DriveMoE: Mixture-of-Experts for Vision-Language-Action Model in End-to-End Autonomous Driving

Supplementary Material

5. Related Work

5.1. VLM/VLA in End-to-end Autonomous Driving

The advancement of Large Language Models (LLMs) has significantly accelerated the development of Vision-Language Models (VLMs) for autonomous driving. Leveraging powerful generalization, open-set reasoning, and scalability, these models have become influential paradigms for end-to-end driving tasks.

DriveGPT-4 [51] pioneers the integration of multimodal LLMs into end-to-end driving by introducing a vision–language–action framework that jointly performs control prediction and natural-language explanation. DriveLM [42] further explores the reasoning aspect of driving intelligence through graph visual question answering (GVQA). LMDrive [40] formulates perception and planning tasks as sequences of discrete tokens, enabling better interpretability and facilitating cross-domain knowledge transfer. LeapAD [35] adopts a data-driven and knowledge-driven Vision-Language Model (VLM) approach. SimLingo [39] proposes a fully closed-loop VLA architecture aligning language reasoning with low-level control. To enhance learning efficiency, it introduces a data-bucket sampling scheme that focuses on diverse and high-risk scenarios, avoiding overfitting to trivial straight-driving cases.

However, Existing approaches predominantly rely on discrete token-based for driving policy learning, without exploring continuous tokenization or diffusion-based policies that could bridge vision-language understanding and continuous control. Moreover, despite the increasing interest in multimodal skill modeling for complex driving scenes, no prior work has investigated Mixture-of-Experts (MoE) architectures to enhance specialization and generalization across diverse driving domains.

To address this limitation, the embodied AI community has proposed vision-language-action (VLA) models that represent actions as continuous variables instead of discrete tokens. Methods such as OpenVLA [28], Diffusion Policy [6] and π_0 [3] demonstrate strong performance by modeling continuous action distributions through sequence prediction and global optimization. Nevertheless, these approaches often rely on task-specific policies or instruction-conditioned models, which struggle to generalize across the long-tail distribution of behaviors seen in complex driving environments.

5.2. Mixture-of-Experts in Large Language Models

Recent progress in Mixture-of-Experts (MoE) architectures has demonstrated remarkable efficiency and scalability in large language models (LLMs). Sparse Mixture-of-Experts (MoE) architectures have become a mainstream approach for scaling LLMs. By replacing the standard feedforward layers in Transformers with expert modules, models like DeepSeekMoE [8] and Mixtral-8x7B [8] improve task specialization and representation capacity while maintaining inference efficiency through conditional computation. In robotics, MoE architectures have also been used to address task heterogeneity and long-tailed data distributions. For example, MENTOR [16] replaces the MLP backbone with MoE layers to enable gradient routing among modular experts, helping mitigate gradient interference in multi-task learning. Despite promising results in language modeling and robot policy learning, the use of MoE in end-to-end autonomous driving remains underexplored.

Motivated by these gaps, our metheds is the first end-to-end autonomous driving framework that integrates MoE at both the vision and action levels.

6. Conditional Flow Matching Loss

Following prior work [3, 32, 33], our method predicts future action trajectories in a denoising manner using a conditional flow matching loss,

$$L^\tau(\theta) = \mathbb{E}_{p(\mathbf{A}_t|\mathbf{o}_t),\, q(\mathbf{A}_t^\tau|\mathbf{A}_t)} \left\| \mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t) - \mathbf{u}(\mathbf{A}_t^\tau \mid \mathbf{A}_t) \right\|^2 \tag{7}$$

where subscripts denote timesteps and superscripts denote flow matching timesteps, with $\tau \in [0, 1]$. We sample noisy actions $\mathbf{A}_t^\tau = \tau \mathbf{A}t + (1-\tau)\epsilon$ and train the network to output a denoising flow $\mathbf{v}_\theta(\mathbf{A}_t^\tau, o_t)$ that matches the ground-truth direction $\mathbf{u}(\mathbf{A}_t^\tau \mid \mathbf{A}_t) = \epsilon - \mathbf{A}_t$. This formulation enables the model to learn the underlying trajectory distribution by aligning with a continuous stochastic process, rather than relying on pointwise supervision. It is particularly suitable for multimodal or uncertain planning scenarios in autonomous driving, where modeling smooth action trajectories is critical.

7. Implementation Details

Vision Routing Annotations: We introduce additional camera-view importance annotations into the Bench2Drive [23] dataset. This annotation approach is

both inexpensive and straightforward, yet it significantly improves model performance through efficient and effective utilization of multi-camera inputs. The details about camera annotation rules refer to Appendix 9.

Action Routing Annotations: We maintain skill definitions consistent with Bench2Drive [23] setup. There are five driving skills: Merging, Overtaking, Emergency Brake, Give Way, and Traffic Sign.

Drive-π_0: We utilize 2 sequential front-view images as input to our model to effectively estimate the velocities of surrounding traffic agents. Additionally, the input state incorporates both current and historical information, including position, velocity, acceleration, and heading angle, enabling the model to predict 10 future waypoints accurately.

DriveMoE: We utilize 2 sequential front-view images combined with a dynamically selected camera view as inputs to our model. The sequential front-view images primarily capture temporal changes to model the velocities of surrounding traffic agents, while the dynamic view is obtained by selecting the Top-1 view from the vision router, which enhances spatial perception according to driving context. The input state representation remains consistent with the π_0 framework, including current and historical position, velocity, acceleration, and heading angle information. In the action model, we employ 1 shared expert and 6 non-shared experts. During the training and inference, the top-3 experts selected by the action router are utilized to generate the final trajectory prediction consisting of 10 future waypoints. We adopt a two-stage post-training strategy for our model:

Training Stage 1. We train the model for 12 epochs. The Vision-Language Model (VLM) component is initialized from the pretrained weights of Paligemma-3b-pt-224 [2]. The VLM and Action MoE experts are optimized separately using two optimizers, both configured as follows: learning rate = 5×10^{-5}, and warmup steps enabled. Gradient clipping is applied with a maximum gradient norm of 1.0. Gradient accumulation is used to simulate a batch size of 128. To balance different loss components effectively, we set the vision router loss weight λ_0 to 10.0, action router loss weight λ_2 to 10.0, flow matching loss weight λ_1 to 1.0.

Training Stage 2. We continue training for an additional 6 epochs, initializing from the checkpoint obtained at the end of Stage 1. In this stage, input camera views and action experts are dynamically selected based on outputs from the routers.the vision router loss weight λ_0 to 5.0, action router loss weight λ_2 to 5.0, flow matching loss weight λ_1 to 1.0, emphasizing trajectory learning. Other hyperparameters remain consistent with Stage 1.

PID Controller. All methods use the same PID controller for fair comparison in closed-loop evaluation. The PID controller module takes as input the current vehicle speed and the future trajectory predicted by the model, consisting of 10 waypoints, and outputs throttle, brake, and steering angle commands. Specifically, for the steering control, the PID gains are: $K_P^{\text{turn}} = 1.25$, $K_I^{\text{turn}} = 0.75$, $K_D^{\text{turn}} = 0.3$ For speed control, the PID gains are: $K_P^{\text{speed}} = 5.0$, $K_I^{\text{speed}} = 0.5$, $K_D^{\text{speed}} = 1.0$. The desired vehicle speed is computed from the 7th waypoint of the predicted trajectory, whereas the steering angle is determined using the 10th waypoint. This configuration ensures stable and responsive vehicle control aligned with the model's trajectory predictions.

8. Discussion on Unsupervised View Selection and Knowledge Distillation

About unsupervised view selection, unsupervised camera view learning could further reduce annotation costs and improve generalization. Our supervised annotations are inexpensive (simple heuristics from trajectory/map) and provide stable training for initial Vision MoE exploration. Nevertheless, the methods DySS [54] and LightVLA [27] offer valuable insights that could inspire image-level pruning. Potential extensions include image-level token pruning after view selection.

About Knowledge Distillation, Combining DriveMoE with knowledge distillation [14] is a promising direction—our architecture could serve as a teacher model to distill skill-specific compact students for deployment.

9. Annotation for Router

Vision Router: We developed a set of heuristic rules based on annotation information from the Bench2Drive dataset to identify special driving scenarios, enabling effective camera-view-level supervision. We select camera views contextually, defaulting to the rear view if no critical view is identified. The Camera Annotation Rules are,

- **Intersection Turning:** When the ego-vehicle is required to turn at an intersection (i.e., `is_in_junction` is true and the current command is either "turn left" or "turn right"), we annotate the front-side camera view pointing toward the intended exit of the intersection.
- **Lane Change:** When a lane change is required, identified by conditions such as the current command being "change left" or "change right," an obstacle appearing within a certain distance ahead in the current lane, or the ego-vehicle not being in the target lane, the annotation depends on lane direction:
  - If the target lane is in the same direction as the ego-vehicle's current movement, we annotate the corresponding rear-side camera.
  - If the ego-vehicle must temporarily occupy the opposing lane, we annotate the corresponding front-side camera.
- **Highway Merging and Cut-in:** In scenarios such as highway merging or vehicle cut-ins (scenario labeled as "merging" or "cut-in"), we determine the merging lo-

cation based on the ego-vehicle's lane position and distance to the junction, annotating the side camera facing the merging location.

- **Yielding to Emergency Vehicles:** If a high-speed emergency vehicle is present in the scenario, the ego-vehicle must yield, and we annotate the camera facing the direction of the approaching emergency vehicle.

Action Router: As shown in Table 8, Bench2Drive [23] divides 44 scenarios into 5 skills.

10. Experiment on nuScenes

Table 9. Open-loop planning performance in nuScenes

Method	L2 (m)↓				Collision (%)↓			
	1s	2s	3s	Avg.	1s	2s	3s	Avg.
UniAD [15]	0.48	0.74	1.07	0.76	0.12	**0.13**	0.28	**0.17**
VAD-Base [26]	**0.41**	**0.70**	**1.05**	**0.72**	**0.07**	0.17	0.41	0.22
Drive-π_0	0.51	0.73	1.11	0.78	0.14	0.18	0.39	0.24
DriveMoE	0.45	**0.70**	1.08	0.74	0.11	0.15	**0.26**	**0.17**

Following the same skill definitions as Bench2Drive, DriveMoE achieves competitive L2 error while significantly reducing collision rate (Table 9), demonstrating strong generalization to real-world scenarios. Same to [12], open-loop evaluation cannot reflect driving performance.

11. Hyperparameters & Efficienc

As shown in Table 10, the increased cost primarily stems from processing one additional camera view and the MoE modules with Top-3 expert activation. We will clarify this in our revision.

Table 10. Comparison of model scale and inference cost.

Method	View Num	View	DS↑	SR(%)↑	Parameters (M)	FLOPs (G)	Latency(ms)↓	Training Cost
Drive-π_0	2	Fixed	63.26	31.82	2606	3400	240	80 GPU-hours
Drive-π_0	6	Fixed	62.27	31.36	2606	7576	700	320 GPU-hours
DriveMoE	2	Dynamic	**74.22**	**48.64**	3008	3896	260	120/80 GPU-hours

Table 8. **Skill Set & Scenarios**

Skill	**Scenario**
Merging	CrossingBicycleFlow, EnterActorFlow, HighwayExit, InterurbanActorFlow, HighwayCutIn, InterurbanAdvancedActorFlow, MergerIntoSlowTrafficV2, MergeIntoSlowTraffic, NonSignalizedJunctionLeftTurn, NonSignalizedJunctionRightTurn, NonSignalizedJunctionLeftTurnEnterFlow, ParkingExit, LaneChange, SignalizedJunctionLeftTurn, SignalizedJunctionRightTurn, SignalizedJunctionLeftTurnEnterFlow
Overtaking	Accident, AccidentTwoWays, ConstructionObstacle, ConstructionObstacleTwoWays, HazardAtSideLaneTwoWays, HazardAtSideLane, ParkedObstacleTwoWays, ParkedObstacle, VehicleOpenDoorTwoWays
Emergency Brake	BlockedIntersection, DynamicObjectCrossing, HardBreakRoute, OppositeVehicleTakingPriority, OppositeVehicleRunningRedLight, ParkingCutIn, PedestrianCrossing, ParkingCrossingPedestrian, StaticCutIn, VehicleTurningRoute, VehicleTurningRoutePedestrian, ControlLoss
Give Way	InvadingTurn, YieldToEmergencyVehicle
Traffic Sign	EnterActorFlow, CrossingBicycleFlow, NonSignalizedJunctionLeftTurn, NonSignalizedJunctionRightTurn, NonSignalizedJunctionLeftTurnEnterFlow, OppositeVehicleTakingPriority, OppositeVehicleRunningRedLight, PedestrianCrossing, SignalizedJunctionLeftTurn, SignalizedJunctionRightTurn, SignalizedJunctionLeftTurnEnterFlow, TJunction, VanillaNonSignalizedTurn, VanillaSignalizedTurnEncounterGreenLight, VanillaSignalizedTurnEncounterRedLight, VanillaNonSignalizedTurnEncounterStopsign, VehicleTurningRoute, VehicleTurningRoutePedestrian