed and cost-effectiveness. With its extensive world knowledge and generalization capabilities, GPT-3.5 Turbo can effectively comprehend our task with just a single example and perform reliably.

Table 4.1 shows the prompt provided to the GPT-3.5 Turbo model via the OpenAI API to generate statements used as text targets. Table 4.2 showcases few example statements generated by the model, highlighting it’s capabilities. The model is utilized to pre-compute full-sentence scene descriptions for each of the 850 scenes in the nuScenes dataset prior to training.

nuScenes Scene Description	GPT-3.5 Scene Statements
Sitting construction worker, parked bicycle, ped walking along road, construction vehicle, worker sitting on back of van	The ego vehicle sees a construction worker sitting. The ego vehicle notices a parked bicycle. A pedestrian is walking along the road. There is a construction vehicle nearby. A worker is sitting on the back of a van.
Parking lot, peds, parked cars, parked scooters, eletronic display	The ego vehicle is in a parking lot. There are pedestrians, parked cars, and parked scooters around. An electronic display is present.
Bus indicates turn but does not go, overtake bus	The ego vehicle sees a bus indicating a turn but not moving. The ego vehicle overtakes the bus.
Night, rain, high speed, scooter, dense traffic at other lane	The ego vehicle is driving at high speed at night in the rain. A scooter is nearby. There is dense traffic in the other lane.
Stationary ego vehicle, person getting out of ego vehicle, parking lot, peds	The ego vehicle is stationary. A person is getting out of the ego vehicle in a parking lot. There are pedestrians nearby.

Table 4.2: Few example statements generated by GPT-3.5 on nuScenes data. GPT expands irregular abbreviations, fixes typos and forms full-sentences reordering and logically connecting words from given scene description.

4.1.2 Model Architecture

Following the pre-training model architecture of BLIP, this work utilizes a VLM with unified multimodal encoder-decoder transformer, as depicted in Figure 4.1. Contrastively to BLIP, images are substituted with BEV feature maps, pre-computed using multi-view images from nuScenes. The BEV feature map is input into a compact Vision Transformer (ViT) module with a transformer encoder of depth three. Given that the images are replaced with BEV feature maps, the ViT is trained from scratch. The 50x50 BEV feature map is divided into 5x5 patches. As in the original ViT architecture, each patch is flattened and then embedded by a linear projection layer common to all patches. Learnable 1D positional embeddings are added to these patch embeddings to retain positional information, resulting in 100 patch embeddings, each being 768-dimensional. These embeddings are processed by the transformer

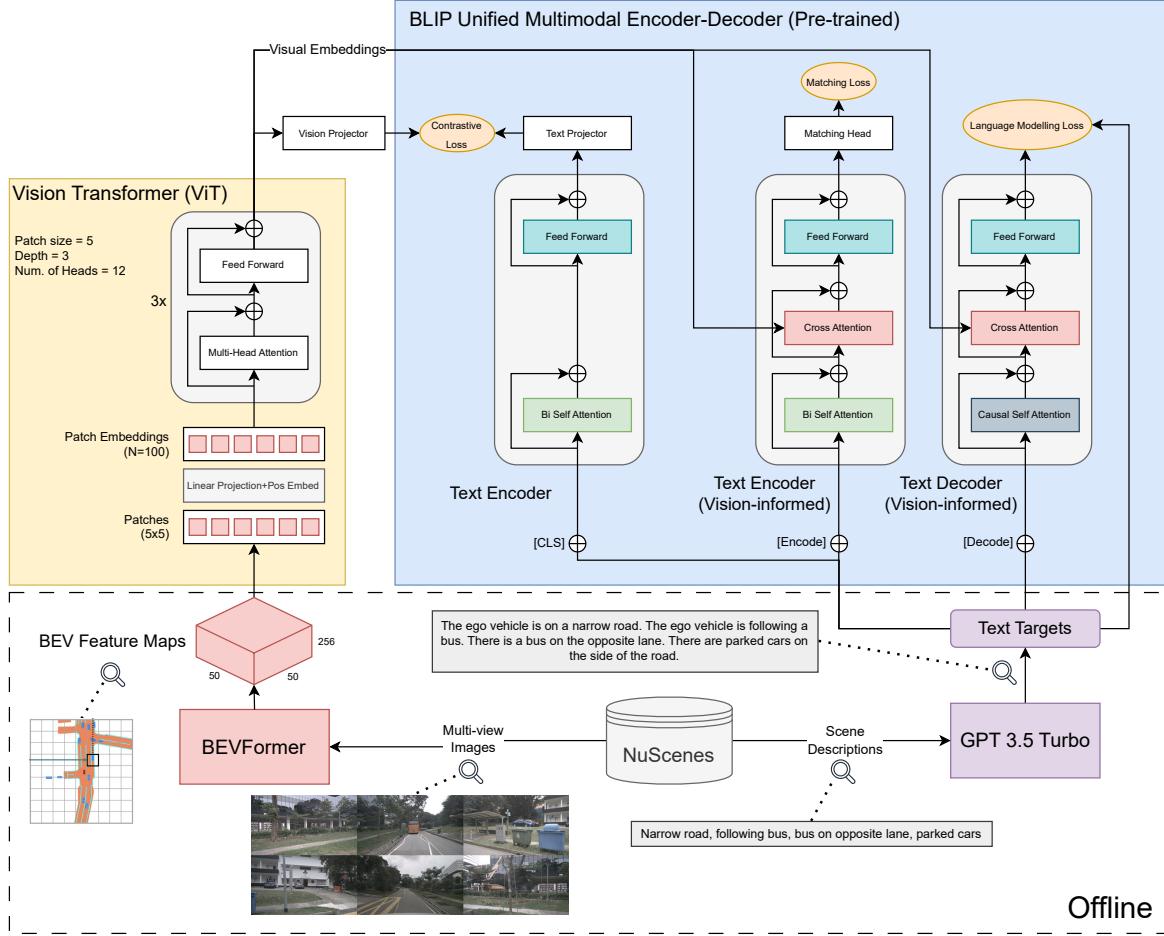


Figure 4.1: The architecture of the pre-training model. The bottom section illustrates offline data generation steps using BEVFormer and GPT-3.5. The upper right section shows the unified multimodal encoder-decoder with pre-trained weights from BLIP. The upper left section depicts the compact vision transformer architecture, trained from scratch with BEV feature maps.

encoder with a 12-head attention mechanism and a feed-forward layer three times, refining the visual embeddings derived from BEV features.

The unified multimodal encoder-decoder transformer module strictly follows the BLIP implementation and is initialized with pre-trained BLIP weights, trained on the image captioning task. Adhering to the original transformer architecture, the module comprises bidirectional and causal self-attention layers, a cross-attention layer, and a feed-forward layer. It operates in three configurations: text encoder, vision-informed text encoder, and vision-informed text decoder.

The text encoder activates bidirectional self-attention, taking whole context into account from both directions. Given the [CLS] token and the text target, it computes text features to be compared with visual features through a contrastive loss after both have undergone corresponding projections. Projectors are consisting of a single fully-connected linear layer without activation. The purpose of the text encoder is to encode the text targets in alignment with the visual embeddings.

The vision-informed text encoder functions similarly to the text encoder, sharing the bidirectional self-attention and feed-forward layer. In addition, it activates a cross-attention layer between the bidirectional self-attention and the feed-forward layer, integrating visual embed-

dings into the computation. The vision-informed text encoder takes an [Encode] token and the text target, aiming to predict whether the text targets and visual targets match, via a single linear layer matching head. The goal of the vision-informed text encoder is to further refine the alignment of visual and text features.

The vision-informed text decoder is similar to its encoder counterpart, with the only difference being the use of causal self-attention instead of bidirectional self-attention, enabling it to process the text in an auto-regressive manner. Given the [Decode] token and text targets, it aims to regenerate text targets, optimized via a language modeling loss.

4.1.3 Losses

During pre-training, three objectives are jointly optimized following the BLIP pre-training framework. Each BEV-text pair requires three forward passes through the transformer modules, to compute the three losses. The final loss is calculated as the sum of these losses, which are delineated below.

Contrastive Loss

Contrastive Loss acts as the objective of the text encoder, aiming to align the feature space of vision transformer and text transformer by encouraging positive visual-text pairs to have similar representations in contrast to negative pairs. For contrastive loss, BLIP follows the loss introduced by ALBEF[Li+21], where a momentum encoder, which are the moving average version of base encoders, is introduced to produce features, and soft labels are created from the momentum encoders as training targets to account for the potential positives in the negative pairs. Visual-text similarity is calculated as:

$$s(V, T) = g_v(\mathbf{v}_{cls})^\top g_w(\mathbf{w}_{cls}) \quad (4.1)$$

where v_{cls} and w_{cls} are embeddings of [CLS] tokens from vision and text respectively while g_v and g_w are linear transformations that map the [CLS] embeddings to normalized lower-dimensional representations. Two queues to are used store the most recent M visual-text representations from the momentum encoders. The normalized features from the momentum encoders are denoted as $g'_v(\mathbf{v}'_{cls})$ and $g'_w(\mathbf{w}'_{cls})$. Similarity in between visual and text embeddings are then defined as:

$$s(V, T) = g_v(\mathbf{v}_{cls})^\top g'_w(\mathbf{w}'_{cls}) \quad (4.2)$$

For each visual and text, softmax-normalized vision-to-text and text-to-vision similarity is then calculated as:

$$p_m^{v2t}(V) = \frac{\exp(s(V, T_m)/\tau)}{\sum_{m=1}^M \exp(s(V, T_m)/\tau)}, \quad p_m^{t2v}(T) = \frac{\exp(s(T, V_m)/\tau)}{\sum_{m=1}^M \exp(s(T, V_m)/\tau)} \quad (4.3)$$

where τ is a learnable temperature parameter. The final vision-text contrastive loss is defined as:

$$\mathcal{L}_c = \frac{1}{2} \mathbb{E}_{(V,T) \sim \mathcal{D}} [\mathcal{H}(\mathbf{y}^{v2t}(V), \mathbf{p}^{v2t}(V)) + \mathcal{H}(\mathbf{y}^{t2v}(T), \mathbf{p}^{t2v}(T))] \quad (4.4)$$

where $y^{v2t}(V)$ and $y^{t2v}(T)$ denote the ground-truth one-hot similarity, where negative pairs have a probability of 0 and the positive pair has a probability of 1, and \mathcal{H} is the cross-entropy between p and y .

Matching Loss

Matching Loss acts as the objective of the visual-grounded text encoder. It is a binary classification task, where the model uses a linear head to predict whether a visual-text pair belong together or not. The aim is to further align representations of visual-text pair to a finer degree. Matching loss is calculated as:

$$\mathcal{L}_m = \mathbb{E}_{(V,T) \sim \mathcal{D}} \mathcal{H}(\mathbf{y}^m, \mathbf{p}^m(V, T)) \quad (4.5)$$

where \mathbf{y}^m is a 2-dimensional one-hot vector representing the ground-truth label and \mathbf{p}^m are predictions.

Language Modelling Loss

Language Modelling Loss acts as the objective of the text decoder, which aims to generate text given visual embeddings from the BEV features. It is essentially a next token prediction task that optimizes a cross entropy loss, which trains the model to maximize the likelihood of the text in an auto-regressively.

4.2 Fine-tuning

The novel VQA task introduced by DriveLM is selected as a downstream objective for a comprehensive traffic scene understanding challenge. Following BLIP’s downstream architecture for VQA tasks, the pre-trained model architecture is modified to generate answers given questions and BEV feature maps. With these modifications, the unified multimodal encoder-decoder is reconfigured to function as a question encoder and an answer decoder. Both the ViT and the unified multimodal encoder-decoder are initialized with weights from the pre-training stage. The VQA model is then trained using question-answer pairs from the DriveLM-nuScenes dataset, with multi-view images replaced by corresponding BEV feature maps.

4.2.1 Model Architecture

For the VQA task, the unified multimodal encoder-decoder is modified to operate in two configurations as illustrated in Figure 4.2: question encoder and answer decoder.

The question encoder is essentially the vision-informed text encoder from the pre-training architecture. It activates bidirectional self-attention, cross-attention, and feed-forward layers. The question encoder takes the [Encode] token followed by the question to generate question embeddings, utilizing both visual and textual embeddings in the cross-attention mechanism. These resulting question embeddings integrate information from both the question and the visual context, refined for visual question answering.

The answer decoder is structurally similar to the text decoder from the pre-training architecture, employing the same set of layers. However, in the cross-attention mechanism, it processes the refined question embeddings generated by the question encoder, rather than visual embeddings from BEV feature maps. Given the [Decode] token and the answers, it aims to regenerate answers, optimized through language modeling loss during training. During inference, it initially receives only the [Decode] token and then generates the answer auto-regressively.

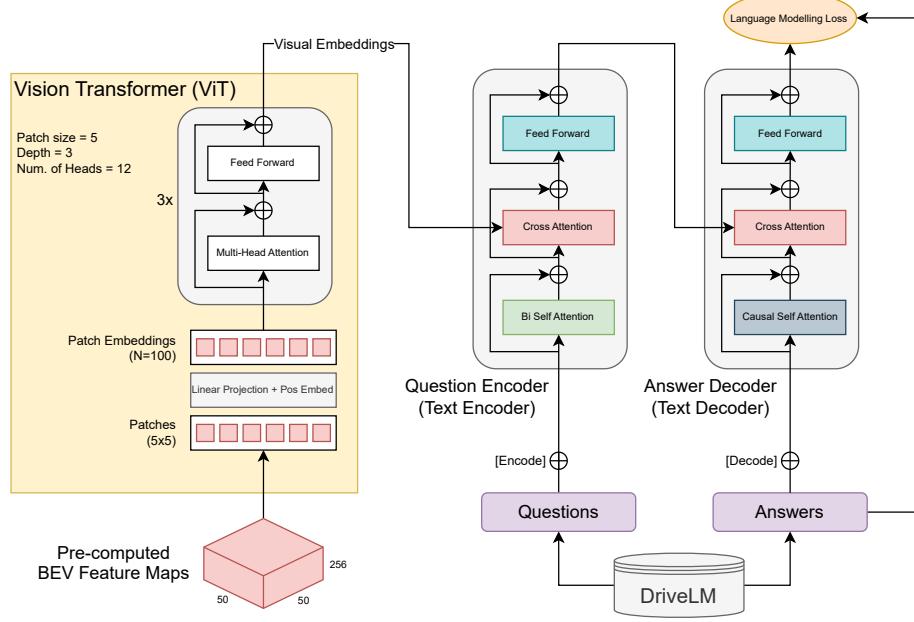


Figure 4.2: The architecture of the VQA model used for the fine-tuning on DriveLM task. Left section shows the vision transformer, initialized with the weights from the pre-training stage. Right section illustrates the reconfiguration of text encoder and text decoder as question encoder and answer decoder respectively.

4.2.2 Visual Feature Extraction Strategies

The modular structure of the BLIP framework allows easy integration of various vision encoders. To evaluate the effectiveness of different sets of vision-grounded features for traffic scene understanding, multiple visual feature extraction strategies, in addition to the vision transformer, are explored as illustrated in Figure 4.3

As an alternative to ViT-derived features, scene features are extracted from the BEV feature map using a simple average pooling operation to represent the global information of the scene. Specifically, the 50x50 features from the BEV grid are reduced to 5x5 using pooling with a kernel size of 10. The resulting features are then flattened and projected to match the text embedding dimension. Notably, scene features are used exclusively in place of the ViT features, never in conjunction.

Beyond the intermediate BEV feature maps, object detections from BEVFormer object detection head are utilized in two ways to enhance object-level perception and understanding capabilities. First, the 50 most confident object detection proposals are selected from BEVFormer. The BEV features at the centers of these 50 object detections are projected to the text embedding dimension, running parallel to the ViT or scene features, essentially acting as skip features. This bypass ensures that object-specific information in the BEV feature map is preserved for cross-attention. Additionally, features of these 50 objects from the BEVFormer object detection head are incorporated. To maintain the positional information of the objects in the BEV map, two 1D positional encodings for the x and y axes, representing the object centers, are added to the object features. As with the other features, these object features are also projected to the text embedding dimension.

Results of these visual feature extraction strategies in various combinations are presented at the Chapter 5:Experiments, and their implications are thoroughly discussed in the Chapter 6:Discussion.

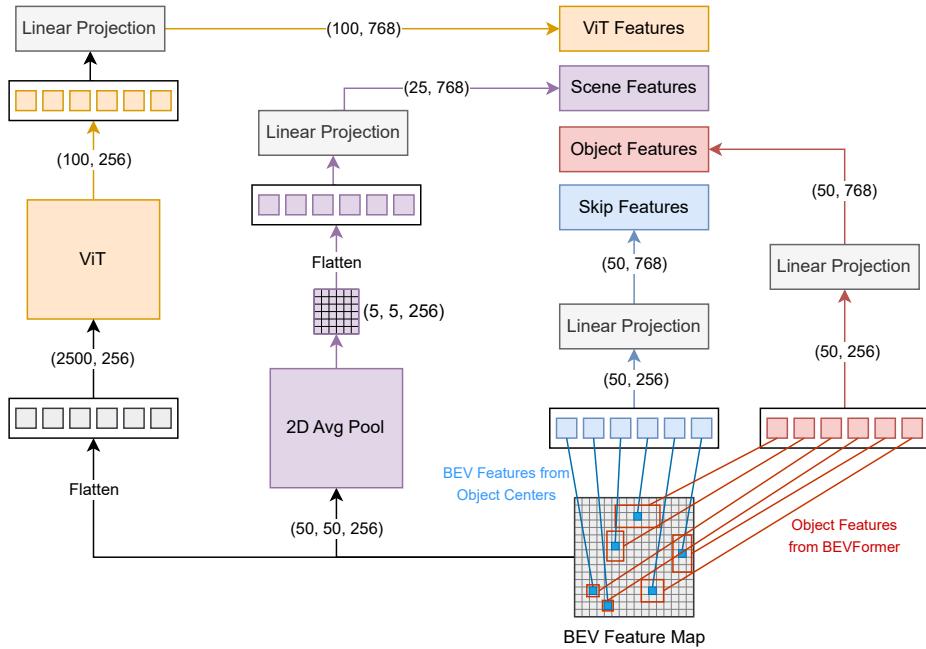


Figure 4.3: A diagram illustrating four different sets of visual features and their implementations. The primary visual features, highlighted in Figure 4.2, are obtained from the BEV feature map using ViT. An alternative set of features, designed to represent global scene information, is derived via a simple average pooling operation. BEV features located at the centers of object detections are directly projected in parallel with ViT, serving as skip connections for object-level understanding. Additionally, object features from the BEVFormer object detection head are utilized for object-level information, following positional encoding and linear projection.

Chapter 5

Experiments

This chapter presents the results of the BEVBlip models, which leverage distinct sets of visual features on the comprehensive VQA task specifically designed for autonomous driving applications, as introduced by DriveLM. Prior to a detailed presentation of the results, the chapter outlines the experimental setup, including implementation, training and inference details. This is followed by an clarification of the different test sets employed in the acquisition of the results, providing a clear context for the evaluation. The chapter then transitions into a detailed description of the evaluation metrics, which are categorized into language scores and the DriveLM metrics, for a clearer interpretation of the results. Subsequently, the chapter presents both quantitative and qualitative results derived from a variety of methods, with tables and figures exemplifying various aspects. The interpretation and discussion of these results are deferred to the following Chapter 6:Discussion.

5.1 Experimental Setup

For the pre-training stage, BEV feature maps are divided according to the BEVFormer's methodology. A total of 28,130 out of 34,149 BEV feature maps, along with their corresponding scene statements by GPT, are utilized as the training dataset, while the remaining 6,019 feature maps are set aside for validation purposes. During this stage, the batch size is fixed at 10 due to VRAM limitations. Following the BLIP pre-training strategy, the AdamW optimizer is employed with an initial learning rate of 5e-6, a minimum learning rate of 1e-6, and a learning rate decay rate of 0.9, accompanied by a weight decay of 0.05. Initially, the learning rate undergoes a warmup period of 3,000 steps, starting from the minimum learning rate. The model is pre-trained over 21 epochs, which takes approximately 6 hours on a GPU.

The DriveLM-nuScenes dataset comprises 377,956 QA pairs in its public training set. The QA pairs are divided by their corresponding scenes into training, validation, and test splits in the ratios of 75%, 5%, and 20% of the total dataset size, respectively. This splitting ensures that the scenes in these subsets are entirely distinct, preventing any information leakage. Fine-tuning is conducted in two sequential training phases. Initially, the model is trained on the main dataset using an initial learning rate of 5e-6, a minimum learning rate of 5e-7, a learning rate decay rate of 0.9, and a weight decay of 0.05 for a maximum of 10 epochs. For each experiment, the best epoch is selected based on the validation split performance. Subsequently, a second training phase is carried out on a smaller dataset derived from the main dataset, focusing on multiple-choice questions created via a script provided by DriveLM. This secondary training aims to emphasize the multiple-choice questions prevalent in the DriveLM private test split. The second phase begins with the weights from the first phase and is conducted with an initial learning rate of 3e-6, a minimum learning rate of 5e-7, a learning

rate decay rate of 0.95, and a weight decay of 0.05 for up to 30 epochs, with the best epoch again selected based on validation performance.

During testing and inference, answers are generated auto-regressively by the vision-informed text decoder using the nucleus (top-p) sampling technique with p set to 0.9. A temperature of 0.8 is applied to achieve more consistent results, and a repetition penalty of 1.1 is enforced. The maximum generation length is set to 300, while the minimum is set to 1 to accommodate single-letter answers for multiple-choice questions. As a final step, light post-processing is applied to the generated text to format c-tags appropriately for correct evaluation, as well as to ensure proper capitalization and punctuation.

This chapter presents results from five distinct experiments, each employing a different set of visual features as detailed in Chapter 4. All experiments are executed under identical data and resource conditions. The experiments, including pre-training, fine-tuning, and inference stages, are conducted using a single NVIDIA RTX 3090 GPU with 24 GB of VRAM. The objective is to investigate the advantages of various approaches. The "Method" column in Table 5.2, Table 5.3, and Table 5.4 specifies the visual feature extraction strategies utilized by each respective method. Quantitative results in Figure 5.2 and Figure 5.3 are derived from the top-performing method as identified in Table 5.2.

5.2 Metrics

This section details the metrics used in the evaluation of question-answer pairs, categorized into two groups: language scores and DriveLM metrics. Language scores encompass various word-level assessment techniques commonly used in natural language processing to evaluate the performance of generated candidate text against reference text. DriveLM metrics encompass a specific set of metrics employed by DriveLM, tailored to the proposed task, assessing the performance of generated answers in terms of semantic similarity to ground truth, object detection and localization, and multiple-choice questions.

5.2.1 Language Scores

Language metrics are critical tools in the evaluation of natural language processing systems, providing quantitative measures to assess the quality of generated text against reference texts. This section explores three widely recognized language scores used for evaluation in this work: BLEU, ROUGE-L, and CIDEr. Each of these metrics provides unique insights into different aspects of text generation, from precision and recall to semantic relevance and informativeness, aimed at various tasks from machine translation and summarization to image captioning.

BLEU

BLEU is one of the earliest and most widely used metrics for evaluating machine translation and other text generation tasks. Proposed by [Pap+02], BLEU score measures the similarity between a candidate text generated by model and one or more reference texts. It is best suited for tasks where matching the reference text closely is crucial, such as machine translation. The core idea behind BLEU is precision, which assesses the proportion of n-grams in the candidate text that appear in the reference text. An n-gram is a contiguous sequence of n items from a given sample of text.

The BLEU score is computed as follows:

$$\text{BLEU}_N = \text{BP} \cdot \exp \left(\sum_{n=1}^N \frac{1}{n} \log p_n \right) \quad (5.1)$$

where N is the length of the largest n-gram to be compared and BP is the brevity penalty to prevent the system from producing overly short translations. It is calculated based on the length of the candidate text relative to the reference text. In this study, BLEU scores with N ranging from 1 to 4 are used.

ROUGE-L

ROUGE-L is a variant of the ROUGE metrics introduced by [Lin04], which is particularly designed to evaluate the recall aspect of text generation systems, making it well-suited for summarization tasks. Unlike BLEU, which focuses on n-gram precision, ROUGE-L measures the longest common sub-sequence (LCS) between the candidate and reference texts. The LCS captures the longest sequence of words that appear in both candidate and reference text in the same order, which helps in evaluating the fluency and informativeness of the generated text.

ROUGE-L score is calculated based on LCS precision (P), which is the length of the LCS divided by the length of the candidate text, and LCS recall (R), the length of the LCS divided by the length of the reference text. The ROUGE-L score is defined as the harmonic mean of precision and recall:

$$\text{ROUGE-L} = \frac{2 \cdot R \cdot P}{R + P} \quad (5.2)$$

ROUGE-L is sensitive to the order of words, making it a good measure for evaluating the syntactic and semantic coherence of generated text.

CIDEr

CIDEr, introduced by [VZP15] is a more recent language metric primarily designed for image captioning tasks. CIDEr measures the consensus between a candidate caption and a set of reference captions by evaluating the importance of n-grams in the context of the reference set. Unlike BLEU and ROUGE, CIDEr incorporates term frequency-inverse document frequency (TF-IDF) weighting to account for the significance of n-grams.

the CIDEr score is calculated as:

$$\text{CIDEr} = \frac{1}{M} \sum_{i=1}^M \frac{\sum_j \text{TF-IDF}(n_{ij}, C) \cdot \text{TF-IDF}(n_{ij}, R_i)}{\left\| \sum_j \text{TF-IDF}(n_{ij}, C) \right\| \left\| \sum_j \text{TF-IDF}(n_{ij}, R_i) \right\|} \quad (5.3)$$

where C and R_i denote the candidate and i -th reference texts, respectively. $\text{TF-IDF}(n_{ij}, C)$ and $\text{TF-IDF}(n_{ij}, R_i)$ are the TF-IDF scores for the n-gram n_{ij} in the candidate and i -th reference texts. M denotes the total number of reference texts, which in our case is equal to 1 since DriveLM-nuScenes provide only a single answer for each question.

5.2.2 DriveLM Metrics

In addition to language scores for evaluating predicted answer text against the ground truth answer text, DriveLM employs three specific scores to assess performance across various aspects of the DriveLM VQA task. For evaluating answers to multiple-choice questions, a straightforward accuracy score is utilized. For assessing object detection and localization performance, a Match score that incorporates L2 distance is proposed, complemented by GPT-based evaluation. Additionally, GPT scoring is used as a standalone metric for evaluating answers to planning questions, where semantic alignment is crucial.

Accuracy Score

Accuracy score is a straightforward metric used in DriveLM to evaluate multiple-choice and yes/no questions. It simply checks whether the predicted answer exactly matches the ground truth answer, as there can only be one correct answer. Over the test set, it computes the fraction of correct answers.

Match Score

Match score is another metric used in DriveLM, for evaluating the object grounding performance of the VLMs. As discussed in section 3.2, the DriveLM-nuScenes dataset encodes key objects within each frame with specific tags in the format $< c, CAM, x, y >$, where c is the identifier, CAM indicates the camera where the object’s center point is located, and x, y represent the coordinates of the 2D bounding box in the respective camera’s coordinate system. Match score compares the center locations of key objects that appear in the candidate text to the objects in the ground truth text by calculating the L2 distance, which is the Euclidean distance between two points. Objects within a certain threshold are considered matched and counted as true positives. Predicted objects that are not matched to the ground truth objects are marked as false positives, while ground truth objects that remain unmatched by the predictions are marked as false negatives. The matching F1 score is then calculated as:

$$R = \frac{TP}{TP + FN}, \quad P = \frac{TP}{TP + FP}, \quad F1 = \frac{2 \cdot R \cdot P}{R + P} \quad (5.4)$$

The final match score is computed as the average of the matching F1 score and the GPT score. The F1 score evaluates the performance of object localization without taking the classification or depiction into account, while the GPT score complements it by evaluating the identification and description of the object as detailed in the candidate text.

GPT Score

DriveLM utilizes GPT’s robust reasoning capabilities for the semantic evaluation of predicted answer text based on the ground truth answer text, thereby complementing traditional language metrics focusing on the word-level assessment of the text. To generate the GPT score, GPT-3.5 is prompted as illustrated in Table 5.1, to assign a numerical score between 0 and 100 to a predicted answer, with higher scores indicating closer alignment to the ground truth. The temperature is set to 0.6 to ensure greater consistency in scoring.

Figure 5.1 illustrates the benefits of GPT-based evaluation over traditional language scores, particularly in scenarios where the predicted answer text diverges syntactically from the ground truth text but remains semantically aligned.



Reference Text	Keep going at the same speed, decelerate gradually without braking.					
Candidate Text	Decelerate gradually without braking and maintain the current speed.					
	GPT	ROUGE-L	BLEU-1	BLEU-2	BLEU-3	BLEU-4
	85	0.417	0.596	0.447	0.371	0.295
Reference Text	There is a brown SUV to the back of the ego vehicle, a black sedan to the back of the ego vehicle, and a green light to the front of the ego vehicle.					
Candidate Text	There is a white truck to the front left of the ego vehicle, a black sedan to the back of the ego vehicle, a white commercial vehicle to the front of the ego vehicle, and a green light to the front of the ego vehicle.					
	GPT	ROUGE-L	BLEU-1	BLEU-2	BLEU-3	BLEU-4
	75	0.791	0.666	0.639	0.611	0.588

Figure 5.1: Example evaluations from GPT and language scores. Top exhibits an example case when the candidate text is syntactically different from reference text but semantically aligned. Bottom showcases another example other way around. While language scores heavily rely on word similarity and order, GPT assessment tends to be more aligned with human evaluation.

```

model="gpt-3.5-turbo",
messages=[
    {
        "role": "system",
        "content": "An evaluator who rates my answer based on the correct
                    answer.",
    },
    {
        "role": "user",
        "content": f"Rate my answer based on the correct answer out of 100,
                    with higher scores indicating that the answer is
                    closer to the correct answer, and you should be
                    accurate to single digits like 62, 78, 41,etc. Output
                    the number only. This is the correct answer: {GT}.
                    This is my answer: {prediction}",
    },
]

```

Table 5.1: Prompt given to GPT-3.5 to evaluate predicted answer based on the correct answer out of 100 accurate to the single digit precision.

The GPT score is incorporated into the Match Score for evaluating prediction questions and is also used as a standalone metric for evaluating planning questions.

Final Score

After each score is calculated and normalized to the same (0, 1) range, a total language score is calculated as:

$$\text{Language} = \frac{1}{3} \cdot \text{ROUGE-L} + \frac{1}{3} \cdot \text{CIDEr} + \frac{1}{3} \cdot \frac{1}{4} \sum_{n=1}^4 \text{BLEU}_n \quad (5.5)$$

Subsequently, a final score is calculated as a weighted average of all scores:

$$\text{Final} = 0.4 \cdot \text{GPT} + 0.2 \cdot \text{Language} + 0.2 \cdot \text{Match} + 0.2 \cdot \text{Accuracy} \quad (5.6)$$

5.3 Results

5.3.1 Quantitative Results

Five distinct experiments are evaluated on the test split, which includes questions on perception, prediction, planning, and behavior. The comparative results are detailed in Table 5.2 and Table 5.3.

The first row presents the outcomes of the initial experiment, wherein the model is implemented as comprehensively described in Chapter 4:Implementation. This model employs the ViT to harness information from the complete BEV feature map and undergoes a pre-training phase as outlined in Section 4.1. During the fine-tuning phase, it incorporates both skip features and object features, as detailed in Section 4.2.2. The second experiment follows the

same model architecture as the first, with the only difference being the omission of the BEV pre-training stage. This modification facilitates an evaluation of the pre-training's efficacy. The subsequent experiments do not utilize the ViT and therefore bypass the pre-training process. The third experiment solely employs the object features derived from BEVFormer object detection. This setup is intended to validate the significance of BEV features by examining the effects of their complete exclusion. The fourth experiment adds the simple pooling-based scene features to the framework of the third experiment. This approach aims to determine whether re-integrating global-level, low-resolution BEV information can improve model performance. The fifth and final experiment replaces the object features from the fourth experiment with the skip features. This experiment is designed to assess the contribution of object features by analyzing the impact of their removal.

Method					DriveLM Metrics			Language Scores						
Pretrain	ViT	Scene	Skip	Obj	Accuracy	(Planning)	(Prediction)	(Perception)					CIDEr	Final
						GPT	Match	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L		
✓	✓	-	✓	✓	0.586	66.732	35.436	0.760	0.679	0.603	0.527	0.688	0.091	0.544
-	✓	-	✓	✓	0.630	67.225	35.547	0.711	0.635	0.563	0.491	0.610	0.089	0.552
-	-	-	-	✓	0.542	63.649	35.804	0.730	0.653	0.579	0.506	0.681	0.093	0.522
-	-	✓	-	✓	0.591	67.651	36.113	0.706	0.632	0.562	0.491	0.678	0.094	0.547
-	-	✓	✓	-	0.610	66.404	36.697	0.707	0.632	0.560	0.488	0.678	0.083	0.546

Table 5.2: Quantitative comparison of results acquired with various experiments utilizing a different set of visual feature extraction strategies on our the test split containing QA pairs from scenes that were not included in the training of the models. The "Method" column specifies the visual feature extraction strategies utilized by each method. Accuracy and language scores range from 0 to 1, while GPT and Match scores range from 0 to 100. Higher scores indicate better performance across all metrics. The highest scores for each metric across all methods are highlighted in bold text.

Method					Accuracy				
Pretrain	ViT	Scene	Skip	Obj	Prediction	Perception	Behavior	Overall	
✓	✓	-	✓	✓	0.982	0.958	0.563	0.586	
-	✓	-	✓	✓	0.977	1.000	0.604	0.630	
-	-	-	-	✓	0.981	0.864	0.522	0.542	
-	-	✓	-	✓	0.980	0.943	0.561	0.591	
-	-	✓	✓	-	0.979	0.938	0.597	0.610	

Table 5.3: A closer look to the Accuracy performance of methods from Table 5.2, detailing the Accuracy score over different categories of questions.

Method					DriveLM Metrics			Language Scores						
Pretrain	ViT	Scene	Skip	Obj	Accuracy	GPT	Match	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	CIDEr	Final
✓	✓	-	✓	✓	0.549	57.104	34.576	0.733	0.654	0.578	0.503	0.690	0.104	0.495

Table 5.4: Quantitative results of the main approach on the private test split of DriveLM-nuScenes.

5.3.2 Qualitative Results

Figure 5.2 and Figure 5.3 showcase qualitative results for the VQA task, with answers generated by the top-performing method identified in Table 5.2. The results are split between two figures to display frames from different traffic scenes and to encompass various question

types included in the dataset for better representation. However, the questions shown are not exhaustive, as many other question types are not covered. The generated answers in both figures are color-coded to enhance the figures' clarity. The colors indicate the correctness of the generated information based on human evaluation: green for correct information, red for incorrect information, and orange for partially correct information. It is important to note that this color-coding does not reflect the actual quantitative evaluation presented in Table 5.2.



Question	GT Answer	Generated Answer
What's your comment on this scene?	There is rainwater on the glass, so the ego vehicle should slow down.	The water droplets on the glass affect the driver's line of sight.
Predict the behavior of the ego vehicle.	The ego vehicle is going straight. The ego vehicle is driving fast.	The ego vehicle is going straight. The ego vehicle is driving very fast.
Which object is most likely to be occluded by <c1,CAM_BACK,1568.3,505.0>? Would this object affect the ego vehicle? Based on this object, what action of the ego vehicle is dangerous?	None, no, none.	None, no, none.
What actions taken by the ego vehicle can lead to a collision with <c1,CAM_BACK,1568.3,505.0>?	No such action will lead to a collision.	Reversing to the right rear can lead to a collision with <c1,CAM_BACK,1301.7,483.3>.
What is the probability of colliding with <c2,CAM_BACK,981.8,499.1> after the ego vehicle goes straight and keeps the same speed?	Low.	Low.
In this scenario, what are safe actions to take for the ego vehicle?	Keep going at the same speed, accelerate, and proceed ahead.	Keep going at the same speed, decelerate gradually without braking.
What object would consider <c3,CAM_BACK,1290.0,472.5> to be most relevant to its decision?	None.	None.

Figure 5.2: Qualitative results on another sample frame from our test split, which includes QA pairs from a scene that is excluded from model training, acquired by the top-performing method identified in Table 5.2. Top section shows the multi-view image of the frame. Colors in the generated answers represent the correctness of information based on human evaluation: green indicates correct information, red indicates incorrect information, and orange indicates partially correct information. This color-coding enhances the figure's clarity and is not related to the model's actual quantitative evaluation.



Question	GT Answer	Generated Answer
What are the important objects in the current scene? Those objects will be considered for the future reasoning and driving decision.	There is a traffic police to the front of the ego vehicle, a black car to the back of the ego vehicle, a white SUV to the front right of the ego vehicle, and a traffic police to the front of the ego vehicle. The IDs of these objects are <c1,CAM_FRONT,1071.7,510.0>, <c2,CAM_BACK,1061.7,650.0>, <c3,CAM_FRONT_RIGHT,582.5,475.8>, and <c4,CAM_FRONT,1176.7,496.7>.	There is a white truck to the front of the ego vehicle, a pedestrian wearing a green shirt to the front of the ego vehicle, and a black suv to the back of the ego vehicle. The IDs of these objects are <c1,CAM_FRONT,1531.7,553.3>, <c2,CAM_FRONT,110.0,594.2>, and <c3,CAM_BACK,1386.4,590.6>.
What actions taken by the ego vehicle can lead to a collision with <c1,CAM_FRONT,1071.7,510.0>?	Accelerating and going straight.	Accelerating and going straight can lead to a collision with <c1,CAM_FRONT,1261.7,556.7>.
What actions could the ego vehicle take based on <c1,CAM_FRONT,1071.7,510.0>? Why take this action and what's the probability?	The action is to decelerate gradually without braking. The reason for taking this action is to avoid a collision, which has a high probability.	The action is to decelerate gradually without braking . The reason for taking this action is to maintain a safe distance , which is high .
Predict the behavior of the ego vehicle. Please select the correct answer from the following options: A. The ego vehicle is slightly steering to the right. The ego vehicle is driving fast. B. The ego vehicle is steering to the right. The ego vehicle is driving very fast. C. The ego vehicle is going straight. The ego vehicle is driving slowly. D. The ego vehicle is going straight. The ego vehicle is driving with normal speed.	C	C
What is the moving status of object <c1,CAM_FRONT,1071.7,510.0>? Please select the correct answer from the following options: A. Going ahead. B. Turn left. C. Back up. D. Reverse parking.	A	A
Is <c1,CAM_FRONT,1071.7,510.0> a traffic sign or a road barrier?	No.	No.
In this scenario, what are safe actions to take for the ego vehicle?	Brake gently to a stop or decelerate gradually without braking.	Decelerate gradually without braking and turn right.

Figure 5.3: Qualitative results on a sample frame from our test split, which includes QA pairs from a scene that is excluded from model training, acquired by the top-performing method identified in Table 5.2. Top section shows the multi-view image of the frame. Colors in the generated answers represent the correctness of information based on human evaluation: green indicates correct information, red indicates incorrect information, and orange indicates partially correct information. This color-coding enhances the figure's clarity and is not related to the model's actual quantitative evaluation.

Chapter 6

Discussion

This chapter discusses the quantitative and qualitative results presented in Chapter 5 altogether, interpreting the findings from in terms of object grounding abilities, quality and correctness of visual descriptions, language generation capabilities, and effectiveness of experienced visual features. Final section interprets the results from the CVPR 2024 Autonomous Driving Challenge, comparing the BEVBlip approach to top-ranking methods. Further research directions to address the limitations are suggested in Chapter 7.

6.1 Object Grounding

The model struggles with object grounding when utilizing c-tags, particularly concerning the precise localization of objects within their respective camera views. As illustrated by the first question in Figure 5.2, the model consistently and correctly identifies the object's position relative to the ego vehicle and determines the appropriate camera identifier where the object's center is located. For instance, when the object's position is described as "to the front" in the answer text, the correct camera identifier "CAM_FRONT" is accurately included in the c-tag. However, the subsequent coordinates following the camera identifier often fail to meet the threshold distance set by the Matching score, leading to mismatches. This suggests that the model is not leveraging visual cues effectively to ground objects in the text but rather relies on frequently occurring numerical patterns in the dataset as a form of guessing, which occasionally results in correct matches. This limitation is further highlighted in the second question in Figure 5.2 and fourth question in Figure 5.3, where the model fails to replicate the c-tag mentioned in the question within the generated answer.

6.2 Visual Description of Objects

Another aspect to evaluate in the prediction questions is the model's capability to linguistically describe visual attributes of the objects. As illustrated in the first question of Figure 5.2, the model frequently confuses cars with SUVs and occasionally misclassifies SUVs as trucks. The accuracy regarding the vehicle's color is generally sufficient, particularly for black and white, which are prevalent in the dataset and relatively easy to distinguish. However, accuracy diminishes for less common colors, with the model tending to predict white for lighter shades and black for darker tones.

The challenges encountered in describing visual attributes, similar to those in image captioning tasks, are evident here due to the variety of ways to describe an object. This is

exemplified in the first question where the traffic police officer in front of the ego vehicle is identified as "a pedestrian wearing a green shirt." While this description is partially accurate, acknowledging the presence of a person with a green top, the model demonstrates successful integration of visual knowledge. This aspect of visual description is incorporated into the Matching score calculation by the GPT, which considers intricate details and prior world knowledge. For example, GPT assigns a higher score to a "blue shirt" prediction for a police officer, as blue is commonly associated with law enforcement attire. Although GPT accounts even for such nuances, it does not have access to the input image and only compares the prediction to the ground truth answer, often resulting in a low score in these scenarios.

6.3 Language Generation

The model's language generation capabilities are primarily assessed using traditional language scores. Table 5.2 presents the language scores for each method, which employs different sets of visual features. Overall, the scores are sufficiently high across all methods, with minimal variation among the metrics. The highest language scores are achieved by the BEV pre-trained model. This can be attributed to the multimodal encoder-decoder's ability to generate sentences that are structurally more relevant to traffic environments, at the expense of some generalization as a result of the pre-training stage.

Language scores primarily evaluate syntactic similarity and do not accurately represent how well the generated answer semantically aligns with the correct answer. For example, in the first question of Figure 5.3, the generated answer, "The water droplets on the glass affect the driver's line of sight," is entirely correct given the images. However, it differs significantly from the only ground truth reference sentence, "There is rainwater on the glass, so the ego vehicle should slow down," in structure. Such discrepancies result in lower language scores due to the lack of matching words and phrases.

6.4 Multiple-choice Questions

The model demonstrates adequate performance in multiple-choice question accuracy. It is important to note that the multiple-choice questions related to prediction and perception are considerably easier compared to those related to behavior. This disparity arises because prediction and perception questions exhibit limited variation in both the questions and the ground truth answers.

Behavior questions, which require the model to predict the future behavior of the ego vehicle, demand a more comprehensive and holistic understanding of the scene. As highlighted in Table 5.3, the model relying solely on object features perform significantly worse in accuracy for both behavior and perception-based multiple-choice questions compared to models that utilize BEV features in various ways.

6.5 Visual Feature Extraction

Overall, considering the final score, the non-pre-trained model configuration utilizing ViT, skip, and object features emerges as the most performant approach. Features reflecting the global characteristics of the environment, such as ViT and scene features, prove to be highly

valuable. This is evident from the difference between the method utilizing only object features and the one utilizing both object and scene features, as shown in Table 5.2. In representing global information, ViT performs better than scene features as anticipated. However, even simple scene features acquired through pooling significantly improve the results, suggesting that there may be other efficient methods to integrate scene-level features apart from ViT.

The higher score of the non-pre-trained model compared to the pre-trained version proposed in this study, especially in terms of accuracy and GPT metrics, can be attributed to the small dataset used in the pre-training stage. This limitation likely diminishes the initial world knowledge present in the already pre-trained BLIP model, leading to poorer generalization.

6.6 CVPR 2024 Autonomous Driving Challenge

Table Table 5.4 presents the quantitative results from the Driving with Language track of CVPR 2024 Autonomous Driving Challenge [CVP24]. The generally lower scores are due to the evaluation being conducted on a private test split, which likely contains more challenging questions and edge cases, making direct comparison with the results from Table 5.2 difficult. As of this writing, the questions remain private, preventing qualitative analysis. According to the challenge leaderboard, the top-ranking methods from the teams NJU-ImagineLab [LL24] and MMTM+ [MMT24] achieve final scores of 0.600 and 0.597, accuracy scores of 0.734 and 0.747, and GPT scores of 65.251 and 65.587, respectively.

The MMTM+ team trains an adapter network between a frozen ViT with 2.5 billion parameters and a frozen LLM, leveraging 16 NVIDIA A100 GPUs. They also incorporate the newer OmniDrive-nuScenes [Wan+24] dataset with 400K QA pairs in addition to DriveLM to enhance their method. Without the OmniDrive-nuScenes dataset, their accuracy drops to 0.59 and the final score to 0.55, which is close to our results. The leading NJU-ImagineLab team trains and fully fine-tunes InternVL, a large VLM they developed, consisting of a 20 billion parameter LLM and a 6 billion parameter ViT, using 64 NVIDIA A100 GPUs.

In contrast, BEVBlip, with approximately 0.3 billion parameters, is trained on a single GPU. Despite the significantly smaller model size and limited computational resources -up to 100 times less than those of the top methods- BEVBlip achieves results comparable to the leading approaches, highlighting the efficiency and effectiveness of BEVBlip.

Chapter 7

Future Work

This chapter outlines the limitations related to the implementation of the proposed BEVBlip method in this thesis and provides recommendations for future work. Several techniques and research directions can potentially improve the performance of the proposed methods. These include, but are not limited to, increasing model size, utilizing parameter-efficient fine-tuning techniques, enhancing data generation processes, and employing improved datasets.

7.1 Increasing Model Size

As detailed in Chapter 4 and further examined in Chapter 6, the proposed BEVBlip approach is significantly constrained by the limited processing power, memory, and storage capacity available for this work. While this study emphasizes exploiting the capabilities of a lightweight approach, the impact of scale is substantial, as evidenced by the competitive results discussed in Section 4.2.2.

The starting point of proposed method is to leverage spatio-temporal BEV feature maps instead of images. However, BEVBlip employs BEVFormer-tiny, the smallest available BEVFormer version, to extract BEV feature maps. In comparison to BEVFormer-base, BEVFormer-tiny is severely limited in several respects: it utilizes a smaller ResNet-50 rather than ResNet-101 as the image backbone, a reduced encoder with three transformer layers instead of six, a smaller input image size reduced down to 800x450 from 1600x900, and single-scale features rather than multi-scale features. Additionally, it incorporates fewer previous frames to construct the historical BEV. These constraints lead to considerably weaker BEV representations, within a BEV feature map with a much smaller grid size of 50x50 and a lower spatial resolution of 2.048 meters, compared to the base model's 200x200 grid with a finer resolution of 0.512 meters. Due to limitations in computational power and memory capacity, this study pre-computes and stores BEV feature maps instead of integrating BEVFormer directly within the pipeline. Despite this, the use of BEVFormer-tiny was necessary to accommodate storage limitations, as larger maps occupy significantly more space.

Future work is undoubtedly going to benefit from higher quality BEV feature maps that possess more expressive features and finer resolutions. The larger BEVFormer-base, the newer BEVFormerV2 [Yan+22], or any other more recent model with improved performance, such as MV2D [Wan+23c] or StreamPETR [Wan+23a], can be utilized as query-based BEV feature extractors. The proposed framework is designed to accommodate the integration of any model as the extractor. With sufficient resources, the BEV extractor could be incorporated directly into the pipeline rather than relying on pre-computation. This would allow for the extractor to be fine-tuned alongside the pre-training and fine-tuning objectives of the proposed approach, further refining the features for the VQA task.

Larger models can also be employed within the BEVBlip approach. Results in Table 5.2 indicate that the ViT outperforms pooling-based scene features, suggesting there is substantial room for improvement in extracting visual embeddings from BEV feature maps. Particularly with larger and more expressive BEV feature maps, the approach is likely to benefit from more sophisticated embedding modules. Future work could utilize a larger ViT, in contrast to the compact ViT with depth 3 and patch size 5x5 used in the current approach.

This work explores the incorporation of a limited set of various visual feature extraction strategies. However, investigating more efficient and effective methods for extracting visual features beyond the ViT and other tested approaches is reserved for future work. Given that one of the most evident shortcomings of the proposed approach lies in its object grounding capabilities, integrating object detection information, such as predicted bounding boxes and classification labels, is likely to significantly enhance performance in future work.

7.2 Parameter-Efficient Fine-Tuning

There are alternative methods to enhance performance without requiring extensive hardware resources. Recently, there has been significant interest in the development of Parameter-Efficient Fine-Tuning (PEFT) techniques. As LLMs and VLMs continue to grow at ever-increasing pace, reaching tens of billions of parameters, the computational and memory demands for full fine-tuning become impractical. On the other hand, smaller models often fail to compete with their larger counterparts, necessitating the use of PEFT techniques.

PEFT offers a promising approach to the limitations encountered in this work. Traditional fine-tuning of large pre-trained models involves updating all parameters, which demands substantial resources often unavailable. PEFT techniques, however, focus on updating only a small subset of parameters, thereby significantly reducing computational and storage requirements while maintaining high performance. These techniques are widely adopted in methods utilizing LLMs and VLMs, including those in the autonomous driving domain.

Several PEFT techniques have gained widespread adoption. One of the earliest and most common is the use of adapter modules, introduced by [Hou+19]. This technique involves inserting small, trainable bottleneck layers within the transformer layers of a pre-trained large model. Only the parameters of these adapter modules are updated during fine-tuning, leaving the rest of the model’s parameters frozen, making the fine-tuning process more efficient, while also reducing the risk of overfitting. Another prevalent technique is Low-Rank Adaptation (LoRA) [Hu+21], which fine-tunes the model by updating a low-rank subset of the weight matrices, effectively capturing changes in the model weights with low-rank updates. Instead of modifying the model parameters, another technique called prompt tuning modifies the input data to the model by prepending or appending learnable tokens, guiding the model to perform specific tasks without altering its core parameters. More recent PEFT techniques include Bias Fine-Tuning (BitFit) [ZRG22], which focuses on updating only the bias terms in the model’s layers, and IA3 [Liu+22], which introduces trainable vectors to perform element-wise re-scaling of internal activations within the transformer model. These methods often perform comparably to full fine-tuning and sometimes even surpass it.

PEFT methods are particularly relevant for our work, as they allow for the adaptation of large models to specific tasks without incurring prohibitive hardware costs. Implementing these PEFT methods in future work could significantly enhance our model’s capabilities and performance without exceeding current resource limitations. Although the employment of

larger models within the framework through PEFT techniques was considered during the development of this work, time constraints prevented the implementation of such experiments. Therefore, the development and application of these techniques are left for future work.

7.3 Improving Data Generation

As illustrated in Table 5.2 and discussed in Chapter 6, the proposed BEV pre-training stage did not effectively improve the model's performance as intended and possibly undermined the initial world knowledge present in the pre-trained BLIP model. This issue can be attributed to the size and quality of the data used in the pre-training stage. For aligning BEV and text features, this work used scene statements generated by GPT-3.5 Turbo from nuScenes scene descriptions. However, this scene description data was very sparse, consisting of only a few words and phrases that did not comprehensively reflect the scene in detail. Furthermore, each scene had only one description, which was used as the text target for all frames within that scene, despite most of the text target content being irrelevant to most frames. This sparsity and low quality of data hindered the potential effectiveness of the pre-training stage.

At the time of this work's development, GPT-3.5 Turbo was the fastest and most cost-effective option available. However, LLMs are rapidly advancing, becoming more capable and affordable. For instance, by the conclusion of this work, the newly released GPT-4o-mini [Ope24] had already rendered GPT-3.5 Turbo obsolete by offering a faster, more capable, and multimodal model at a significantly lower cost, opening up numerous possibilities. Utilizing these newer multimodal models can facilitate the generation of advanced text targets from both ground truth text and scene images, effectively serving as a form of knowledge distillation.

The spatio-temporal reasoning capabilities of these multimodal LLMs can be leveraged to align BEV and text features in the pre-training stage by providing spatial and temporal information through sequences of multi-view images. Furthermore, future work can generate much more complex datasets for use in various training stages. This improvement in data quality and complexity will likely enhance the pre-training stage's effectiveness, addressing the limitations observed in this work.

7.4 Utilizing Improved Datasets

During the implementation phase of this work, autonomous driving datasets aimed at VQA, such as NuScenes-QA, NuScenes-MQA, and DriveLM-nuScenes, were just beginning to emerge. Although DriveLM-nuScenes, the primary dataset used for the fine-tuning stage, offers a comprehensive VQA task, it nevertheless suffers from a lack of variation in its answers and questions, particularly regarding multiple-choice questions. This lack of variation potentially leads to poor generalization of the trained models and sub-optimal evaluation accuracy. As this thesis was being developed and written, these datasets were releasing more comprehensive and improved versions, and new nuScenes-based VQA datasets focused on traffic scene understanding, such as OmniDrive-nuScenes, were emerging. Integrating these datasets intelligently is likely to facilitate a more holistic understanding of the traffic environment, encompassing various aspects of perception, prediction, and planning. For instance, the advantageous effects of combining DriveLM-nuScenes and OmniDrive-nuScenes have already been demonstrated by the MMTM+ team, as explained in Section 6.6.

Extending beyond nuScenes, recent out-of-domain datasets such as MAPLM, may also offer higher quality data in greater volumes, which is crucial for any deep learning approach. Integrating and fusing all these datasets, particularly with the help of multimodal large models like GPT, is likely to unlock enormous potential for future research. Utilizing these comprehensive datasets will not only enhance the robustness of the models but also improve their ability to generalize across diverse traffic scenarios. This integration promises to provide a more nuanced and detailed understanding of the traffic environment, contributing to more effective, reliable and interpretable autonomous driving systems.

Chapter 8

Conclusion

This research has investigated the integration of VLMs to enhance traffic environment understanding within autonomous driving systems. By employing BEV feature maps and leveraging the efficient BLIP framework, the study explores the potential of multimodal approaches to improve the generalization, reasoning and interpretability capabilities of autonomous driving systems. The proposed BEVBlip model shows promising performance on a comprehensive VQA task that is tailored for traffic scenarios, underscoring the potential of combining visual and linguistic information to advance the accuracy and transparency of autonomous driving systems regarding traffic environment understanding.

The motivation for this research arises from the significant challenges faced by current autonomous driving systems, particularly their limited ability to generalize to unfamiliar scenarios and the opaque nature of their decision-making processes. Traditional end-to-end approaches often function as black boxes, obscuring the rationale behind driving decisions and complicating failure diagnosis. In contrast, VLMs present a promising solution by integrating extensive linguistic and visual knowledge, enabling a more comprehensive understanding of traffic scenes and human-like reasoning capabilities. This study explores the rather young and unexplored territory of incorporating visual and language features effectively in the autonomous driving domain, particularly for scene understanding and reasoning.

Adapting the BLIP framework, this study integrates BEV feature maps with multimodal encoder-decoders. The pre-training stage involved aligning BEV feature maps with image features, while the fine-tuning stage optimized the model for the VQA task proposed by the DriveLM. Various strategies for visual feature extraction were explored, resulting in a model that integrates visual and textual data to answer traffic-related questions on prediction, perception, planning and ego vehicle behavior. Performance metrics indicate that the proposed method has the potential to effectively integrate visual features in understanding and responding to complex traffic scenarios, without using extensive resources.

However, the research identified several limitations, particularly in object grounding and visual attribute description. The model occasionally struggled with precise localization of objects and misclassification of vehicle types and colors. These challenges highlight the need for further refinement of the visual feature extraction and alignment processes. Future work should focus on increasing the model size, utilizing more advanced BEV feature extractors, and exploring parameter-efficient fine-tuning techniques to enhance the model's capabilities without incurring prohibitive computational costs. Additionally, leveraging newer multimodal large language models can provide richer and more detailed textual descriptions, facilitating better alignment between visual and textual features.

The findings of this research demonstrate that integration of VLMs holds great promise for traffic scene understanding and suggest pathways for developing more advanced and reliable

autonomous driving technologies capable of navigating complex and dynamic traffic environments with greater accuracy and transparency. Future work will build on these foundations to develop even more effective ways to utilize VLMs in this domain.

List of Figures

1.1	A set of examples from NuScenes-MQA [Ino+24] dataset, showing question-answer pairs under corresponding multi-view camera images from various traffic scenes. Questions include object detection, localization and counting. Key information such as cameras, objects, numbers or locations are enclosed within markups.	2
1.2	High-level outline of the proposed approach, utilizing BLIP [Li+22a] as a vision-language framework to address the DriveLM [Sim+23] VQA task for traffic scene understanding. The proposed approach replaces images with feature maps in the Bird’s Eye View (BEV) perspective, acquired via BEVFormer [Li+22c] from multi-view camera images surrounding the vehicle. The bottom part illustrates the pre-training step for feature alignment, which includes offline BEV feature map acquisition and text target generation using GPT-3.5 [Bro+20] from nuScenes scene descriptions. The top part depicts the fine-tuning step for VQA using the DriveLM dataset and various visual feature extraction strategies. For more detailed architecture illustrations, refer to Figure 4.1, Figure 4.2, and Figure 4.3 in Chapter 4	4
2.1	Implementation of residual block with a residual connection in ResNet [He+15].	6
2.2	A high level overview of object detection pipeline of R-CNN [Gir+14].	6
2.3	The architecture of YOLOv1 [Red+16].	7
2.4	The original encoder-decoder transformer architecture from [Vas+17]. On the left is the encoder with multi-head attention, processing input embeddings. On the right is the decoder, with a masked multi-head attention, processing shifted output embeddings. In the decoder’s multi-head attention, the key and value matrices are derived from the encoder, while the queries originate from within the decoder.	8
2.5	The architecture of ViT [Dos+20]. Left side illustrates the image classification pipeline. Right side details the architecture of the transformer encoder.	10
2.6	The object detection pipeline of DETR [Car+20]. Set of image features extracted via a CNN backbone are embedded by a transformer encoder. A transformer decoder, using object queries, then attends to the encoder output to produce output embeddings, which are subsequently used by prediction heads to perform object detection.	10
2.7	The architecture of BEVFormer [Li+22c]. The left section illustrates 2D multi-camera feature extraction using a convolutional backbone. The middle section details the transformer layers, which utilize BEV from the previous timestamp and BEV queries in 3D space, attending to multi-camera features through spatial cross-attention to generate the BEV feature representation. The right section demonstrates the operation of spatial cross-attention and temporal self-attention mechanisms.	13

2.8 Pre-training model architecture and objectives of BLIP [Li+22a]. The left-most transformer block illustrates a ViT as the image encoder. The remaining three transformer blocks represent the three configurations of the unified multimodal encoder-decoder, with same-colored layers indicating shared weights. The text encoder employs bidirectional self-attention layers. The image-grounded text encoder activates additional cross-attention layers to attend to image features from the image encoder. The image-grounded text decoder replaces the bidirectional self-attention layers of the encoder with causal self-attention layers.	16
3.1 An overall Composition of DriveLM-nuScenes [Sim+23], illustrating example perception, prediction and planning questions from the dataset. Key objects are encoded using c tags that contain camera affiliations, and the center coordinates of their 2D bounding boxes within the corresponding camera frame.	21
4.1 The architecture of the pre-training model. The bottom section illustrates offline data generation steps using BEVFormer and GPT-3.5. The upper right section shows the unified multimodal encoder-decoder with pre-trained weights from BLIP. The upper left section depicts the compact vision transformer architecture, trained from scratch with BEV feature maps.	26
4.2 The architecture of the VQA model used for the fine-tuning on DriveLM task. Left section shows the vision transformer, initialized with the weights from the pre-training stage. Right section illustrates the reconfiguration of text encoder and text decoder as question encoder and answer decoder respectively.	29
4.3 A diagram illustrating four different sets of visual features and their implementations. The primary visual features, highlighted in Figure 4.2, are obtained from the BEV feature map using ViT. An alternative set of features, designed to represent global scene information, is derived via a simple average pooling operation. BEV features located at the centers of object detections are directly projected in parallel with ViT, serving as skip connections for object-level understanding. Additionally, object features from the BEVFormer object detection head are utilized for object-level information, following positional encoding and linear projection.	30
5.1 Example evaluations from GPT and language scores. Top exhibits an example case when the candidate text is syntactically different from reference text but semantically aligned. Bottom showcases another example other way around. While language scores heavily rely on word similarity and order, GPT assessment tends to be more aligned with human evaluation.	35
5.2 Qualitative results on another sample frame from our test split, which includes QA pairs from a scene that is excluded from model training, acquired by the top-performing method identified in Table 5.2. Top section shows the multi-view image of the frame. Colors in the generated answers represent the correctness of information based on human evaluation: green indicates correct information, red indicates incorrect information, and orange indicates partially correct information. This color-coding enhances the figure’s clarity and is not related to the model’s actual quantitative evaluation.	38

List of Tables

3.1	A non-exhaustive overview of recent works integrating VLMs in autonomous driving.	19
4.1	Prompt given to GPT-3.5 to generate statements used as text targets. Writing from the perspective of the ego vehicle and explicitly referring to it as 'the ego vehicle' helps stitching the given information together. Using simple present tense regularizes the verbs. The model is also given the instruction not to add new information to prevent hallucination. Alongside the instruction, a single example is also provided as a chat history to enhance responses.	24
4.2	Few example statements generated by GPT-3.5 on nuScenes data. GPT expands irregular abbreviations, fixes typos and forms full-sentences reordering and logically connecting words from given scene description.	25
5.1	Prompt given to GPT-3.5 to evaluate predicted answer based on the correct answer out of 100 accurate to the single digit precision.	36
5.2	Quantitative comparison of results acquired with various experiments utilizing a different set of visual feature extraction strategies on our the test split containing QA pairs from scenes that were not included in the training of the models. The "Method" column specifies the visual feature extraction strategies utilized by each method. Accuracy and language scores range from 0 to 1, while GPT and Match scores range from 0 to 100. Higher scores indicate better performance across all metrics. The highest scores for each metric across all methods are highlighted in bold text.	37
5.3	A closer look to the Accuracy performance of methods from Table 5.2, detailing the Accuracy score over different categories of questions.	37
5.4	Quantitative results of the main approach on the private test split of DriveLM-nuScenes.	37

Bibliography

- [Bro+20] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165.
- [Cae+20] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Lioing, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. “nuScenes: A multimodal dataset for autonomous driving”. In: *CVPR*. 2020.
- [Cao+24] Cao, X., Zhou, T., Ma, Y., Ye, W., Cui, C., Tang, K., Cao, Z., Liang, K., Wang, Z., Rehg, J. M., and Zheng, C. “MAPLM: A Real-World Large-Scale Vision-Language Benchmark for Map and Traffic Scene Understanding”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2024, pp. 21819–21830.
- [Car+20] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. *End-to-End Object Detection with Transformers*. 2020. arXiv: 2005.12872.
- [Che+19] Chen, Y.-C., Li, L., Yu, L., Kholy, A. E., Ahmed, F., Gan, Z., Cheng, Y., and Liu, J. *UNITER: UNiversal Image-Text Representation Learning*. 2019. arXiv: 1909.11740 [cs.CV]. URL: <https://arxiv.org/abs/1909.11740>.
- [Cho+23] Choudhary, T., Dewangan, V., Chandhok, S., Priyadarshan, S., Jain, A., Singh, A. K., Srivastava, S., Jatavallabhula, K. M., and Krishna, K. M. *Talk2BEV: Language-enhanced Bird’s-eye View Maps for Autonomous Driving*. 2023. arXiv: 2310.02251.
- [CVP24] CVPR. *CVPR 2024 Autonomous Grand Challenge*. 2024. URL: https://opendrivelab.com/challenge2024/#driving_with_language.
- [DT05] Dalal, N. and Triggs, B. “Histograms of oriented gradients for human detection”. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*. Vol. 1. 2005, 886–893 vol. 1. doi: 10.1109/CVPR.2005.177.
- [Dev+19] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805.
- [Dos+20] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. arXiv: 2010.11929 [cs.CV].

- [Fu+23] Fu, D., Li, X., Wen, L., Dou, M., Cai, P., Shi, B., and Qiao, Y. *Drive Like a Human: Rethinking Autonomous Driving with Large Language Models*. 2023. arXiv: 2307.07162.
- [Gem23] Gemini. *Gemini: A Family of Highly Capable Multimodal Models*. 2023. arXiv: 2312.11805.
- [Gir15] Girshick, R. *Fast R-CNN*. 2015. arXiv: 1504.08083.
- [Gir+14] Girshick, R., Donahue, J., Darrell, T., and Malik, J. *Rich feature hierarchies for accurate object detection and semantic segmentation*. 2014. arXiv: 1311.2524.
- [GGT24] Gopalkrishnan, A., Greer, R., and Trivedi, M. *Multi-Frame, Lightweight and Efficient Vision-Language Models for Question Answering in Autonomous Driving*. 2024. arXiv: 2403.19838.
- [GV22] Gosala, N. and Valada, A. “Bird’s-Eye-View Panoptic Segmentation Using Monocular Frontal View Images”. In: *IEEE Robotics and Automation Letters* 7.2 (Apr. 2022), pp. 1968–1975. ISSN: 2377-3774. DOI: 10.1109/lra.2022.3142418. URL: <http://dx.doi.org/10.1109/LRA.2022.3142418>.
- [He+15] He, K., Zhang, X., Ren, S., and Sun, J. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385.
- [Hou+19] Houlsby, N., Giurgiu, A., Jastrzebski, S., Morrone, B., Laroussilhe, Q. de, Gesmundo, A., Attariyan, M., and Gelly, S. *Parameter-Efficient Transfer Learning for NLP*. 2019. arXiv: 1902.00751 [cs.LG]. URL: <https://arxiv.org/abs/1902.00751>.
- [Hu+21] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. *LoRA: Low-Rank Adaptation of Large Language Models*. 2021. arXiv: 2106.09685.
- [Hu+23] Hu, H., Wang, F., Su, J., Wang, Y., Hu, L., Fang, W., Xu, J., and Zhang, Z. *EALSS: Edge-aware Lift-splat-shot Framework for 3D BEV Object Detection*. 2023. arXiv: 2303.17895.
- [Hua+22] Huang, J., Huang, G., Zhu, Z., Ye, Y., and Du, D. *BEVDet: High-performance Multi-camera 3D Object Detection in Bird-Eye-View*. 2022. arXiv: 2112.11790.
- [Ino+24] Inoue, Y., Yada, Y., Tanahashi, K., and Yamaguchi, Y. “NuScenes-MQA: Integrated Evaluation of Captions and QA for Autonomous Driving Datasets Using Markup Annotations”. In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*. Jan. 2024, pp. 930–938.
- [Jai+22] Jain, K., Chhangani, V., Tiwari, A., Krishna, K. M., and Gandhi, V. *Ground then Navigate: Language-guided Navigation in Dynamic Scenes*. 2022. arXiv: 2209.11972.
- [Jia+21] Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q. V., Sung, Y., Li, Z., and Duerig, T. *Scaling Up Visual and Vision-Language Representation Learning With Noisy Text Supervision*. 2021. arXiv: 2102.05918 [cs.CV]. URL: <https://arxiv.org/abs/2102.05918>.
- [Jin+23] Jin, B., Liu, X., Zheng, Y., Li, P., Zhao, H., Zhang, T., Zheng, Y., Zhou, G., and Liu, J. *ADAPT: Action-aware Driving Caption Transformer*. 2023. arXiv: 2302.00673.

- [Kap+20] Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. *Scaling Laws for Neural Language Models*. 2020. arXiv: 2001.08361 [cs.LG]. URL: <https://arxiv.org/abs/2001.08361>.
- [Key+23] Keysan, A., Look, A., Kosman, E., Gürsun, G., Wagner, J., Yao, Y., and Rakitsch, B. *Can you text what is happening? Integrating pre-trained language encoders into trajectory prediction models for autonomous driving*. 2023. arXiv: 2309.05282.
- [Kim+21] Kim, S. W., Philion, J., Torralba, A., and Fidler, S. *DriveGAN: Towards a Controllable High-Quality Neural Simulation*. 2021. arXiv: 2104.15060.
- [KSH12] Krizhevsky, A., Sutskever, I., and Hinton, G. E. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by Pereira, F., Burges, C., Bottou, L., and Weinberger, K. Vol. 25. Curran Associates, Inc., 2012. URL: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.
- [Lan+19] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. 2019. arXiv: 1909.11942 [cs.CL]. URL: <https://arxiv.org/abs/1909.11942>.
- [LL24] Li, J. and Lu, T. *Driving with InternVL*. 2024.
- [Li+23] Li, J., Li, D., Savarese, S., and Hoi, S. *BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models*. 2023. arXiv: 2301.12597 [cs.CV]. URL: <https://arxiv.org/abs/2301.12597>.
- [Li+22a] Li, J., Li, D., Xiong, C., and Hoi, S. *BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation*. 2022. arXiv: 2201.12086 [cs.CV].
- [Li+21] Li, J., Selvaraju, R. R., Gotmare, A. D., Joty, S., Xiong, C., and Hoi, S. *Align before Fuse: Vision and Language Representation Learning with Momentum Distillation*. 2021. arXiv: 2107.07651 [cs.CV]. URL: <https://arxiv.org/abs/2107.07651>.
- [Li+22b] Li, Y., Ge, Z., Yu, G., Yang, J., Wang, Z., Shi, Y., Sun, J., and Li, Z. *BEVDepth: Acquisition of Reliable Depth for Multi-view 3D Object Detection*. 2022. arXiv: 2206.10092.
- [Li+22c] Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Yu, Q., and Dai, J. *BEVFormer: Learning Bird’s-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers*. 2022. arXiv: 2203.17270 [cs.CV].
- [Lin04] Lin, C.-Y. “ROUGE: A Package for Automatic Evaluation of Summaries”. In: *Annual Meeting of the Association for Computational Linguistics*. 2004. URL: <https://api.semanticscholar.org/CorpusID:964287>.
- [Liu+22] Liu, H., Tam, D., Muqeeth, M., Mohta, J., Huang, T., Bansal, M., and Raffel, C. *Few-Shot Parameter-Efficient Fine-Tuning is Better and Cheaper than In-Context Learning*. 2022. arXiv: 2205.05638 [cs.LG]. URL: <https://arxiv.org/abs/2205.05638>.
- [Liu+19] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. *RoBERTa: A Robustly Optimized BERT Pre-training Approach*. 2019. arXiv: 1907.11692 [cs.CL]. URL: <https://arxiv.org/abs/1907.11692>.

- [Lu+19] Lu, J., Batra, D., Parikh, D., and Lee, S. *ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks*. 2019. arXiv: 1908.02265 [cs.CV]. URL: <https://arxiv.org/abs/1908.02265>.
- [Ma+23] Ma, Y., Wang, T., Bai, X., Yang, H., Hou, Y., Wang, Y., Qiao, Y., Yang, R., Manocha, D., and Zhu, X. *Vision-Centric BEV Perception: A Survey*. 2023. arXiv: 2208.02797 [cs.CV]. URL: <https://arxiv.org/abs/2208.02797>.
- [Mic20] Microsoft. *Turing-NLG: A 17-billion-parameter language model by Microsoft*. 2020. URL: <https://www.microsoft.com/en-us/research/blog/turing-nlg-a-17-billion-parameter-language-model-by-microsoft/>.
- [MMT24] MMTM+. *Enhancing Vision Language Models for Autonomous Driving with Multi-view Multi-dataset Fusion*. 2024.
- [Nie+23] Nie, M., Peng, R., Wang, C., Cai, X., Han, J., Xu, H., and Zhang, L. *Reason2Drive: Towards Interpretable and Chain-based Reasoning for Autonomous Driving*. 2023. arXiv: 2312.03661.
- [Ope24] OpenAI. *GPT-4o mini: advancing cost-efficient intelligence*. 2024. URL: <https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>.
- [Ope+24] OpenAI, Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., and Altman, S. *GPT-4 Technical Report*. 2024. arXiv: 2303.08774 [cs.CL]. URL: <https://arxiv.org/abs/2303.08774>.
- [Pan+20] Pan, B., Sun, J., Leung, H. Y. T., Andonian, A., and Zhou, B. “Cross-View Semantic Segmentation for Sensing Surroundings”. In: *IEEE Robotics and Automation Letters* 5.3 (July 2020), pp. 4867–4873. ISSN: 2377-3774. DOI: 10.1109/lra.2020.3004325. URL: <http://dx.doi.org/10.1109/LRA.2020.3004325>.
- [Pap+02] Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. “BLEU: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL ’02. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: <https://doi.org/10.3115/1073083.1073135>.
- [PF20] Philion, J. and Fidler, S. *Lift, Splat, Shoot: Encoding Images From Arbitrary Camera Rigs by Implicitly Unprojecting to 3D*. 2020. arXiv: 2008.05711.
- [Qia+23] Qian, T., Chen, J., Zhuo, L., Jiao, Y., and Jiang, Y.-G. *NuScenes-QA: A Multi-modal Visual Question Answering Benchmark for Autonomous Driving Scenario*. 2023. arXiv: 2305.14836 [cs.CV].
- [Rad+21] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020.
- [Rad+18] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. “Improving language understanding by generative pre-training”. In: (2018).
- [Rad+19] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. “Language Models are Unsupervised Multitask Learners”. In: 2019. URL: <https://api.semanticscholar.org/CorpusID:160025533>.
- [Ram+21] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. *Zero-Shot Text-to-Image Generation*. 2021. arXiv: 2102.12092.

- [Red+16] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640.
- [Ren+16] Ren, S., He, K., Girshick, R., and Sun, J. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2016. arXiv: 1506.01497.
- [Sho+19] Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. *Megatron-LM: Training Multi-Billion Parameter Language Models Using Model Parallelism*. 2019. arXiv: 1909.08053 [cs.CL]. URL: <https://arxiv.org/abs/1909.08053>.
- [Sim+23] Sima, C., Renz, K., Chitta, K., Chen, L., Zhang, H., Xie, C., Luo, P., Geiger, A., and Li, H. *DriveLM: Driving with Graph Visual Question Answering*. 2023. arXiv: 2312.14150 [cs.CV].
- [SZ14] Simonyan, K. and Zisserman, A. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. arXiv: 1409.1556.
- [Sze+14] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. *Going Deeper with Convolutions*. 2014. arXiv: 1409.4842.
- [Tou+23] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. *LLaMA: Open and Efficient Foundation Language Models*. 2023. arXiv: 2302.13971.
- [Vas+17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. “Attention is all you need”. In: *Advances in neural information processing systems* 30 (2017).
- [VZP15] Vedantam, R., Zitnick, C. L., and Parikh, D. *CIDEr: Consensus-based Image Description Evaluation*. 2015. arXiv: 1411.5726 [cs.CV]. URL: <https://arxiv.org/abs/1411.5726>.
- [VJ01] Viola, P. and Jones, M. “Rapid object detection using a boosted cascade of simple features”. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. Vol. 1. 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517.
- [Wan+23a] Wang, S., Liu, Y., Wang, T., Li, Y., and Zhang, X. *Exploring Object-Centric Temporal Modeling for Efficient Multi-View 3D Object Detection*. 2023. arXiv: 2303.11926 [cs.CV]. URL: <https://arxiv.org/abs/2303.11926>.
- [Wan+24] Wang, S., Yu, Z., Jiang, X., Lan, S., Shi, M., Chang, N., Kautz, J., Li, Y., and Alvarez, J. M. *OmniDrive: A Holistic LLM-Agent Framework for Autonomous Driving with 3D Perception, Reasoning and Planning*. 2024. arXiv: 2405.01533 [cs.CV]. URL: <https://arxiv.org/abs/2405.01533>.
- [Wan+23b] Wang, X., Zhu, Z., Huang, G., Chen, X., Zhu, J., and Lu, J. “DriveDreamer: Towards Real-world-driven World Models for Autonomous Driving”. In: *arXiv preprint arXiv:2309.09777* (2023).
- [Wan+21] Wang, Y., Guizilini, V., Zhang, T., Wang, Y., Zhao, H., and Solomon, J. *DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries*. 2021. arXiv: 2110.06922.
- [Wan+23c] Wang, Z., Huang, Z., Fu, J., Wang, N., and Liu, S. *Object as Query: Lifting any 2D Object Detector to 3D Detection*. 2023. arXiv: 2301.02364 [cs.CV]. URL: <https://arxiv.org/abs/2301.02364>.

- [Wu+23] Wu, D., Han, W., Wang, T., Liu, Y., Zhang, X., and Shen, J. *Language Prompt for Autonomous Driving*. 2023. arXiv: 2309.04379 [cs.CV]. URL: <https://arxiv.org/abs/2309.04379>.
- [Xu+24] Xu, Z., Zhang, Y., Xie, E., Zhao, Z., Guo, Y., Wong, K.-Y. K., Li, Z., and Zhao, H. *DriveGPT4: Interpretable End-to-end Autonomous Driving via Large Language Model*. 2024. arXiv: 2310.01412.
- [Yan+22] Yang, C., Chen, Y., Tian, H., Tao, C., Zhu, X., Zhang, Z., Huang, G., Li, H., Qiao, Y., Lu, L., Zhou, J., and Dai, J. *BEVFormer v2: Adapting Modern Image Backbones to Bird’s-Eye-View Recognition via Perspective Supervision*. 2022. arXiv: 2211.10439 [cs.CV]. URL: <https://arxiv.org/abs/2211.10439>.
- [Yan+21] Yang, W., Li, Q., Liu, W., Yu, Y., Ma, Y., He, S., and Pan, J. “Projecting Your View Attentively: Monocular Road Scene Layout Estimation via Cross-view Transformation”. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 15531–15540. DOI: 10.1109/CVPR46437.2021.01528.
- [Yu+19] Yu, Z., Yu, J., Cui, Y., Tao, D., and Tian, Q. “Deep Modular Co-Attention Networks for Visual Question Answering”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019.
- [ZRG22] Zaken, E. B., Ravfogel, S., and Goldberg, Y. *BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models*. 2022. arXiv: 2106.10199 [cs.LG]. URL: <https://arxiv.org/abs/2106.10199>.
- [Zho+23] Zhou, X., Liu, M., Zagar, B. L., Yurtsever, E., and Knoll, A. C. *Vision Language Models in Autonomous Driving and Intelligent Transportation Systems*. 2023. arXiv: 2310.14414 [cs.CV].