SmolDriver: LLM Guided Efficient Autonomous Driving Model

Arian Firoozi
arianfirooziM@gmail.com

Mehdi Jamalkhah
mehdijmlkh1382@gmail.com

1 Problem statement

In previous years, autonomous driving (AD) has become a challenging problem due to its system 2 nature and hard generalization (Chen et al., 2025), (Li et al., 2025). Recently, with the emergence of large language models (LLMs), many attempts have been made to provide a general solution for real-time and reliable driving (Wu et al., 2024), (Yang et al., 2024), (Cui et al., 2023). Still, most of those models lack one of the mentioned aspects. We examined state-of-the-art methods and came to the conclusion that integrating Vision Language Models into autonomous driving comes with two primary effects: firstly, the models, including LLMs or VLMs are more adaptive to unseen circumstances and generate better results due to their vast knowledge and common sense, secondly, many Language models require powerful and costly hardware and have too much computational overhead. For these reasons, we decided to take a hybrid approach by guiding the planning model with a LLM, leveraging its general adaptability while maintaining the memory and time efficiency of small models similar to DiMA (Hegde et al., 2025).

We decided to implement a framework that leverages the power of LLMs and provides insight and scene understanding that can later be used for guiding a smaller model to perform autonomous driving. We originally aimed to implement the scene planner as a transformer model but due to time constrained we limited our work to LLM part only, aiming fine-tuning LLaVA (Liu et al., 2024) and evaluate its scene understanding. Later on, we changed the baseline once more since we could not find the financial support to acquire a decent GPU (more specifically the university was closed and we did not have access to laboratory environment). The final model we worked on is SmolVLM2 (Marafioti et al., 2025) due to its small size and accurate predictions.

2 What you proposed vs. what you accomplished

The original project was designed to involve a VAD guided by an LLM for generating low level navigation instructions, but since the proposed timeline was limited, we decided to limit our project to designing and implementing the LLM part of the project. We changed the base line again to SmolVLM to reduce the training computational power to match our limited resources. The workflow included following steps:

- **Prepare dataset**: acquring data from DriveLM dataset and reprocessing to get it ready for the model
- **Prepare modules and ensure they function as expected**: implementation and test of various modules including Q-Formers, multi-qformer module and vision model
- ~~Implement teacher-student framework~~: this part has been excluded based on the instructors' recommendation due to time constraints and its limited relevance to the course.
- **Swap the vision model**: swapping the vision model of the SmolVLM model and testing the inference for incompatibilities
- **Train the new vision model via knowledge transfer**: perform knowledge transfer between the new vision model and the original one
- **Fine tune the model**: fine tuning the new model via LoRA and fine tuning the vision model in an end-to-end manner

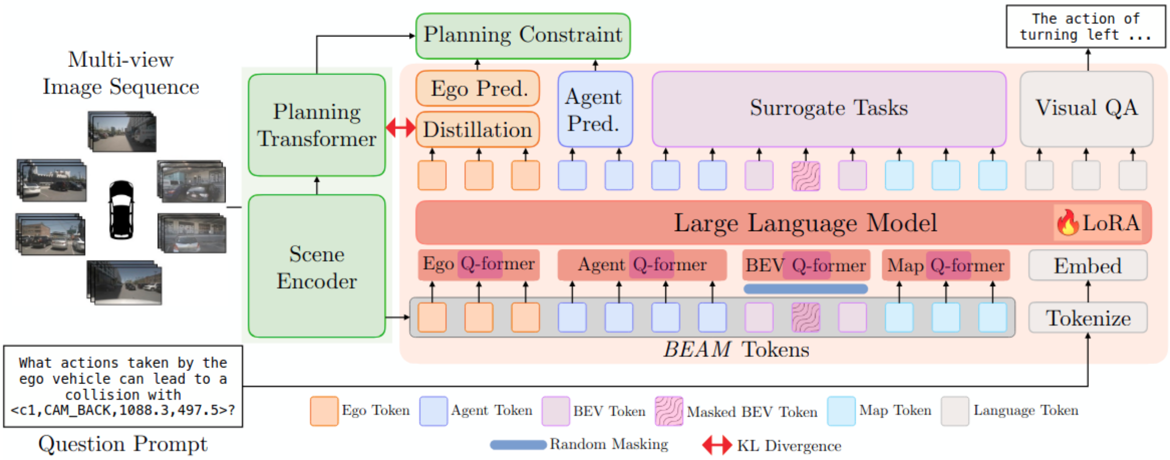

Figure 1: DiMA Framework(Hegde et al., 2025)

- ~~Try to generalize the model~~: We excluded this part too. As we mentioned earlier, this section was intended to be carried out on a highend GPU, but we failed to acquire the necessary hardware.
- **Link two datasets and feed more data to the model**: find the DriveLM samples in NuScenes and extract map and Lidar information in order to equip the model with bird-eye-view (BEV) and map tokens. this stage was partially done, but we did not included the full dataset and therefore did not use it in training because of the huge size of the Nuscenes dataset and unnecessary overhead
- **Final tests**: test and compare the model with the baseline
- **Final report**: Report and document the work

3 Related work

Behavior planning in Autonomous Driving refers to the task of determining high-level actions that a vehicle should take to reach its destination which is then converted into low level actions such as steering and increasing speed in an end-to-end driving network. Over the recent years, serious improvements were achieved in behavior planning with integrating transformer based models into the architectures of the models. Models like UniAD (Hu et al., 2023) and VAD (Jiang et al., 2023) used transformers in both decision making and scene representation (by integrating methods like BEV-Former (Li et al., 2022)) and outperformed RL-based methods in closed and open loop driving. While many traditional models aimed to achieve a safe and generalizable AD model that could adapt to difficult conditions that may appear outside of a controlled environment, these methods lacked the necessary world knowledge to be able to adapt to different conditions. GPT-Driver (Mao et al., 2023) overcame this problem by using a GPT-3.5 model to plan the vehicle movement. Although it surpassed the UniAD and VAD (which were the SOTA methods for Autonomous Driving at time) in accuracy, but it was far too slow to have practical use. Methods such as TOKEN (Tian et al., 2024) and DriveLM (Sima et al., 2023) further improved the accuracy of LLMs in driving. Specifically, DriveLM used question-answer pairs to fine-tune the model and forced the LLM to gain deep understanding of scene and motion of both ego vehicle and other agents and TOKEN tokenized the world into object-level knowledge used DriveLM dataset and added more QA pairs, but both of these models lacked the real-time necessity. To overcome the performance issues while keeping a LLM in the training loop, DiMA (Hegde et al., 2025) offered a framework in which a VLM was fine-tuned similar to TOKEN, but instead of using LLM directly the knowledge gained by VLM was distilled into a transformer based planning network. Unfortunately, we did not come across any public implementation of this network and decided to re-build the network with slight modifications and experiment with different modules to see the effects of replacing Q-Former, scene encoder and planning transformer shown in Fig. 1. DiMA is built upon MQT (Hu et al., 2024) which is an extension of the LLaVA (Liu et al., 2023) framework. The overall architecture of LLaVA

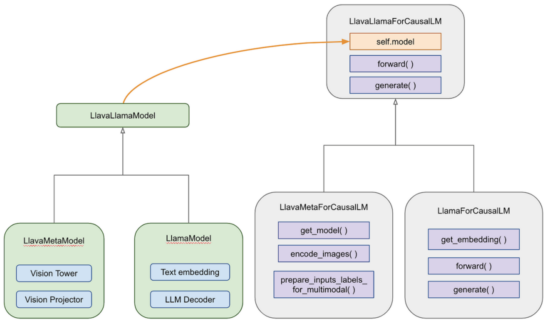

Figure 2: LLaVA model architecture(Liang, 2025)

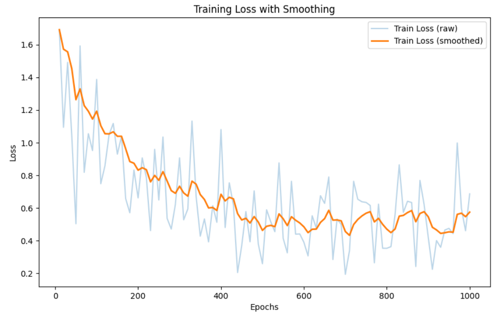

Figure 3: Fine-tuning loss for SmolVLMv2

is shown in Figure 2, and it uses Vicuna LLM (Zheng et al., 2023) (which is obtained by fine-tuning LLaMA). MQT framework adds a layer after the CLIP (Radford et al., 2021) vision tower and enables the model to learn with encodings of varying length. Similar to DiMA, we modify this layer to take multiple images as input instead of a single image to take advantage of multiple views of the scene and enhance scene understanding of the model.

The final baseline of the project is **SmolVLM2**: a compact and resource-efficient vision language model based on SmolLM2 (Allal et al., 2025) which only uses 4.9GB of RAM during inference and is suitable for full finetuning in a Google Colab environment. This VLM has been trained on a diverse dataset including OCR and Documents, Captioning, Chart Understanding, Reasoning, Table Understanding and most importantly for us, Visual Question Answering (Marafioti et al., 2025). SmolVLM is the state-of-the-art VLM in its parameter size, and outperforms larger models such as MolmoE 7B (Deitke et al., 2024) in various single image tasks. Furthermore, SmolVLM is much easier to change by loading and accessing its weights directly unlike LlaVA which required changing some of its inner functions that could only be done by changing its source code (which we did by the way). Complete separation of vision and text logic in SmolVLM allowed us to change the parameters with small adjustments in configuration and modules. The only disadvantage of using a smaller model is that it is more sensitive to input and building a precise vision model is essential in order to get optimal results. We will address this issue by healing steps after the replacement of the new vision model. Although our intention was to find a balance between computational efficiency and accuracy of the model, our current method has more emphasis on computational efficiency and we do not expect to outperform much larger models since we only had access to extremely limited resources.

4 Dataset

We fine-tuned our VLM using supervised visual question answering (VQA) using the DriveLM dataset (Sima et al., 2023), which contains a curated 4K-sample subset of the nuScenes (Caesar et al., 2020) dataset annotated with 300K question-answer pairs. These QA pairs cover key aspects of autonomous driving, including perception, prediction, planning, and ego-vehicle behavior. By training on these QA pairs, especially the preception-related samples, the model extracts more meaningful and context-aware features from the input camera images. This, in turn, is expected to improve performance on downstream autonomous driving tasks such as prediction and planning.

5 Baselines

Since this project focuses exclusively on the fine-tuning of the VLM, we need to establish a baseline specifically for this component. To that end, we use BERTScore and BLEU metrics. We consider the performance of the pre-trained VLM without any fine-tuning as our baseline and perform an additional comparison with a fine-tuned SmolVLMv2. The fine-tuning is done using DriveLM dataset with 1000 samples and since the loss didn't improve after 500 samples, we stopped the fine-tuning process and didn't fed all of the DriveLM dataset to the model. Loss function of the model is pictured in Figure 3.

6 Our approach

Initially, our approach was to distill an LLM into a VAD model. However, given the hardware limitations and irrelevance of the transformer-based VAD to the course, we decided to narrow the scope of the project to focus on the design and implementation of the LLM component. Specifically, we aimed to design a VLM on a question-answering (QA) task, where the model receives an input image from a car's cameras along with a corresponding question about the scene. The objective is for the model to generate accurate answers based on the visual content. The training and fine-tuning process is intended to help the model learn to extract meaningful visual features, which can later facilitate the distillation of the LLM into a VAD in future work. We tried several methods for manipulating the LLaVA architecture in a way that it supports several **Matryoshka Query Transformers**. We used MQT-LLaVA model (Hu et al., 2024) in which the LLaVA model has been equipped with a single Query Transformer (or resampler in the code). We first tried to change the loaded model with torch functions, but that approach lacked the required flexibility for our task (for example it was not possible to modify generate function). Therefore, we decided to modify library code and change the necessary functions prior to loading. After a comprehensive study of the model architecture, we modified the query abstractor inside LlavaMetaModel which is a submodule of the more complex LLaVA architecture. The modified layer is placed after the vision tower of the VLM, and modifies the image features generated by this layer and prepares it for LLM using attention mechanism. In our approach, the forward function is modified to generate separate features for each image and pass it to the MQT layer. MQT layer then routes each of the features to a resampler using the MultResampler layer that we implemented and all the results are concatenated and passed to the next layers. After completing this step, we loaded the model and as we initially assessed, 16GB of GPU RAM on Google Colab was not enough to train the model. As we discussed earlier, we abandoned this method and moved on to a new baseline. For the various aforementioned reasons, we chose **SmolVLM 2B instruct** as our baseline. We examined the overall structure of the model and its training method and concluded that the model has the potential to be an effective baseline for our task. The model consists of vision and LLM components that operated sequentially, but almost independently. We examined the vision model and designed our own vision model to match the input and output shapes. Our vision model consists of a vision tower for feature extractor and a multi-qformer part to process those features. When our pipeline receives data, each batch of N images are resized to match the initial size of our vision tower. The vision tower we chose to use is Google's visual transformer (Wu et al., 2020) and is a well known and widely used ViT and extracts image features with reasonable accuracy and overhead. The extracted features are then directed into N qformer (resampler) modules that are the designed in MQT as depicted in Figure 4 with minor modifications. All of the resamplers are cascaded in a parallel fashion using a custom class and can receive images with independent semantic and structural features.

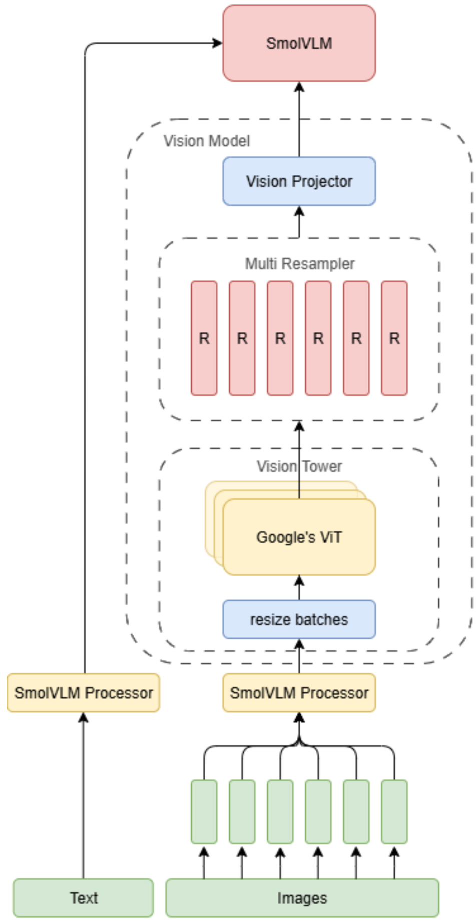

Figure 4: SmolDriver (ours)

To train the new vision model, we used knowledge transfer principles. We separated the original vision tower from the model and used it as a

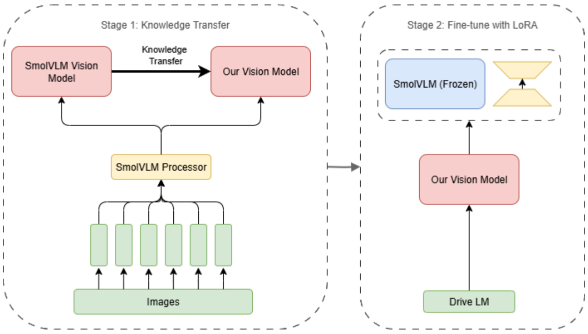

Figure 5: Training Stages of Our Model

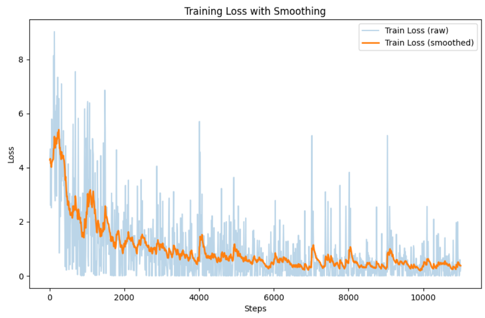

Figure 6: SmolDriver Training Loss

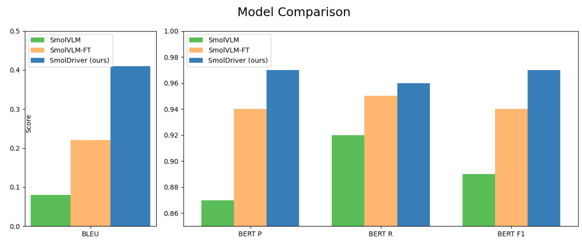

Figure 7: Metric Comparison Across Models

teacher model, guiding the new vision model on our dataset. We first tested this method on a single image and after proving that this training approach is effective, we proceeded to train the model on the entire dataset to teach the new vision model to replicate the answers given by the original model. After this initial training, we fine tuned the model in an end-to-end fashion by installing the vision model on VLM and freezing the LLM parameters to unleash the potential of multi-qformer modules. This new model proves to be more useful to our tasks since each qformer is specializing in a specific data type instead of relying on previously employed monolithic general model. This method can also be used to combine images with different modalities such as radar and LiDAR images, but we did not explore this feature and limited our scope to 2D camera images only. We fine-tuned the model on our QA dataset, giving the model all six images from six different angles of the vehicle: back, back left, back right, front, front left and front right and a question about the scene, then model was expected to generate the proper answer. This fine-tuning completed by using Quantized-LoRA for all linear modules.

7 Error analysis and Results

All of our experiments were done on a 16GB T4 GPU on Google Colab. We used image samples from DriveLM dataset and Nuscenes to train and evaluate out models. The dataset was split with a 1/1/8 ration for evaluation, test and train, although we did not use evaluation in some parts of experiment that we were not concerned about the accuracy training process (for example in knowledge transfer phase). Training loss is show in Figure 6 and the overall loss continued to drop through the whole training unlike SmolVLM-FT which saturated after only 500-600 samples were given. We evaluated our results in two ways: First, we evaluated the models on metrics including BLEU, Bert Precision, Bert Recall and Bert F1, Second, we randomely selected some generated samples and compared the answers to labels. Metrics are shown in Figure 7 and Table 1. We can observe that our SmolDriver outperformed both baselines in all metrics, showing the effectiveness of our implementation. Some samples of the manual evaluation is shown in Table 2. We observed that the raw SmolVLM model struggled with answering Yes/No questions and had a tendency to give comprehensive (and often wrong) answers to questions. Although this was fixed in the Fine-tuned SmolVLM, the accuracy was still lower than SmolDriver. For example, in indexes 30 and 203 it failed to give the correct answer to a Yes/No question while SmolDriver successfully answered the questions. Despite its acceptable performance, our method had some crucial mistakes on some questions, especially more sophisticated questions that required reasoning and future prediction.

8 Contributions of group members

Our contribution is listed below. All the code will be made publicly available once it finished in this repository. Our datasets, models, and results are currently available at this and this link.

- Mehdi: dataset preparation, a few bug

Model	BLEU	BERT P	BERT R	BERT F1
SmolVLM	0.085	0.870	0.918	0.893
SmolVLM-FT	0.214	0.945	0.941	0.943
SmolDriver (ours)	**0.406**	**0.972**	**0.961**	**0.967**

Table 1: Comparison of BLEU and BERT metrics across models

fixes in vision model integration, fine-tuning SmolVLM, unsuccessful attempt to fine-tune LLaVA using different approaches, fine-tuning SmolDriver, evaluation of SmolDriver

- Arian: vision model implementation, integration of vision model into SmolVLM, knowledge transfer, failed attempt to load LlaVA with multi-qformer, fine-tuning SmolVLM-FT, evaluation of SmolVLM-FT and SmolVLM, charts and figures

9 Conclusion

It is obvious that changing fundamental parts of a model after pretraining is a challenging task, but the scope of the work had to be done went beyond our expectations. While changing larger models such as LlaVA is remarkably hard and requires careful examination of the model parameters, even changing smaller models like SmolVLM that is designed by a much simpler process is still a demanding task. The process includes finding solutions to internal mistmatches, dependency errors, dataset incompatibilities and model configurations, and generally poorly documented model structures plays a major role specially when dealing with less studied models. While this task is difficult, its essential to truly change a VLM's structure if the task demands an expert VLM. While our approach yielded promising results, the rare but fatal mistakes shows that an additional safety mechanism must be implemented on top of the LLM to stop the vehicle from moving with a rule-based manner at the time of emergency. Our goal was to create a fast and reliable LLM for behavior planning in autonomous driving and while the results are promising, our model is not even close to perfect. Autonomous driving has been a major challenge for years and many methods have been used in the past but none of the current models is suitable for a true autonomous vehicle. As for future improvements, we suggest training the model in a diverse dataset including the extreme conditions as our current model has only been trained on images from normal weather and traffic conditions. A safety mechanism for extreme cases must be employed if the model is going to be used in real-world applications, and data other than images (such as radars and lidars) can be fed to the model in order to enhance the scene understanding in order to create a safe and reliable AD model.

10 Data and Code Availability

All of the code used in the project can be found in github.com/MehdiJmlkh/LLM-Project, and trained models and data are available at huggingface.co/mehdijmlkh and huggingface.co/arianfiroozi.

References

Allal, L. B., Lozhkov, A., Bakouch, E., Blázquez, G. M., Penedo, G., Tunstall, L., Marafioti, A., Kydlíček, H., Lajarín, A. P., Srivastav, V., Lochner, J., Fahlgren, C., Nguyen, X.-S., Fourrier, C., Burtenshaw, B., Larcher, H., Zhao, H., Zakka, C., Morlon, M., Raffel, C., von Werra, L., and Wolf, T. (2025). Smollm2: When smol goes big – data-centric training of a small language model.

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving.

Chen, Q., Qin, L., Liu, J., Peng, D., Guan, J., Wang, P., Hu, M., Zhou, Y., Gao, T., and Che, W. (2025). Towards reasoning era: A survey of long chain-of-thought for reasoning large language models.

Cui, C., Ma, Y., Cao, X., Ye, W., Zhou, Y., Liang, K., Chen, J., Lu, J., Yang, Z., Liao, K.-D., Gao, T., Li, E., Tang, K., Cao, Z., Zhou, T., Liu, A., Yan, X., Mei, S., Cao, J., Wang, Z., and Zheng, C. (2023). A survey on multimodal large language models for autonomous driving.

Deitke, M., Clark, C., Lee, S., Tripathi, R., Yang, Y., Park, J. S., Salehi, M., Muennighoff, N., Lo, K., Soldaini, L., Lu, J., Anderson, T., Bransom, E., Ehsani, K., Ngo, H., Chen, Y., Patel, A., Yatskar, M., Callison-Burch, C., Head, A., Hendrix, R., Bastani, F., VanderBilt, E., Lambert, N., Chou, Y., Chheda, A., Sparks, J., Skjonsberg, S., Schmitz, M., Sarnat, A., Bischoff, B., Walsh, P., Newell, C., Wolters, P., Gupta, T., Zeng, K.-H., Borchardt, J., Groeneveld, D., Nam, C., Lebrecht, S., Wittlif, C., Schoenick, C., Michel, O., Krishna, R., Weihs, L., Smith, N. A., Hajishirzi, H., Girshick, R., Farhadi, A., and Kembhavi, A. (2024). Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models.

Hegde, D., Yasarla, R., Cai, H., Han, S., Bhattacharyya, A., Mahajan, S., Liu, L., Garrepalli, R., Patel, V. M., and Porikli, F. (2025). Distilling multi-modal large language models for autonomous driving.

Hu, W., Dou, Z.-Y., Li, L. H., Kamath, A., Peng, N., and Chang, K.-W. (2024). Matryoshka query transformer for large vision-language models.

Hu, Y., Yang, J., Chen, L., Li, K., Sima, C., Zhu, X., Chai, S., Du, S., Lin, T., Wang, W., Lu, L., Jia, X., Liu, Q., Dai, J., Qiao, Y., and Li, H. (2023). Planning-oriented autonomous driving.

Jiang, B., Chen, S., Xu, Q., Liao, B., Chen, J., Zhou, H., Zhang, Q., Liu, W., Huang, C., and Wang, X. (2023). Vad: Vectorized scene representation for efficient autonomous driving.

Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Yu, Q., and Dai, J. (2022). Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers.

Li, Z.-Z., Zhang, D., Zhang, M.-L., Zhang, J., Liu, Z., Yao, Y., Xu, H., Zheng, J., Wang, P.-J., Chen, X., Zhang, Y., Yin, F., Dong, J., Guo, Z., Song, L., and Liu, C.-L. (2025). From system 1 to system 2: A survey of reasoning large language models.

Liang, A. (2025). Understanding llava architecture & code: A detailed explanation. https://medium.com/@liang/understanding-llava-architecture-code-a-detailed-explanation-c84da9fe4a4e. Accessed: 2025-09-03.

Liu, H., Li, C., Li, Y., and Lee, Y. J. (2024). Improved baselines with visual instruction tuning.

Liu, H., Li, C., Wu, Q., and Lee, Y. J. (2023). Visual instruction tuning.

Mao, J., Qian, Y., Ye, J., Zhao, H., and Wang, Y. (2023). Gpt-driver: Learning to drive with gpt.

Marafioti, A., Zohar, O., Farré, M., Noyan, M., Bakouch, E., Cuenca, P., Zakka, C., Allal, L. B., Lozhkov, A., Tazi, N., Srivastav, V., Lochner, J., Larcher, H., Morlon, M., Tunstall, L., von Werra, L., and Wolf, T. (2025). Smolvlm: Redefining small and efficient multimodal models. *arXiv preprint arXiv:2504.05299*.

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. (2021). Learning transferable visual models from natural language supervision.

Sima, C., Renz, K., Chitta, K., Chen, L., Zhang, H., Xie, C., Luo, P., Geiger, A., and Li, H. (2023). Drivelm: Driving with graph visual question answering. *arXiv preprint*, arXiv:2312.14150:1, 2, 4, 5, 8, 14, 19.

Tian, R., Li, B., Weng, X., Chen, Y., Schmerling, E., Wang, Y., Ivanovic, B., and Pavone, M. (2024). Tokenize the world into object-level knowledge to address long-tail events in autonomous driving.

Wu, B., Xu, C., Dai, X., Wan, A., Zhang, P., Yan, Z., Tomizuka, M., Gonzalez, J., Keutzer, K., and Vajda, P. (2020). Visual transformers: Token-based image representation and processing for computer vision.

Wu, J., Huang, C., Huang, H., Lv, C., Wang, Y., and Wang, F.-Y. (2024). Recent advances in reinforcement learning-based autonomous driving behavior planning: A survey.

Yang, Z., Jia, X., Li, H., and Yan, J. (2024). Llm4drive: A survey of large language models for autonomous driving.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E. P., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging llm-as-a-judge with mt-bench and chatbot arena.

Index	Question	Answer	SmolDriver	SmolVLM	SmolVLM-FT
30	Are there moving cars to the back of the ego car?	Yes.	**Yes.**	No, there are no moving cars to the back of the ego car.	No.
45	Is there any traffic element in the front view?	Yes, there are traffic elements in the front view.	**Yes, there are traffic elements in the front view.**	**Yes, there is a traffic light in the front view.**	**Yes.**
48	Are there parked cars to the back left of the ego car?	No.	Yes.	The image does not provide enough information to determine if there are parked cars to the back left of the ego car.	Yes.
58	Are there parked trailers to the back of the ego car?	Yes.	No.	No, there are not any parked trailers to the back of the ego car.	No.
94	What is the traffic signal that the ego vehicle should pay attention to?	None.	**None.**	The traffic signal that the ego vehicle should pay attention to is the one at the intersection.	**None.**
203	Are there standing pedestrians to the back right of the ego car?	Yes.	**Yes.**	No, there are no standing pedestrians to the back right of the ego car.	No.
248	What are objects to the back left of the ego car?	There is one truck, two barriers, many cars, and one traffic cone to the back left of the ego car.	There are two pedestrians to the back left of the ego car.	There are two orange traffic cones and one orange traffic barrier to the back left of the ego car.	The objects to the back left of the ego car are a white building, a white truck, and a white building.
421	Are there moving construction vehicles to the front left of the ego car?	No.	**No.**	**There are no moving construction vehicles to the front left of the ego car.**	**No.**

Table 2: Comparison of answers