BOLT: BEHAVIOR-ORIENTED LEARNING FOR AUTONOMOUS DRIVING DECISION-MAKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Autonomous driving in mixed traffic requires understanding surrounding drivers' behavioral tendencies and how they affect ego decisions. However, trajectory supervision alone does not explicitly distinguish these tendencies. Appropriate decisions must also translate into motion that respects vehicle limits. We propose BOLT, a behavior-aware decision-making framework that combines explicit behavioral supervision with kinematically informed trajectory refinement. We construct nuPlan-Behavior, a behavior-annotated extension of nuPlan with motion-derived pseudo-labels. These labels supervise a continuous-time encoder with prototype memory, whose continuous behavioral representations guide surrounding-agent prediction and ego trajectory generation through cross-attention. At inference, differential-flatness-informed refinement adjusts the selected ego trajectory using motion and obstacle-proximity penalties. Evaluations on nuScenes, nuPlan, and CommonRoad demonstrate leading prediction accuracy and closed-loop decision-making performance. The results show that behavioral representations complement scene encoding of motion histories, while trajectory refinement further improves closed-loop performance in challenging scenarios.

1 INTRODUCTION

Autonomous driving (AD) systems must make safe and effective decisions in mixed traffic, where autonomous vehicles (AVs) share the same road with human-driven vehicles. A central difficulty is that these environments evolve through interactions among drivers with different behavioral tendencies. Each driver's actions can influence the choices of others, giving driving a strategic dimension that physical motion modeling alone does not capture (Cao et al., 2026). At a merge, for example, a neighboring driver may yield to create space or accelerate to claim a gap. The same geometric opportunity can therefore call for different ego responses, depending on how surrounding drivers are likely to behave (Liao et al., 2024b; 2025; Tang et al., 2026).

Recent learning-based approaches have shown strong capabilities in modeling traffic scenes and generating diverse trajectories (Tan et al., 2025; Sun et al., 2026). However, trajectory-fitting objectives alone provide no explicit supervision for distinguishing the driving tendencies underlying observed motion. Beyond describing past movement, historical trajectories provide clues about how drivers behave during interaction (Zhang et al., 2026). This raises a question: *How can AVs learn behavioral tendencies from observed motion and use them to guide their decisions in mixed traffic?*

We address this question by learning behavioral representations that connect driving history with autonomous decision-making. Motion-derived categories, such as aggressive, normal, and conservative driving, provide explicit supervision for distinguishing broad behavioral tendencies. These categories guide the learning of continuous features, which are combined with the current scene to inform both the anticipation of surrounding motion and the generation of ego actions. In this way, behavioral understanding becomes part of the decision process: evidence about how surrounding drivers have behaved helps the AV determine how to respond (Li et al., 2024).

In addition, the resulting decisions must also be translated into motion that the vehicle can execute. An appropriate response to surrounding traffic may still be represented by a trajectory that requires excessive acceleration, sharp turns, or abrupt changes in motion (Wang et al., 2025b; Rao et al., 2026b). Incorporating behavioral information does not by itself prevent these violations of vehicle motion limits. This motivates complementing behavior-informed decision-making with trajectory

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

refinement that explicitly accounts for motion limits and obstacle proximity, helping translate the intended response into more physically feasible and smoother motion.

To this end, this study propose BOLT, which brings together three complementary aspects of autonomous driving decision-making: **(1) Behavioral representation learning**, where motion-derived behavioral pseudo-labels supervise a closed-form continuous-time encoder with prototype memory to obtain continuous representations of recent driving tendencies; **(2) Behavior-informed decision-making**, where the learned features provide behavioral context for anticipating surrounding traffic and generating ego trajectories through cross-attention; and **(3) Physically informed trajectory refinement**, where optimization informed by differential flatness refines the selected ego trajectory at inference time using penalties on motion-limit violations and obstacle proximity.

Overall, the main contributions of this paper are summarized as follows:

- We construct **nuPlan-Behavior**, an extension of nuPlan-mini with behavioral pseudo-labels derived through GMM-LDA modeling, and introduce a behavior-supervised learning approach that combines continuous-time encoding with prototype memory to learn continuous behavioral representations from historical motion for decision-making.
- We propose **BOLT**, a behavior-oriented decision-making framework that uses continuous behavioral representations to guide motion forecasting and ego trajectory generation, with differential-flatness-informed refinement improving the physical feasibility of ego motion.
- We validate BOLT on nuScenes, nuPlan, and CommonRoad, demonstrating top performance in motion forecasting and closed-loop planning. Ablation studies further demonstrate the benefits of incorporating behavioral information into ego decision-making.

2 Related Work

Behavior-Aware Motion Forecasting. Human driving tendencies shape agent interactions, motivating behavior-aware trajectory prediction. BAT incorporates traffic-psychology-inspired behavioral cues (Liao et al., 2024b), while HLTP uses a cognitive-inspired teacher-student framework with adaptive perception (Liao et al., 2024a). WAKE combines wavelet modeling with kinematic information (Wang et al., 2025a), and NDPNet models near and distant horizons with kinematic constraints (Hu et al., 2026). To enrich scene context, CoT-Drive distills language-model knowledge from semantic descriptions (Liao et al., 2026), while RAG-TP retrieves scene and motion priors for multimodal prediction (Wang et al., 2026). AMD addresses rare motion patterns through adaptive momentum, contrastive learning, and clustering (Rao et al., 2025), whereas Den-TP targets underrepresented high-density scenarios through density-balanced data selection and evaluation (Yang et al., 2026). Our approach learns explicitly supervised behavior representations from a behavior-annotated extension of nuPlan and fuses them with motion features through cross-attention.

Behavior-Aware Trajectory Planning. Learning-based planners combine interaction modeling, structured supervision, and generative trajectory modeling. GameFormer uses hierarchical game-theoretic reasoning for joint prediction and planning (Huang et al., 2023). PLUTO strengthens imitation learning through structured queries and auxiliary objectives (Cheng et al., 2024a), while PlanScope separates decisions across temporal scales (Xin et al., 2026). Diffusion Planner jointly models ego and surrounding-agent trajectories with flexible behavioral guidance (Zheng et al., 2025); PlannerRFT applies closed-loop reinforcement fine-tuning (Li et al., 2026), and Flow Planner combines trajectory tokenization with flow matching (Tan et al., 2025). For personalized driving, StyleDrive provides style annotations and a benchmark (Hao et al., 2026), while SDD Planner combines style-aware encoding with dynamic diffusion guidance (Pei et al., 2026). Our framework conditions both prediction and planning on learned behavior features, followed by differential-flatness-based refinement (Han et al., 2024) to account for vehicle motion constraints, safety, and comfort.

3 nuPlan-Behavior Dataset

We introduce **nuPlan-Behavior**, a behavior-annotated extension of nuPlan-mini containing 443,218 samples. It augments the original data with scenario-level behavioral pseudo-labels for aggressive, normal, and conservative driving, providing explicit supervision for learning behavioral representations from motion history. The dataset construction consists of three steps.

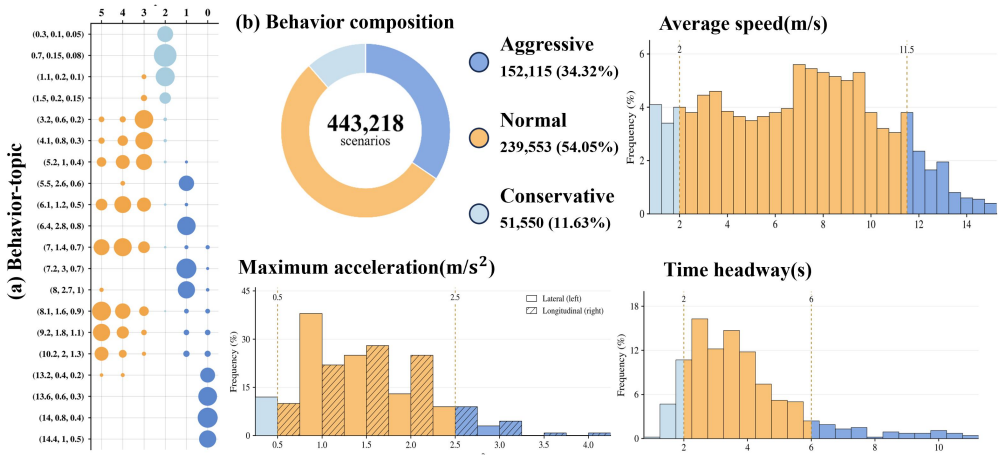

Figure 1: Dataset statistics and topic interpretation. (a) LDA topic-token distributions, with circle area proportional to token probability. Tokens encode speed and longitudinal/lateral acceleration. Topics 0-1, 2, and 3-5 represent aggressive (blue), conservative (light blue), and normal (orange) driving, respectively. (b) Behavior category counts and distributions of mean speed, time headway, and maximum longitudinal/lateral acceleration. Dashed lines mark reference thresholds.

Step 1: Motion Tokenization. We first encode motion observations as discrete tokens. For scenario d at time t, we define $\mathbf{x}_{d,t} = [v_{d,t}, a^{\text{lon}}_{d,t}, a^{\text{lat}}_{d,t}]^\top$, comprising speed, longitudinal acceleration, and lateral acceleration. We fit a Gaussian mixture model (GMM) with $K = 20$ components and assign each observation to its most probable component. Formally,

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}\big(\mathbf{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\big), \quad w_{d,t} = \arg\max_{k \in \{1,\dots,K\}} \pi_k \mathcal{N}\big(\mathbf{x}_{d,t}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\big), \tag{1}$$

Here, π_k, $\boldsymbol{\mu}_k$, and $\boldsymbol{\Sigma}_k$ denote the mixture weight, mean, and covariance of component k, respectively. The resulting token $w_{d,t}$ encodes the local motion state.

Step 2: Scenario-Level Topic Modeling. We use latent Dirichlet allocation (LDA) to summarize the composition of motion tokens within each scenario, treating scenarios as documents and tokens as words. With $L = 6$ latent topics, the generative process can be formally represented as follows:

$$\boldsymbol{\phi}_\ell \sim \text{Dir}(\boldsymbol{\beta}), \quad \boldsymbol{\theta}_d \sim \text{Dir}(\boldsymbol{\alpha}), \quad z_{d,t} \sim \text{Cat}(\boldsymbol{\theta}_d), \quad w_{d,t} \sim \text{Cat}\big(\boldsymbol{\phi}_{z_{d,t}}\big). \tag{2}$$

where $\boldsymbol{\phi}_\ell$ is the token distribution for topic $\ell \in \{0, \dots, L-1\}$, $\boldsymbol{\theta}_d$ describes the topic mixture for scene d, and $z_{d,t}$ is the latent topic assignment. The vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are Dirichlet prior parameters.

Step 3: Behavioral Interpretation and Label Assignment. We assign behavioral semantics to the inferred topics by manually examining their highest-probability motion tokens and the characteristics of the corresponding GMM components (Fig. 1(a)). In the fitted LDA model, topics 0 and 1 are mapped to aggressive driving, topic 2 to conservative driving, and topics 3, 4, and 5 to normal driving. Denoting this mapping by g, we label each scene according to its dominant inferred topic:

$$\ell^\star_d = \arg\max_{\ell \in \{0,\dots,L-1\}} \theta_{d,\ell}, \qquad y_d = g(\ell^\star_d). \tag{3}$$

The resulting dataset contains 152,115 aggressive (34.32%), 239,553 normal (54.05%), and 51,550 conservative (11.63%) samples. Fig. 1(b) summarizes the category counts and distributions of mean speed, time headway, and maximum longitudinal and lateral acceleration. For behavior classification, samples are randomly divided into training, validation, and test subsets in a 7:2:1 ratio. These annotations support the training and evaluation of the behavior encoder, whose continuous representations inform downstream decision-making. Further details are provided in Appendix A.

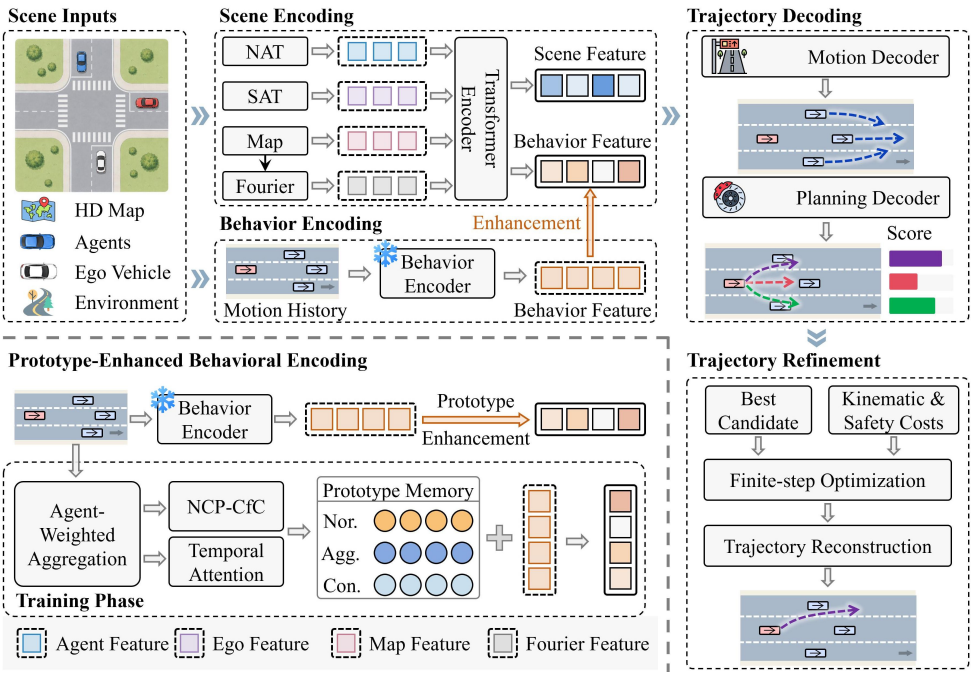

Figure 2: Overview of BOLT. Prototype-enhanced behavioral features support joint prediction and planning with differential-flatness-informed refinement. Snowflakes denote frozen encoders.

4 Methodology

4.1 Preliminaries

Problem Formulation. We study behavior-aware decision-making in mixed traffic, focusing on surrounding-agent trajectory prediction and ego trajectory generation. Consider a scene comprising an ego vehicle indexed by 0, N_a surrounding agents, and N_s static obstacles. The observed scene is $X = (\{H_i\}_{i=0}^{N_a}, \{O_s\}_{s=1}^{N_s}, M)$, where H_i contains the states of agent i over T_h historical steps, O_s describes the s-th static obstacle, and M provides vectorized map and traffic information. Given X, the model uses behavioral representations inferred from motion histories to predict surrounding-agent trajectories $\widehat{Y}_{1:N_a}$ and generate K scored ego trajectory candidates $\mathcal{G} = \{(\gamma^{(k)}, \sigma^{(k)})\}_{k=1}^{K}$ over T_f future steps. Here, $\gamma^{(k)}$ denotes the k-th ego candidate and $\sigma^{(k)}$ its selection score.

Overall Framework. Fig. 2 illustrates the pipeline of BOLT. A scene encoder captures agent motion and environmental context, while the behavior branch extracts representations $\mathbf{b}_i = B_\phi(\mathcal{H}_i)$ for the ego and surrounding motor vehicles. Here, $\mathcal{H}_i$ includes the motion history and neighboring-agent context of vehicle i. The behavior branch is pretrained on nuPlan-Behavior with prototype memory enhancement and remains frozen during downstream training. The prediction and planning decoders then integrate behavioral and scene features through cross-attention to produce surrounding-agent trajectories $\widehat{Y}_{1:N_a}$ and scored ego candidates $\mathcal{G}$, respectively. At inference, we select $\gamma^{(k^\star)}$ with $k^\star = \arg\max_k \sigma^{(k)}$ and refine it through finite-step optimization informed by differential flatness. This refinement incorporates kinematic and obstacle-proximity costs to obtain the final ego trajectory $\widehat{\gamma}$.

4.2 Prototype-Enhanced Behavioral Encoding

The behavior branch learns continuous representations through contextual temporal encoding and prototype memory enhancement (Fig. 2(b)). Behavioral pseudo-labels supervise representation learning, while the resulting features condition downstream prediction and planning.

Contextual Temporal Encoding. For focal vehicle i, let $\mathbf{h}_{n,t}^{(i)}$ denote the state of input agent n at historical step t in $\mathcal{H}_i$. We first aggregate agent states using learned importance weights and then

summarize their temporal evolution through continuous-time encoding and attention pooling:

$$\mathbf{a}_t^{(i)} = W_{\mathrm{a}} \sum_n \omega_{n,t}^{(i)} \mathbf{h}_{n,t}^{(i)} + \mathbf{b}_{\mathrm{a}}, \qquad \mathbf{z}_i = f_{\mathrm{p}} \left(\sum_{t=1}^{T_h} \beta_t^{(i)} \, \mathrm{LN}(\mathbf{r}_t^{(i)}) \right). \tag{4}$$

Here, $\omega_{n,t}^{(i)} = \frac{\exp f_{\mathrm{a}}(\mathbf{h}_{n,t}^{(i)})}{\sum_j \exp f_{\mathrm{a}}(\mathbf{h}_{j,t}^{(i)})}$ normalizes the learned scalar scores across input agents, and W_{a} and $\mathbf{b}_{\mathrm{a}}$ project the aggregated states. The temporal feature $\mathbf{r}_t^{(i)} = [\phi_{\text{NCP-CfC}}(\mathbf{a}_{1:T_h}^{(i)})]_t$ is produced by a closed-form continuous-time network (Hasani et al., 2022) with NCP wiring (Lechner et al., 2020). The operator LN applies layer normalization at each step, $\beta_t^{(i)}$ denotes an attention weight normalized over time, and f_{p} projects the pooled features into $\mathbf{z}_i$. Encoder parameters are shared across focal vehicles. Further encoding and attention details are provided in Appendix B.

Evidence-based Prototype Retrieval. Then, we freeze the feature extractor and construct a prototype $\mathbf{c}_c = |\mathcal{I}_c|^{-1} \sum_{j \in \mathcal{I}_c} \mathbf{z}_j$ for each of the $C = 3$ behavior categories, where $\mathcal{I}_c$ contains training samples of class c. A shared evidence mapping $A(\mathbf{z}) = \mathrm{ReLU}(\phi_{\mathrm{MLP}_\alpha}(\mathbf{z})) + \mathbf{1}$ parameterizes a Dirichlet distribution for each feature. We retrieve prototype information via following process:

$$d_{i,c} = D_{\mathrm{KL}}(\mathrm{Dir}(A(\mathbf{z}_i)) \,\|\, \mathrm{Dir}(A(\mathbf{c}_c))), \qquad a_{i,c} = \frac{(d_{i,c} + \epsilon_{\mathrm{KL}})^{-1}}{\sum_{j=1}^{C} (d_{i,j} + \epsilon_{\mathrm{KL}})^{-1}}. \tag{5}$$

Here, $\epsilon_{\mathrm{KL}} > 0$ ensures numerical stability. Prototypes with smaller divergence receive larger weights, yielding a continuous mixture of class-level features. The retrieved memory is fused with the history-dependent embedding through the following process:

$$\mathbf{m}_i = \sum_{c=1}^{C} a_{i,c} \mathbf{c}_c, \mathbf{b}_i = s_i \left(\mathbf{z}_i + \mathbf{g}_i \odot \mathbf{m}_i \right), \tag{6}$$

where $\mathbf{g}_i$ is a learned componentwise gate and s_i is a prototype-distance scale. Their detailed definitions, the retrieval details, and training objectives, are given in Appendix C.

4.3 Behavior-Informed Decision-Making

We integrate the learned behavioral representations with scene context to support surrounding-agent prediction and ego trajectory generation. The two tasks share the scene encoding and use separate cross-attention modules to incorporate behavioral information.

Scenario Encoding. As shown in Fig. 2(a), we encode the observed scene $X = (\{H_i\}_{i=0}^{N_a}, \{O_s\}_{s=1}^{N_s}, M)$ into ego, surrounding-agent, obstacle, and map tokens. A Neighborhood Attention Transformer (NAT) processes temporal differences in surrounding-agent histories, while a self-attention encoder processes H_0 with stochastic feature masking during training. Static obstacles are embedded by an MLP, and map features combine pointwise aggregation, semantic embeddings, and Fourier features. The resulting tokens are jointly contextualized as follows:

$$E_{\mathrm{scene}} = \phi_{\mathrm{enc}}([E_0 \| E_A \| E_O \| E_M] + \phi_{\mathrm{F}}(P)), \tag{7}$$

where E_0, E_A, E_O, and E_M denote the corresponding token embeddings with a common feature dimension d. The operator $\|$ concatenates tokens, and $\phi_{\mathrm{F}}(P)$ provides Fourier positional embeddings from their coordinates P. The surrounding-agent rows of E_{scene} form $E_{\mathrm{agent}} \in \mathbb{R}^{N_a \times d}$.

Behavioral Tokens. The behavior branch produces $\mathbf{b}_i = B_\phi(\mathcal{H}_i)$ for the ego and surrounding agents. A trainable projection P_{b} maps these representations to dimension d, yielding token sets:

$$S_{\mathrm{veh}} = [P_{\mathrm{b}} \mathbf{b}_i]_{i \in \mathcal{V}}, \qquad S_{\mathrm{all}} = [P_{\mathrm{b}} \mathbf{b}_i]_{i \in \{0\} \cup \mathcal{V}}, \tag{8}$$

where $\mathcal{V} \subseteq \{1, \ldots, N_a\}$ indexes surrounding motor vehicles, and the projected vectors are stacked along the token dimension. Prediction uses S_{veh}, while ego trajectory generation uses S_{all} to access both ego and surrounding-vehicle behavior. Task-specific projection indices are omitted for brevity.

Behavior-Conditioned Prediction. As illustrated in Fig. 2(c), each contextualized agent token queries the surrounding-vehicle behavior tokens to retrieve behavioral information:

$$\widehat{Y}_{1:N_a} = \phi_{\mathrm{pred}}(\phi_{\mathrm{CA}}(E_{\mathrm{agent}}, S_{\mathrm{veh}}, S_{\mathrm{veh}})). \tag{9}$$

Here, ϕ_{CA} denotes multi-head cross-attention, with its arguments specifying queries, keys, and values, respectively. Each agent can attend to multiple vehicle behavior tokens. The prediction outputs include pedestrians and bicycles, although these agents are not assigned behavior labels.

Behavior-Conditioned Planning. Reference-lane features and learnable longitudinal queries define K initial candidate queries $Q_{\mathrm{init}} \in \mathbb{R}^{K\times d}$. These queries first retrieve behavioral information from S_{all} and then interact with the scene representation. Formally,

$$Q_{\mathrm{beh}} = Q_{\mathrm{init}} + \eta\, \phi_{\mathrm{CA}}(Q_{\mathrm{init}}, S_{\mathrm{all}}, S_{\mathrm{all}}), \quad Z_{\mathrm{plan}} = \phi_{\mathrm{dec}}(Q_{\mathrm{beh}}, E_{\mathrm{scene}}). \tag{10}$$

The coefficient η scales the residual behavioral contribution. Prediction and planning use separate cross-attention parameters. The decoder ϕ_{dec} combines query self-attention with cross-attention to the scene tokens. Its outputs are mapped to candidate trajectories and scores:

$$(\gamma^{(k)}, \sigma^{(k)}) = (\phi_{\mathrm{traj}}(Z_{\mathrm{plan},k}), \phi_{\mathrm{score}}(Z_{\mathrm{plan},k}))\,, \qquad k = 1, \ldots, K. \tag{11}$$

The resulting candidates form $\mathcal{G}$ and represent planning modes. The three behavior categories serve as supervision for behavior pretraining. More details for this module are provided in Appendix D.

4.4 Kinematically Informed Trajectory Refinement

We refine the highest-scoring candidate $\gamma^{(k^\star)}$, where $k^\star = \arg\max_k \sigma^{(k)}$, using a speed-heading parameterization informed by the kinematic bicycle model's differential-flatness relations. The candidate initializes $\mathbf{u} = \{(v_t, \theta_t)\}_{t=1}^{T_f}$. For $v_t > 0$ and $t = 1, \ldots, T_f - 1$, the estimates are as follows:

$$a_t^{\mathrm{lon}} = \frac{v_{t+1} - v_t}{\Delta t}, \quad \kappa_t = \frac{\Delta\theta_t}{v_t \Delta t}, \quad a_t^{\mathrm{lat}} = v_t^2 \kappa_t, \quad \delta_t = \arctan(L\kappa_t), \tag{12}$$

where L is the wheelbase, Δt is the sampling interval, and $\Delta\theta_t = \mathrm{atan2}(\sin(\theta_{t+1} - \theta_t), \cos(\theta_{t+1} - \theta_t))$ is the wrapped heading increment. These relations distinguish geometric turning limits from lateral acceleration. The same sequence reconstructs ego positions by discrete integration:

$$\mathbf{p}_t = \mathbf{p}_0 + \Delta t \sum_{\tau=1}^{t} v_\tau [\cos\theta_\tau, \sin\theta_\tau]^\top, \qquad t = 1, \ldots, T_f, \tag{13}$$

where $\mathbf{p}_0$ is the observed ego position. We then update $\mathbf{u}$ for a fixed number of optimization steps to minimize $J(\mathbf{u}; X, \widehat{Y}) = \mathcal{L}_{\mathrm{lon}} + \mathcal{L}_{\mathrm{lat}} + \mathcal{L}_{\mathrm{curv}} + \mathcal{L}_{\mathrm{jerk}} + \mathcal{L}_{\mathrm{speed}} + \mathcal{L}_{\mathrm{static}} + \mathcal{L}_{\mathrm{dynamic}}$. The first five terms penalize acceleration and curvature violations, jerk, and speeding; the last two penalize proximity to static obstacles and predicted agents. Refinement yields $\widehat{\gamma}$ while keeping network parameters and agent forecasts fixed. Cost definitions appear in Appendix E.

5 Experiments

5.1 Experimental Setup

Benchmarks. We pretrain the behavior encoder on nuPlan-Behavior and evaluate BOLT on nuScenes (Caesar et al., 2020) for prediction, nuPlan (Caesar et al., 2022) (Val14, Test14, and Test14-hard) for closed-loop planning, and CommonRoad (Althoff et al., 2017) for simulation.

Evaluation Metrics. We report minimum average and final displacement errors (minADE and minFDE) and miss rate (MR) for prediction. For nuPlan planning, we report closed-loop scores under non-reactive (CLS-NR) and reactive (CLS-R) simulation (Caesar et al., 2022).

Implementation Details. Experiments use a server with two NVIDIA A100 GPUs. The downstream model is trained for 25 epochs, including three warm-up epochs, with a batch size of 32 and an initial learning rate of 10^{-3}. The pretrained behavior branch remains frozen, and trajectory refinement is applied only at inference. Further details are provided in Appendices F-H.

5.2 Comparison to State-of-the-arts (SOTA) Methods

Behavior Recognition. On nuPlan-Behavior, the full behavior model (CfC+M) achieves 94.8% accuracy against the behavioral pseudo-labels, compared with 81.1% for a two-layer MLP and 94.4% for CfC without prototype memory. Detailed comparisons and analysis are provided in Appendix G.

Table 1: Trajectory prediction results on nuScenes dataset (lower is better). Bold and underlined values indicate the best and second-best results, respectively, with ties marked identically. The behavior-feature ablation (w/o BF) is excluded from ranking. "-" denotes an unreported value.

Model	Venue	$\text{minADE}_5 \downarrow$	$\text{minADE}_{10} \downarrow$	$\text{minFDE}_1 \downarrow$	$\text{minFDE}_5 \downarrow$	$\text{MR}_5 \downarrow$
AutoBot Girgis et al. (2022)	ICLR 2022	1.37	1.03	8.19	3.40	0.62
LaPred Kim et al. (2021)	CVPR 2021	1.47	1.12	8.12	3.37	0.53
PGP (Deo et al., 2022)	PMLR 2022	1.30	1.03	7.17	2.52	0.61
GoHome (Gilles et al., 2022)	ICRA 2022	1.42	1.15	6.99	-	0.57
EMSIN (Ren et al., 2024)	IEEE TFS 2024	1.77	1.28	9.06	3.56	0.54
WAKE (Wang et al., 2025a)	TPAMI 2025	1.24	1.09	7.02	2.96	0.55
AMD (Rao et al., 2025)	ICCV 2025	1.23	1.06	6.99	2.43	0.50
NEST (Wang et al., 2025c)	AAAI 2025	1.18	-	6.87	2.39	0.50
SeFlow (Zhang et al., 2024)	ECCV 2024	1.38	**0.98**	7.89	-	0.60
SAIL (Rao et al., 2026a)	TR-C 2026	1.18	1.03	**6.45**	2.42	0.50
BOLT (w/o BF)	-	1.21	1.02	6.81	2.35	0.51
BOLT	-	**1.17**	1.00	6.62	**2.33**	**0.48**

Motion Forecasting. As shown in Table 1, BOLT achieves the best minADE_5, minFDE_5, and MR_5, and ranks second in minADE_{10} and minFDE_1 among the compared methods on nuScenes. Relative to the best baseline values, it reduces minFDE_5 and MR_5 by 2.5% and 4.0%, respectively. Compared with BOLT (w/o BF), incorporating behavioral features improves all five metrics, including a 5.9% reduction in MR_5. These improvements support the use of behavioral representations learned from motion histories as complementary context for predicting surrounding-agent trajectories.

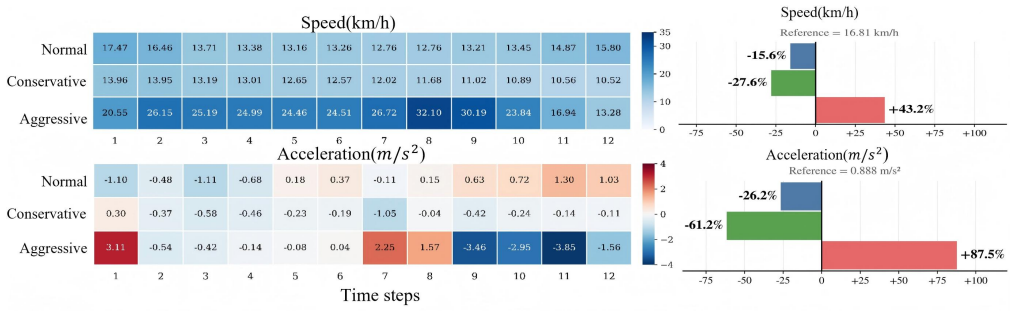

Figure 3: Ego motion profiles during turning under normal, conservative, and aggressive initializations in CommonRoad. Left: heatmaps of recorded speed and acceleration; Right: percentage deviations of mean speed and mean absolute acceleration from their respective pooled means.

Closed-loop Decision-Making. As shown in Table 2, BOLT improves CLS-R and CLS-NR by 5.15% and 2.34% over CarPlanner (Zhang et al., 2025) on Val14, the strongest learned baseline in this comparison. It also outperforms imitation-learning planners PlanTF (Cheng et al., 2024b) and PLUTO (Cheng et al., 2024a), prediction-based GC-PGP (Hallgarten et al., 2023), interaction-aware GameFormer (Huang et al., 2023), and occupancy-guided OccDriver (Huang et al., 2026) in both settings. Against rule-based baselines, BOLT surpasses IDM (Treiber et al., 2000) and achieves higher CLS-NR than PDM-Closed (Dauner et al., 2023), while closely matching the latter's CLS-R. Beyond aggregate scores, BOLT exceeds the best baseline TTC and progress scores by 1.08% and 1.37%, respectively. These concurrent gains indicate strong safety-related performance alongside progress, consistent with combining behavioral context and kinematically informed refinement. As shown in Table 3, BOLT ranks among the top two planners across all four evaluations on Test14 and Test14-hard subsets, outperforming raster-based RasterModel (Karnchanachari et al., 2024) and vector-based UrbanDriver (Scheel et al., 2022) in both settings. CommonRoad simulations further show lower speeds and milder acceleration changes under conservative initialization, and earlier junction traversal with stronger acceleration and braking under aggressive initialization.

Joint Performance in Challenging Driving Scenarios. Table 3 reports closed-loop results on Test14-hard and Test14. BOLT ranks among the top two planners across all four evaluations, achieving the best results. It leads both CLS-R and CLS-NR on Test14-hard, while ranking first in CLS-R and second in CLS-NR on Test14. These results demonstrate competitive decision-making performance across standard and challenging scenarios under both reactive and non-reactive simulation.

Table 2: Closed-loop planning results on the *Val14* split of the nuPlan benchmark. We report the overall closed-loop score under the reactive setting and the non-reactive setting. **Bold** and underlined text indicate the best and second-best results, respectively.

Planner	CLS-R	CLS-NR	Collisions	TTC	Drivable	Comfort	Progress	Speed
Log-Replay	81.24	93.68	98.76	94.40	98.07	99.27	98.99	96.47
PDM-Open	54.24	53.53	74.54	69.08	87.89	**99.54**	69.86	97.72
PDM-Closed	**92.12**	92.84	98.07	93.30	**99.82**	95.52	92.13	**99.83**
GC-PGP	51.39	61.09	85.87	80.18	89.72	90.00	60.32	99.34
GameFormer	79.78	82.95	94.32	86.77	94.87	93.39	89.04	98.67
RasterModel	64.37	66.92	86.97	81.46	85.04	81.46	80.60	98.03
IDM	77.33	79.31	90.92	83.49	94.04	94.40	86.16	97.33
PlanTF	76.95	85.30	94.13	90.73	96.79	93.67	89.83	97.78
Pluto	80.01	91.32	96.18	93.85	98.53	91.93	87.13	98.13
CarPlanner	87.56	91.45	96.38	91.42	97.73	93.17	91.26	95.48
OccDriver	83.8	89.6	97.1	93.8	-	96.9	88.5	-
BOLT	92.07	**93.59**	**98.42**	**94.86**	**99.82**	95.85	**93.39**	98.62
BOLT (w/o BF)	91.97	92.17	98.24	94.61	99.74	94.69	93.15	98.61
BOLT (w/o DFTR)	90.04	91.41	97.86	94.07	99.46	92.30	92.74	98.37

Table 3: NuPlan results on Test14-hard and Test14. Bold and underlined values indicate the best and second-best planners. BF: Behavioral feature; DFTR: Differential-Flatness-Informed Trajectory Refinement.

Planner	Test14-hard		Test14	
	CLS-R↑	CLS-NR↑	CLS-R↑	CLS-NR↑
GC-PGP	39.63	43.22	51.39	55.99
RasterModel	52.16	49.47	67.54	69.66
PDM-Open	35.83	33.51	57.23	52.80
UrbanDriver	49.95	50.40	61.02	63.27
PDM-Closed	75.19	65.08	91.64	90.05
PDM-Hybrid	76.07	65.99	91.28	90.10
GameFormer	67.05	68.70	79.31	80.80
IDM	62.26	56.15	72.42	70.39
PlanTF	61.61	69.70	80.59	86.48
PLUTO	76.88	80.08	90.03	91.92
CarPlanner	-	-	91.10	**94.07**
BOLT	**76.92**	**81.05**	**91.73**	93.87
BOLT (w/o BF)	75.87	80.92	91.14	92.83
BOLT (w/o DFTR)	72.73	77.28	89.61	91.92

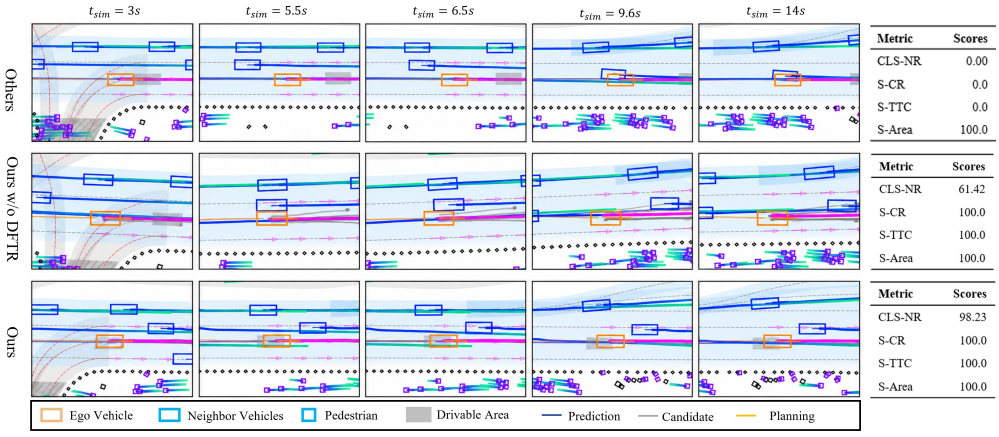

Figure 4: Closed-loop comparison in nuPlan: PLUTO (Others), BOLT without DFTR, and full BOLT (top to bottom). Columns show successive times, with scores reported on the right.

5.3 Ablation Studies

Effect of Behavior Features. Adding behavior features improves all five nuScenes prediction metrics (Table 1), including reductions of 3.3% in minADE_5 and 5.9% in MR_5 relative to BOLT (w/o BF). Both closed-loop scores also improve across Val14, Test14, and Test14-hard (Tables 2 and 3), although the gains vary across splits and simulation modes. Since motion histories are already available to the scene encoder, these improvements support the value of extracting explicit behavioral representations as complementary context for prediction and decision-making.

Effect of Trajectory Refinement. With behavior features retained, DFTR improves both closed-loop scores on all three splits (Tables 2 and 3). Its largest gains occur on Test14-hard, where CLS-R and CLS-NR improve by 5.8% and 4.9%, respectively. On Val14, higher comfort and safety-related scores align with its motion and obstacle-proximity costs. With network parameters and surrounding-agent forecasts fixed, these results highlight the value of refining the selected ego trajectory at inference. Together, the ablations support contributions at two stages: behavioral context guides trajectory generation, while refinement improves closed-loop performance.

Effects of Ego Initialization. We evaluate BOLT under normal, conservative, and aggressive ego initializations at the same CommonRoad junction, varying initial speed and acceleration (Fig. 5). The aggressive case reaches the outgoing road earliest, followed by the normal and conservative

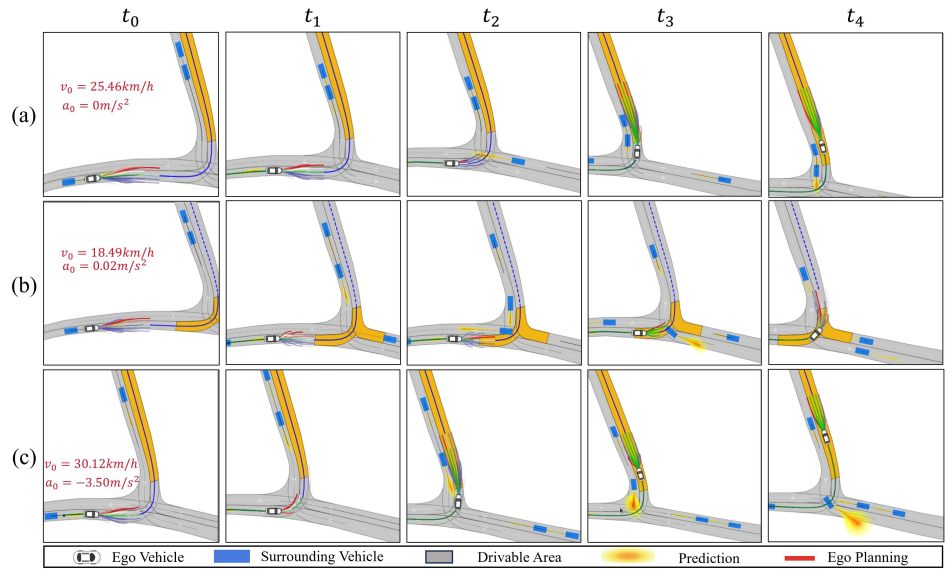

Figure 5: Visualization of BOLT at a CommonRoad junction under (a) normal, (b) conservative, and (c) aggressive initializations. Surrounding-agent predictions and ego plans are shown from t_0 to t_4.

cases. During the turn (Fig. 3), the conservative case maintains lower speeds and milder acceleration changes, while the aggressive case shows higher speeds over most of the interval and stronger acceleration and braking, with the normal case generally in between. These observations illustrate how initial motion conditions shape traversal timing and motion profiles within the same road layout.

5.4 Qualitative Comparison

Fig. 4 compares PLUTO, BOLT without DFTR, and full BOLT. Both BOLT variants attain full collision-avoidance, TTC, and drivable-area scores, whereas PLUTO receives zero collision-avoidance and TTC scores. With behavioral conditioning retained, DFTR further improves CLS-NR by 59.9%. This gain shows that refining the selected trajectory with motion and proximity costs can improve overall driving performance even when the displayed safety indicators have reached their maximum. In CommonRoad (Fig. 5), the aggressive case reaches the outgoing lane by t_2, while the conservative case is still turning at t_4, with the normal case progressing between them. These differences highlight the timing and pace of maneuver execution within a shared road layout: the same turning task admits distinct motion profiles under different ego initializations. The behavior-feature ablations complement these observations by showing that learned behavioral representations improve prediction and planning even when motion histories are already available to the scene encoder. Together, these results support extracting behavioral context from observed motion and refining the resulting ego trajectories for execution. More visualizations appear in Appendix H.

6 Conclusion

This work examines how behavioral representations and trajectory refinement support autonomous driving decision-making in mixed traffic. To this end, we propose BOLT, a behavior-aware framework that integrates explicit behavioral supervision with kinematically informed trajectory refinement. We construct nuPlan-Behavior through GMM-LDA-based pseudo-labeling and develop a closed-form continuous-time encoder with prototype memory to extract continuous behavioral features. Cross-attention integrates these features with scene context for surrounding-agent prediction and ego trajectory generation. A differential-flatness-informed optimization module further refines the selected ego trajectory using motion and obstacle-proximity costs. Experiments support the effectiveness of this design for prediction and closed-loop decision-making. Current limitations include coarse behavioral supervision and simulation-based closed-loop evaluation. Future research directions include richer behavioral annotations and validation in real-world driving environments.

7 AI USE STATEMENT

We used generative AI tools to assist with literature searches and to improve the clarity and readability of the manuscript. The authors verified the cited references and reviewed all AI-assisted text for accuracy and consistency with the methods and experimental findings. We take full responsibility for the final content of this paper, including its claims, results, and references.

8 ETHICS STATEMENT

This study uses publicly available datasets and evaluates the proposed approach using offline benchmarks and simulation. Although the research aims to improve behavior-aware autonomous driving, the reported results do not establish safety guarantees for real-world deployment. Further validation under diverse traffic conditions and appropriate safety assessment are necessary before practical use.

9 REPRODUCIBILITY STATEMENT

The main paper describes the proposed architecture, learning objectives, and evaluation protocols. The appendix provides additional implementation details, including data preprocessing, training hyperparameters, and hardware configurations. All datasets used in this study are publicly available. We will publicly release our code upon acceptance to facilitate reproduction of the reported results.

REFERENCES

Matthias Althoff, Markus Koschi, and Stefanie Manzinger. Commonroad: Composable benchmarks for motion planning on roads. In *Proc. of the IEEE Intelligent Vehicles Symposium*, pp. 719–726, 2017.

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles, 2022. URL https://arxiv.org/abs/2106.11810.

Yuxiao Cao, Yue Duan, Yucong Wang, Xiangrui Zeng, and Zhouping Yin. Interactive game-theoretic decision with self-evolution risk assessment for autonomous vehicles: A cut-in case study in dense traffic scenarios. *Transportation Research Part C: Emerging Technologies*, 186: 105621, 2026. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2026.105621. URL https://www.sciencedirect.com/science/article/pii/S0968090X26001099.

Jie Cheng, Yingbing Chen, and Qifeng Chen. Pluto: Pushing the limit of imitation learning-based planning for autonomous driving. *arXiv preprint arXiv:2404.14327*, 2024a.

Jie Cheng, Yingbing Chen, Xiaodong Mei, Bowen Yang, Bo Li, and Ming Liu. Rethinking imitation-based planners for autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 14123–14130. IEEE, 2024b.

Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with misconceptions about learning-based vehicle motion planning. In *Conference on Robot Learning*, pp. 1268–1281. PMLR, 2023.

Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In *Conference on Robot Learning*, pp. 203–212. PMLR, 2022.

Thomas Gilles, Stefano Sabatini, Dzmitry Tsishkou, Bogdan Stanciulescu, and Fabien Moutarde. Gohome: Graph-oriented heatmap output for future motion estimation. In *2022 international conference on robotics and automation (ICRA)*, pp. 9107–9114. IEEE, 2022.

Roger Girgis, Florian Golemo, Felipe Codevilla, Martin Weiss, Jim Aldon D'Souza, Samira Ebrahimi Kahou, Felix Heide, and Christopher Pal. Latent variable sequential set transformers for joint multi-agent motion prediction. In *International Conference on Learning Representations*, 2022.

Marcel Hallgarten, Martin Stoll, and Andreas Zell. From prediction to planning with goal conditioned lane graph traversals. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 951–958. IEEE, 2023.

Zhichao Han, Yuwei Wu, Tong Li, Lu Zhang, Liuao Pei, Long Xu, Chengyang Li, Changjia Ma, Chao Xu, Shaojie Shen, and Fei Gao. An efficient spatial-temporal trajectory planner for autonomous vehicles in unstructured environments. *IEEE Transactions on Intelligent Transportation Systems*, 25(2):1797–1814, 2024. doi: 10.1109/TITS.2023.3315320.

Ruiyang Hao, Bowen Jing, Haibao Yu, and Zaiqing Nie. Styledrive: Towards driving-style aware benchmarking of end-to-end autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 4627–4635, 2026.

Ramin Hasani, Mathias Lechner, Alexander Amini, Lucas Liebenwein, Aaron Ray, Max Tschaikowski, Gerald Teschl, and Daniela Rus. Closed-form continuous-time neural networks. *Nature Machine Intelligence*, November 2022. ISSN 2522-5839.

Hua Hu, Zikang Zhou, Qian Zhou, Zihao Wen, Junjie Hu, Xinhong Chen, Zhengmin Jiang, Yung-Hui Li, and Jianping Wang. Perceiving the near, reasoning the distant: Coherent long-horizon trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24875–24884, June 2026.

Zhao Huang, Bowen Zhang, Zhongzhu Li, and Di Lin. Occdriver: Future occupancy guided dual-branch trajectory planner in autonomous driving. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 141940–141962, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e59bf9a9077c361ed57180fc79c5b8d8-Paper-Conference.pdf.

Zhiyu Huang, Haochen Liu, and Chen Lv. Gameformer: Game-theoretic modeling and learning of transformer-based interactive prediction and planning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3903–3913, October 2023.

Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 629–636. IEEE, 2024.

ByeoungDo Kim, Seong Hyeon Park, Seokhwan Lee, Elbek Khoshimjonov, Dongsuk Kum, Junsoo Kim, Jeong Soo Kim, and Jun Won Choi. Lapred: Lane-aware prediction of multi-modal future trajectories of dynamic agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14636–14645, 2021.

Mathias Lechner, Ramin Hasani, Alexander Amini, Thomas A Henzinger, Daniela Rus, and Radu Grosu. Neural circuit policies enabling auditable autonomy. *Nature Machine Intelligence*, 2(10): 642–652, 2020.

Hongchen Li, Tianyu Li, Jiazhi Yang, Haochen Tian, Caojun Wang, Lei Shi, Mingyang Shang, Zengrong Lin, Gaoqiang Wu, Zhihui Hao, Xianpeng Lang, Jia Hu, and Hongyang Li. Plannerrft: Reinforcing diffusion planners through closed-loop and sample-efficient fine-tuning. *arXiv preprint arXiv:2601.12901*, 2026.

Zhenning Li, Zhiyong Cui, Haicheng Liao, John Ash, Guohui Zhang, Chengzhong Xu, and Yinhai Wang. Steering the future: Redefining intelligent transportation systems with foundation models. *Chain*, 1(1):46–53, 2024.

Haicheng Liao, Yongkang Li, Zhenning Li, Chengyue Wang, Zhiyong Cui, Shengbo Eben Li, and Chengzhong Xu. A cognitive-based trajectory prediction approach for autonomous driving. *IEEE Transactions on Intelligent Vehicles*, 9(4):4632–4643, 2024a. doi: 10.1109/TIV.2024.3376074.

Haicheng Liao, Zhenning Li, Huanming Shen, Wenxuan Zeng, Dongping Liao, Guofa Li, and Chengzhong Xu. Bat: Behavior-aware human-like trajectory prediction for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 10332–10340, 2024b.

Haicheng Liao, Zhenning Li, Kaiqun Zhu, Keqiang Li, and Chengzhong Xu. Sa-tpˆ{2}: A safety-aware trajectory prediction and planning model for autonomous driving. *IEEE Transactions on Robotics*, 2025.

Haicheng Liao, Hanlin Kong, Bonan Wang, Chengyue Wang, Kanye Ye Wang, Zhengbing He, Chengzhong Xu, and Zhenning Li. Cot-drive: Efficient motion forecasting for autonomous driving with llms and chain-of-thought prompting. *IEEE Transactions on Artificial Intelligence*, 7(2): 625–641, 2026. doi: 10.1109/TAI.2025.3564594.

Shuo Pei, Yong Wang, Yuanchen Zhu, Chen Sun, Qin Li, Yanan Zhao, and Huachun Tan. Safe and stylized trajectory planning for autonomous driving via diffusion model, 2026. URL https://arxiv.org/abs/2602.04329.

Bin Rao, Haicheng Liao, Yanchen Guan, Chengyue Wang, Bonan Wang, Jiaxun Zhang, and Zhenning Li. Amd: Adaptive momentum and decoupled contrastive learning framework for robust long-tail trajectory prediction. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 28849–28858, 2025. doi: 10.1109/ICCV51701.2025.02679.

Bin Rao, Haicheng Liao, Chengyue Wang, Keqiang Li, Zhenning Li, and Hai Yang. Sail: Scene-aware adaptive iterative learning for long-tail trajectory prediction in autonomous vehicles. *Transportation Research Part C: Emerging Technologies*, 190:105757, 2026a. ISSN 0968-090X. doi: https://doi.org/10.1016/j.trc.2026.105757. URL https://www.sciencedirect.com/science/article/pii/S0968090X26002457.

Bin Rao, Chengyue Wang, Haicheng Liao, Qianfang Wang, Yanchen Guan, Jiaxun Zhang, Xingcheng Liu, Meixin Zhu, Kanye Ye Wang, and Zhenning Li. Differentiable semantic meta-learning framework for long-tail motion forecasting in autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 962–970, 2026b.

Yilong Ren, Zhengxing Lan, Lingshan Liu, and Haiyang Yu. Emsin: Enhanced multistream interaction network for vehicle trajectory prediction. *IEEE Transactions on Fuzzy Systems*, 33(1):54–68, 2024.

Oliver Scheel, Luca Bergamini, Maciej Wolczyk, Błażej Osiński, and Peter Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients. In *Conference on Robot Learning*, pp. 718–728. PMLR, 2022.

Jiawei Sun, Xibin Yue, Jiahui Li, Tianle Shen, Chengran Yuan, Shuo Sun, Sheng Guo, Quanyun Zhou, and Marcelo H Ang Jr. Impact: Behavioral intention-aware multimodal trajectory prediction with adaptive context trimming. *IEEE Robotics and Automation Letters*, 11(1):610–617, 2026. doi: 10.1109/LRA.2025.3634880.

Tianyi Tan, Yinan Zheng, Ruiming Liang, Zexu Wang, Kexin Zheng, Jinliang Zheng, Jianxiong Li, Xianyuan Zhan, and Jingjing Liu. Flow matching-based autonomous driving planning with advanced interactive behavior modeling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Yihong Tang, Haicheng Liao, Tong Nie, Junlin He, Ao Qu, Kehua Chen, Wei Ma, Zhenning Li, Lijun Sun, and Chengzhong Xu. E3ad: An emotion-aware vision-language-action model for human-centric end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10610–10620, 2026.

Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

Chengyue Wang, Haicheng Liao, Zhenning Li, and Chengzhong Xu. Wake: Towards robust and physically feasible trajectory prediction for autonomous vehicles with wavelet and kinematics synergy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):3126–3140, 2025a. doi: 10.1109/TPAMI.2025.3529259.

Chengyue Wang, Haicheng Liao, Zhenning Li, and Chengzhong Xu. Wake: Towards robust and physically feasible trajectory prediction for autonomous vehicles with wavelet and kinematics synergy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(4):3126–3140, 2025b.

Chengyue Wang, Haicheng Liao, Bonan Wang, Yanchen Guan, Bin Rao, Ziyuan Pu, Zhiyong Cui, Cheng-Zhong Xu, and Zhenning Li. Nest: A neuromodulated small-world hypergraph trajectory prediction model for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 808–816, 2025c.

Ziyi Wang, Yang Zhang, Guijian Tang, Chao Zhang, Shibo Zhang, Xueqiong Li, and Shaowu Yang. Rag-tp: A general framework for vehicle trajectory prediction via retrieval-augmented generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24865–24874, June 2026.

Ren Xin, Jie Cheng, Hongji Liu, and Jun Ma. Planscope: Learning to plan within decision scope for urban autonomous driving. *IEEE Robotics and Automation Letters*, 11(3):3246–3253, 2026. doi: 10.1109/LRA.2026.3655195.

Ruining Yang, Yi Xu, Yun Fu, and Lili Su. Den-tp: A density-balanced data curation and evaluation framework for trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10632–10641, June 2026.

Dehai Zhang, Mingzhe Zhang, Ping Wang, Fengxiang Guo, Bowen Chen, and Zheng Chen. Style-modulated risk field for interaction-aware trajectory prediction in autonomous driving. *IEEE Transactions on Vehicular Technology*, pp. 1–13, 2026. doi: 10.1109/TVT.2026.3724901.

Dongkun Zhang, Jiaming Liang, Ke Guo, Sha Lu, Qi Wang, Rong Xiong, Zhenwei Miao, and Yue Wang. Carplanner: Consistent auto-regressive trajectory planning for large-scale reinforcement learning in autonomous driving. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 17239–17248, 2025.

Qingwen Zhang, Yi Yang, Peizheng Li, Olov Andersson, and Patric Jensfelt. SeFlow: A self-supervised scene flow method in autonomous driving. In *European Conference on Computer Vision (ECCV)*, pp. 353–369. Springer, 2024. doi: 10.1007/978-3-031-73232-4_20.

Yinan Zheng, Ruiming Liang, Kexin ZHENG, Jinliang Zheng, Liyuan Mao, Jianxiong Li, Weihao Gu, Rui Ai, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Diffusion-based planning for autonomous driving with flexible guidance. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=wM2sfVgMDH.

A Behavior Dataset Construction Details

A.1 Motion features and annotation unit

We construct behavioral annotations from the motion histories of nuPlan-mini. Each scenario d contains T_d observations of speed, longitudinal acceleration, and lateral acceleration:

$$\begin{aligned} \mathbf{X}_d &= (\mathbf{x}_{d,1}, \ldots, \mathbf{x}_{d,T_d}), \\ \mathbf{x}_{d,t} &= [v_{d,t}, a_{d,t}^{\text{lon}}, a_{d,t}^{\text{lat}}]^\top. \end{aligned} \tag{14}$$

These features characterize the observed motion within each sampled period. Labels are assigned at the scenario level and do not represent persistent characteristics of individual drivers.

A.2 Motion tokenization

We fit the $K = 20$ component GMM defined in Eq. equation 1 using expectation maximization. For each observation, the posterior responsibility of component k is computed as follows:

$$r_{d,t,k} = \frac{\pi_k \mathcal{N}(\mathbf{x}_{d,t}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_{d,t}; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{15}$$

We then assign each observation to the component with the largest responsibility and construct the discrete token sequence as follows:

$$\begin{aligned} w_{d,t} &= \arg \max_{k \in \{1,\ldots,K\}} r_{d,t,k}, \\ \mathbf{w}_d &= (w_{d,1}, \ldots, w_{d,T_d}). \end{aligned} \tag{16}$$

The responsibility denominator is shared across components, so this assignment is equivalent to Eq. equation 1. The resulting vocabulary describes local motion states; behavioral categories are assigned after topic inference.

A.3 Latent topic inference

We apply the LDA model in Eq. equation 2, treating each scenario as a document and its motion tokens as words. The topic and token index sets are $\mathcal{L} = \{0, \ldots, L-1\}$ and $\mathcal{K} = \{1, \ldots, K\}$, respectively, with $L = 6$ and $K = 20$. For scenario d, $\boldsymbol{\theta}_d$ denotes the topic distribution, while $\boldsymbol{\phi}_\ell$ denotes the token distribution of topic ℓ. Their Dirichlet prior parameters are $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_{L-1})$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_K)$, respectively, with strictly positive entries. LDA models token composition without explicitly representing temporal order.

Let $z_{d,t} \in \mathcal{L}$ denote the latent topic assigned to token $w_{d,t}$. Under collapsed Gibbs sampling, $\boldsymbol{\theta}_d$ and $\boldsymbol{\phi}_\ell$ are integrated out, and each topic assignment is sampled from the following conditional distribution:

$$\begin{aligned} &p\big(z_{d,t} = \ell \mid \mathbf{z}_{\neg(d,t)}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta}\big) \\ &\quad \propto \Big(n_{d,\ell}^{\neg(d,t)} + \alpha_\ell\Big) \, \frac{n_{\ell,w_{d,t}}^{\neg(d,t)} + \beta_{w_{d,t}}}{n_{\ell,\cdot}^{\neg(d,t)} + \sum_{k=1}^{K} \beta_k}. \end{aligned} \tag{17}$$

Here, $\mathbf{w}$ denotes all observed tokens, and $\mathbf{z}_{\neg(d,t)}$ denotes all topic assignments except that of the current token. The count $n_{d,\ell}$ records the number of tokens assigned to topic ℓ in scenario d, while $n_{\ell,k}$ records the number of occurrences of token k assigned to topic ℓ across all scenarios. The superscript $\neg(d,t)$ indicates that the current token is excluded from the corresponding counts. The total count for topic ℓ and the number of tokens in scenario d are given by:

$$n_{\ell,\cdot} = \sum_{k=1}^{K} n_{\ell,k}, \qquad T_d = \sum_{\ell=0}^{L-1} n_{d,\ell}. \tag{18}$$

The two factors in Eq. equation 17 favor topics that are prevalent within the scenario and assign high probability to the observed token, respectively. The resulting weights are normalized over $\ell \in \mathcal{L}$ to obtain the sampling probabilities.

Given the sampled topic assignments, the conditional posterior means of the scenario-specific topic distributions and the topic-specific token distributions are computed as follows:

$$\begin{aligned} \widehat{\theta}_{d,\ell} &= \frac{n_{d,\ell} + \alpha_\ell}{T_d + \sum_{j=0}^{L-1} \alpha_j}, \\ \widehat{\phi}_{\ell,k} &= \frac{n_{\ell,k} + \beta_k}{n_{\ell,\cdot} + \sum_{j=1}^{K} \beta_j}. \end{aligned} \tag{19}$$

Here, $\widehat{\theta}_{d,\ell}$ represents the estimated proportion of topic ℓ in scenario d, and $\widehat{\phi}_{\ell,k}$ represents the estimated probability of token k under topic ℓ. These distributions are used for dominant-topic selection and semantic interpretation, respectively. For notational simplicity, the estimation hats are omitted in the main text.

A.4 Topic interpretation and label assignment

We interpret each topic using its token probabilities and the motion characteristics of the corresponding GMM components. Topic 0 places substantial probability on higher-speed tokens, whereas Topic 2 is dominated by near-stationary motion. Topic 1 includes tokens reflecting longitudinal and lateral motion changes and is assigned to the aggressive category through semantic interpretation.

The resulting topic-to-category mapping is

$$g(\ell) = \begin{cases} \text{aggressive}, & \ell \in \{0, 1\}, \\ \text{conservative}, & \ell = 2, \\ \text{normal}, & \ell \in \{3, 4, 5\}. \end{cases} \tag{20}$$

For each scenario, we first identify the topic with the largest estimated probability and then apply the semantic mapping:

$$\begin{aligned} \ell_d^\star &= \arg\max_{\ell \in \mathcal{L}} \widehat{\theta}_{d,\ell}, \\ y_d &= g(\ell_d^\star). \end{aligned} \tag{21}$$

Topic probabilities are not summed across categories. The annotations therefore represent model-derived behavioral pseudo-labels based on each scenario's dominant topic.

A.5 Dataset statistics and partitioning

The annotation process produces 443,218 labeled samples. Normal behavior accounts for 54.05% of the samples, followed by aggressive behavior at 34.32% and conservative behavior at 11.63%. These proportions characterize the constructed dataset and should not be interpreted as population-level frequencies of driver types.

For behavior classification experiments, the labeled samples are randomly divided into training, validation, and test subsets in a ratio of $7 : 2 : 1$.

B Behavior Encoder and Stage-1 Training

This appendix details the contextual temporal encoder introduced in Section 4.2. In Stage 1, the encoder and an initial classification head are jointly trained on nuPlan-Behavior to learn a continuous behavior embedding. The encoder combines agent-weighted aggregation, NCP-CfC temporal modeling, and attentive temporal pooling. Its output $\mathbf{z}_i$ subsequently serves as the query feature for prototype memory enhancement.

B.1 INPUT REPRESENTATION

For focal vehicle i, the input $\mathcal{H}_i$ contains its historical states and neighboring-agent context. We represent these observations as $\mathbf{H}^{(i)} \in \mathbb{R}^{N_i \times T_h \times D_{\text{state}}}$, where N_i includes the focal vehicle and its contextual agents, T_h is the historical sequence length, and D_{state} is the state dimension. Each vector $\mathbf{h}_{n,t}^{(i)} \in \mathbb{R}^{D_{\text{state}}}$ contains the position, velocity, heading, and dimensions of input agent n at time step t. The superscript (i) identifies the focal-vehicle-specific input; the encoder parameters are shared across focal vehicles. Batch indices are omitted for clarity.

B.2 AGENT-WEIGHTED AGGREGATION

At each historical step, a shared scoring network $f_{\text{a}} : \mathbb{R}^{D_{\text{state}}} \to \mathbb{R}$ assigns a scalar score to each input agent. The scores are normalized across agents and used to construct a contextual state vector:

$$\begin{aligned} e_{n,t}^{(i)} &= f_{\text{a}}(\mathbf{h}_{n,t}^{(i)}), \\ \omega_{n,t}^{(i)} &= \frac{\exp(e_{n,t}^{(i)})}{\sum_{j=1}^{N_i} \exp(e_{j,t}^{(i)})}, \\ \mathbf{a}_t^{(i)} &= W_{\text{a}} \left(\sum_{n=1}^{N_i} \omega_{n,t}^{(i)} \mathbf{h}_{n,t}^{(i)} \right) + \mathbf{b}_{\text{a}}. \end{aligned} \tag{22}$$

Here, $W_{\text{a}} \in \mathbb{R}^{D_{\text{a}} \times D_{\text{state}}}$ and $\mathbf{b}_{\text{a}} \in \mathbb{R}^{D_{\text{a}}}$ project the aggregated state into the temporal encoder's input space. Since $\sum_n \omega_{n,t}^{(i)} = 1$, the aggregation forms a weighted summary of agent states at each step, yielding the contextual sequence $\mathbf{a}_{1:T_h}^{(i)} \in \mathbb{R}^{T_h \times D_{\text{a}}}$.

B.3 NCP-CFC TEMPORAL MODELING

The contextual sequence is processed by a closed-form continuous-time network with Neural Circuit Policy wiring. We denote the resulting temporal mapping by $\phi_{\text{NCP-CfC}}$:

$$\begin{aligned} \mathbf{R}_i &= \phi_{\text{NCP-CfC}} \left(\mathbf{a}_{1:T_h}^{(i)} \right), \\ \mathbf{r}_t^{(i)} &= [\mathbf{R}_i]_t, \qquad \bar{\mathbf{r}}_t^{(i)} = \text{LN}(\mathbf{r}_t^{(i)}). \end{aligned} \tag{23}$$

The output $\mathbf{R}_i \in \mathbb{R}^{T_h \times D_{\text{r}}}$ contains a temporal feature for each historical step. CfC provides the recurrent temporal computation, while NCP specifies its sparse connectivity. Layer normalization is applied to each output before temporal attention. Thus, the temporal features encode the evolution of the aggregated context rather than treating historical observations independently.

B.4 TEMPORAL ATTENTION AND FEATURE PROJECTION

We summarize the encoded history using learned temporal weights. A shared linear scoring function assigns an importance score to each normalized temporal feature:

$$\begin{aligned} u_t^{(i)} &= \mathbf{w}_\beta^\top \bar{\mathbf{r}}_t^{(i)} + b_\beta, \\ \beta_t^{(i)} &= \frac{\exp(u_t^{(i)})}{\sum_{\tau=1}^{T_h} \exp(u_\tau^{(i)})}, \\ \mathbf{q}_i &= \sum_{t=1}^{T_h} \beta_t^{(i)} \bar{\mathbf{r}}_t^{(i)}, \qquad \mathbf{z}_i = f_{\text{p}}(\mathbf{q}_i). \end{aligned} \tag{24}$$

Here, $\mathbf{w}_\beta \in \mathbb{R}^{D_{\text{r}}}$ and $b_\beta \in \mathbb{R}$ are trainable parameters, and $f_{\text{p}} : \mathbb{R}^{D_{\text{r}}} \to \mathbb{R}^{D_{\text{z}}}$ is the feature projection network. Agent weights $\omega_{n,t}^{(i)}$ aggregate participants within each step, whereas temporal weights $\beta_t^{(i)}$ aggregate information across the observed history. Their composition produces the sequence-level behavior embedding $\mathbf{z}_i \in \mathbb{R}^{D_{\text{z}}}$.

B.5 Stage 1: supervised representation learning.

Let $E_{\theta_{\text{enc}}}$ denote the complete feature extractor defined above, parameterized by θ_{enc}, and let $g^{(1)}_{\eta_1}$ denote the Stage 1 classification head, parameterized by η_1. For a labeled training sample $(\mathcal{H}_i, y_i)$, $\mathcal{H}_i$ denotes the historical input and $y_i \in \{1, \dots, C\}$ denotes its behavior label, where $C = 3$. The feature embedding, classification logits, and class probabilities are computed as follows:

$$\begin{aligned} \mathbf{z}_i &= E_{\theta_{\text{enc}}}(\mathcal{H}_i), \\ \boldsymbol{\ell}^{(1)}_i &= g^{(1)}_{\eta_1}(\mathbf{z}_i), \\ \mathbf{p}^{(1)}_i &= \mathrm{softmax}(\boldsymbol{\ell}^{(1)}_i). \end{aligned} \tag{25}$$

Here, $\mathbf{z}_i$ is the continuous behavior embedding, $\boldsymbol{\ell}^{(1)}_i \in \mathbb{R}^C$ contains the classification logits, and $\mathbf{p}^{(1)}_i \in [0,1]^C$ contains the corresponding class probabilities. The superscript (1) identifies Stage 1.

The feature extractor and classification head are jointly trained by minimizing the following objective:

$$\mathcal{L}_{\text{stage1}} = \lambda_{\text{CE}}\mathcal{L}_{\text{CE}} + \lambda_{\text{EDL}}\mathcal{L}_{\text{EDL}}. \tag{26}$$

Here, $\mathcal{L}_{\text{CE}}$ is the cross-entropy loss for behavior classification, and $\mathcal{L}_{\text{EDL}}$ is the evidential learning loss. The nonnegative coefficients λ_{CE} and λ_{EDL} control their relative contributions.

After Stage 1, the feature extractor is frozen, and the embeddings extracted from the training set are used to construct the behavior prototypes. The continuous embedding $\mathbf{z}_i$ then serves as the input to the memory-enhanced head in Stage 2. The initial classification head provides supervision during Stage 1, while downstream behavioral conditioning uses continuous features rather than hard class predictions.

C Prototype Memory and Stage-2 Training

This appendix details prototype construction, evidence-based retrieval, and memory-enhanced behavior learning. After Stage 1, the feature extractor $E_{\theta_{\text{enc}}}$ is frozen. Stage 2 learns to supplement its history-dependent embeddings with class-level information stored in a prototype memory.

C.1 Prototype construction

Let $\mathcal{D}_{\text{tr}} = \{(\mathcal{H}_j, y_j)\}_{j=1}^{N_{\text{tr}}}$ denote the behavior training set. We extract its embeddings using the frozen encoder and compute one prototype for each behavior category:

$$\begin{aligned} &\mathbf{z}_j = E_{\theta_{\text{enc}}}(\mathcal{H}_j), \\ &\mathcal{I}_c = \{j : y_j = c\}, \qquad \mathbf{c}_c = \frac{1}{|\mathcal{I}_c|}\sum_{j \in \mathcal{I}_c} \mathbf{z}_j, \\ &\mathbf{C} = [\mathbf{c}_1, \dots, \mathbf{c}_C]^\top \in \mathbb{R}^{C \times D_z}. \end{aligned} \tag{27}$$

Here, $C = 3$ corresponds to aggressive, normal, and conservative behavior. The prototypes summarize the class-conditional training embeddings and remain fixed during Stage 2. Validation and test samples are excluded from their construction.

C.2 Evidence-space comparison

A shared evidence mapping transforms the query embedding and each prototype into Dirichlet parameters:

$$\begin{aligned} &A(\mathbf{z}) = \mathrm{ReLU}\big(\phi_{\text{MLP}_\alpha}(\mathbf{z})\big) + \mathbf{1}, \\ &\boldsymbol{\alpha}_i = A(\mathbf{z}_i), \qquad \boldsymbol{\nu}_c = A(\mathbf{c}_c). \end{aligned} \tag{28}$$

For a parameter vector $\boldsymbol{\alpha}$, the corresponding Dirichlet mean is $\alpha_k / \sum_{\ell=1}^{C} \alpha_\ell$. The mapping therefore compares queries and prototypes through their induced distributions over behavior categories.

We quantify the discrepancy between query i and class prototype c using the directed KL divergence: $d_{i,c} = D_{\mathrm{KL}}(\mathrm{Dir}(\boldsymbol{\alpha}_i) \| \mathrm{Dir}(\boldsymbol{\nu}_c))$. For arbitrary positive parameter vectors $\boldsymbol{\alpha}$ and $\boldsymbol{\nu}$, this divergence is

$$\begin{aligned} &D_{\mathrm{KL}}(\mathrm{Dir}(\boldsymbol{\alpha}) \parallel \mathrm{Dir}(\boldsymbol{\nu})) \\ &\quad = \log\Gamma(S_\alpha) - \sum_{k=1}^{C} \log\Gamma(\alpha_k) - \log\Gamma(S_\nu) + \sum_{k=1}^{C} \log\Gamma(\nu_k) \\ &\qquad + \sum_{k=1}^{C} (\alpha_k - \nu_k)\,[\psi(\alpha_k) - \psi(S_\alpha)]\,, \end{aligned} \tag{29}$$

where $S_\alpha = \sum_k \alpha_k$, $S_\nu = \sum_k \nu_k$, and Γ and ψ denote the gamma and digamma functions, respectively. The query distribution is the first argument throughout.

C.3 Prototype retrieval

The divergences are converted into normalized inverse-distance weights, which determine the contribution of each prototype:

$$\begin{aligned} a_{i,c} &= \frac{(d_{i,c} + \epsilon_{\mathrm{KL}})^{-1}}{\sum_{j=1}^{C} (d_{i,j} + \epsilon_{\mathrm{KL}})^{-1}}, \\ \mathbf{m}_i &= \sum_{c=1}^{C} a_{i,c} \mathbf{c}_c. \end{aligned} \tag{30}$$

The constant $\epsilon_{\mathrm{KL}} > 0$ prevents division by zero. Since $a_{i,c} > 0$ and $\sum_c a_{i,c} = 1$, the retrieved feature $\mathbf{m}_i$ is a convex combination of the behavior prototypes. Smaller divergence produces greater contribution, allowing multiple categories to inform the representation without a hard prototype assignment. These weights serve as retrieval coefficients rather than calibrated class probabilities.

C.4 Gated memory fusion

The retrieved memory is combined with the query through a componentwise gate and a prototype-distance scale:

$$\begin{aligned} \mathbf{g}_i &= \tanh\big(\phi_{\mathrm{MLP}_g}(\mathbf{z}_i)\big)\,, \\ s_i &= \frac{1}{\lambda_s} \min_{c \in \{1,\dots,C\}} \|\mathbf{z}_i - \mathbf{c}_c\|_2, \\ \mathbf{b}_i &= s_i\,(\mathbf{z}_i + \mathbf{g}_i \odot \mathbf{m}_i)\,, \end{aligned} \tag{31}$$

where $\lambda_s > 0$ controls the distance scale and $\odot$ denotes elementwise multiplication. The gate has values in $(-1, 1)$ and permits both additive and subtractive memory contributions. The scalar s_i rescales the entire fused feature according to its query's distance from the nearest prototype. It increases with this distance and is not interpreted as a calibrated confidence estimate.

C.5 Stage 2: memory-enhanced classification

A classification head maps the enhanced representation to behavior logits:

$$\boldsymbol{\ell}_i^{(2)} = W_{\mathrm{cls}}^{(2)} \mathbf{b}_i + \mathbf{b}_{\mathrm{cls}}^{(2)}. \tag{32}$$

Stage 2 optimizes the evidence mapping, the gating network, and the final classifier while keeping $E_{\theta_{\mathrm{enc}}}$ and $\mathbf{C}$ fixed. Classification supervision thus learns how to retrieve and integrate prototype information within the fixed embedding space.

After Stage 2, the complete behavior branch is frozen. The continuous feature $\mathbf{b}_i$, rather than the classification logits or a hard behavior label, is supplied to the downstream prediction and planning modules.

Training objectives. Both stages use classification and evidential supervision. For stage $q \in \{1, 2\}$, let $\boldsymbol{\ell}_i^{(q)}$ denote the classification logits and $\mathbf{y}_i$ the one-hot behavior label. We define the softmax probabilities and classification evidence as:

$$\begin{aligned} \mathbf{p}_i^{(q)} &= \mathrm{softmax}(\boldsymbol{\ell}_i^{(q)}), \\ \boldsymbol{\alpha}_i^{(q)} &= \mathrm{ReLU}(\boldsymbol{\ell}_i^{(q)}) + \mathbf{1}, \\ S_i^{(q)} &= \sum_{c=1}^{C} \alpha_{i,c}^{(q)}, \qquad \mu_{i,c}^{(q)} = \frac{\alpha_{i,c}^{(q)}}{S_i^{(q)}}. \end{aligned} \tag{33}$$

Here, $\boldsymbol{\mu}_i^{(q)}$ is the Dirichlet mean. The classification evidence is derived from the final logits; it is distinct from the query evidence used for memory retrieval.

For a mini-batch $\mathcal{B}$, let $\mathbf{y}_i \in \{0, 1\}^C$ denote the one-hot behavior label of sample i, and let $q \in \{1, 2\}$ index the training stage. The predicted class probabilities are denoted by $\mathbf{p}_i^{(q)}$, while $\boldsymbol{\alpha}_i^{(q)} \in \mathbb{R}_{>0}^C$ contains the predicted Dirichlet parameters. The corresponding Dirichlet mean is $\boldsymbol{\mu}_i^{(q)} = \boldsymbol{\alpha}_i^{(q)} / S_i^{(q)}$, where $S_i^{(q)} = \sum_{c=1}^{C} \alpha_{i,c}^{(q)}$.

We combine cross-entropy supervision with an evidential objective consisting of a squared error on the Dirichlet mean and a prior regularizer. The losses and the overall training objective are defined as follows:

$$\begin{aligned} \mathcal{L}_{\mathrm{CE}}^{(q)} &= -\frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \sum_{c=1}^{C} y_{i,c} \log p_{i,c}^{(q)}, \\ \mathcal{L}_{\mathrm{EDL}}^{(q)} &= \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \Big[\|\mathbf{y}_i - \boldsymbol{\mu}_i^{(q)}\|_2^2 \\ &\qquad + \rho(e)\, D_{\mathrm{KL}}\Big(\mathrm{Dir}(\boldsymbol{\alpha}_i^{(q)}) \,\|\, \mathrm{Dir}(\mathbf{1}) \Big) \Big], \\ \mathcal{L}_{\mathrm{stage}q} &= \lambda_{\mathrm{CE}}(e) \mathcal{L}_{\mathrm{CE}}^{(q)} + \lambda_{\mathrm{EDL}}(e) \mathcal{L}_{\mathrm{EDL}}^{(q)}. \end{aligned} \tag{34}$$

Here, $|\mathcal{B}|$ is the mini-batch size, and $\mathbf{1} \in \mathbb{R}^C$ is the all-ones vector, so that $\mathrm{Dir}(\mathbf{1})$ is the uniform distribution over the probability simplex. The nonnegative coefficient $\rho(e)$ controls the strength of the KL regularizer within the evidential loss.

The outer loss weights vary with the epoch index e within each training stage according to the following schedule:

$$\big(\lambda_{\mathrm{CE}}(e), \lambda_{\mathrm{EDL}}(e)\big) = \begin{cases} (1.0,\, 0.0), & e < 20, \\ (0.7,\, 0.3), & 20 \le e < 25, \\ (0.2,\, 0.8), & e \ge 25. \end{cases} \tag{35}$$

This schedule initially uses only cross-entropy supervision and progressively increases the weight of the evidential objective.

D Behavior-Conditioned Prediction and Planning: Architecture and Objectives

This appendix details the scene encoders, behavior-conditioning interfaces, prediction and planning decoders, and joint training objectives introduced in Section 4.3. Given $X = (\{H_i\}_{i=0}^{N_a}, \{O_s\}_{s=1}^{N_s}, M)$, the downstream model combines contextual scene features with frozen behavior representations to predict surrounding-agent motion and generate scored ego trajectories.

D.1 Scene Encoding

Surrounding-agent motion encoding. For surrounding agent i, let ΔH_i denote the temporal differences of its historical motion sequence. A Neighborhood Attention Transformer (NAT) encodes

these differences using local temporal attention. Its output is combined with an embedding of the agent's semantic category:

$$\begin{aligned} \mathbf{e}_i &= \phi_{\mathrm{NAT}}(\Delta H_i) + \phi_{\mathrm{type}}(c_i), \qquad i = 1, \dots, N_a, \\ E_A &= [\mathbf{e}_1, \dots, \mathbf{e}_{N_a}]^\top \in \mathbb{R}^{N_a \times d}. \end{aligned} \tag{36}$$

Here, c_i denotes the agent type, such as vehicle, pedestrian, or bicycle, rather than a driving-behavior label. The temporal-difference input describes local motion changes, while the semantic embedding distinguishes participant types.

Ego and static-obstacle encoding. The ego encoder retains the historical state sequence H_0 rather than replacing it with temporal differences. A dedicated self-attention module processes the sequence under stochastic feature masking during training. Static obstacles are independently embedded by an MLP:

$$\begin{aligned} \mathbf{e}_0 &= \phi_{\mathrm{ego}}(M(H_0)) + \phi_{\mathrm{type}}(c_0), \\ \mathbf{e}_s^O &= \phi_O(O_s), \qquad s = 1, \dots, N_s, \\ E_0 &= \mathbf{e}_0^\top, \qquad E_O = [\mathbf{e}_1^O, \dots, \mathbf{e}_{N_s}^O]^\top. \end{aligned} \tag{37}$$

M denotes the training-time feature-masking operation, and all output tokens have width d.

Map encoding. Let $M = \{M_j\}_{j=1}^{N_m}$ contain vectorized map elements. For element j, a pointwise MLP followed by elementwise max pooling produces a geometric descriptor. Discrete semantic attributes s_j and continuous attributes r_j are embedded separately:

$$\begin{aligned} \phi_{\mathrm{geom}}(M_j) &= \operatorname*{MaxPool}_{\mathbf{m} \in M_j} \phi_{\mathrm{pt}}(\mathbf{m}), \\ \mathbf{e}_j^M &= \phi_{\mathrm{geom}}(M_j) + \phi_{\mathrm{sem}}(s_j) + \phi_{\mathrm{F}}^{\mathrm{attr}}(r_j), \\ E_M &= [\mathbf{e}_1^M, \dots, \mathbf{e}_{N_m}^M]^\top. \end{aligned} \tag{38}$$

The pooling operation is invariant to the ordering of points within each element. The resulting tokens combine map geometry, semantic information, and Fourier-encoded continuous attributes.

Global interaction encoding. We concatenate the ego, surrounding-agent, obstacle, and map tokens and add Fourier positional embeddings:

$$E_{\mathrm{scene}} = \phi_{\mathrm{enc}}([E_0 \| E_A \| E_O \| E_M] + \phi_{\mathrm{F}}(P)) \,. \tag{39}$$

Here, P contains the coordinates associated with the tokens, and $\|$ denotes concatenation along the token dimension. The Transformer encoder contextualizes the heterogeneous inputs through scene-wide interactions. Its surrounding-agent token subset is denoted by $E_{\mathrm{agent}} \in \mathbb{R}^{N_a \times d}$.

D.2 Behavior Tokens and Cross-Attention

The pretrained behavior branch produces $\mathbf{b}_i = B_{\phi^\star}(\mathcal{H}_i)$, where $\phi^\star$ remains fixed throughout downstream training. The surrounding-vehicle and ego-inclusive token sets are

$$S_{\mathrm{veh}} = [P_{\mathrm{b}} \mathbf{b}_i]_{i \in \mathcal{V}}, \qquad S_{\mathrm{all}} = [P_{\mathrm{b}} \mathbf{b}_i]_{i \in \{0\} \cup \mathcal{V}}. \tag{40}$$

Here, $\mathcal{V} \subseteq \{1, \dots, N_a\}$ indexes surrounding motor vehicles. The projection notation suppresses task-specific indexing; the projection parameters belong to the trainable downstream network rather than the frozen behavior branch.

For query, key, and value matrices Q, K, and V, multi-head cross-attention is defined as

$$\begin{aligned} U_h(Q, K, V) &= \mathrm{softmax}_{\mathrm{row}}\left(\frac{(QW_Q^h)(KW_K^h)^\top}{\sqrt{d_k}} + \mathcal{M}_{\mathrm{att}} \right)(VW_V^h), \\ \mathrm{CA}(Q, K, V) &= \mathrm{Concat}_{\mathrm{feat}}\big(U_1(Q, K, V), \dots, U_{N_{\mathrm{h}}}(Q, K, V)\big) W_{\mathrm{out}}. \end{aligned} \tag{41}$$

N_{h} is the number of heads, and d_k is the per-head key dimension. The attention mask $\mathcal{M}_{\mathrm{att}}$ excludes invalid keys. $\mathrm{Concat}_{\mathrm{feat}}$ concatenates head outputs along the feature dimension, and W_{out} projects them to the common width d. Prediction and planning use separate attention parameters.

D.3 Prediction and Planning Decoders

Behavior-conditioned prediction. Contextualized surrounding-agent tokens query the surrounding-vehicle behavior tokens:

$$\widehat{Y}_{1:N_a} = \phi_{\text{pred}}(\text{CA}(E_{\text{agent}}, S_{\text{veh}}, S_{\text{veh}})) . \tag{42}$$

The prediction head maps the fused features to future agent trajectories. Each agent query can retrieve information from multiple vehicle behavior tokens; the attention does not impose a one-to-one correspondence between queries and behavior keys. Pedestrians and bicycles retain prediction outputs without being assigned vehicle behavior categories.

Candidate-query construction. Let E_{ref} denote reference-lane features and Q_{long} the learnable longitudinal queries. The planner combines them into K candidate queries:

$$Q_{\text{init}} = \phi_{\text{q}}(E_{\text{ref}}, Q_{\text{long}}) \in \mathbb{R}^{K\times d}. \tag{43}$$

The construction operator ϕ_{q} follows the adopted planner's reference-lane and longitudinal-query parameterization. The resulting candidates represent planning modes, independently of the three behavior categories used to pretrain the behavior encoder.

Behavior and scene fusion. The initial queries first retrieve behavior information from both the ego and surrounding motor vehicles:

$$Q_{\text{beh}} = Q_{\text{init}} + \eta\, \phi_{CA}(Q_{\text{init}}, S_{\text{all}}, S_{\text{all}}). \tag{44}$$

The residual coefficient η scales the behavioral contribution while retaining the initial query features.

The behavior-conditioned queries then pass through a stack of planning decoder layers:

$$\begin{aligned} &Z^{(0)} = Q_{\text{beh}}, \\ &Z^{(\ell)} = \mathcal{D}_\ell \left(Z^{(\ell-1)}, E_{\text{scene}} \right), \qquad \ell = 1, \dots, L_{\text{dec}}, \\ &Z_{\text{plan}} = Z^{(L_{\text{dec}})}, \\ &\gamma^{(k)} = \phi_{\text{traj}}(Z_{\text{plan},k}), \qquad \sigma^{(k)} = \phi_{\text{score}}(Z_{\text{plan},k}). \end{aligned} \tag{45}$$

Each $\mathcal{D}_\ell$ combines query self-attention, cross-attention to the scene tokens, and feed-forward transformations. The trajectory and scoring heads produce $\mathcal{G} = \{(\gamma^{(k)}, \sigma^{(k)})\}_{k=1}^{K}$. At inference, the candidate indexed by $k^\star = \arg\max_k \sigma^{(k)}$ initializes the subsequent trajectory-refinement stage.

D.4 Joint Training Objectives

We adopt the training objectives of PlanScope (Xin et al., 2026) and retain the contrastive auxiliary objective used in its contrastive-learning configuration:

$$\mathcal{L}_{\text{plan}} = \lambda_{\text{pred}}\mathcal{L}_{\text{pred}} + \lambda_{\text{cls}}\mathcal{L}_{\text{cls}} + \lambda_{\text{reg}}\mathcal{L}_{\text{ego-reg}} + \lambda_{\text{col}}\mathcal{L}_{\text{col}} + \lambda_{\text{ds}}\mathcal{L}_{\text{ds}} + \lambda_{\text{ctr}}\mathcal{L}_{\text{contrastive}}. \tag{46}$$

Here, $\mathcal{L}_{\text{pred}}$ supervises surrounding-agent trajectories using Smooth L1 loss, and $\mathcal{L}_{\text{cls}}$ applies cross-entropy supervision to candidate scores, with the candidate closest to the expert trajectory as the target. $\mathcal{L}_{\text{ego-reg}}$ supervises ego trajectories with time-dependent normalization. $\mathcal{L}_{\text{col}}$ penalizes insufficient clearance using a circle-based vehicle representation, while $\mathcal{L}_{\text{ds}}$ supervises trajectory components at multiple decision scopes using wavelet-decomposed expert trajectories. Finally, $\mathcal{L}_{\text{contrastive}}$ encourages scene representations to distinguish positive and negative augmentations. The coefficients $\lambda_\cdot$ balance these objectives; their underlying formulations follow PlanScope.

During joint training, we update the scene backbone, behavior projections, and prediction and planning decoders, while keeping the pretrained behavior branch frozen. Trajectory refinement is applied only at inference.

E Trajectory Refinement Objectives

This appendix details the inference-time refinement objective introduced in Section 4.4. The highest-scoring candidate initializes the speed-heading sequence $\mathbf{u} = \{(v_t, \theta_t)\}_{t=1}^{T_f}$. The network parameters and surrounding-agent forecasts remain fixed while the ego trajectory is refined.

E.1 Motion Parameterization

For each future step, the velocity vector and reconstructed position are

$$\mathbf{v}_t = v_t[\cos\theta_t, \sin\theta_t]^\top, \qquad \mathbf{p}_t = \mathbf{p}_0 + \Delta t \sum_{\tau=1}^{t} \mathbf{v}_\tau. \tag{47}$$

Here, v_t is the nonnegative forward speed, θ_t is the heading, and Δt is the sampling interval. We use $[x]_+ = \max(0, x)$ for all hinge penalties. Acceleration and curvature terms are evaluated over $t = 1, \ldots, T_f - 1$, while jerk is evaluated over $t = 1, \ldots, T_f - 2$.

E.2 Motion and Comfort Costs

Longitudinal acceleration. Longitudinal acceleration is estimated from consecutive speed samples. We penalize acceleration and braking magnitudes exceeding the prescribed limit:

$$\mathcal{L}_{\text{lon}} = \frac{1}{T_f - 1} \sum_{t=1}^{T_f-1} \left[\left| \frac{v_{t+1} - v_t}{\Delta t} \right| - a_{\max}^{\text{lon}} \right]_+ . \tag{48}$$

The threshold $a_{\max}^{\text{lon}} > 0$ specifies the allowable longitudinal acceleration magnitude. Samples within this limit contribute zero penalty.

Lateral acceleration. We first compute the wrapped heading increment $\Delta\theta_t = \operatorname{atan2}(\sin(\theta_{t+1} - \theta_t), \cos(\theta_{t+1} - \theta_t))$. Using the bicycle-model relation $a^{\text{lat}} = v\dot{\theta}$, the lateral penalty is

$$\mathcal{L}_{\text{lat}} = \frac{1}{T_f - 1} \sum_{t=1}^{T_f-1} \left[\left| \frac{v_t \Delta\theta_t}{\Delta t} \right| - a_{\max}^{\text{lat}} \right]_+ . \tag{49}$$

This term couples turning rate with speed, discouraging sharp heading changes at high speed. The threshold $a_{\max}^{\text{lat}} > 0$ controls the allowable lateral acceleration magnitude.

Curvature. For $v_t > 0$, curvature is approximated by $\kappa_t = \Delta\theta_t / (v_t \Delta t)$. We penalize curvature magnitudes above $\kappa_{\max}$:

$$\mathcal{L}_{\text{curv}} = \frac{1}{T_f - 1} \sum_{t=1}^{T_f-1} \left[\left| \frac{\Delta\theta_t}{v_t \Delta t} \right| - \kappa_{\max} \right]_+ . \tag{50}$$

Under the kinematic bicycle model, $\delta_t = \arctan(L\kappa_t)$, where L is the wheelbase. The curvature threshold therefore corresponds to a steering-angle magnitude of $\arctan(L\kappa_{\max})$. Unlike the lateral acceleration penalty, this term directly regulates turning geometry.

Cartesian jerk. To discourage rapid changes in acceleration, we penalize the norm of the discrete Cartesian jerk:

$$\mathcal{L}_{\text{jerk}} = \frac{1}{T_f - 2} \sum_{t=1}^{T_f-2} \left\| \frac{\mathbf{v}_{t+2} - 2\mathbf{v}_{t+1} + \mathbf{v}_t}{\Delta t^2} \right\|_2 . \tag{51}$$

Because $\mathbf{v}_t$ depends on both speed and heading, this objective responds to changes in velocity magnitude and direction. It complements the acceleration-limit penalties by encouraging smoother motion even within their admissible ranges.

Speed-limit compliance. Let $v_t^{\max}$ denote the applicable speed limit. The speeding penalty is

$$\mathcal{L}_{\text{speed}} = \frac{1}{T_f} \sum_{t=1}^{T_f} [v_t - v_t^{\max}]_+ . \tag{52}$$

Only speed-limit exceedances contribute to this term; motion below the limit is not penalized.

E.3 Obstacle-Proximity Costs

Static clearance. Let $\Psi(\mathbf{p})$ denote the signed distance to occupied geometry, with positive values in free space. We encourage a clearance margin $d_{\text{safe}} > 0$ along the reconstructed trajectory:

$$\mathcal{L}_{\text{static}} = \frac{1}{T_f} \sum_{t=1}^{T_f} [d_{\text{safe}} - \Psi(\mathbf{p}_t)]_+. \tag{53}$$

Positions with clearance below d_{safe} incur a penalty, which increases further inside occupied geometry. The field is constructed from the static scene information and evaluated at the reconstructed ego positions.

Acceleration-bounded agent regions. For surrounding agent n, let $\mathbf{p}_{n,0}$, $\mathbf{v}_{n,0}$, and $\theta_{n,0}$ denote its observed position, velocity, and heading. We construct a local frame using

$$R_n = \begin{bmatrix} \cos\theta_{n,0} & -\sin\theta_{n,0} \\ \sin\theta_{n,0} & \cos\theta_{n,0} \end{bmatrix}, \qquad \begin{bmatrix} v_n^{\text{lon}} \\ v_n^{\text{lat}} \end{bmatrix} = R_n^\top \mathbf{v}_{n,0}. \tag{54}$$

For coordinate $q \in \{\text{lon}, \text{lat}\}$, let $a_n^{q,-}$ and $a_n^{q,+}$ be its lower and upper acceleration bounds. At elapsed time $\tau_t = t\Delta t$, the displacement bounds are

$$s_{n,t}^{q,-} = v_n^q \tau_t + \frac{1}{2} a_n^{q,-} \tau_t^2, \qquad s_{n,t}^{q,+} = v_n^q \tau_t + \frac{1}{2} a_n^{q,+} \tau_t^2. \tag{55}$$

These bounds define a time-dependent region

$$\mathcal{R}_{n,t} = \left\{ \mathbf{p}_{n,0} + R_n \mathbf{s} : \begin{array}{l} s^{\text{lon}} \in [s_{n,t}^{\text{lon},-}, s_{n,t}^{\text{lon},+}], \\ s^{\text{lat}} \in [s_{n,t}^{\text{lat},-}, s_{n,t}^{\text{lat},+}] \end{array} \right\}. \tag{56}$$

The acceleration bounds depend on the participant type. The region approximates possible agent positions under independent acceleration bounds in a fixed local frame; it does not represent a full vehicle-footprint or steering-constrained reachable set.

Dynamic proximity. Let $\widehat{\mathbf{p}}_{n,t}$ be the predicted position of agent n extracted from $\widehat{Y}_{1:N_a}$. We activate a Gaussian proximity penalty when the ego position lies inside $\mathcal{R}_{n,t}$:

$$\mathcal{L}_{\text{dynamic}} = \frac{1}{T_f} \sum_{t=1}^{T_f} \sum_{n=1}^{N_a} \mathbb{I}[\mathbf{p}_t \in \mathcal{R}_{n,t}] \exp\left(-\frac{\|\mathbf{p}_t - \widehat{\mathbf{p}}_{n,t}\|_2^2}{2\sigma_{\text{d}}^2} \right). \tag{57}$$

Here, $\sigma_{\text{d}} > 0$ controls the spatial extent of the proximity penalty. The region indicator selects potentially interacting agent-time pairs, while the Gaussian term assigns greater cost to smaller separation from the predicted position. Agent forecasts remain unchanged during ego refinement.

E.4 Combined Objective and Refinement

The complete objective is the sum of the seven costs:

$$\begin{aligned} J(\mathbf{u}; X, \widehat{Y}) = \mathcal{L}_{\text{lon}} + \mathcal{L}_{\text{lat}} + \mathcal{L}_{\text{curv}} + \mathcal{L}_{\text{jerk}} \\ + \mathcal{L}_{\text{speed}} + \mathcal{L}_{\text{static}} + \mathcal{L}_{\text{dynamic}}. \end{aligned} \tag{58}$$

Starting from the selected candidate, a fixed number of optimization steps updates the speed-heading sequence. The final positions are recovered using Eq. equation 47, yielding the refined trajectory $\widehat{\gamma}$. The objectives provide soft penalties on sampled motion and proximity quantities rather than a hard feasibility guarantee.

F Experimental Settings

We evaluate behavior recognition on nuPlan-behavior using classification accuracy (Acc) and uncertainty rate (UR):

$$\begin{aligned} \text{Acc} &= \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(\hat{y}_i = y_i), \\ \text{UR} &= \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(\max_{k \in \{1,\dots,C\}} p(y_i = k \mid \mathbf{x}_i) < \tau \right), \end{aligned} \tag{59}$$

Table 4: Behavior recognition results on nuPlan-behavior. Each result cell reports Acc↑/UR↓. Best and second-best results are bold and underlined, respectively.

Model	Params	Overall	Aggressive	Normal	Conservative
MLP	196K	0.811/0.595	0.293/0.713	0.857/0.586	<u>0.927</u>/0.708
CFC	141K	<u>0.944</u>/<u>0.232</u>	<u>0.874</u>/<u>0.375</u>	**0.967**/<u>0.173</u>	0.919/<u>0.324</u>
CFC+M	146K	**0.948/0.191**	**0.901/0.254**	<u>0.958</u>/**0.164**	**0.950/0.238**

where N is the number of evaluated samples, C is the number of behavior classes, y_i and $\hat{y}_i$ are the ground-truth and predicted labels, respectively, and $\mathbb{I}(\cdot)$ is the indicator function. Acc measures classification correctness, while UR measures the proportion of predictions whose maximum class probability falls below a confidence threshold τ. Higher Acc and lower UR indicate better classification accuracy and fewer low-confidence predictions, respectively.

We evaluate trajectory prediction on nuScenes using minADE_5, minADE_{10}, minFDE_1, minFDE_5, and MR_5. The displacement metrics measure agreement between predicted and logged future trajectories, while the miss rate measures the proportion of predictions that fail the benchmark's specified criterion. Closed-loop planning is evaluated on nuPlan Val14, Test14, and Test14-hard under reactive (CLS-R) and non-reactive (CLS-NR) simulation, with higher scores indicating better performance. Detailed Val14 sub-metrics are reported for CLS-NR.

Behavior pretraining. The behavior recognition model is trained on nuPlan-behavior, which is randomly partitioned into training, validation, and test sets with a ratio of 7:2:1. The training set is used for parameter optimization, the validation set for performance monitoring and hyperparameter selection, and the test set for held-out evaluation. Both training stages use Adam with an initial learning rate of 5×10^{-4} and a batch size of 32, and each stage runs for 30 epochs. The two-stage training procedure and its objectives are detailed in the methods appendix.

Hardware and downstream training. Training and inference experiments are conducted on a server equipped with two NVIDIA A100 GPUs. The downstream prediction/planning model is trained for 25 epochs, including three warm-up epochs, with an initial learning rate of 10^{-3}, weight decay of 10^{-4}, and a batch size of 32. The pretrained behavior branch remains frozen during downstream training, and trajectory refinement is applied only at inference.

G Behavior Recognition and Model Size

Figure 6 compares an MLP, a CfC temporal encoder, and its prototype-memory variant (CfC+M) on the behavior-annotated nuPlan-mini data. As shown in Table 4 Accuracy increases from 81.1% for the MLP to 94.4% for CfC, a 13.3-percentage-point improvement. This comparison supports the value of the temporal encoder in the evaluated architecture. Adding memory increases accuracy to 94.8% and reduces UR from 0.232 to 0.191, a 17.7% relative reduction in the reported uncertainty metric. Parameter count increases from 141K to 146K, approximately 3.5%.

The class-wise effects are uneven. Accuracy for aggressive behavior increases from 87.4% to 90.1%, and for conservative behavior from 91.9% to 95.0%, while accuracy for normal behavior decreases from 96.7% to 95.8%. Prototype memory therefore improves the two behavioral extremes in this comparison while introducing a small trade-off for the normal category. Lower UR alone does not establish better probability calibration. These results characterize behavior recognition and parameter overhead; the downstream ablations in Table 2 and 3 provide the separate evidence for planning utility.

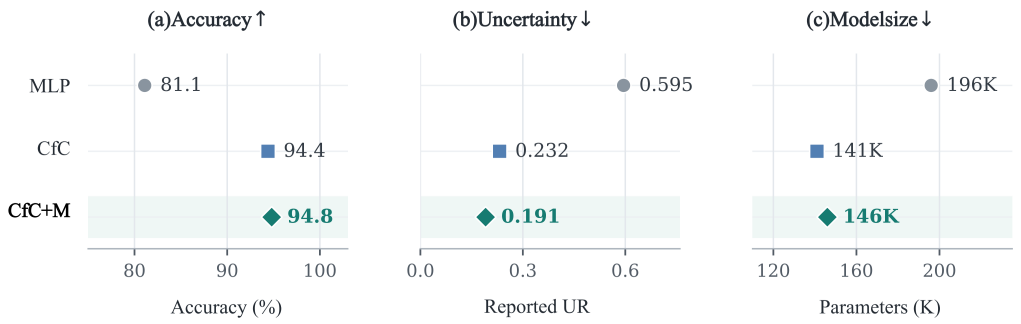

Figure 6: Behavior recognition and encoder size on behavior-annotated nuPlan-mini. CfC+M adds prototype memory to the CfC encoder. Accuracy is in percent, UR is the reported uncertainty metric, and K denotes thousands of parameters.

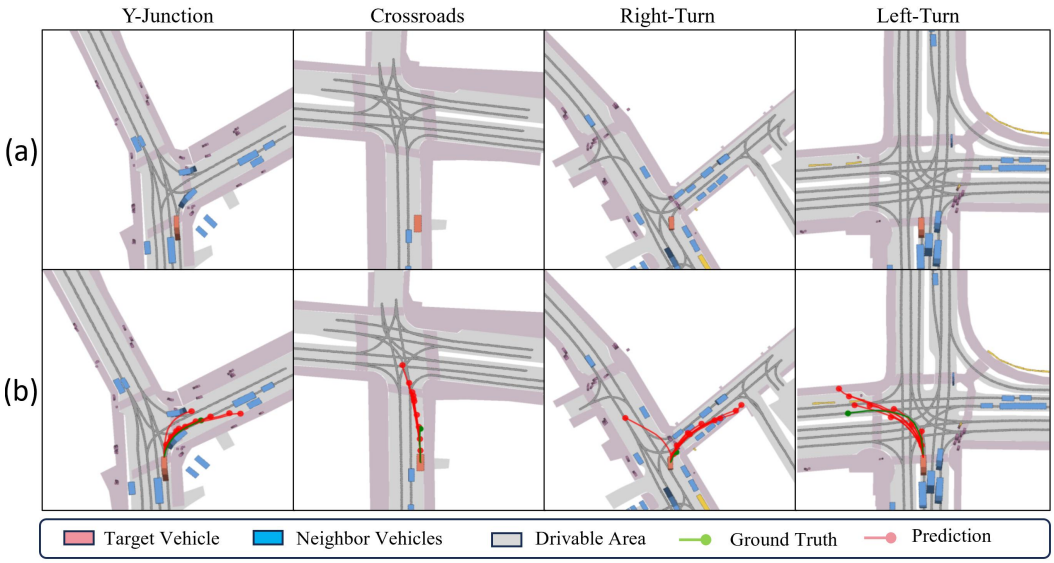

Figure 7: Trajectory prediction examples on nuScenes. The top row shows scene context, and the bottom row shows BOLT's predictions.

H Extended Qualitative Analysis

H.1 nuScenes Prediction Examples

Fig. 7 presents BOLT's predictions in four urban scenarios: a Y-junction, a crossroads, a right turn, and a left turn. The top row shows the scene context, while the bottom row overlays predicted trajectories and ground-truth future motion. The examples illustrate how predictions vary with road geometry and capture alternative future paths and travel distances.

Y-junction and crossroads. At the Y-junction, the predicted trajectories bend toward the outgoing branch followed by the ground truth, with variations in curvature and endpoint location. At the crossroads, the predictions primarily continue along the target vehicle's current direction, while differing in how far the vehicle travels. These examples highlight two aspects of prediction: identifying a plausible continuation and estimating progress along that path.

Right and left turns. In the right-turn scene, several predictions follow the ground-truth turning direction, while another extends toward a different road branch. In the left-turn scene, the predictions

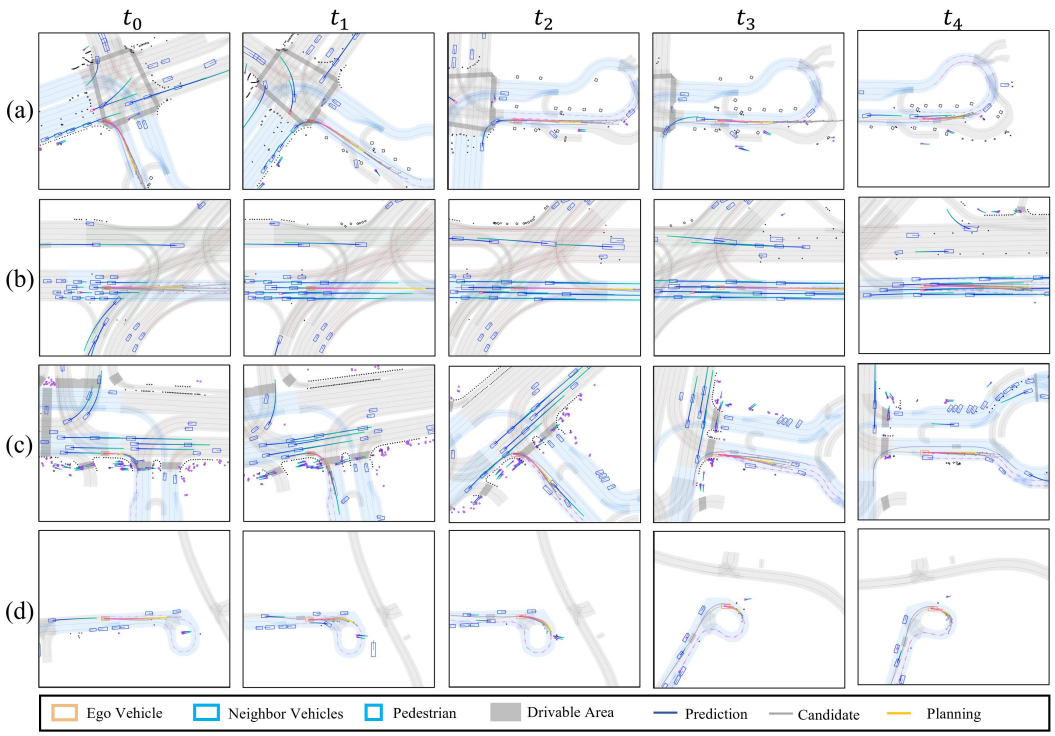

Figure 8: Closed-loop nuPlan rollouts: (a) a right-turn and lane-transition scene, (b) dense-traffic straight driving, (c) a right turn near pedestrians, and (d) a roundabout. Columns show successive times t_0-t_4. The legend identifies ego and surrounding participants, candidate trajectories, predictions, and the selected plan. View orientation and centering vary across snapshots.

capture the overall turning maneuver but spread across different endpoints on the outgoing road. The presence of predictions close to the ground truth alongside more distant alternatives illustrates both multimodal coverage and remaining uncertainty. These visualizations complement the aggregate metrics in Table 1 by showing trajectory alignment and variation in individual scenes.

H.2 nuPLAN CLOSED-LOOP ROLLOUTS

Fig. 8 shows five successive snapshots for each of four scenarios. The views are recentered or rotated as the ego moves; progress should therefore be interpreted relative to the road and other participants rather than from displacement in image pixels.

Turning and lane transition. In row (a), the ego proceeds from the junction approach into the curved outgoing road. Candidate trajectories initially span several possible continuations, and the selected plan follows the turn as the local scene evolves. The sequence illustrates successive prediction and planning updates across a substantial change in heading and road geometry.

Dense-traffic straight driving. Row (b) contains multiple vehicles in adjacent lanes with overlapping prediction and candidate trajectories. The selected ego plan continues along the traffic corridor over the displayed snapshots. This case emphasizes interaction in a geometrically simple route: planning must account for nearby vehicle motion even when the overall maneuver remains straight driving.

Turning near pedestrians. In row (c), pedestrians are visible around the junction while the ego follows the curved route. The scene requires the planner to retain non-vehicle participants in its contextual representation and proximity evaluation. Dedicated behavior features are assigned to motor vehicles, while pedestrian motion remains represented through the scene and prediction branches. This case illustrates how vehicle behavior conditioning operates alongside a scene representation that includes other participant types.

Roundabout. Row (d) follows the approach and progression around a tight curved road segment. The selected trajectory adapts to the changing local heading while surrounding vehicles remain in the scene. This geometry motivates the curvature and lateral-motion terms used in refinement and provides a complementary setting to the straight-driving example.

Across these examples, BOLT produces successive plans in different geometric and interaction settings. Quantitative support for the behavior and refinement components comes from the ablations and the scored comparisons, rather than from visual smoothness alone.