

---

# Bidirectional Motion Transformer for Safety-Critical Traffic Scenario Generation

---

Anonymous Author(s)

Affiliation

Address

email

## Abstract

1        Scenario-based testing is essential for validating the performance of autonomous  
2        driving (AD) systems. However, such testing is limited by the scarcity of long-  
3        tailed, safety-critical scenarios in existing datasets collected in the real world.  
4        To tackle the data issue, we propose the Adv-BMT framework, which augments  
5        real-world scenarios with diverse and realistic adversarial interactions. The core  
6        component of Adv-BMT is a bidirectional motion transformer (BMT) model  
7        to perform inverse traffic motion predictions, which takes the last frame of the  
8        scenario as input and reconstruct the traffic in the inverse of chronological order  
9        till the initial time step. The Adv-BMT framework is a two-staged pipeline: it first  
10       conducts adversarial initializations and then inverse motion predictions. Different  
11       from previous work, we do not need any collision data for pretraining and are able  
12       to generate realistic and diverse collision interactions. Our experimental results  
13       validate the quality of generated collision scenarios by Adv-BMT: training in  
14       our augmented dataset would reduce episode collision rates by 20% compared to  
15       previous work. The code will be made available.

## 16    1 Introduction

17    In recent years, autonomous driving (AD) agents have achieved unprecedented performance in urban  
18    environments [4, 7, 19]. However, handling corner traffic situations, especially collision scenarios,  
19    remains a major challenge. One cause is that safety-critical scenarios are missing from real-world  
20    driving datasets due to high costs and risks of data collections. Without enough collision training data,  
21    it is hard for the autonomous driving (AD) planners and prediction models learn to drive safely in  
22    challenging and risky scenarios. This motivates the need for simulating collision traffic that emulates  
23    real-world traffic accidents.

24    One key question is regarding the desired qualities of safety-critical traffic simulations. Previous  
25    work [11] has argued that safety-critical scenarios should be both realistic and diverse, as well  
26    as challenging. Realism means that adversarial behaviors be consistent with real-world traffic  
27    distributions and align with human driving patterns; while diversity refers to different collision  
28    interactions on limited number of original driving logs. Previous works [11, 21, 14, 10] has enhanced  
29    the realism of simulated adversarial interactions. They leverage learned real-world traffic priors and  
30    optimized predictions on collision-encouraging objectives. However, from our observations and  
31    evaluations, generated behaviors from these methods have a lack of diversity and a relatively low  
32    collision success rates.

33    In order to address the limitations of previous work, we design our framework, namely the Adv-  
34    BMT, for realistic and diverse collision interaction augmentations from real-world data. Adv-BMT  
35    has a motion prediction unit that is conditioned on the final state of a traffic scene. This allows  
36    Adv-BMT to "imagine" various collision situations from the current driving log information, and

pass these collision frames to the prediction unit to reconstruct the complete interaction trajectories. Furthermore, to ensure realistic interactions, we implement a rule-based check to filter out unrealistic candidates with sharp turnings and unrealistic position changes. With this design, the prediction unit is able to reconstruct diverse collisions proposed by Adv-BMT adversarial initializer, and discard implausible generations.

In the Adv-BMT’s motion prediction unit is our bidirectional motion transformer (BMT) for traffic simulations. Similar to existing models [20, 12, 22], BMT tokenizes the continuous motion trajectories into discretized tokens and performs the next token prediction in an auto-regressive manner. The difference lies on the tokenization process. BMT processes two sets of motion tokens, one for forward prediction and one for reverse prediction with respect to chronological order. Through training, BMT learns to predict from the current frame to the future timestep in forward predictions, as well as from the current frame to the from the past frame in reverse predictions. On one set of training scenarios, we train two sets of motion tokens for bidirectional next token predictions. Leveraging the ability of BMT for reverse prediction with optionally forward predictions (details in section 3.5), we achieve diverse and realistic collision interaction augmentations.

As a summary, Adv-BMT is a two-staged pipeline: first, it initializes diverse collision events by introducing an adversary agent colliding with ego vehicle; then, it reconstructs the adversarial trajectories via BMT’s reverse predictions. In the output, The adversarial agent is added to the original scenario, maintaining realistic interactions with surrounding traffic. An overview of our framework is illustrated in Fig. 2. We summarize our contributions as follows: (1) We introduce the bidirectional motion transformer with temporally reversible motion tokenizations; (2) We develop Adv-BMT for controllable safety-critical traffic simulations; (3) We validate the quality of generated scenarios and show improvements in driving safety.

## 2 Related Work

**Driving Motion Predictions and Simulations** Recent work in traffic simulation have utilized transformer-based sequence modeling upon discretized motion representations, and perform motion token prediction in an auto-regressive manner. Trajenglish [9] discretizes motion representation using a k-disk based tokenization scheme to represent position and angle difference for relative movement in the agent’s local frame; but the performance is sensitive to agent ordering, and agents decoded first might not be as reactive to the rest of the predicted agents. MTR++ [13] avoids discrete tokenization and directly represent motion on continuous space on Gaussian mixture distributions. This design is efficient for multi-modal generations but the diversity across modes rely on the learned intention queries, and need to align well with the behavioral semantics. SMART [20] introduces decoder-only architecture on motion prediction for real-time simulations. MotionLM [12] tokenizes driving motion discretizing trajectory deltas and model on temporally causal transformer decoder. BehaviorGPT [22] introduces the next patch prediction task, predicting a chunk (i.e. one second) of future motions at once instead of a single step (0.1 second) prediction, which achieves much higher inference efficiency and reduces compounding errors.

**Safety-Critical Scenario Generations** In previous work, STRIVE [11] enhanced diversity through modeling traffic a latent space using variational autoencoders but required computationally expensive per-scenario optimization. CAT [21] leverages a AD planner and develop a simulator-based closed-loop reinforcement learning environment, where adversarial behaviors are dynamically generated online using the given planner. MixSim [15] generates realistic reactions by exploring future goals and re-simulates the inferred trajectories using a trained policy for interactions; however, the re-simulations are exponentially expensive to the number of agents and thus impossible to use for generating interactive traffic online. Furthermore, Advsim [17] generates adversarial motions by directly perturbing actor trajectories in real-world traffic scenarios using a kinematic model and optimize on a black-box adversarial loss in a liDAR-based system. Safesim [3] uses a diffusion model with a test-time collision-sensitive guidance loss to control the collision type and adversarial agent selections; however, the quality of generations highly depend on initial collisions data collections, whose performance might not be diverse and realistic enough.

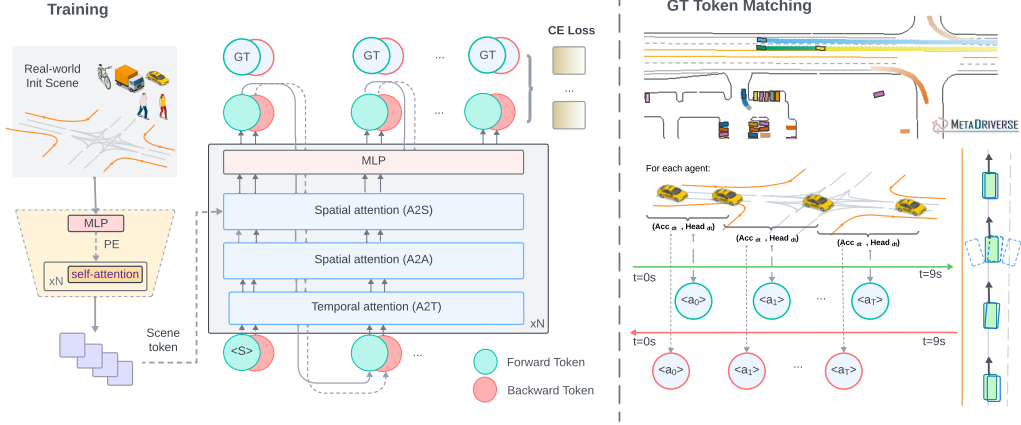


Figure 1: Framework of BMT. (1) BMT learns from real-world driving dataset. BMT has a transformer-based model architecture, with the initial prediction states as input, and predict the following sequence of motion tokens. (2) We tokenize the ground-truth trajectories from the training scenarios by matching the minimum corner distance.

### 3 Methodology

In this section, we introduce the bidirectional motion prediction task as a next-token prediction problem and then describe the key designs in the Adv-BMT framework.

#### 3.1 Problem Formulation

Consider a scene with  $N$  agents. The trajectory of agent  $i$  is denoted by  $\tau^i = \{\tau_0^i, \tau_1^i, \dots, \tau_T^i\}$ , where each state  $\tau_t^i \in \mathbb{R}^d$  includes motion attributes such as position, velocity, and heading. We define a prediction direction indicator  $D \in \{\text{forward}, \text{reverse}\}$ . For each direction  $D$ , we construct one ordered sequence of motion tokens:

$$\mathbf{Z}^{i,D} = \{z_0^i, z_1^i, \dots, z_T^i\}, \quad z_k^i = \begin{cases} \text{Tokenize}(\tau_k^i), & D = \text{forward} \\ \text{Tokenize}(\tau_{T-k}^i), & D = \text{reverse} \end{cases} \quad (1)$$

The goal is to model the next-token prediction distribution over discrete motion tokens in either temporal direction, conditioned on the sequence history and the scene context  $\mathcal{M}$ :

$$P(z_{k+1}^1, z_{k+1}^2, \dots, z_{k+1}^N \mid \mathbf{Z}_{0:k}, \mathcal{M}, D) \quad (2)$$

Here,  $\mathbf{Z}_{0:k}$  denotes history token sequence for all agents up to step  $k$  in time direction  $D$ . The decoder autoregressively generates the next token  $z_{k+1}^i$  by attending to past tokens and scene token embedding  $\mathcal{M}$ .

When  $D = \text{forward}$ , the model behaves as a standard motion predictor. When  $D = \text{reverse}$ , the model conducts reverse-time inference and predicts earlier states, conditioned on the final agent configurations. Together, the model is able to perform controllable generations of traffic outcomes such as collisions.

#### 3.2 Temporally-Reversible Motion Tokenizations

Our bidirectional motion tokenization is derived from a bicycle dynamics model. It includes two sets of tokenizations: forward and reverse. Both are defined over the same shared token space—a discretized grid of acceleration and yaw rate pairs. Formally, we uniformly quantize the control space of accelerations  $a \in [-a_{\max}, a_{\max}]$  and yaw rates  $\delta \in [-\delta_{\max}, \delta_{\max}]$  into  $K$  bins each, yielding a total of  $K^2$  discrete motion tokens.

During forward tokenization, we simulate all  $K^2$  candidate motion tokens starting from each agent state at time  $t$ , then select the token whose predicted next state  $\hat{\tau}_{t+1}$  best matches the true next state  $\tau_{t+1}$ , minimizing the reconstruction error in position, heading, and optionally velocity. This process maps the continuous trajectory into a sequence of discrete motion tokens  $\{z_0, z_1, \dots, z_T\}$ . Conversely, in reverse tokenization process, we reverse the temporal order of the trajectory and treat the final state (e.g., a collision) as the starting point. For each step, we simulate each token’s effect reversely in time (by inverting the time step,  $\Delta t \rightarrow -\Delta t$ ), and find the token that reconstructs the previous state  $\tau_t$  from  $\tau_{t+1}$ . This generates a reversed token sequence  $\{\tilde{z}_0, \tilde{z}_1, \dots, \tilde{z}_T\}$ , where  $\tilde{z}_0$  corresponds to the final state in forward time. Both directions use the same motion token space but differ in the simulation direction and token interpretation.

### 3.3 Min-error Token Matching

We define 33 evenly spaced values for acceleration and headings correspondingly, resulting in 1089 unique (acceleration, heading) pairs, each pair represents a token. With a 0.5-second time interval, a search is conducted from candidate token vocabularies, and selects the candidate token that minimizes the distance between the ground-truth and candidate contours at the current frame. For either reverse prediction or forward time, we build our tokens step-by-step in the order of prediction time direction, starting from the corresponding last frame and reconstructing till its first predicted frame. At each step with current agent position, heading, and velocity with the next state, we search over a discrete grid of (acceleration, steering angle) pairs for the best candidate motion token to minimize the reconstruction error for the  $i$ -th candidate action for agent  $n$ :

$$a_n^* = \arg \min_i \left( \mathcal{E}_n^{(i)} = \frac{1}{P} \sum_{p=1}^P \left\| \mathbf{c}_n^{(i,p)} - \hat{\mathbf{c}}_n^{(p)} \right\|_2 \right) \quad (3)$$

where  $\mathbf{c}_n^{(i,p)} \in \mathbb{R}^2$  is the  $p$ -th polygon corner of the reconstructed contour under candidate  $i$ ,  $\hat{\mathbf{c}}_n^{(p)} \in \mathbb{R}^2$  is the corresponding ground-truth contour point,  $P$  is the total number of contour points, and  $a_n^*$  is the index of the best discrete action with the minimal error.

### 3.4 Bidirectional Motion Transformer (BMT)

The main architecture is shown in Fig. 1. The same model can work for forward prediction or reverse predictions.

**Scene Encoder** BMT has a scene encoding component, which is used to obtain embeddings for scenario contexts during motion predictions. We use MLP encoders to get separate embeddings for map polylines, traffic lights, and agent initializations. Then, we use Fourier-encoded edge features [16] to represent the spatial and directional information between these encoded environment entities, which are then passed to the transformer encoder with self-attention layers for the relational encodings.

**Motion Predictor** The prediction decoder predicts subsequent motion tokens in an autoregressive manner with only initial agent information for the first frame, along with the scene embedding obtained from the Scene Encoder. The motion decoder incorporates self-attention over the initial agent token embeddings, and three relation computations separately: agent-to-agent (a2a), agent-to-time (a2t), and agent-to-scene (a2s), with each relation embedding then passed to its cross-attention layers. The output agent embeddings are concatenated and repeated a number of times. The output agent motion embeddings are mapped to the vocabulary of discretized motion tokens through MLPs with logits.

**Learning Objectives** In training, BMT predicts the sequence of motion tokens and use the cross-entropy objective to match the joint action distribution of the observed behaviors in the dataset during training:

$$\max_{\pi_\theta} \mathbb{E}_{\mathcal{D}} \left[ \sum_{t=1}^{T_{\text{pred}}} \sum_{i \in \mathcal{I}} \log \pi_\theta \left( a_{t,i}^{\text{GT}} \mid o_{t,i} \right) \right] \quad (4)$$

During inference, we generate motion tokens autoregressively using nucleus sampling (top- $p$  sampling) to enhance diversity and maintain plausibility. To prevent the exposure bias during autoregressive inference, our model rolls out upon predicted trajectory reconstructed using our chosen motion tokens, instead of on the ground-truth agent trajectories in a given range of timesteps. We illustrate the training process in Fig. 1

### 3.5 Adv-BMT

The overview of Adv-BMT is illustrated in Fig.2. The framework mainly consists of three steps: (1) it adds an adversarial agent (ADV) with a specified collision state into the current scenario; (2) it predicts reversely for the adversarial trajectory; (3) it performs rule-based checks and reject physically implausible ones.

**Initialize Diverse Collisions** While existing works select a neighbor agent and modify the behavior to attack the ego agent, Adv-BMT inserts new agents (ADV) with diverse collision initializations for different opponent interactions. We visualize an example result in 3. Formally, we define the collision state as  $(\text{position}_{\text{coll}}, T_{\text{coll}}, \text{speed}_{\text{coll}}, \text{heading}_{\text{coll}})$ . The collision time  $T_{\text{coll}}$  is sampled from the first second to the last timestep of the ego trajectory length. Similarly, ADV’s collision headings are randomly sampled. ADV’s collision position can be calculated from  $\text{heading}_{\text{coll}}$  and the ego vehicle’s position at  $T_{\text{coll}}$ , with the ADV contour intersecting with the ego vehicle’s contour. Last but not least,  $\text{speed}_{\text{coll}}$  is also calculated from sampling a variable offset from the ego vehicle’s speed at  $T_{\text{coll}}$ .

**Reverse Prediction on Adversarial Initializations** We use the sampled collision states as inputs of BMT for reverse predictions, and obtain the trajectory to augment the original scene with a new adv agent. During BMT inference, we teacher-force the ground-truth token sequence of ego agent and traffic agents, to keep the originality of the scenario as much as possible (since for BMT could predict agents to driving on a different lane choice, an makes it a totally different scenario). By teacher-forcing non-collided vehicle agent behaviors, we reconstruct the adversarial agent trajectory to fit in the input scenario. Reconstructing entirely new scenarios is out of the scope of our current work, but BMT could realize both normal and safety-critical utilizing different prediction directions.

**Rule-based Rejection Samplings** We design Adv-BMT to have diverse initializations, but do not guarantee the realism of the collision outcomes. To address this issue, we implement a rule-based rejection mechanism for ADV candidates. The process is straightforward: we first measure the driving distance and average speed of ADV candidates; if it moves too short and mostly wanders at the designated position waiting for the ego vehicle, then it is considered invalid. Meanwhile, we check the max curvature (the rate of change for heading): given a candidate ADV trajectory, we compute the curvature constraint using:  $\kappa_t = \Delta\theta_t / \Delta s_t$ , where  $\Delta\theta_t$  is the absolute heading change between time steps, and  $\Delta s_t$  is the displacement between consecutive positions. Adv-BMT rejects a prediction if  $\max_t(\kappa_t) > \kappa_{\text{threshold}}$ , where  $\kappa_{\text{threshold}} = 0.8$  is a predefined curvature limit that we found useful. We check the curvature to ensure it does not exceed our defined threshold, and reject candidate ADVs with unrealistic changes in headings. With our straightforward rule-based rejection sampling mechanism, we are able to maintain only realistic collision events between the ADV and the ego vehicle.

**Forward refinement for reactive traffic** In real-world collision events, neighbor vehicles will be affected by the collision vehicles and react like sudden lane changes and brakings. With this concern, we implemented a variation of Adv-BMT with a further forward prediction to re-simulate traffic agents upon predicted ADV trajectory, with teacher-forcing predicted adversarial trajectory. In experiments, we refer it as Adv-BMT (Bi) to indicate bi-directions.

## 4 Experiments

In section 4.1, we evaluate BMT model on motion prediction performance. In section 4.2, we compare Adv-BMT with other safety-critical methods. In section 4.3 we train an reinforcement learning agent in augmented datasets. For augmentation, we use real-world traffic data from the Waymo Open Motion Datasets (WOMD) [5].

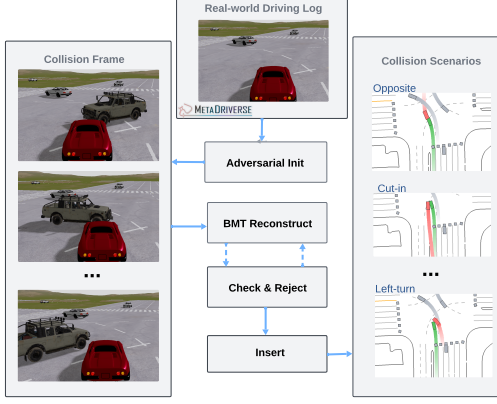


Figure 2: Adv-BMT overview.

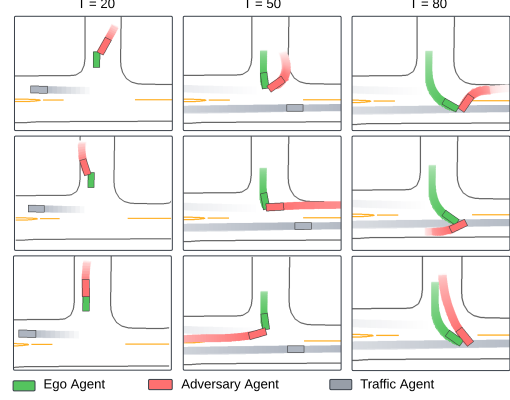


Figure 3: Diverse collision scenarios generated by Adv-BMT.

Table 1: Evaluation for BMT on scenario realism on different prediction modes.

	Context	SFDE ↓	SADE ↓	minSFDE ↓	minSADE ↓	FDD ↑	ADD ↑	Collision
Forward	1s	<b>3.043</b>	<b>1.019</b>	1.908	<b>0.693</b>	9.241	3.240	<b>0.026</b>
Forward	0s	3.180	1.297	2.144	0.927	<b>10.263</b>	4.177	0.028
Reverse	0s	3.411	1.380	2.158	0.975	8.234	2.913	0.031
Bidirections	0s	3.059	2.111	<b>1.831</b>	1.536	7.618	<b>6.270</b>	0.035

#### 4.1 Realism of Adv-BMT

**Setting** We measure scenario generation on four settings for our BMT model: forward prediction with or w/o agent history, Reverse prediction without agent history, and bidirectional predictions (first reverse then forward). We measure on open-loop scenario generation metrics: scenario final displacement errors (SFDE) and scenario average displacement errors (SADE), which measures the average displacement error at the final step and average according to all time steps on the best prediction mode; final displacement diversity (FDD) and average displacement diversity (ADD) for prediction diversities, measuring the spread of final positions and trajectory variations over time across different prediction modes; besides, we measure the overall collision rates for realistic interactions. The results are presented in Table 1. Prediction Mode is set to 6.

**Analysis** Results from Table 1 indicate that BMT predictions in both direction modes exhibit realism in scenario generation. Furthermore, by conditioning on a 1-second history, the forward prediction achieves better SADE and SFDE scores, indicating more temporally consistent predicted trajectories. Due to the design of Adv-BMT, which initializes ADV-ego collisions only for the last frame, the reverse prediction doesn't have access to any longer agent histories. Compared to single-pass prediction, Bidirectional prediction exhibit lower prediction accuracy in ADE metrics, indicating a harm in the realism of generations.

In terms of prediction diversities, forward prediction exhibits higher FDD and ADD scores than the reverse prediction. This indicates that predicting histories is more constrained, but predicting future motions encourages more explorations for variant directions. This further validates the necessity of Adv-BMT's diverse collision initialization design to force different collision outcomes. Despite these differences, the overall collision rate remains reasonable and comparable for all prediction modes, and indicates that both models effectively generate realistic interactions.

#### 4.2 Safety-critical Evaluation on Generation

We compare Adv-BMT with three baseline methods, namely STRIVE [11], CAT [21], and SEAL [14]. We evaluate the quality and efficiency of generated collisions among all methods. We measure the adversarial attack success rate (SDC-ADV collision rate), collision rates between adversary agents and traffic agents (ADV-BV Collision rate), indicating the realistic interactions. Additionally, we evaluate



Figure 4: Comparison among adversarial behaviors of different methods.

Table 2: Evaluation for Safety-critical Scenarios

	SDC-ADV Collision $\uparrow$	ADV-BV Collision $\downarrow$	Velocity JSD $\downarrow$	Accel. JSD $\downarrow$	FDD $\uparrow$	SDD $\uparrow$
CAT [21]	0.470	0.271	0.137	0.480	0.0	0.0
SEAL [14]	0.582	<b>0.261</b>	0.239	0.561	0.0	0.0
STRIVE [11]	0.534	0.358	<b>0.117</b>	<b>0.432</b>	0.004	0.001
Adv-BMT	<b>1.000</b>	0.391	0.142	0.435	0	<b>2.344</b>
Adv-BMT (Bi)	0.997	0.373	0.188	0.534	<b>0.038</b>	2.257

the difference between the predicted distribution and the ground-truth distribution on velocities (VEL JSD) and accelerations (ACC JSD). These two metrics indicate the realism of generated adversarial motions.

**Settings** We use a random set of 500 WOMD scenarios, each as a base environment for adversarial scenario generation. For each scenario, each method predicts six different adversarial trajectories. To be consistent with baseline methods in this evaluation, we force Adv-BMT to choose an existing neighbor agent and modify its behavior as the predicted adversary.

**Quantitative Results** Table 2 shows the quantitative results for scenario generation metrics of all methods. Adv-BMT outperforms others in terms of adversarial attack success rate. This validates Adv-BMT’s design of collision initialization + reverse predictions, which couldn’t achieve by other methods. While Adv-BMT generates highly adversarial behaviors, it also preserves diversity of traffic interactions compared to baselines. Note tha FDD and SDD metrics suggest that our baselines generate nearly the same adversarial trajectory on the given scene, while Adv-BMT is able to generate diverse behaviors at different prediction mode. The JSD metrics suggests a slight better realism on velocities and accelerations for baselines compared to Adv-BMT, showing the trade-off between diversity and velocity/acceleration realism. Our BMT model is able to achieve realistic motion predictions indicted by Waymo Open Sim Agent Challenge metrics [8], which we append in our appendix. Overall, our safety-critical evaluations indicate that Adv-BMT is an efficient framework for realistic and diverse adversarial behaviors.

**Visualizations** We simulate Adv-BMT scenarios in MetaDrive [7] simulators using both BEV and 3D rendering. In Fig. 3, we show diverse Adv-BMT adversarial behaviors in the collision directions on the same ego agent in an intersection scenario from WOMD. Adversarial agents (in red) come from different lanes and collide with the ego agent at different angles and facets. Besides, the generated adversarial agent follows the traffic rules and maintains realistic driving patterns. Apart from diverse collision directions, Adv-BMT generates collisions at diverse timings. Adv-BMT adversarial agents cover collisions with the ego before entering the intersection, during the middle of the intersection, and after turning to the main lane. With a limited number of real-world maps and logs, Adv-BMT is able to generate diverse collision behaviors and timings, making it suitable for AD testing and adversarial training. A visual comparison among Adv-BMT and baseline methods is shown in Fig. 4, where other methods fail to generate an adversarial trajectory but Adv-BMT is able to insert a realistic ADV and triggers collisions .

**Generation Speed** Adv-BMT and CAT generate safety-critical scenarios significantly faster than other baselines. Adv-BMT generates one scenario in 1.02s compared to SEAL’s 2.36s and STRIVE’s 9.53s, achieving great efficiency gains. With details shown in Table 7 in Appendix.

### 4.3 Adversarial Learning

To further validate the value of Adv-BMT scenarios in downstream autonomous driving (AD) tasks, we train a reinforcement learning agent within augmented scenarios with collision interactions. To determine the quality of augmented training scenarios, we measuring the driving performance and safety of the learned AD agent compared to learning in original training set. In our experiment, we conduct two sets of trainings. (1) open-loop training: agent is trained in the fixed training set with adversarial scenarios generated based on ground-truth ego trajectories. (2) closed-loop training: an adaptive adversarial agent attacks on current ego agent based on its recent rollout trajectory records. Results of both trainings are shown in table 3 and table 4 respectively.

**Settings** The training set contains 500 real-world scenarios randomly selected from WOMD training set. Each waymo scenario is augmented with one collision scenario, so that the ratio between waymo and safety-critical scenes is 1:1. For each method, we have one set of augmented training set. For the adaptive adversarial learning experiment, we implement an adaptive generator for Adv-BMT and CAT [21] (discard other methods due to lower inference time). We train a TD3 agent for 1 million steps using 8 random seeds to ensure robustness. We conduct the training in the MetaDrive simulator and use the Twin Delayed DDPG (TD3) algorithm for reinforcement learning (hyper-parameters listed in Appendix). We measure the average reward, average cost, average route completion rate (Compl.), and average episode cost (cost sum) for driving performance measure. We further conducted ablation studies with results shown in Appendix.

**Quantitative Results** To evaluate the impact of adversarial training using Adv-BMT-generated scenarios, we assess policy performance across two distinct validation environments: (1) 100 Waymo Validation Environments, which consist of unmodified real-world driving scenarios from WOMD validation set, and (2) 100 Adv-BMT Environments, which is the augmented collision scenarios from the 100 validation scenes. Table 3 presents the evaluation results, demonstrating how training with Adv-BMT improves policy resilience in both standard and adversarial settings. A detailed analysis of these results follows in the subsequent sections.

**Analysis** 1) Improved Policy Robustness in Adversarial Settings. Table 3 presents the evaluation results comparing policies trained with Adv-BMT-generated scenarios against baselines in both real-world and safety-critical validation environments. Adv-BMT demonstrates superior performance in reducing collision rates while maintaining high completion rates. In the Collision Validation Environments, Adv-BMT-trained policies achieve the lowest collision rate while preserving competitive completion rates. This highlights the effectiveness of Adv-BMT in exposing policies to diverse and high-risk scenarios, leading to improved safety performance. Furthermore, Adv-BMT achieves the lowest sum cost, suggesting that policies trained on Adv-BMT-generated data are more efficient in handling safety-critical interactions. Besides, the results from adaptive generators shows a slight improvement in performance compared to training in fixed scenario sets.



Table 3: Open-loop learning evaluation.

(a) Evaluation in the Waymo Validation Environments

Training Scenarios	Reward $\uparrow$	Cost $\downarrow$	Compl. $\uparrow$	Coll. $\downarrow$	Cost Sum $\downarrow$
Waymo [5]	32.03 $\pm$ 4.27	0.39 $\pm$ 0.07	0.72 $\pm$ 0.05	0.14 $\pm$ 0.02	1.41 $\pm$ 0.35
CAT [21]	30.37 $\pm$ 3.89	0.39 $\pm$ 0.05	0.71 $\pm$ 0.05	0.14 $\pm$ 0.02	1.73 $\pm$ 0.39
STRIVE [11]	31.30 $\pm$ 3.59	0.40 $\pm$ 0.04	0.73 $\pm$ 0.05	0.13 $\pm$ 0.03	1.51 $\pm$ 0.40
SEAL [14]	29.94 $\pm$ 5.14	0.39 $\pm$ 0.05	0.71 $\pm$ 0.04	0.12 $\pm$ 0.02	1.63 $\pm$ 0.44
Adv-BMT	31.47 $\pm$ 3.21	0.38 $\pm$ 0.03	0.73 $\pm$ 0.04	<b>0.11 <math>\pm</math> 0.02</b>	<b>1.35 <math>\pm</math> 0.40</b>
Adv-BMT (Bi)	<b>33.22 <math>\pm</math> 1.83</b>	<b>0.36 <math>\pm</math> 0.03</b>	<b>0.74 <math>\pm</math> 0.03</b>	0.12 $\pm$ 0.02	1.39 $\pm$ 0.22

(b) Evaluation in the Adv-BMT Environments

Training Scenarios	Reward $\uparrow$	Cost $\downarrow$	Compl. $\uparrow$	Coll. $\downarrow$	Cost Sum $\downarrow$
Waymo [5]	37.01 $\pm$ 6.16	0.64 $\pm$ 0.09	0.60 $\pm$ 0.07	0.30 $\pm$ 0.02	2.96 $\pm$ 0.63
CAT [21]	36.77 $\pm$ 4.95	0.62 $\pm$ 0.05	0.62 $\pm$ 0.05	0.29 $\pm$ 0.02	3.09 $\pm$ 0.56
STRIVE [11]	37.72 $\pm$ 5.38	0.63 $\pm$ 0.06	0.63 $\pm$ 0.06	0.29 $\pm$ 0.04	2.92 $\pm$ 0.68
SEAL [14]	35.74 $\pm$ 6.36	0.67 $\pm$ 0.08	0.60 $\pm$ 0.06	0.31 $\pm$ 0.01	2.97 $\pm$ 0.34
Adv-BMT	37.33 $\pm$ 3.57	0.62 $\pm$ 0.03	0.63 $\pm$ 0.04	<b>0.25 <math>\pm</math> 0.05</b>	<b>2.41 <math>\pm</math> 0.43</b>
Adv-BMT (Bi)	<b>39.55 <math>\pm</math> 2.94</b>	<b>0.59 <math>\pm</math> 0.04</b>	<b>0.65 <math>\pm</math> 0.02</b>	0.27 $\pm$ 0.04	2.74 $\pm$ 0.54

Table 4: Closed-loop learning results

(a) Waymo Validation Environments

Adaptive Generator	Reward $\uparrow$	Cost $\downarrow$	Comp. $\uparrow$	Coll. $\downarrow$	Cost Sum $\downarrow$
GT	55.08 $\pm$ 0.0	0.04 $\pm$ 0.0	0.97 $\pm$ 0.0	0.00 $\pm$ 0.0	0.04 $\pm$ 0.0
CAT	32.15 $\pm$ 2.89	<b>0.38 <math>\pm</math> 0.03</b>	0.74 $\pm$ 0.04	0.10 $\pm$ 0.00	2.02 $\pm$ 0.24
Adv-BMT	<b>33.13 <math>\pm</math> 4.11</b>	0.39 $\pm$ 0.03	0.74 $\pm$ 0.03	<b>0.09 <math>\pm</math> 0.00</b>	<b>1.25 <math>\pm</math> 0.52</b>

(b) Adv-BMT Environments

Adaptive Generator	Reward $\uparrow$	Cost $\downarrow$	Comp. $\uparrow$	Coll. $\downarrow$	Cost Sum $\downarrow$
GT	60.53 $\pm$ 0.00	0.78 $\pm$ 0.00	0.95 $\pm$ 0.00	0.74 $\pm$ 0.00	8.09 $\pm$ 0.00
CAT	39.47 $\pm$ 3.88	0.62 $\pm$ 0.05	0.63 $\pm$ 0.04	0.22 $\pm$ 0.02	2.51 $\pm$ 0.45
Adv-BMT	<b>40.40 <math>\pm</math> 6.39</b>	<b>0.57 <math>\pm</math> 0.04</b>	0.63 $\pm$ 0.05	0.22 $\pm$ 0.04	<b>2.48 <math>\pm</math> 0.97</b>

In the open-loop learning, among the baselines, STRIVE achieves the highest reward (37.72) in collision environments, but this comes at the cost of a higher collision rate (0.29). SEAL and CAT exhibit higher costs and collision rates, which suggests that adversarial training leads to suboptimal behaviors. In contrast, agent training with Adv-BMT attacks results in improvement on both safety and task completion. We conduct an ablation study on Adv-BMT plus the forward refinement method, and results indicate a slightly better performance in average rewards, and average cost compared to Adv-BMT. During closed-loop learning shown in table 4 we can see the significant decrease in average episode cost, however reward and cost remains consistent with open-loop learning. In evaluation with safety-critical environments, there is a slight improvement over rewards, cost, as well as episode sum, comparing to open-loop learned agents.

## 5 Conclusion and Limitations

**Conclusion** This work introduces Adv-BMT, a novel safety-critical scenario generation framework that leverages BMT model to bring diverse and realistic adversarial collisions into real-world traffic scenarios. Adv-BMT consists of three key components: (1) collision initialization, which samples a potential collision outcome as the last frame; (2) reverse prediction, which generates the adversarial pre-collision trajectories; (3) rule-based filtering, which ensures adversarial agents' motions remain realistic and feasible. We evaluate Adv-BMT across multiple benchmarks and experiments and demonstrate that training with generated adversarial scenarios improves AD agent performance in both held-out log-replay and collision test sets with a clear margin.

**Limitations** Following limitations of our current work will be potentially addressed in the future: (1) we only consider adversarial behaviors among vehicle agents, but not among other agent types like pedestrians and bicycles. (2) we only evaluate our adversarial scenarios in RL policies, but adversarial scenarios can also be leveraged in human-in-the-loop learning.

## References

- [1] 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, 2020.
- [2] 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.
- [3] Wei-Jer Chang, Francesco Pittaluga, Masayoshi Tomizuka, Wei Zhan, and Manmohan Chandraker. Safe-sim: Safety-critical closed-loop traffic simulation with diffusion-controllable adversaries, 2024.
- [4] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator, 2017.
- [5] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles Qi, Yin Zhou, Zoey Yang, Aurelien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan, Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting for autonomous driving : The waymo open motion dataset, 2021.
- [6] Quanyi Li, Zhenghao Peng, Lan Feng, Zhizheng Liu, Chenda Duan, Wenjie Mo, and Bolei Zhou. Scenario-net: Open-source platform for large-scale traffic scenario simulation and modeling, 2023.
- [7] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning, 2022.
- [8] Nico Montali, John Lambert, Paul Mouglin, Alex Kuefler, Nick Rhinehart, Michelle Li, Cole Gulino, Tristan Emrich, Zoey Yang, Shimon Whiteson, Brandyn White, and Dragomir Anguelov. The waymo open sim agents challenge, 2023.
- [9] Jonah Philion, Xue Bin Peng, and Sanja Fidler. Trajenglish: Traffic modeling as next-token prediction, 2024.
- [10] Joshua Ransiek, Johannes Plaum, Jacob Langner, and Eric Sax. Goose: Goal-conditioned reinforcement learning for safety-critical scenario generation, 2024.
- [11] Davis Rempe, Jonah Philion, Leonidas J. Guibas, Sanja Fidler, and Or Litany. Generating useful accident-prone driving scenarios via a learned traffic prior, 2022.
- [12] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S. Refaat, Rami Al-Rfou, and Benjamin Sapp. Motionlm: Multi-agent motion forecasting as language modeling, 2023.
- [13] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying, 2024.
- [14] Benjamin Stoler, Ingrid Navarro, Jonathan Francis, and Jean Oh. Seal: Towards safe autonomous driving via skill-enabled adversary learning for closed-loop scenario generation, 2025.
- [15] Simon Suo, Kelvin Wong, Justin Xu, James Tu, Alexander Cui, Sergio Casas, and Raquel Urtasun. Mixsim: A hierarchical framework for mixed reality traffic simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9622–9631, June 2023.
- [16] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains, 2020.
- [17] Jingkan Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. Advsim: Generating safety-critical scenarios for self-driving vehicles, 2023.
- [18] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting, 2023.
- [19] Wayne Wu, Honglin He, Jack He, Yiran Wang, Chenda Duan, Zhizheng Liu, Quanyi Li, and Bolei Zhou. Metaurban: An embodied ai simulation platform for urban micromobility, 2024.
- [20] Wei Wu, Xiaoxin Feng, Ziyang Gao, and Yuheng Kan. Smart: Scalable multi-agent real-time motion generation via next-token prediction, 2024.

- 374 [21] Linrui Zhang, Zhenghao Peng, Quanyi Li, and Bolei Zhou. Cat: Closed-loop adversarial training for safe  
375 end-to-end driving, 2023.
- 376 [22] Zikang Zhou, Haibo Hu, Xinhong Chen, Jianping Wang, Nan Guan, Kui Wu, Yung-Hui Li, Yu-Kai Huang,  
377 and Chun Jason Xue. Behaviorgpt: Smart agent simulation for autonomous driving with next-patch  
378 prediction, 2024.

In the appendix, we include additional technical details and results upon the main paper due to limit of page length. Appendix A gives details on the model architecture and trainings, Appendix B gives details for experiments, and Appendix C and D provides additional results to supplement those in the main paper. In Appendix E, we claim the broader impacts of our methodology.

## A Model Details

### A.1 BMT Architecture

**Scenario Format and Preprocessing** The model takes real-world scenarios with the format from ScenarioNet [6], which unifies real-world datasets including [1] [2] [18].

The model performs a series of preprocessing on both static map features and dynamic agent trajectories. We compute the global boundary to extract a consistent map center and heading. Each map feature is decomposed into a sequence of vectorized segments, where each vector is represented by a start and end point in 3D coordinates. These vectors are then augmented with directional relative positions, segment headings, and lengths. Semantic attributes are binary indicators that encode feature types. The map feature types (lane, crosswalk, broken line, yellow line, stop sign, etc.) are encoded for semantic information. Features are then centralized to map center. Traffic light states are also encoded and aligned with the steps. Similarly, agent trajectories are also centralized for agent feature encode, with information of (position, heading, velocity, shape, and type) extracted across time for temporal sequences. Besides, there is reordering for ego agent and agents to be predicted. We extract a 16-dimensional state vector at each timestep. The preprocessing ensures all trajectories are represented in a consistent spatial frame and translated to an ego-centric coordinate system.

**Tokenization** We formulate motion prediction as an auto-regressive next-token prediction problem, where each motion token corresponds to a discretized control input for a fixed time interval. Tokenization process maps continuous agent motion (acceleration and yaw rate) into discrete 2D bins. The tokenization process considers candidate tokens sampled from a fixed grid of bins, and simulates the resulting motion over a short duration, and then selects the best-matching action by minimizing the contour alignment error between the predicted agent shape and the ground-truth position and heading. We adopt a simplified version of the bicycle model to parameterize agent motion using longitudinal acceleration and yaw rate within predefined bounds: acceleration is limited to the range of  $[-10, 10]$  m/s<sup>2</sup>, and the yaw rate is constrained to  $[-\pi/2, \pi/2]$  rad/s. With predicted motion token sequences, the trajectory can be reconstructed by mapping the token back to the acceleration and yaw angle change.

In both forward and reverse directions, motion tokens are decoded into continuous trajectories using the same tokenizer. Given an initial state  $\tau_t = (x_t, y_t, \theta_t, v_t)$ , the forward decoding process simulates the agent’s next state  $\tau_{t+1}$  by applying a tokenized control action  $z_t = (a_t, \delta_t) \in \mathcal{A}$ , where  $\mathcal{A}$  denotes the discrete token space. This is repeated autoregressively over the sequence of predicted tokens to reconstruct the full trajectory  $\tau_{t:t+T}$ . In contrast, reverse decoding starts from a known future state  $\tau_{t+1}$ , the model evaluates all possible token candidates  $z_t \in \mathcal{A}$ , simulates their inverse dynamics using  $\Delta t \rightarrow -\Delta t$ , and selects the token that best reconstructs the preceding state  $\tau_t$ . The ability to operate in both directions is a key distinction of our approach: forward prediction enables open-loop simulation of future behaviors, while reverse prediction allows us to trace back from a desired outcome (e.g., a collision state) to plausible initiating actions.

**Decoding Architecture.** The decoder follows a GPT-like structure composed of stacked cross-attention layers. Each layer integrates three structured attention modules: agent-to-agent (A2A), agent-to-temporal (A2T), and agent-to-scene (A2S). These modules attend over dynamically constructed graphs defined by spatial or temporal adjacency. Relational information across modalities is captured via multiple embeddings: Agent-to-Agent Relation Embedding, Agent-to-Time Relation Embedding, and Agent-to-Scene Relation Embedding, each encoding context-specific spatial information. For token construction, several embeddings are used, including the Agent Type Embedding, Agent Shape Embedding, Agent ID Embedding, and the Motion Token Embedding which maps discrete control tokens. A Continuous Motion Feature Embedding is applied to embed acceleration and yaw rate attributes. Auxiliary embeddings include the Special Token Embedding for indicating sequence boundaries, and the Backward Prediction Indicator Embedding to distinguish between forward and

backward prediction modes. For each attention edge, a relative relation embedding is computed using a Fourier encoder and added to the key and/or value vectors. The input agent token  $\mathbf{X} \in \mathbb{R}^{B \times T \times N \times d}$  (batch size  $B$ , time steps  $T$ , number of agents  $N$ , hidden dimension  $d$ ) is progressively updated across layers by aggregating contextual features from other agents, their temporal history, and relevant map elements.

**Motion Token Embedding** Specifically, each input agent token to the BMT motion decoder is constructed by summing of embeddings of:

- a. the motion token from the previous step (representing discretized acceleration and yaw rate),
- b. agent shape (length, width, height),
- c. agent type (e.g., vehicle, pedestrian),
- d. agent identifier (embedded optionally),
- e. special token type (e.g., <start>, <end>, or padding),

as well as a continuous motion delta feature embedded via a Fourier encoder. These components are projected into the same hidden dimension and summed to form the input motion token embedding.

**Prediction Heads and Outputs** After processing through all decoding layers, the final hidden state for each valid token is passed through a two-layer MLP head to produce logits over the motion token space:

$$\text{MLP}(\mathbf{h}) = W_2 \cdot \phi(W_1 \cdot \mathbf{h}) \in \mathbb{R}^{K^2},$$

where  $\phi(\cdot)$  denotes the GELU activation and  $K^2$  is the number of discrete motion tokens (from a  $K \times K$  acceleration–yaw bin grid).

The resulting logits are used to predict the next motion token at each time step. During inference, we generate motion tokens using nucleus (top- $p$ ) sampling. Given the predicted logits  $\hat{\mathbf{z}}_{b,t,n} \in \mathbb{R}^{|\mathcal{A}|}$  at each valid location  $(b, t, n)$ , we first compute the probability distribution via softmax:

$$P(a \mid \mathbf{h}_{b,t,n}) = \frac{\exp(\hat{z}_{b,t,n,a})}{\sum_{a'} \exp(\hat{z}_{b,t,n,a'})}, \quad \text{for } a \in \mathcal{A}.$$

We then sort the action space  $\mathcal{A}$  in descending order of  $P(a \mid \mathbf{h}_{b,t,n})$  and identify the smallest prefix set  $\mathcal{A}_p \subseteq \mathcal{A}$  such that  $\sum_{a \in \mathcal{A}_p} P(a \mid \mathbf{h}_{b,t,n}) \geq p$ , where  $p \in (0, 1)$  is the sampling threshold (e.g.,  $p = 0.9$ ). The next motion token is sampled from  $\mathcal{A}_p$  according to the renormalized distribution. Top- $p$  sampling encourages both diversity and plausibility, avoiding low-probability outliers.

**Trajectory Reconstruction** Our model makes a prediction in the interval of 5 time steps (0.5 seconds). To simulate the effect of a motion token over a fixed time step  $\Delta t = 0.5$  s, we adopt midpoint integration based on a simplified bicycle model. In forward predictions, given a current state  $\mathbf{s}_t = (x_t, y_t, \theta_t, v_t)$ , the model computes the next speed and heading as  $v_{t+1} = v_t + a \cdot \Delta t$  and  $\theta_{t+1} = \theta_t + \omega \cdot \Delta t$ . The average speed and heading are then defined as  $\bar{v} = \frac{v_t + v_{t+1}}{2}$  and  $\bar{\theta} = \left( \frac{\theta_t + \theta_{t+1}}{2} \right)$ . In reverse prediction, the process is reverted. In the backward direction, the process is inverted. Given a future state  $\mathbf{s}_{t+1}$ , the model enumerates all possible token candidates and inverts the dynamics:  $v_t = v_{t+1} - a \cdot \Delta t$  and  $\theta_t = \theta_{t+1} - \omega \cdot \Delta t$ . The average quantities  $\bar{v}$  and  $\bar{\theta}$  are computed similarly and used to derive the previous position:

$$x_t = x_{t+1} - \bar{v} \cdot \cos(\bar{\theta}) \cdot \Delta t, \quad y_t = y_{t+1} - \bar{v} \cdot \sin(\bar{\theta}) \cdot \Delta t.$$

**Decoder Training Loss** The decoder produces a logit tensor  $\hat{\mathbf{z}} \in \mathbb{R}^{B \times T \times N \times |\mathcal{A}|}$ , where  $|\mathcal{A}|$  is the number of motion tokens (i.e., discretized acceleration–yaw pairs). The supervision target is the ground-truth token sequence  $\mathbf{z}^* \in \mathbb{N}^{B \times T \times N}$ , derived by tokenizing agent trajectories. A binary mask  $\mathbf{m} \in \{0, 1\}^{B \times T \times N}$  specifies which tokens are valid and should contribute to the training loss. The training objective is computed over all valid entries using the cross-entropy loss:

$$\mathcal{L}_{\text{main}} = \frac{1}{\sum_{b,t,n} m_{b,t,n}} \sum_{b,t,n} m_{b,t,n} \cdot \text{CE}(\hat{z}_{b,t,n}, z_{b,t,n}^*),$$

where CE denotes the standard cross-entropy loss between the predicted logits and the ground-truth discrete token.

Table 5: BMT Model Parameters.

Component	Parameters	Size (MB)
Scene Encoder	902,080	3.44
Map Polyline Encoder	22,656	0.09
Traffic Light Embedding MLP	1,024	0.00
Scene Relation Embedding	117,184	0.45
Scene Transformer Encoder	744,448	2.84
Scene Encoder Output Projection	16,512	0.06
Scene Output Pre-Normalization	256	0.00
Motion Decoder	4,385,025	16.73
Multi-Cross Attention Decoder	2,881,536	10.99
Motion Prediction Head	157,121	0.60
Motion Prediction Pre-Normalization	256	0.00
Agent-to-Agent Relation Embedding	418,432	1.60
Agent-to-Time Relation Embedding	418,432	1.60
Agent-to-Scene Relation Embedding	117,184	0.45
Agent Type Embedding	640	0.00
Motion Token Embedding	139,520	0.53
Agent Shape Embedding	17,152	0.07
Agent ID Embedding	16,384	0.06
Continuous Motion Feature Embedding	217,600	0.83
Special Token Embedding	512	0.00
Reverse Prediction Indicator Embedding	256	0.00
Total	5,287,105	20.17

**Reverse Prediction** During reverse prediction, the model measures metrics separately for forward and reverse token predictions. Let  $\mathbf{b} \in \{0, 1\}^{B \times T \times N}$  be a binary indicator for whether each token comes from reverse prediction. Then we compute separate metrics:

$$\text{Accuracy}^{\text{reverse}} = \frac{\sum m_{b,t,n} \cdot b_{b,t,n} \cdot \mathbf{1}[\hat{z}_{b,t,n} = z_{b,t,n}^*]}{\sum m_{b,t,n} \cdot b_{b,t,n}},$$

$$\text{Entropy}^{\text{reverse}} = \frac{1}{\sum m \cdot b} \sum m_{b,t,n} \cdot b_{b,t,n} \cdot \mathcal{H}(\hat{z}_{b,t,n}),$$

with analogous expressions for forward prediction (i.e., for  $1 - b_{b,t,n}$ ).

**Metrics** To measure the quality and diversity of the model’s predictions during training, we track the perplexity:

$$\text{Perplexity} = \exp \left( - \sum_{a \in \mathcal{A}} \bar{p}_a \log(\bar{p}_a + \epsilon) \right), \quad \text{where} \quad \bar{p}_a = \frac{1}{M} \sum_{i=1}^M \mathbf{1}[\hat{z}_i = a],$$

and  $M$  is the number of valid tokens. We also track the number of distinct tokens used by both predictions and ground truth:

$$\text{Cluster} = \sum_{a \in \mathcal{A}} \mathbf{1}[\bar{p}_a > 0].$$

**Total Loss** The total loss is the sum of all enabled components:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{main}} + \lambda_{\text{map}} \mathcal{L}_{\text{map}} + \lambda_{\text{tg}} \mathcal{L}_{\text{tg-total}},$$

with default weights  $\lambda_{\text{map}} = \lambda_{\text{tg}} = 1$ .

## A.2 Training Details

Our model has 5.2 million trainable parameters, with detailed break downs indicatd in Table 5. We trained BMT model on the training set of Waymo Open Motion Datasets [5]. WOMD contains 480K

Table 6: BMT Training settings.

Forward Prediction		Reverse Prediction	
Hyper-parameter	Value	Hyper-parameter	Value
Training steps	10E6	Training steps	15E6
Batch sizes	2	Batch size	2
Training Time (h)	185	Training Time (h)	310
Sampling Topp	0.95	Sampling Topp	0.95
Sampling temperature	1.0	Sampling temperature	1.0
Learning Rates	3E-4	Learning Rates	3E-4

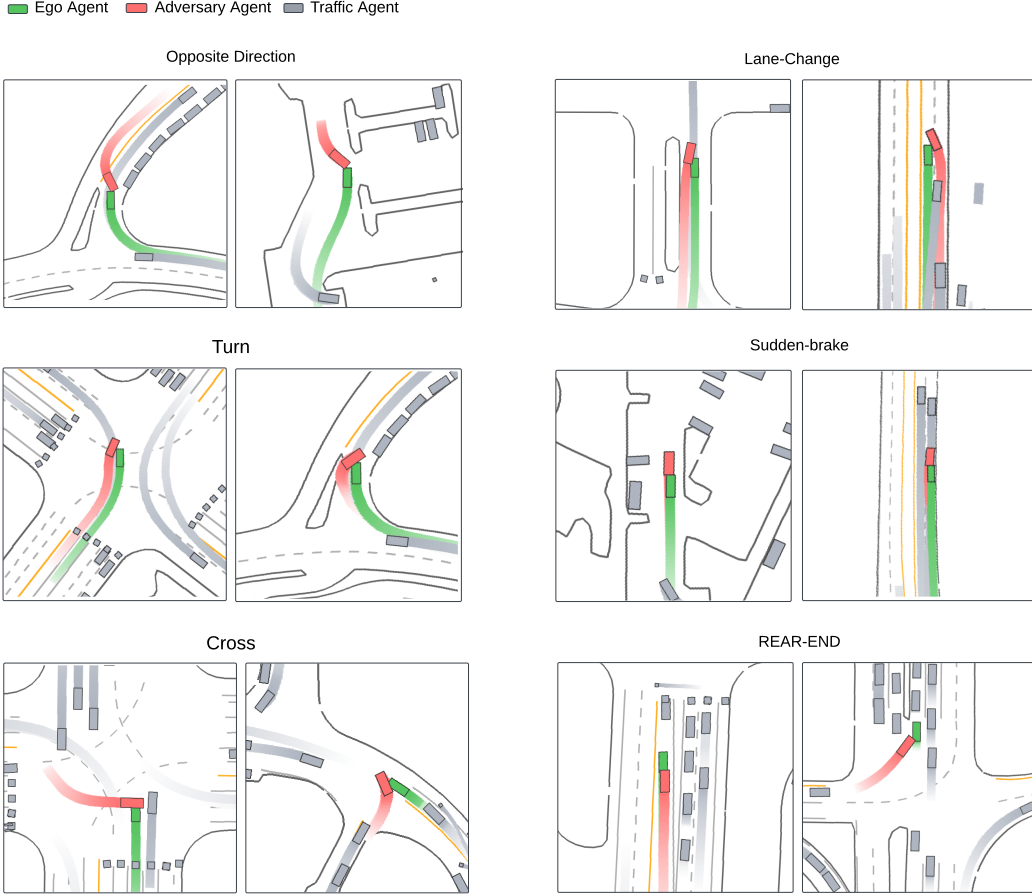


Figure 5: Diverse adversarial behaviors generated by Adv-BMT.

486 real-world traffic with each scenario of length 9 seconds; traffic are composed by agents of vehicle,  
 487 pedestrian, and bicycle; Each scenario comes with a high-fidelity road map. During training, we use 8  
 488 NVIDIA RTX A6000 GPUs for our model training and fine-tunings. We trained BMT in two stages,  
 489 each with hyper-parameters indicated in Table 6. In the first stage, we pre-trained BMT for forward  
 490 prediction only with 1 million steps. Then, we fine-tuned BMT with reverse motion prediction in  
 491 fine-tuning with totally 1.5 million steps. We use AdamW optimizer for learning rate scheduling.

## 492 B Experiment Details

493 **Training Environment** We conduct our reinforcement learning experiments using the MetaDrive  
 494 ScenarioEnv [7], which provides standardized driving environments for training and evaluating au-

Table 7: RL training settings.

Adv-BMT		TD3	
Hyper-parameter	Value	Hyper-parameter	Value
Scenario Horizon	9s	Discounted Factor	0.99
History Horizon	0s	Train Batch Size	1024
Collision Step	1s–9s	Learning Rate	1E-4
Prediction Mode	8	Policy Delay	200
Policy Training Steps	10E6	Target Network	0.005

Generation Time	
Method	Time
CAT [21]	0.80s
SEAL [14]	2.36s
STRIVE [11]	9.53s
Adv-BMT	0.74s

505 autonomous agents. Each environment encodes sensor observations including LiDAR-based surround-  
 506 ings and physical dynamics. Specifically, the observation space consists of three key components: (i)  
 507 Ego State, which contains the ego vehicle’s current physical state such as speed, heading, and steer-  
 508 ing; (ii) Navigation, which provides waypoints derived from a precomputed route to the destination,  
 509 including relative distance and direction to the next target checkpoint; and (iii) Surrounding, which  
 510 encodes nearby traffic objects and obstacles using a vectorized representation of 72 LiDAR beams  
 spanning a 50-meter range.

502 Actions are continuous and correspond to low-level vehicle control commands. The agent outputs a  
 503 2D normalized vector, which is then mapped to steering angle, throttle (acceleration), and brake signal.  
 504 The environment includes a compositional reward structure combining driving progress, collision  
 505 penalties, and road boundary violations. Driving reward is measured by forward lane progress, while  
 506 penalties are applied for collisions with other vehicles or drifting off-road.

507 **Hyper-parameters** The settings of our open-loop and closed-loop adversarial RL experiments  
 508 are shown in table 7. Note that in our closed-loop learning, Adv-BMT takes one frame of agent  
 509 information as input for adversarial generations, whereas all baseline methods take one second agent  
 510 history.

## 511 C Quantitative Results

### 512 C.1 Waymo Open Sim Agent Challenge

513 We evaluated BMT results on 400 WOMB validation scenarios using Waymo Open Sim Agent  
 514 Challenge (WOSAC) 2024 [8]. Evaluation results are summarized in Table 8. Within the metrics, for  
 515 minADE metrics, smaller values indicate better more accurate predictions, and for the rest metrics,  
 516 the larger score means better performance. Within the results, we can see that forward prediction  
 517 achieves much better performance compared to reverse prediction across all metrics, except similar  
 518 performance in Angular speed, angular acceleration, distance to nearest object, and TTC. In these  
 519 mentioned metrics. Note that the training time for forward prediction and reverse prediction are  
 520 similar (10E6 and 15E6 steps respectively). The WOSAC results indicate that our BMT model is  
 521 better at predicting future motion than predicting history motion. We will submit to the official  
 522 WOSAC leaderboard for further reference in the future.



Table 8: WOSAC Evaluation results of BMT.

Metrics	Reverse	Forward
Linear speed	0.375	0.393
Linear acceleration	0.394	0.405
Angular speed	0.441	0.428
Angular acceleration	0.594	0.593
Distance to nearest object	0.405	0.388
Collision	0.521	0.951
Time to Collide	0.840	0.840
Distance to road edge	0.675	0.683
Offroad	0.564	0.934
Realism	0.554	0.753
Kinematic	0.451	0.455
Interactive	0.566	0.801
Map	0.596	0.862
minADE	2.148	1.344
Metametric	0.554	0.753

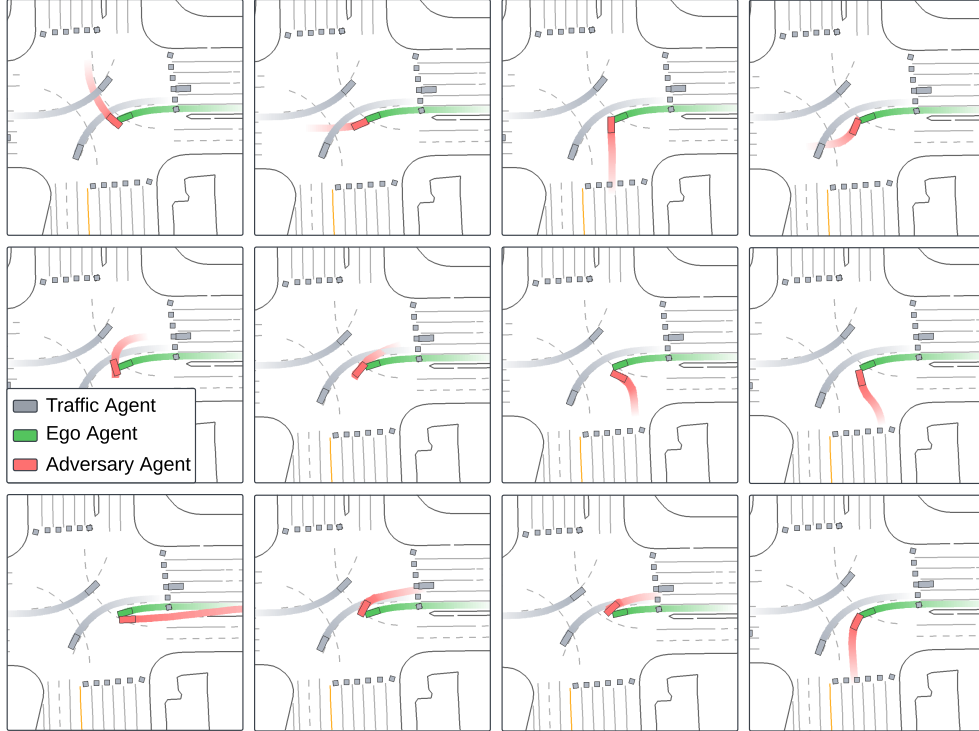


Figure 6: Diverse adversarial behaviors generated on one driving log by Adv-BMT.

## 523 D Qualitative Results

524 **Visualizations** In Fig. 5, we demonstrate six pairs of qualitative results by Adv-BMT. In different  
525 scenarios, our opponent behavior exhibits in different category of safety-critical driving patterns.  
526 This means that our augmented opponent agents have plausible interactions with realistic driving  
527 trajectories. Besides, in Fig. ??, we demonstrate diverse collisions from just one single scenario. This  
528 is the power of Adv-BMT compares to other baseline work, which generates the same or similar  
529 adversarial behavior with the same driving log.

530 **Demo Video** We submit a video within our supplementary materials. Here we put visualizations  
531 with case studies via more animated visualizations of Adv-BMT scenarios, which simulates different  
532 types of vehicles, pedestrians and bicycle agents.

## 533 **E Broader Impacts**

534 Our work introduces a novel model for generating safety-critical traffic scenarios, aiming to improve  
535 the safety reliability and driving robustness of AD systems. By modeling both forward and reverse  
536 motion trajectories, our framework enables controllable and diverse simulation of rare and high-risk  
537 traffic events. Our framework, Adv-BMT, benefits the development and testing of safer autonomous  
538 agents by exposing failure cases under challenging interactions. However, generating adversarial  
539 scenarios may potentially raise concerns about potential misuse, such as crafting unrealistic or  
540 malicious simulations. To address this, our approach is designed for research and evaluation within  
541 closed simulation environments. We encourage responsible usages of our model and encourage  
542 integrating them into safety validation pipelines with appropriate regulations.

## NeurIPS Paper Checklist

### 1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [Yes]

Justification: Yes, our main claims are supported by our experiments and arguments in the paper.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

### 2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: We provide a limitation section at the end of the paper.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

### 3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We don't have proofs and theories in this paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

#### 4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: We discuss all details and information needed to reproduce the experimental results in our paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

#### 5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: Yes, we publish our code and the dataset is public already.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

## 6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: The training and testing details are specified in our main paper and also appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

## 7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We report error bars in our experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

## 8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We discuss information of computer resources needed to reproduce in our appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

## 9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: This paper comply with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

## 10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The broader impacts are addressed in our supplementary materials.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

## 11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Our work poses no high risk for misuse of released data and model.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

## 12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: Our work is unrelated in using existing assets.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, [paperswithcode.com/datasets](https://paperswithcode.com/datasets) has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset’s creators.

### 13. New assets

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: No new assets are introduced in our work.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

### 14. Crowdsourcing and research with human subjects

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: Our work does not engage in crowdsourcing and research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

### 15. Institutional review board (IRB) approvals or equivalent for research with human subjects

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: Our work does not engage in research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

### 16. Declaration of LLM usage



854 Question: Does the paper describe the usage of LLMs if it is an important, original, or  
855 non-standard component of the core methods in this research? Note that if the LLM is used  
856 only for writing, editing, or formatting purposes and does not impact the core methodology,  
857 scientific rigorousness, or originality of the research, declaration is not required.

858 Answer: [NA]

859 Justification: LLM usage is not an important, original, or non-standard component of our  
860 core methodology in this paper.

861 Guidelines:

- 862 • The answer NA means that the core method development in this research does not
- 863 involve LLMs as any important, original, or non-standard components.
- 864 • Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>)
- 865 for what should or should not be described.