

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

5 A Model Details

6 A.1 BMT Architecture

7 **Scenario Format and Preprocessing** The model takes real-world scenarios with the format from
8 ScenarioNet [4], which unifies real-world datasets including [1] [2] [9].

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

21 **Tokenization** We formulate motion prediction as an auto-regressive next-token prediction problem,
22 where each motion token corresponds to a discretized control input for a fixed time interval. Tok-
23 enization process maps continuous agent motion (acceleration and yaw rate) into discrete 2D bins.
24 The tokenization process considers candidate tokens sampled from a fixed grid of bins, and simulates
25 the resulting motion over a short duration, and then selects the best-matching action by minimizing
26 the contour alignment error between the predicted agent shape and the ground-truth position and
27 heading. We adopt a simplified version of the bicycle model to parameterize agent motion using
28 longitudinal acceleration and yaw rate within predefined bounds: acceleration is limited to the range
29 of $[-10, 10]$ m/s², and the yaw rate is constrained to $[-\pi/2, \pi/2]$ rad/s. With predicted motion token
30 sequences, the trajectory can be reconstructed by mapping the token back to the acceleration and yaw
31 angle change.

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

42 **Decoding Architecture.** The decoder follows a GPT-like structure composed of stacked cross-
43 attention layers. Each layer integrates three structured attention modules: agent-to-agent (A2A), agent-
44 to-temporal (A2T), and agent-to-scene (A2S). These modules attend over dynamically constructed
45 graphs defined by spatial or temporal adjacency. Relational information across modalities is captured
46 via multiple embeddings: Agent-to-Agent Relation Embedding, Agent-to-Time Relation Embedding,
47 and Agent-to-Scene Relation Embedding, each encoding context-specific spatial information. For
48 token construction, several embeddings are used, including the Agent Type Embedding, Agent Shape
49 Embedding, Agent ID Embedding, and the Motion Token Embedding which maps discrete control
50 tokens. A Continuous Motion Feature Embedding is applied to embed acceleration and yaw rate
51 attributes. Auxiliary embeddings include the Special Token Embedding for indicating sequence
52 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 1: 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 1. We trained BMT model on the training set of Waymo Open Motion Datasets [3]. WOMD contains 480K

Table 2: 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

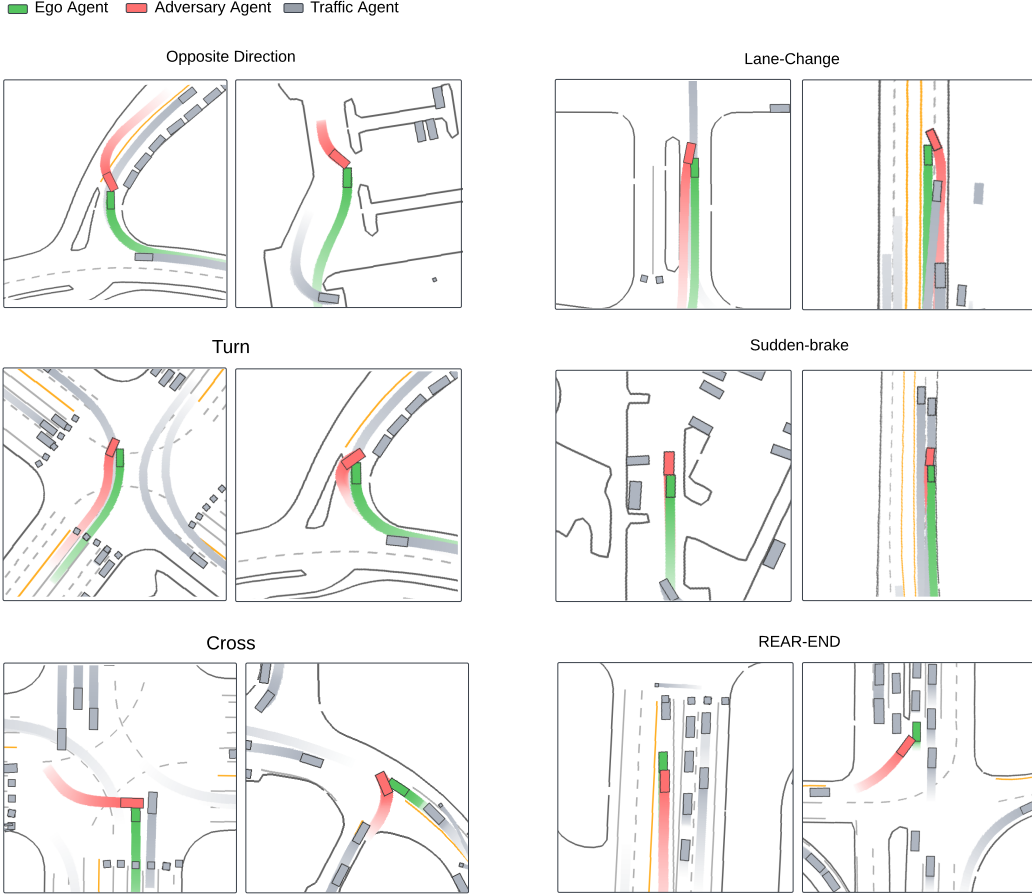


Figure 1: Diverse adversarial behaviors generated by Adv-BMT.

108 real-world traffic with each scenario of length 9 seconds; traffic are composed by agents of vehicle,
 109 pedestrian, and bicycle; Each scenario comes with a high-fidelity road map. During training, we use 8
 110 NVIDIA RTX A6000 GPUs for our model training and fine-tunings. We trained BMT in two stages,
 111 each with hyper-parameters indicated in Table 2. In the first stage, we pre-trained BMT for forward
 112 prediction only with 1 million steps. Then, we fine-tuned BMT with reverse motion prediction in
 113 fine-tuning with totally 1.5 million steps. We use AdamW optimizer for learning rate scheduling.

114 B Experiment Details

115 **Training Environment** We conduct our reinforcement learning experiments using the MetaDrive
 116 ScenarioEnv [5], which provides standardized driving environments for training and evaluating au-

Table 3: 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 [10]	0.80s
SEAL [8]	2.36s
STRIVE [7]	9.53s
Adv-BMT	0.74s

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

124 Actions are continuous and correspond to low-level vehicle control commands. The agent outputs a
 125 2D normalized vector, which is then mapped to steering angle, throttle (acceleration), and brake signal.
 126 The environment includes a compositional reward structure combining driving progress, collision
 127 penalties, and road boundary violations. Driving reward is measured by forward lane progress, while
 128 penalties are applied for collisions with other vehicles or drifting off-road.

129 **Hyper-parameters** The settings of our open-loop and closed-loop adversarial RL experiments
 130 are shown in table 3. Note that in our closed-loop learning, Adv-BMT takes one frame of agent
 131 information as input for adversarial generations, whereas all baseline methods take one second agent
 132 history.

133 C Quantitative Results

134 C.1 Waymo Open Sim Agent Challenge

135 We evaluated BMT results on 400 WOMB validation scenarios using Waymo Open Sim Agent
 136 Challenge (WOSAC) 2024 [6]. Evaluation results are summarized in Table 4. Within the metrics, for
 137 minADE metrics, smaller values indicate better more accurate predictions, and for the rest metrics,
 138 the larger score means better performance. Within the results, we can see that forward prediction
 139 achieves much better performance compared to reverse prediction across all metrics, except similar
 140 performance in Angular speed, angular acceleration, distance to nearest object, and TTC. In these
 141 mentioned metrics. Note that the training time for forward prediction and reverse prediction are
 142 similar (10E6 and 15E6 steps respectively). The WOSAC results indicate that our BMT model is
 143 better at predicting future motion than predicting history motion. We will submit to the official
 144 WOSAC leaderboard for further reference in the future.

Table 4: 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

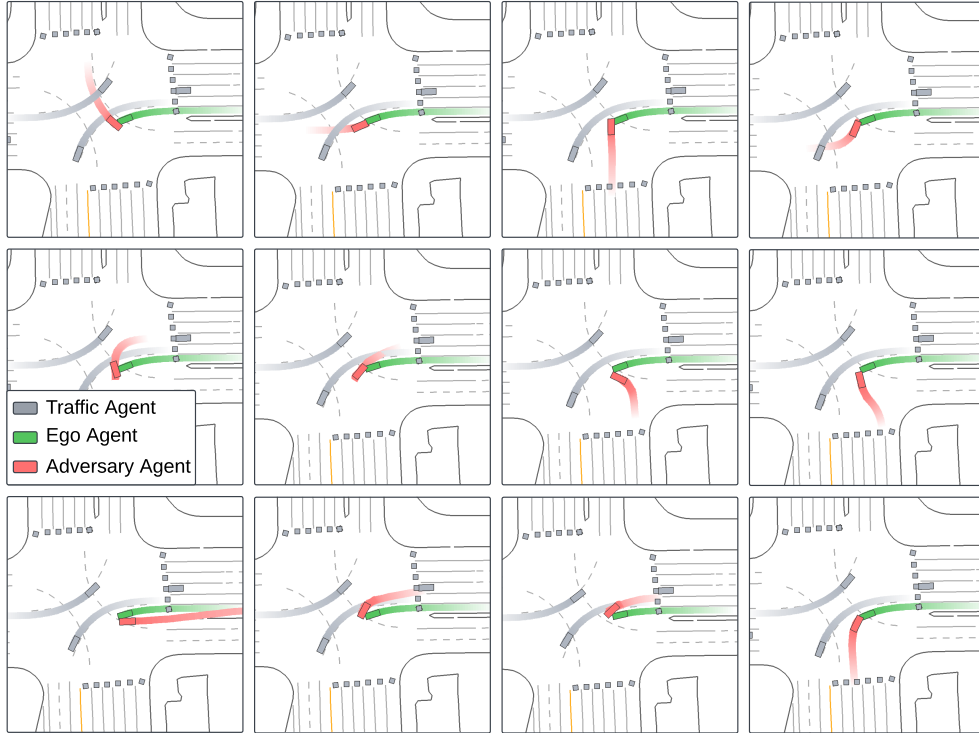


Figure 2: Diverse adversarial behaviors generated on one driving log by Adv-BMT.

145 D Qualitative Results

146 **Visualizations** In Fig. 1, we demonstrate six pairs of qualitative results by Adv-BMT. In different
147 scenarios, our opponent behavior exhibits in different category of safety-critical driving patterns.
148 This means that our augmented opponent agents have plausible interactions with realistic driving
149 trajectories. Besides, in Fig. ??, we demonstrate diverse collisions from just one single scenario. This
150 is the power of Adv-BMT compares to other baseline work, which generates the same or similar
151 adversarial behavior with the same driving log.

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

155 E Broader Impacts

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

165 References

- 166 [1] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan,
167 Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving,
168 2020.
- 169 [2] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher,
170 Oscar Beijbom, and Sammy Omari. Nuplan: A closed-loop ml-based planning benchmark for autonomous
171 vehicles, 2022.
- 172 [3] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai,
173 Ben Sapp, Charles Qi, Yin Zhou, Zoey Yang, Aurelien Chouard, Pei Sun, Jiquan Ngiam, Vijay Vasudevan,
174 Alexander McCauley, Jonathon Shlens, and Dragomir Anguelov. Large scale interactive motion forecasting
175 for autonomous driving : The waymo open motion dataset, 2021.
- 176 [4] Quanyi Li, Zhenghao Peng, Lan Feng, Zhizheng Liu, Chenda Duan, Wenjie Mo, and Bolei Zhou. Scenari-
177 onet: Open-source platform for large-scale traffic scenario simulation and modeling, 2023.
- 178 [5] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive:
179 Composing diverse driving scenarios for generalizable reinforcement learning, 2022.
- 180 [6] Nico Montali, John Lambert, Paul Mougins, Alex Kuefler, Nick Rhinehart, Michelle Li, Cole Gulino,
181 Tristan Emrich, Zoey Yang, Shimon Whiteson, Brandyn White, and Dragomir Anguelov. The waymo open
182 sim agents challenge, 2023.
- 183 [7] Davis Rempe, Jonah Philion, Leonidas J. Guibas, Sanja Fidler, and Or Litany. Generating useful accident-
184 prone driving scenarios via a learned traffic prior, 2022.
- 185 [8] Benjamin Stoler, Ingrid Navarro, Jonathan Francis, and Jean Oh. Seal: Towards safe autonomous driving
186 via skill-enabled adversary learning for closed-loop scenario generation, 2025.
- 187 [9] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal,
188 Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and
189 James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting, 2023.
- 190 [10] Linrui Zhang, Zhenghao Peng, Quanyi Li, and Bolei Zhou. Cat: Closed-loop adversarial training for safe
191 end-to-end driving, 2023.