MD4AD: BASELINE ANALYSIS

Patrick Chen* **Daniel Yang*** **Junhong Zhou*** **Tianzhi Li***
{bochunc, danielya, junhong2, tianzhil}@andrew.cmu.edu

1 BASELINE MODELS AND METRICS (2 PAGES)

[1 points] for general clarity of exposition.

1.0.1 UNIMODAL BASELINES: Q-ONLY (LANGUAGE MODEL WITHOUT VISION)

- This model only takes the question as input and does not use any images or object detections.
- The language model (LM) tries to answer based on pre-trained knowledge from past data.
- Since it does not receive any visual input, it cannot understand scene-specific details (e.g., *what number of parked trucks are there?*).

Interaction Type: No interaction between text and vision because the model processes only language.

What can the LM infer from the prompt?

- If the prompt contains general knowledge questions (e.g., traffic rules, common driving behaviors), the LM can infer an answer.
- If the prompt contains spatial or object-specific reasoning, the LM will fail.

1.0.2 SIMPLE MULTIMODAL MODELS: CENTERPOINT + LM (USING DETECTED OBJECTS)

- The CenterPoint model detects objects and extracts features such as their position, size, and velocity.
- These detected objects are then given to the language model (LM) along with the text latent.
- The LM can now answer simple questions like *"How many moving pedestrians are there?"* because it receives visual object data.
- However, the model does not process the raw image pixels directly, so it struggles with spatial relationships (e.g., *"What status is the truck to the front left of me?"*).

Interaction Type: The language model interacts with visual object information but does not have access to raw image details.

What can a simple detector figure out about the task?

- Can roughly recognize objects, bounding boxes, and categories (e.g., cars, pedestrians).
- Cannot understand relationships (e.g., "*Is the pedestrian crossing the road?*").
- Cannot infer itentions or context (e.g., "*Does the construction vehicle have the same status as the car?*").

What can the LM infer from the prompt?

- Can **use object detections** to answer simple questions (e.g., "*How many moving pedestrians are there?*").
- Cannot infer **spatial relationships** without explicit modeling.

[*] Everyone Contributed Equally – Alphabetical order

1.0.3 Alternate choices for encoderes/decoders: MSMDFusion + LM (Using LiDAR-Camera Fusion)

- This model fuses LiDAR and camera data to provide a more accurate 3D representation of the scene.
- The LM receives information about object depth, occlusions, and multi-view perspectives.
- This allows it to answer more complex spatial questions, such as *"What number of pedestrians are to the back right of me?"*.
- However, the LM is still limited in understanding scene dynamics without explicit reasoning.

Interaction Type: The LM now has access to richer visual information (from both LiDAR and cameras), making object recognition more reliable.

What can a simple detector figure out about the task?

- Can infer object depth, size, and occlusions in details.
- Improves detection in low-light or adverse weather conditions.
- Still lacks high-level reasoning (e.g., predicting a pedestrian's intent).

What can the LM infer from the prompt?

- Can use richer object features (e.g., "*Is there a pedestrian occluded behind a truck?*").
- Can answer depth-related questions better than single-modality models.

1.1 [0.5 points] Unimodal Baselines

We include the following unimodal baselines: **Q-Only+BUTD** and **Q-Only+MCAN,** we expect these to capture the bias of language.

The Q-Only model is a unimodal baseline that processes only the question as input, without incorporating images or object detections. It relies solely on a pre-trained language model (LM) to generate responses based on prior knowledge. However, due to the lack of visual input, it cannot capture scene-specific details, such as identifying the number of parked trucks in an image. Since there is no interaction between text and vision, the model is limited to answering general knowledge questions, such as those related to traffic rules or common driving behaviors. In contrast, it struggles with spatial reasoning or object-specific queries, as it lacks the necessary visual context to make accurate inferences. To extend its capability, we introduce two variations with distinct QA heads:

1. Unimodal Baseline 1 is **Q-Only+BUTD** whose key insight is using Q-Only feature extraction and BUTD as QA head.
2. Unimodal Baseline 2 is **Q-Only+MCAN** whose key insight is using Q-Only feature extraction and MCAN as QA head.

1.2 [0.5 points] Simple Multimodal Baselines

In addition to the two QA heads: BUTD and MCAN, which maps high-level object features into the text modality, we tested two straightforward feature extraction backbones as baseline models. As there are two QA heads and two feature extraction backbones, there are totally 4 simple multi-modal baselines:

1. Simple Multimodal Baseline 1 is BEVDet Huang et al. (2021)+BUTD **?** whose key insight is using BEVDet feature extraction and BUTD as QA head which advocates for computing bottom-up and top-down attention on salient regions of the image.
2. Simple Multimodal Baseline 2 is BEVDet+MCAN **?** whose key insight is using BEVDet feature extraction and MCAN as QA head which stacks self-attention and cross-attention modules for vision-language feature interaction.

3. Simple Multimodal Baseline 3 is CenterPoint Yin et al. (2021)+BUTD whose key insight is using CenterPoint feature extraction and BUTD QA head.

4. Simple Multimodal Baseline 4 is CenterPoint+MCAN whose key insight is using CenterPoint feature extraction and MCAN QA head.

1.2.1 CenterPoint

CenterPoint is a two-stage 3D object detection model that only takes LiDAR as the input modality. CenterPoint first detects objects in a bird's-eye view (BEV) representation and then refines their locations and attributes.

- **Ignores history:** CenterPoint operates on single frames and does not model temporal dependencies, making it less effective for tracking and reasoning over time.
- **Doesn't have attention:** The model relies on CNN-based feature extraction instead of self-attention mechanisms, limiting its ability to learn long-range dependencies between objects.
- **Doesn't require pretraining:** Unlike transformer-based models, CenterPoint does not rely on large-scale pretraining and can be trained directly on labeled LiDAR datasets.

Since CenterPoint only processes object detections without further multimodal fusion, it struggles with higher-order reasoning tasks that require relational understanding beyond object presence and position. Besides, CenterPoint does not directly process raw camera pixels; instead, it relies on LiDAR-based bird's-eye view (BEV) features. As a result, it can only capture coarse object-level information without detailed spatial and texture features from the original image space.

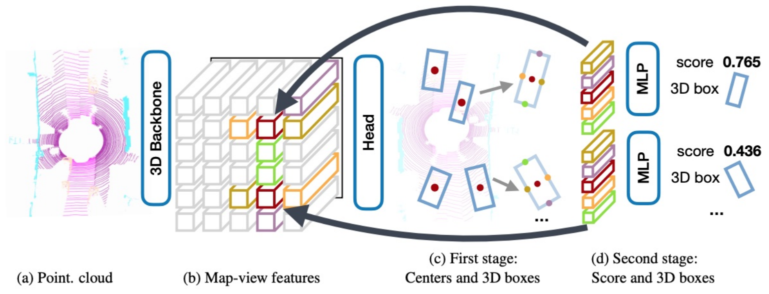

Figure 1: CenterPoint network architecture

1.2.2 BEVDet

BEVDet is a 3D object detection model that takes multi-camera images as input. The model first extracts image-view features, densely predicts pixel-wise depth, and converts the images into point clouds. Then, a pooling operation along the vertical (Z) axis aggregates these depth-aware features into a BEV feature map. Finally, a BEV encoder and detection head, partially based on the CenterPoint model, predict 3D object locations and attributes.

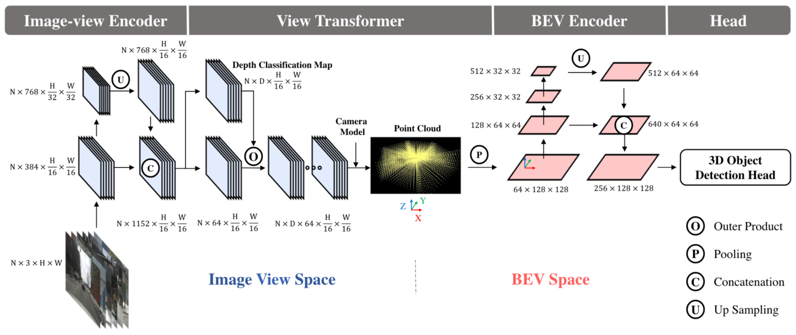

Figure 2: BEVDet network architecture

Since BEVDet indirectly predicts the pixel depths, its effectiveness largely depends on the accuracy of depth estimation and the quality of BEV feature generation. Errors in depth prediction can propagate through the pipeline, affecting the final 3D object detection performance. However, by leveraging multi-camera inputs, BEVDet can potentially capture richer semantic information compared to LiDAR, which primarily provides geometric data. This additional visual context may enhance object recognition and provide better visual features where fine-grained appearance details are important.

1.3 [0.5 POINTS] COMPETITIVE BASELINES

We run 2 competitive baselines available at public repos. These include: (these are models you found in the literature, github, etc with existing checheckpoints – no training. You should be able to run these so you can do analysis on them in R3. Additionally, they need to be rerun if you've done your own data sampling)

1. Competitive Baseline 1 is MSMDFusion **?**+BUTD **?** whose key insight is using BUTD as QA head which advocates for computing bottom-up and top-down attention on salient regions of the image. Bottom-up attention detects salient image regions using a Faster R-CNN object detector, while top-down attention learns to reason over these regions based on the input question, enabling fine-grained grounding between visual content and language.

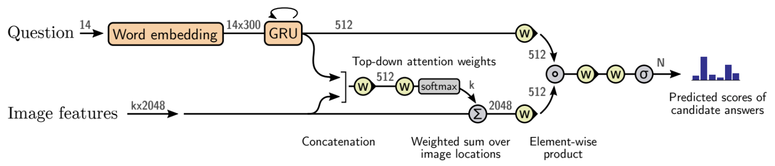
Question
Word embedding
GRU
14x300
Top-down attention weights
softmax
Image features
kx2048
Predicted scores of candidate answers
Concatenation
Weighted sum over image locations
Element-wise product

Figure 3: BUTD network architecture

2. Competitive Baseline 2 is MSMDFusion+MCAN **?** whose key insight is using MCAN as head which stacks self-attention and cross-attention modules for vision-language feature interaction. MCAN employs Modular Co-Attention to dynamically fuse image features and question embeddings at multiple levels, allowing the model to iteratively refine its understanding of both modalities. This hierarchical reasoning mechanism helps capture fine-grained alignments between visual objects and linguistic cues.

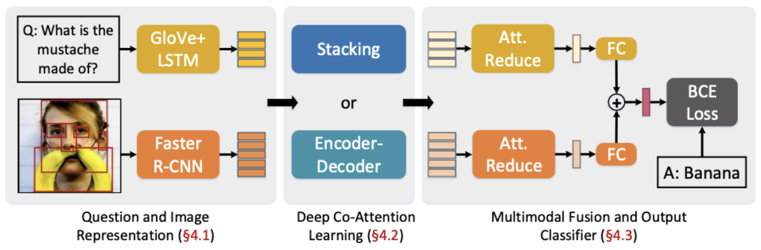
Q: What is the mustache made of?
GloVe+ LSTM
Faster R-CNN
Stacking
or
Encoder-Decoder
Att. Reduce
Att. Reduce
FC
FC
BCE Loss
A: Banana
Question and Image Representation (§4.1)
Deep Co-Attention Learning (§4.2)
Multimodal Fusion and Output Classifier (§4.3)

Figure 4: MCAN network architecture

Models	**Exist**			**Count**			**Object**			**Status**			**Comparison**			**Acc**
	H0	H1	All	H0	H1	All	H0	H1	All	H0	H1	All	H0	H1	All	
Q-Only+BUTD	81.7	78.3	79.9	18.7	18.8	18.8	63.2	40.0	43.4	57.2	49.2	52.0	81.0	64.5	66.0	54.0
BEVDet+BUTD	87.2	80.6	83.7	21.7	20.0	20.9	69.4	45.2	48.8	55.0	50.5	52.0	76.1	66.8	67.6	57.0
CenterPoint+BUTD	87.3	80.8	83.8	21.6	20.2	20.9	67.7	43.5	47.0	67.7	51.1	54.7	76.6	65.1	66.1	56.8
MSMDFusion+BUTD	89.4	81.4	85.1	25.3	21.3	23.2	73.3	48.7	52.3	67.4	55.4	59.5	81.6	67.2	68.5	59.8
Q-Only+MCAN	81.7	78.6	80.1	18.8	18.8	18.8	64.9	40.9	44.5	56.9	45.6	49.5	80.5	65.9	67.3	54.2
BEVDet+MCAN	87.2	81.7	84.2	21.8	19.2	20.4	73.0	47.4	51.2	64.1	49.9	54.7	75.1	66.7	67.4	57.9
CenterPoint+MCAN	87.1	82.4	84.6	21.7	20.8	21.2	69.8	50.9	53.7	64.5	56.3	59.1	75.5	66.8	67.6	59.3
MSMDFusion+MCAN	89.0	82.3	85.4	23.4	21.1	22.2	75.3	50.6	54.3	69.0	56.2	60.6	78.8	68.8	69.7	60.4

Table 1: Top-1 accuracy across different question types in the NuScenes-QA test set. H0 denotes zero-hop and H1 denotes one-hop. (**[1 points]** for formatted table. **[1 points]** Results for for N*2 approaches (N is number of teammates))

2 Results (1 page)

[1 points] For three metrics

2.1 Metric 1: Overall Accuracy (Acc)

Definition: Measures the top-1 accuracy across all test samples, regardless of question type.

Why is it beneficial?

- Provides a high-level evaluation of the model's performance.
- Allows for direct comparison between different models.
- Ensures that models correctly answer a variety of question types.

2.2 Metric 2: H0 and H1 Accuracy

Definition:

- **H0 (Zero-hop)**: Questions that can be answered directly from object detection (e.g., "*Is there a car?*").
- **H1 (One-hop)**: Questions that require reasoning over detected objects (e.g., "*Is the pedestrian behind the truck?*").

Why is it beneficial?

- Differentiates between direct perception (H0) and reasoning-based inference (H1).
- Highlights whether models struggle more with basic object recognition or higher-level reasoning.
- Helps in understanding how different architectures (e.g., CenterPoint, MSMDFusion) contribute to reasoning.

2.3 Metric 3: Accuracy in 5 Question Categories

Definition: Measures accuracy across five specific question categories:

1. **Existence** – Does an object exist in the scene?
2. **Counting** – How many objects are in the scene?
3. **Object Classification** – What type of object is present?
4. **Status Recognition** – What is the state of the object (e.g., moving or parked)?
5. **Comparison** – How do objects compare (e.g., "*Does the construction vehicle have the same status as the car?*")

Each category has:

- **H0 Accuracy** (direct detection-based answers).
- **H1 Accuracy** (requires additional reasoning).
- **Overall Accuracy** (average across H0 and H1).

Why is it beneficial?

- Provides a fine-grained evaluation of model strengths and weaknesses.
- Helps identify whether a model is better at direct perception (H0) or relational reasoning (H1).
- Useful for diagnosing where multimodal fusion (e.g., LiDAR + Camera) improves performance.

3 Model Proposal (1 page)

Include a diagram (e.g. labeled flow chart) of all modules. This is not final!

3.1 [1.5 points] Overall model structure

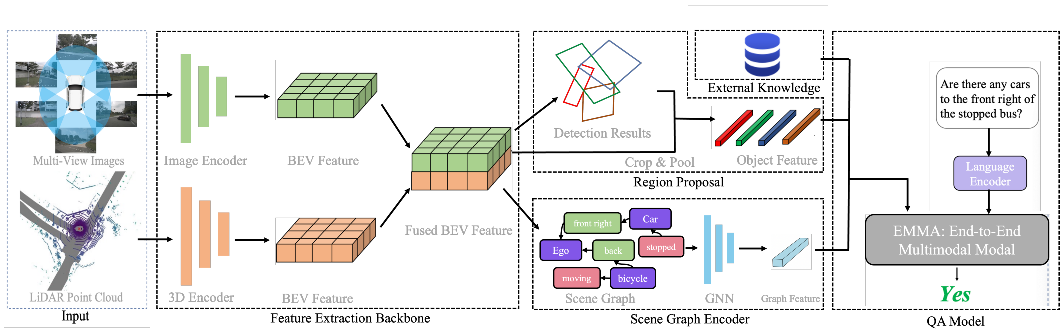

Figure 5: Proposed Model Architecture

The proposed model can be structured into four main components: the MSMDFusion encoder, scene graph encoder, external knowledge encoder, and QA model. Here's a detailed description of each component:

1. **MSMDFusion Encoder**: The MSMDFusion Encoder combines the functionalities of the Image Encoder and Point Cloud Encoder into a single, cohesive module. It processes both 2D visual data from multi-view cameras and 3D spatial data from LIDAR sensors, extracting and fusing features from these modalities into a unified Bird's Eye View (BEV) representation.
2. **Scene Graph Encoder**: This branch focuses on understanding the relationships and interactions between different objects in the scene. It uses graph-based features and possibly a Graph Neural Network (GNN) to model the connections between objects. The scene graph encoder enhances the model's ability to interpret complex scenes by capturing contextual relationships.
3. **External Knowledge Encoder**: The external knowledge encoder integrates additional contextual or domain-specific information to enrich the model's understanding. This could involve incorporating prior knowledge or external data sources to improve the accuracy and relevance of the model's predictions.
4. **QA Model**: The QA (Question Answering) model is responsible for generating responses or predictions based on the integrated features from the image encoder, point cloud encoder, scene graph branch, and knowledge branch. It leverages the multimodal data to provide comprehensive answers or detection results, ensuring that the model can perform end-to-end tasks effectively.

By organizing the model into these four components, it can effectively process and integrate multimodal data, leveraging both visual and spatial information to perform complex tasks such as object detection and scene understanding. The integration of graph-based features and external knowledge further enhances the model's capabilities, making it a robust end-to-end multimodal system.

3.2 [1 points] Encoders

3.2.1 MSMDFusion Encoder

The MSMDFusion Encoder integrates multi-view camera images and LiDAR point clouds to enhance 3D object detection in autonomous driving. It consists of an image encoder (BEVDet-based) that extracts and projects 2D image features into Bird's-Eye View (BEV) space, and a LiDAR encoder (CenterPoint-based) that processes 3D point clouds for precise object localization. By fusing multi-scale and multi-depth features, MSMDFusion improves detection accuracy, depth estimation, and spatial understanding in complex driving scenarios.

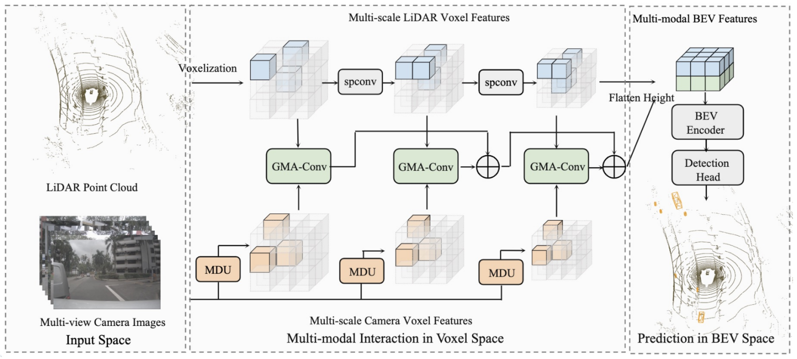

Figure 6: MSMDFusion Encoder

3.2.2 Scene Graph Encoder

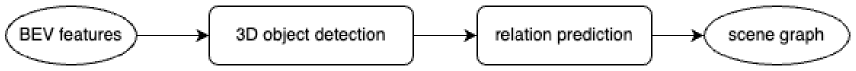

Figure 7: Scene Graph Encoder

The Scene Graph Encoder, as shown in Fig. 7, begins by utilizing an object detector to identify and localize all objects from the Bird's Eye View (BEV) features generated by the MSMDFusion Encoder. Once the objects are detected, the encoder predicts the relationships between every pair of objects, capturing contextual information such as spatial arrangements, interactions, and dependencies.

The ground truth for these relationships is derived from the annotations provided in the NuScenes-QA dataset, following the scene graph construction methods in Qian et al. (2024). We define six relationships between objects, namely *front*, *back*, *front left*, *front right*, *back left*, and *back right*. To determine object relationships, we first project 3D bounding boxes onto the Bird's-Eye-View (BEV). Subsequently, we calculate the angle between the vector connecting the centers of two bounding boxes and the forward direction of the ego-car. The formula is given by

$$\theta = \cos^{-1}\left(\frac{(B_1[:2] - B_2[:2]) \cdot V_{ego}[:2]}{\|B_1[:2] - B_2[:2]\| \|V_{ego}[:2]\|}\right), \tag{1}$$

where $B_i = [x, y, z, x_{size}, y_{size}, z_{size}, \varphi]$ is the 3D bounding box of object i, and $V_{ego} = [v_x, v_y, v_z]$ represents the speed of the ego car. Based on the angle range, the relationship between two objects is defined as

$$relation = \begin{cases} front, & \text{if } -30^\circ < \theta \leq 30^\circ \\ front\ left, & \text{if } 30^\circ < \theta \leq 90^\circ \\ front\ right, & \text{if } -90^\circ \leq \theta \leq -30^\circ \\ back\ left, & \text{if } 90^\circ < \theta \leq 150^\circ \\ back\ right, & \text{if } -150^\circ \leq \theta \leq -90^\circ \\ back, & \text{else.} \end{cases} \tag{2}$$

We define the forward direction of the car as 0° and counterclockwise angles as positive.

3.2.3 External Knowledge Encoder

The External Knowledge Encoder leverages a knowledge graph that captures general traffic rules and typical relationships between driving entities such as vehicles, cyclists, and pedestrians. Having explicitly structured rules helps in inferring right-of-way, identifying potential conflicts, and predicting plausible future behaviors based on traffic norms and spatial interactions. This graph can be generated from traffic rule databases, such as the MUTCD (Manual on Uniform Traffic Control Devices), the Highway Code (UK), and the Federal Motor Vehicle Safety Standards (FMVSS). The external knowledge extraction module retrieves relevant information from the graph, improving 3D object detection, trajectory forecasting, and question-answering tasks for autonomous systems.

By incorporating the knowledge graph, we can enhance scene understanding by providing structured context to our multimodal large language model (MLLM), enabling the model to infer implicit traffic rules, predict interactions between entities, and reason about vehicles' status and intentions more reliably.

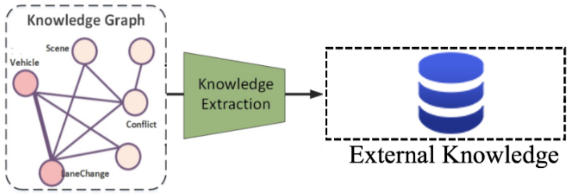

Figure 8: External Knowledge Graph Encoder

3.3 [1 POINTS] Decoders

In our model, we use **OpenEMMA?** as our QA head that is decoder. OpenEMMA is the implementation of EMMAHwang et al. (2024). EMMA (End-to-End Multimodal Model for Autonomous Driving) is a novel end-to-end framework designed for autonomous driving, built on top of a multimodal large language model (MLLM). EMMA reframes driving tasks as visual question answering (VQA) problems, leveraging the extensive world knowledge and reasoning capabilities of pre-trained language models like Gemini. In EMMA, all non-sensor inputs (e.g., high-level navigation commands and vehicle status) and outputs (e.g., future trajectories, detected objects, and road graph elements) are represented as natural language text. This unified language space allows EMMA to jointly handle multiple driving-related tasks using task-specific prompts with Chain of Thoughts design.

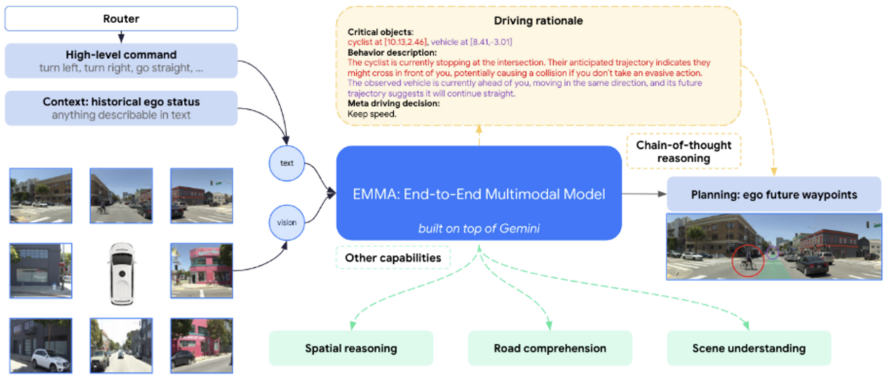
Router
High-level command
turn left, turn right, go straight, ...
Context: historical ego status
anything describable in text
text
vision
Driving rationale
Critical objects:
cyclist at [10.13,2.46], vehicle at [8.41,-3.01]
Behavior description:
The cyclist is currently stopping at the intersection. Their anticipated trajectory indicates they might cross in front of you, potentially causing a collision if you don't take an evasive action.
The observed vehicle is currently ahead of you, moving in the same direction, and its future trajectory suggests it will continue straight.
Meta driving decision:
Keep speed.
Chain-of-thought reasoning
EMMA: End-to-End Multimodal Model
built on top of Gemini
Planning: ego future waypoints
Other capabilities
Spatial reasoning
Road comprehension
Scene understanding

Figure 9: OpenEMMA Model architecture

OpenEMMA is an open-source end-to-end multimodal framework for autonomous driving, replicating the EMMA model. OpenEMMA leverages open soureced pre-trained Multimodal Large Language Models (MLLMs) combined with Chain-of-Thought reasoning to process front camera images and historical ego-vehicle status just EMMA did. To enhance object detection performance, OpenEMMA incorporates a fine-tuned YOLO3D model optimized for 3D bounding box estimation in driving scenarios. The system achieves state-of-the-art performance in trajectory planning and demonstrates strong generalizability across diverse and challenging driving scenes, showcasing EMMA's potential ability in analyzing complex driving scenarios.

3.4 [1 POINTS] Loss Functions

Describe both your primary task loss and three possible auxiliary losses that might improve performance. Justify your choices.

OpenEMMA Loss Function

The OpenEMMA is trained on multiple tasks with a single model as following:

- **Trajectory Prediction**
  Trajectory prediction aims to forecast the future positions of the ego vehicle over a fixed time horizon. This task enables the vehicle to plan its motion while anticipating future maneuvers. The prediction is typically performed in the Bird's Eye View (BEV) space, with each predicted waypoint representing the future (x, y) position at a given timestamp.

  Loss Function: The common loss function for trajectory prediction is the L2 loss between the predicted trajectory and the ground truth trajectory:

  $\mathcal{L}_{\text{traj}} = \frac{1}{T}\sum_{t=1}^{T} \|\mathbf{p}_t - \hat{\mathbf{p}}_t\|_2^2$where $\mathbf{p}_t$ and $\hat{\mathbf{p}}_t$ are the ground truth and predicted coordinates at time t, respectively, and T is the prediction horizon.

- **3D Object Detection**
  3D object detection focuses on detecting traffic participants such as vehicles, pedestrians, and cyclists in the scene. The goal is to predict a 3D bounding box for each object, which includes position (x, y, z), dimensions (w, h, l), and orientation θ. This task is critical for understanding the spatial layout of the scene.

  Loss Function: The loss for 3D object detection typically combines multiple components, including localization loss, size loss, and orientation loss:

  $$\mathcal{L}_{\text{3D}} = \mathcal{L}_{\text{cls}} + \lambda_1\mathcal{L}_{\text{loc}} + \lambda_2\mathcal{L}_{\text{size}} + \lambda_3\mathcal{L}_{\text{yaw}}$$

  where:

  - $\mathcal{L}_{\text{cls}}$: Classification loss (e.g., focal loss) for object category.
  - $\mathcal{L}_{\text{loc}} = \|\mathbf{c} - \hat{\mathbf{c}}\|_1$ for 3D center coordinates.
  - $\mathcal{L}_{\text{size}} = \|\mathbf{s} - \hat{\mathbf{s}}\|_1$ for object size.
  - $\mathcal{L}_{\text{yaw}}$: Orientation loss (e.g., smooth L1 or sine-cosine regression).

- **Road Graph Estimation**
  Road graph estimation is a structured prediction task that reconstructs the road topology in the scene. This involves predicting a set of unordered polylines, each represented as a sequence of waypoints, which collectively form the drivable lanes and road boundaries. Evaluation uses two levels:

  - Lane-level precision and recall: True positives are polylines that match a ground truth polyline within a Chamfer distance threshold (1 meter).
  - Pixel-level precision and recall: Both predicted and ground truth polylines are rasterized into a BEV grid (1m resolution), and matching is done per-pixel.

  Loss Function: The loss can be formulated as a combination of Chamfer Distance Loss (for polyline matching) and Binary Cross Entropy (BCE) (for rasterized segmentation map):

  $$\mathcal{L}_{\text{road}} = \mathcal{L}_{\text{chamfer}} + \lambda\mathcal{L}_{\text{bce}}$$

- Chamfer Distance Loss between predicted and ground truth polylines:

$$\mathcal{L}_{\text{chamfer}} = \frac{1}{N}\sum_{p\in\hat{\mathcal{P}}}\min_{q\in\mathcal{P}}\|p-q\|_2 + \frac{1}{M}\sum_{q\in\mathcal{P}}\min_{p\in\hat{\mathcal{P}}}\|p-q\|_2$$

where $\hat{\mathcal{P}}$ and $\mathcal{P}$ denote predicted and ground truth polylines.

- BCE loss for rasterized BEV grid:

$$\mathcal{L}_{\text{bce}} = -\frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}\left(y_{ij}\log(\hat{y}_{ij}) + (1-y_{ij})\log(1-\hat{y}_{ij})\right)$$

- **Scene Understanding (Temporary Blockage Detection)**
Scene understanding focuses on understanding the semantic meaning of the current driving scene. A key sub-task is temporary blockage detection, where the model determines whether a lane is temporarily blocked. This is framed as a binary classification task ("blocked" or "not blocked"). The model is typically pre-trained on related tasks (like road graph estimation) to enhance its contextual understanding before fine-tuning on this specific classification task.
Loss Function: The loss for this binary classification task is simply:

$$\mathcal{L}_{\text{scene}} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i\log\hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right)$$

where y_i is the ground truth label (1 for blocked, 0 for not blocked), and $\hat{y}_i$ is the predicted probability.

References

Junjie Huang, Guan Huang, Zheng Zhu, Yun Ye, and Dalong Du. Bevdet: High-performance multi-camera 3d object detection in bird-eye-view. In *arXiv preprint arXiv:2112.11790*, 2021. URL `https://arxiv.org/abs/2112.11790`.

Jyh-Jing Hwang, Runsheng Xu, Hubert Lin, Wei-Chih Hung, Jingwei Ji, Kristy Choi, Di Huang, Tong He, Paul Covington, Benjamin Sapp, James Guo, Drago Anguelov, and Mingxing Tan. Emma: End-to-end multimodal model for autonomous driving. *ArXiv*, abs/2410.23262, 2024. URL `https://api.semanticscholar.org/CorpusID:273695673`.

Tianwen Qian, Jingjing Chen, Linhai Zhuo, Yang Jiao, and Yu-Gang Jiang. Nuscenes-qa: A multi-modal visual question answering benchmark for autonomous driving scenario. *arXiv preprint arXiv:2305.14836*, 2024. URL `https://github.com/qiantianwen/NuScenes-QA`.

Tianwei Yin, Xingyi Zhou, and Philipp Krähenbühl. Centerpoint: A unified framework for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. URL `https://arxiv.org/abs/2006.11275`.

A Appendix

You may include other additional sections here.