BehaviorGPT: Smart Agent Simulation for Autonomous Driving with Next-Patch Prediction

Zikang Zhou[1*] **Haibo Hu**[1*] **Xinhong Chen**[1] **Jianping Wang**[1] **Nan Guan**[1]
Kui Wu[2] **Yung-Hui Li**[3] **Yu-Kai Huang**[4] **Chun Jason Xue**[5]
[1]City University of Hong Kong [2]University of Victoria [3]Hon Hai Research Institute
[4]Carnegie Mellon University [5]Mohamed bin Zayed University of Artificial Intelligence
{zikanzhou2-c, haibohu2-c}@my.cityu.edu.hk
{xinhong.chen, jianwang, nanguan}@cityu.edu.hk
wkui@uvic.ca yunghui.li@foxconn.com yukaih2@andrew.cmu.edu
jason.xue@mbzuai.ac.ae

Abstract

Simulating realistic behaviors of traffic agents is pivotal for efficiently validating the safety of autonomous driving systems. Existing data-driven simulators primarily use an encoder-decoder architecture to encode the historical trajectories before decoding the future. However, the heterogeneity between encoders and decoders complicates the models, and the manual separation of historical and future trajectories leads to low data utilization. Given these limitations, we propose BehaviorGPT, a homogeneous and fully autoregressive Transformer designed to simulate the sequential behavior of multiple agents. Crucially, our approach discards the traditional separation between "history" and "future" by modeling each time step as the "current" one for motion generation, leading to a simpler, more parameter- and data-efficient agent simulator. We further introduce the Next-Patch Prediction Paradigm (NP3) to mitigate the negative effects of autoregressive modeling, in which models are trained to reason at the patch level of trajectories and capture long-range spatial-temporal interactions. Despite having merely 3M model parameters, BehaviorGPT won first place in the 2024 Waymo Open Sim Agents Challenge with a realism score of 0.7473 and a minADE score of 1.4147, demonstrating its exceptional performance in traffic agent simulation.

Keywords: Multi-Agent Systems, Transformers, Generative Models, Autonomous Driving

1 Introduction

Autonomous driving has emerged as an unstoppable trend, with its rapid development increasing the demand for faithful evaluation of autonomy systems' reliability [32]. While on-road testing can measure driving performance by allowing autonomous vehicles (AVs) to interact with the physical world directly, the high testing cost and the scarcity of safety-critical scenarios in the real world have hindered large-scale and comprehensive evaluation. As an alternative, validating system safety via simulation has become increasingly attractive [14, 48, 53, 44] as it enables rapid testing in diverse driving scenarios simulated at a low cost. This work focuses on smart agent simulation, i.e., simulating the behavior of traffic participants such as vehicles, pedestrians, and cyclists in the digital world, which is critical for efficiently validating and iterating behavioral policies for AVs.

A good simulator should be realistic, matching the real-world distribution of multi-agent behaviors to support the assessment of AVs' ability to coexist with humans safely. To this end, researchers started

*Equal contribution.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

 https://doi.org/10.52202/079017-2527

by designing naive simulators that mainly replay the driving logs collected in the real world [27, 29]. When testing new driving policies that deviate from the ones during data collection, agents in such simulators often exhibit unrealistic interactions with AVs, owing to the lack of reactivity to AVs' behavior changes. To simulate reactive agents, traditional approaches [14, 28] apply traffic rules to control agents heuristically [45, 26], which may struggle to capture real-world complexity. Recently, the availability of large-scale driving data [6, 15, 50], the emergence of powerful deep learning tools [19, 47, 20], and the prosperity of related fields such as motion forecasting [16, 46, 59, 42, 58], have spurred the development of data-driven agent simulation [44, 4, 24, 52, 56] towards more precise matching of behavioral distribution. With the establishment of standard benchmarks like the Waymo Open Sim Agents Challenge (WOSAC) [32], which systematically evaluates the realism of agent simulation in terms of kinematics, map compliance, and multi-agent interaction, the research on data-driven simulation approaches has been further advanced [49, 35].

Existing learning-based agent simulators [44, 4, 24, 52, 56, 49, 35] mainly mirror the techniques from motion forecasting [16, 46, 59, 42, 58, 41, 18] and opt for an encoder-decoder architecture, presumably due to the similarity between the two fields. Typically, these models use an encoder to extract historical information and a decoder to predict agents' future states leveraging the encoded features. This paradigm requires manually splitting the multi-agent time series into a historical and a future segment, with the two segments being processed by separate encoders and decoders with heterogeneous architecture. For example, MVTA [49] constructs training samples by randomly selecting a "current" timestamp to divide sequences into historical and future components. Others [52, 35] use fixed-length agent trajectories as historical scene context, conditioned on which the multi-agent future is sampled from the decoder. Nonetheless, the benefit of employing heterogeneous modules to separately encode the history and decode the future, at the cost of significantly complicating the architecture, is unclear. Moreover, the manual separation of history and future leads to low utilization of data and computation: as every point in the sequence can be used for the separation, we believe a sample-efficient framework should be able to learn from every possible history-future pair from the sequence in parallel, which cannot be easily achieved by encoder-decoder solutions owing to their heterogeneous processing for the historical and the future time steps.

Inspired by the success of decoder-only Large Language Models (LLMs) [37, 38, 5], we introduce a fully autoregressive Transformer architecture, dubbed BehaviorGPT, into the field of smart agent simulation to overcome the limitations of previous works. By applying homogeneous Transformer blocks [47] to the complete trajectory snippets without differentiating history and future, we arrive at a simpler, more parameter-efficient, and more sample-efficient solution for agent simulation. Utilizing relative spacetime representations [58], BehaviorGPT symmetrically models each agent state in the sequence as if it were the "current" one and tasks each state with modeling subsequent states' distribution during training. As a result, our framework maximizes the utilization of traffic data for autoregressive modeling, avoiding wasting any learning signals available in the time series.

Autoregressive modeling with imitation learning, however, suffers from compounding errors [39] and causal confusion [11]. Concerning the behavior simulation task, we observed that blindly mimicking LLMs' training paradigm of next-token prediction [35], regardless of the difference in tokens' semantics across tasks, will make these issues more prominent. For a next-token prediction model embedding tokens at 10 Hz, a low training loss can be achieved by simply copying and pasting the current token as the next one without performing any long-range interaction reasoning in space or time. To mitigate this issue, we introduce the Next-Patch Prediction Paradigm (NP3) that enables models to reason at the patch level of trajectories, as illustrated in Figure 1. By enforcing models to autoregressively generate the next trajectory patch containing multiple time steps, which requires understanding the high-level semantics of agent behaviors and capturing long-range spatial-temporal interactions, we prevent models from leveraging trivial shortcuts during training. We equip BehaviorGPT with NP3 and attain superior performance on WOSAC [32] with merely 3M model parameters, demonstrating the effectiveness of our modeling framework for smart agent simulation.

Our main contributions are three-fold. First, we propose a fully autoregressive architecture for smart agent simulation, which consists of homogeneous Transformer blocks that process multi-agent long sequences with high parameter and sample efficiency. Second, we develop the Next-Patch Prediction scheme to enhance long-range interaction reasoning, leading to more realistic multi-agent simulation over a long horizon. Third, we achieve remarkable performance on the Waymo Open Motion Dataset, winning first place in the 2024 Waymo Open Sim Agents Challenge.

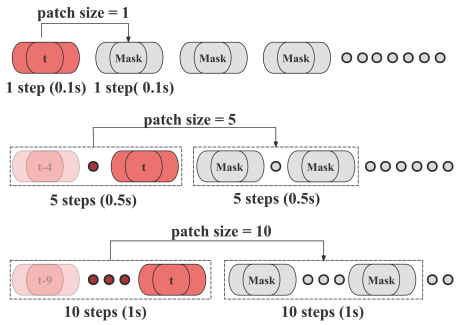

Figure 1: **Next-Patch Prediction Paradigm** with patch sizes of 1, 5, and 10 time steps for trajectories sampled at 10 Hz. The capsules in dark red represent the agent states at the current time step t, while the faded red capsules indicate agents' past states. The grey circles represent the masked agent states required for generation. Our approach groups multi-step agent states as patches, demanding each patch to predict the subsequent patch during training.

2 Related Work

2.1 Multi-Agent Traffic Simulation

Multi-agent traffic simulation is essential for developing and testing autonomous driving systems. From early systems like ALVINN [36] to contemporary simulators such as CARLA [14] and SUMO [28], these platforms have used heuristic driving policies to simulate agents' reactive behaviors [8, 7, 10]. However, they struggle to capture real-world complexity since policies based on simple heuristics are not robust enough to handle all sorts of scenarios. With the availability of large-scale data and deep learning approaches, generative models like VAEs [44], GANs [24], Diffusion [56], and autoregressive models [49, 41, 35] have gained success in generating multi-agent motions, which greatly enhance the realism of simulations. Given the temporal dependency of agent trajectories, autoregressive models naturally fit the simulation task, while others require extra designs to capture such dependencies. Among the existing autoregressive models, two representatives are MotionLM [41] and Trajeglish [35]. Both of them adopt an encoder-decoder paradigm, designing complicated scene context encoders to extract historical information before autoregressive decoding. In contrast, our approach is fully autoregressive similar to decoder-only LLMs [37, 38, 5], which eliminates the need for using heterogeneous modules to process the historical and future time steps and achieves higher efficiency in terms of data and parameters via simpler architectural design.

2.2 Patching Operations in Transformers

The application of patches in Transformer models has demonstrated significant potential across various data modalities. For instance, BERT [12] employs subword tokenization [40] for natural language processing, while ViT [13] segments images into 2D patches for visual understanding. The patching design has also found applications in time-series forecasting [51, 57, 34], aiming at retaining local semantics and reducing computational complexity [34]. Moreover, it has shown the effectiveness in self-supervised learning, which has significantly facilitated representation learning and contributed to excellent fine-tuning results on large datasets [2, 21, 3]. Since the task of agent simulation also involves time-series data, we expect the patching mechanism to help models effectively capture the spatial-temporal interactions in driving scenarios and enhance the realism of the generated motion. Our proposed Next-Patch Prediction Paradigm (NP3) utilizes patch-level tokens in autoregressive modeling and trains each token to generate the next patch that comprises multi-step motions, which shares some similarities to multi-token prediction in LLMs [17].

3 Methodology

This section presents the proposed BehaviorGPT for multi-agent behavior simulation, with Figure 2 illustrating the overall framework. To begin with, we provide the formulation of our map-conditioned, multi-agent autoregressive modeling. Then, we detail the architecture of BehaviorGPT, which adopts a Transformer decoder with a triple-attention mechanism to operate sequences at the patch level. Finally, we present the objective for model training.

https://doi.org/10.52202/079017-2527

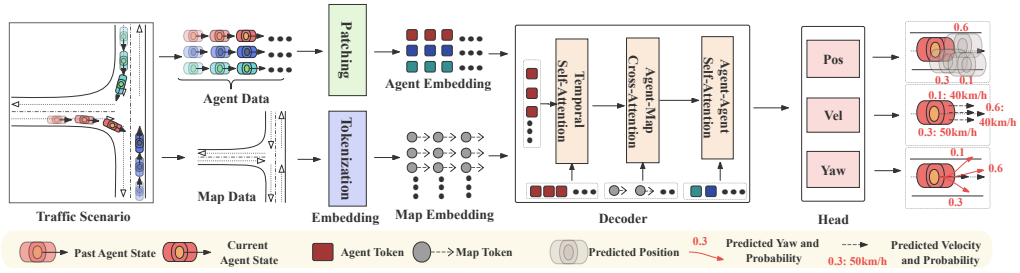

Figure 2: **Overview of BehaviorGPT.** The model takes as input the agent trajectories and the map elements, which are converted into the embeddings of trajectory patches and map polyline segments, respectively. These embeddings are fed into a Transformer decoder for autoregressive modeling based on next-patch prediction, in which the model is trained to generate the positions, velocities, and yaw angles of trajectory patches.

3.1 Problem Formulation

In multi-agent traffic simulation, we aim to simulate agents' future behavior in dynamic and complex environments. Specifically, we define a scenario as the composite of a vector map M and the states of N_{agent} agents over T time steps. At each time step, the state of the i-th agent S_i includes the agent's position, velocity, yaw angle, and bounding box size. The semantic type of agents (e.g., vehicles, pedestrians, and cyclists) are also available. Given the sequential nature of agent trajectories, we formulate the problem as sequential predictions over trajectory patches, where the prediction of each patch will affect the subsequent patches. We define an agent-level trajectory patch as

$$P_i^\tau = S_i^{((\tau-1)\times\ell+1):(\tau\times\ell)}\,,\ i \in \{1,\ldots,N_{\text{agent}}\}\,,\ \tau \in \{1,\ldots,N_{\text{patch}}\}\,, \tag{1}$$

where ℓ is the number of time steps covered by a patch, $N_{\text{patch}} = T/\ell$ indicates the number of patches, and P_i^τ represents the τ-th trajectory patch of the i-th agent, with $S_i^{((\tau-1)\times\ell+1):(\tau\times\ell)}$ denoting the states within the patch. On top of P_i^τ, we use $P^\tau = S_{1:N_{\text{agent}}}^{((\tau-1)\times\ell+1):(\tau\times\ell)}$ to denote the τ-th multi-agent patch, where P^τ incorporates all agents' states at the τ-th patch. Next, we factorize the multi-agent joint distribution over patches along the time axis according to the chain rule:

$$\Pr\left(S_{1:N_{\text{agent}}}^{1:T} \mid M\right) = \prod_{\tau=1}^{N_{\text{patch}}} \Pr\left(P^\tau \mid P^{1:(\tau-1)}, M\right)\,, \tag{2}$$

where $\Pr(S_{1:N_{\text{agent}}}^{1:T} \mid M)$ is the joint distribution of all agents' states over all time steps conditioned on the map M. Further, we factorize over agents the conditional distribution of multi-agent patches based on the assumption that agents plan their motions independently within the horizon of a patch:

$$\Pr\left(P^\tau \mid P^{1:(\tau-1)}, M\right) = \prod_{i=1}^{N_{\text{agent}}} \Pr\left(P_i^\tau \mid P^{1:(\tau-1)}, M\right)\,. \tag{3}$$

Considering the multimodality of agents' behavior within the horizon of a patch, we assume $\Pr(P_i^\tau \mid P^{1:(\tau-1)}, M)$ to be a mixture model consisting of N_{mode} modes:

$$\Pr\left(P_i^\tau \mid P^{1:(\tau-1)}, M\right) = \sum_{k=1}^{N_{\text{mode}}} \pi_{i,k}^\tau \Pr\left(P_{i,k}^\tau \mid P^{1:(\tau-1)}, M\right)\,, \tag{4}$$

where $\pi_{i,k}^\tau$ is the probability of the k-th mode. Given the sequential nature of the states within a patch, we further conduct factorization over the states per mode using the chain rule:

$$\Pr\left(P_{i,k}^\tau \mid P^{1:(\tau-1)}, M\right) = \prod_{t=(\tau-1)\times\ell+1}^{\tau\times\ell} \Pr\left(S_{i,k}^t \mid S_{i,k}^{((\tau-1)\times\ell+1):(t-1)}, P^{1:(\tau-1)}, M\right)\,. \tag{5}$$

Such an autoregressive formulation can be interpreted as planning the patch-level behavior of each agent independently (Eq. (3)), freezing agents' behavior mode per ℓ time steps (Eq. (4)),

and autoregressively unrolling the next state under a specific behavior mode (Eq. (5)). Under this formulation, we can flexibly adjust the replan frequency during inference to control the reactivity of agents. For example, we can let agents execute $\alpha \in \{1, \ldots, \ell\}$ steps of the planned motions and choose a new behavior mode after α steps to react to the change in environments.

3.2 Relative Spacetime Representation

In our autoregressive formulation, we treat each trajectory patch as the "current" patch that is responsible for estimating the next-patch distribution during training, contrasting many existing approaches that designate one current time step per sequence [52, 49, 23]. As a result, it is inefficient to employ the well-established agent- or polyline-centric representation from the field of motion forecasting [46, 59, 33, 42, 25, 54, 43], given that these representations are computed under the reference frames determined by one current time step per sequence. For this reason, we adopt the relative spacetime representation introduced in QCNet [58] to model the patches symmetrically in space and time, achieving simultaneous multi-agent prediction when implementing Eq. (3) and allowing parallel next-patch prediction for the modeling of Eq. (2). Under this representation, the features of each map element and agent state are derived from coordinate-independent attributes, e.g., the semantic category of a map element and the speed of an agent state. On top of this, we effectively maintain the spatial-temporal relationships between input elements via relative positional embeddings. Specifically, we use i and j to index two different input elements and compute the relative spatial-temporal embedding by

$$\mathcal{R}_{j \to i} = \text{MLP}\left(\|\boldsymbol{d}_{j \to i}\|,\, \angle\left(\boldsymbol{n}_i,\, \boldsymbol{d}_{j \to i}\right),\, \Delta\boldsymbol{\theta}_{j \to i},\, \Delta\boldsymbol{z}_{j \to i},\, \Delta\boldsymbol{\tau}_{j \to i}\right), \tag{6}$$

where $R_{j \to i}$ is the relational embedding from j to i, $\|d_{j \to i}\|$ is the Euclidean distance between them, $\angle(n_i, d_{j \to i})$ is the angle between n_i (i.e., the orientation of i) and $d_{j \to i}$ (i.e., the displacement vector from j to i), $\Delta\theta_{j \to i}$/$\Delta\boldsymbol{z}_{j \to i}$ is the relative yaw/height from j to i, and $\Delta\tau_{j \to i}$ is the time difference.

3.3 Map Tokenization and Agent Patching

Before performing spatial-temporal relational reasoning among the input elements of a traffic scenario, we must convert the raw information into high-dimensional embeddings. We first embed map information by sampling points along map polylines every 5 meters and tokenizing the semantic category of each 5-meter segment (e.g., lane centerlines, road edges, and crosswalks) via learnable embeddings. The i-th polyline segment's embedding is denoted by $\hat{M}_i$, which does not include any information about coordinates. On the other hand, we process agent states using attention-based patching to obtain patch-level embeddings of trajectories. For the i-th agent's state S_i^t at time step t, we employ an MLP to transform the speed, the velocity vector's angle relative to the bounding box's heading, the size of the bounding box, and the semantic type of the agent, into a feature vector $\hat{S}_i^t$. To further acquire patch embeddings, we collect the feature vectors of ℓ consecutive agent states and apply the attention mechanism with relative positional embeddings to them:

$$\hat{P}_i^\tau = \text{MHSA}(Q = \hat{S}_i^{\tau \times \ell}, K = V = \{[\hat{S}_i^t,\, \mathcal{R}_i^{t \to (\tau \times \ell)}]\}_{t \in \{(\tau-1) \times \ell + 1, \ldots, \tau \times \ell - 1\}}), \tag{7}$$

where $\hat{P}_i^\tau$ is the patch embedding of the i-th agent at the τ-th patch, $\text{MHSA}(\cdot)$ denotes the multi-head self-attention [47], $[:, :]$ denotes concatenation, and $\mathcal{R}_i^{t \to (\tau \times \ell)}$ indicates the positional embedding of S_i^t relative to $S_i^{\tau \times \ell}$ computed according to Eq. (6). Such an operation can be viewed as aggregating the features of $S_i^{((\tau-1) \times \ell + 1):(\tau \times \ell - 1)}$ into $\hat{S}_i^{\tau \times \ell}$ and using the embeddings fused with high-level semantics as the agent tokens in the subsequent modules.

3.4 Triple-Attention Transformer Decoder

After obtaining map tokens and the patch embeddings of agents, we employ a Transformer decoder [47] with the triple-attention mechanism to model the spatial-temporal interactions among scene elements. As illustrated in Figure 3, the triple-attention mechanism considers three distinct sources of relations in the scene, including the temporal dependencies over the trajectory patches per agent, the regulations of the map elements on the agents, and the social interactions among agents.

Temporal Self-Attention. This module captures the relationships among the trajectory patches of each individual agent. Similar to decoder-only LLMs [37, 38, 5], it leverages the multi-head

 https://doi.org/10.52202/079017-2527

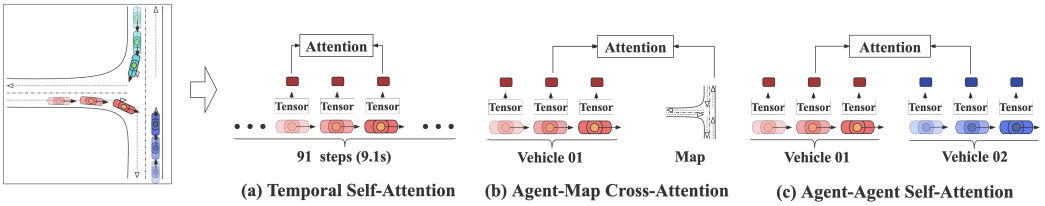

Figure 3: **Triple Attention** applies attention mechanisms to model (a) agents' sequential behaviors, (b) agents' relationships with the map context, and (c) the interactions among agents.

self-attention (MHSA) with a causal mask to enforce each trajectory patch to only attend to the preceding patches of the same agent, accommodating our autoregressive formulation. The temporal MHSA is equipped with relative positional embeddings:

$$F^{\tau}_{a2t,i} = \text{MHSA}(Q = \hat{P}^{\tau}_i, K = V = \{[\hat{P}^{t}_i,\, \mathcal{R}^{(t\times\ell)\to(\tau\times\ell)}_i]\}_{t\in\{1,\dots,\tau-1\}}), \tag{8}$$

where $F^{\tau}_{a2t,i}$ and $\hat{P}^{\tau}_i$ are the temporal-aware feature vector and the patch embedding of the i-th agent at the τ-th patch, respectively, and $\mathcal{R}^{t\times\ell\to\tau\times\ell}_i$ embeds the relative position from $S^{t\times\ell}_i$ to $S^{\tau\times\ell}_i$, which represents the spatial-temporal relationship between the patches P^{t}_i and P^{τ}_i.

Agent-Map Cross-Attention. Unlike natural language which only has a sequence dimension, we must also conduct spatial reasoning to consider the environmental influence on agents' behavior. To facilitate the modeling of agent-map interactions, we apply the multi-head cross-attention (MHCA) to each trajectory patch in the scenario. Considering that a scenario may comprise an explosive number of map polyline segments and that an agent would not be influenced by map elements far away, we filter the key/value map elements in MHCA using the k-nearest neighbors algorithm [42, 54]. The agent-map cross-attention is formulated as

$$F^{\tau}_{a2m,i} = \text{MHCA}(Q = F^{\tau}_{a2t,i}, K = V = \{[\hat{M}_j,\, \mathcal{R}^{\tau\times\ell}_{j\to i}]\}_{j\in\mathcal{N}(i,\tau)}), \tag{9}$$

where $F^{\tau}_{a2m,i}$ is the map-aware feature vector for the i-th agent at the τ-th patch, $\hat{M}_j$ is the embedding of the j-th map polyline segment, $\mathcal{R}^{\tau\times\ell}_{j\to i}$ is the relative positional embedding between the agent state $S^{\tau\times\ell}_i$ and the j-th map polyline segment, and $\mathcal{N}(i,\tau)$ denotes the k-nearest map neighbors of $S^{\tau\times\ell}_i$.

Agent-Agent Self-Attention. We further capture the social interactions among agents by applying the MHSA to the space dimension of the trajectory patches. In this module, we also utilize the locality assumption induced by the k-nearest neighbor selection for better computational and memory efficiency. Specifically, the map-aware features of trajectory patches are refined by

$$F^{\tau}_{a2a,i} = \text{MHSA}(Q = F^{\tau}_{a2m,i}, K = V = \{[F^{\tau}_{a2m,j},\, \mathcal{R}^{\tau\times\ell}_{j\to i}]\}_{j\in\mathcal{N}(i,\tau)}), \tag{10}$$

where $F^{\tau}_{a2a,i}$ is the feature vector enriched with spatial interaction information among agents for the i-th agent at the τ-th patch, $\mathcal{R}^{\tau\times\ell}_{j\to i}$ contains the relative information between the i-th and the j-th agent at the τ-th patch, and $\mathcal{N}(i,\tau)$ filters the k-nearest agent neighbors of $S^{\tau\times\ell}_i$.

Overall Decoder Architecture. Each of the attention layers above is enhanced by commonly used components in Transformers [47], including feed-forward networks, residual connections [19], and Layer Normalization [1] in a pre-norm fashion. To enable higher-order relational reasoning, we stack multiple triple-attention blocks by interleaving the three Transformer layers. We denote the ultimate feature of the i-th agent at the τ-th patch as F^{τ}_i, which will serve as the input of the prediction head for next-patch prediction modeling.

3.5 Next-Patch Prediction Head

Given the interaction-aware patch features output by the Transformer decoder, we develop a next-patch prediction head to model the marginal multimodal distribution of agent trajectories, which estimates the distributional parameters of each patch's successor.

The following describes the process of next-patch prediction regarding the τ-th patch of the i-th agent. Based on the attention output F_i^τ, we intend to estimate the parameters of the next patch's mixture model pre-defined with N_{mode} modes. First, we use an MLP to transform F_i^τ into $\pi_i^{\tau+1} \in \mathbb{R}^{N_{\text{mode}}}$, the mixing coefficient of the modes. In each mode, the conditional distribution of the next agent state, as depicted in Eq. (5), is considered a multivariate marginal distribution that parameterizes the position and velocity components as Laplace distributions and the yaw angle as a von Mises distribution. Based on this formulation, we employ a GRU-based autoregressive RNN [9] to unroll the states within the next patch step by step, with each step being conditioned on the previously predicted states. Specifically, The hidden state $h_{i,k}^{\tau,t}$ of the RNN is initialized with F_i^τ at $t = 1$ for $\forall k \in \{1, \ldots, N_{\text{modes}}\}$. At each step of the rollout, we use an MLP to estimate the location and scale parameters of the next agent state's position and velocity based on the hidden state. On the other hand, the MLP also estimates the location and concentration parameters of the next yaw angle. The location parameters of the newly predicted state, including the 3D positions, the 2D velocities, and the yaw angle, are used to update the RNN's hidden state directly without relying on the predicted scale/concentration parameters for sampling. The whole process is summarized as follows:

$$\begin{aligned} \pi_{i,k}^{\tau+1} &= \text{MLP}([F_i^\tau, Z_k])\,, \\ h_{i,k}^{\tau,1} &= F_i^\tau\,, \\ \mu_{i,k}^{\tau\times\ell+t},\ b_{i,k}^{\tau\times\ell+t},\ \kappa_{i,k}^{\tau\times\ell+t} &= \text{MLP}([h_{i,k}^{\tau,t}, Z_k])\,, \\ h_{i,k}^{\tau,t+1} &= \text{RNN}(h_{i,k}^{\tau,t},\ \text{MLP}(\mu_{i,k}^{\tau\times\ell+t}))\,, \end{aligned} \tag{11}$$

where $\{\mu_{i,k}^{\tau\times\ell+t} \in \mathbb{R}^6\}_{t\in\{1,\ldots,\ell\}}$, $\{b_{i,k}^{\tau\times\ell+t} \in \mathbb{R}^5\}_{t\in\{1,\ldots,\ell\}}$, and $\{\kappa_{i,k}^{\tau\times\ell+t} \in \mathbb{R}\}_{t\in\{1,\ldots,\ell\}}$ are the location, scale, and concentration parameters in the k-th mode, and Z_k is the k-th learnable mode embedding.

3.6 Training Objective

To train BehaviorGPT, we apply the negative log-likelihood loss $\mathcal{L}_{\text{NLL}}$ to the factorized distribution of $\Pr(S_{1:N_{\text{agent}}}^{1:T} \mid M)$ as formulated previously:

$$\mathcal{L}_{\text{NLL}} = \sum_{\tau=1}^{N_{\text{patch}}} \sum_{i=1}^{N_{\text{agent}}} -\log \sum_{k=1}^{N_{\text{mode}}} \pi_{i,k}^{\tau} \prod_{t=(\tau-1)\times\ell+1}^{\tau\times\ell} \Pr\left(S_{i,k}^{t} \mid S_{i,k}^{((\tau-1)\times\ell+1):(t-1)}, P^{1:(\tau-1)}, M\right). \tag{12}$$

Note that each ground-truth trajectory patch is transformed into the viewpoint of its previous patch. During training, we utilize teacher forcing to parallelize the modeling of next-patch prediction and ease the learning difficulty, but we do not use the ground-truth agent states when updating the RNN's hidden states, intending to train the model to recover from its mistakes made in next-state prediction.

4 Experiments

This section first introduces the dataset and the evaluation metrics used in our experiments, followed by presenting the implementation details and the rollout results obtained by BehaviorGPT on the Waymo Open Sim Agents Benchmark [32]. Finally, we conduct ablation studies to further compare and analyze the performance of BehaviorGPT under various settings.

4.1 Dataset and Metrics

Our experiments are conducted on the Waymo Open Motion Dataset (WOMD) [15]. The dataset comprises 486,995/44,097/44,920 training/validation/testing scenarios. Each scenario includes 91-step observations sampled at 10 Hz, totaling 9.1 seconds. Given 11-step initial states of the scenarios, we simulate up to 128 agents and generate 80 simulation steps per agent at 0.1-second intervals in an autoregressive and reactive manner. Each agent requires 32 simulations comprising x/y/z centroid coordinates and a heading value. The results on the test set are obtained by utilizing the full training set, while the performance on the validation set is based on 20% of training data unless specified.

We use various metrics for evaluation. The minADE measures the minimum average displacement error over multiple simulated trajectories, assessing trajectory accuracy. REALISM is the meta-metric

https://doi.org/10.52202/079017-2527

Table 1: Test set results in the 2024 Waymo Open Sim Agents Challenge.

Model	#Param	minADE (↓)	REALISM (↑)	LINEAR SPEED (↑)	LINEAR ACCEL (↑)	ANG SPEED (↑)	ANG ACCEL (↑)	DIST TO OBJ (↑)	COLLISION (↑)	TTC (↑)	DIST TO ROAD EDGE (↑)	OFFROAD (↑)
Linear Extrapolation [32]	-	7.5148	0.3985	0.0434	0.1661	0.2522	0.4393	0.2154	0.3905	0.7555	0.4801	0.4426
TrafficBotsV1.5 [55]	10M	1.8825	0.6988	0.3361	0.3497	0.4512	0.5844	0.3596	0.8083	0.8209	0.6423	0.9134
VBD [23]	12M	1.4743	0.7200	0.3591	0.3664	0.4197	0.5222	0.3683	0.9341	0.8153	0.6508	0.8788
MVTE [49]	>65M	1.6770	0.7302	0.3506	0.3531	0.4974	0.6000	0.3743	0.9049	**0.8310**	0.6655	0.9071
GUMP [22]	523M	1.6041	0.7431	0.3567	**0.4111**	**0.5089**	**0.6353**	0.3707	0.9403	0.8276	0.6686	0.9028
BehaviorGPT (Ours)	3M	**1.4147**	**0.7473**	**0.3615**	0.3365	0.4806	0.5544	**0.3834**	**0.9537**	0.8308	**0.6702**	**0.9349**

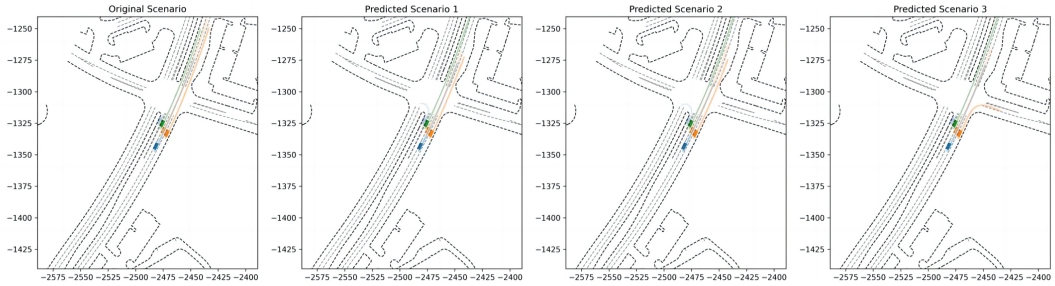

Figure 4: High-quality simulations produced by BehaviorGPT, where multimodal behaviors of agents are simulated realistically.

that expects the simulations to match the real-world distribution. LINEAR SPEED and LINEAR ACCEL evaluate the realism regarding speed and acceleration. Similarly, ANG SPEED and ANG ACCEL measure the realism of angular speed and acceleration. DIST TO OBJ considers the distances to objects, while COLLISION and TTC assess the simulation performance in terms of collision and time to collision. Finally, DIST TO ROAD EDGE and OFFROAD focus on map compliance.

4.2 Implementation Details

The optimal patch size we experimented with is 10, corresponding to 1 second. All hidden sizes are set to 128. Each attention layer has 8 attention heads with 16 dimensions per head. To save training resources, we limit the maximum number of agents per scenario to 128 and restrict the maximum number of neighbors in kNN attention layers to 32. The prediction head produces 16 modes per agent and time step. We train the models for 30 epochs on 8 NVIDIA RTX 4090 GPUs with a batch size of 24, utilizing the AdamW optimizer [31]. The weight decay rate and dropout rate are both set to 0.1. The learning rate is initially set to 5×10^{-4} and decayed to 0 following a cosine annealing schedule [30]. Our results in the 2024 WOSAC are obtained using a single model with 2 decoding blocks and a total of 3M parameters. To produce 32 replicas of rollouts, we randomly sample behavior modes from agents' next-patch distributions until completing the 8-second multi-agent trajectories, and we repeat this process with different random seeds. The final results on the leaderboard are based on a replan rate of 2 Hz, while the ablation studies are based on a 1-Hz replan rate unless specified.

4.3 Quantitative Results

We report the test set results in Table 1. Notably, BehaviorGPT achieves the lowest minADE and the best REALISM, underscoring the model's ability to match the real-world distribution. Its excellent performance on COLLISION and OFFROAD also indicates that the model has successfully captured the agent-agent and agent-map interactions in driving scenarios. Besides the benchmarking results, we also compare the number of model parameters in BehaviorGPT and other baselines. Table 1 demonstrates that BehaviorGPT, with only 3M parameters, achieves more realistic simulation than significantly larger models like MVTE [49] and GUMP [22], which demonstrates the parameter efficiency of our approach. Without employing tricks like data augmentation, model ensemble, or post-processing steps, BehaviorGPT won first place in the 2024 Waymo Open Sim Agents Challenge.

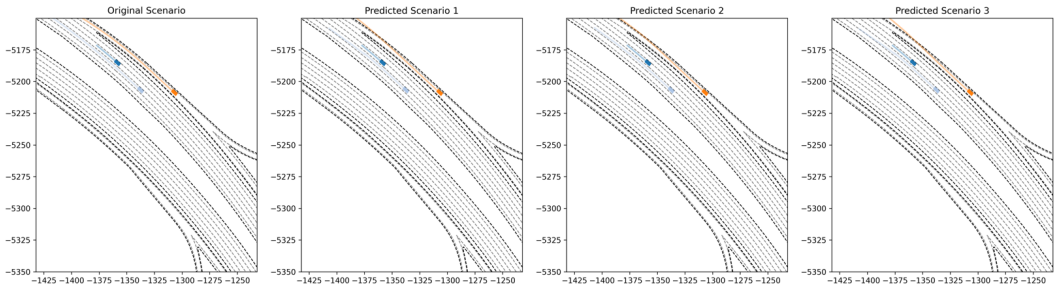

Figure 5: A typical failed case produced by BehaviorGPT, where offroad trajectories are generated owing to the compounding error caused by autoregressive modeling.

4.4 Qualitative Results

Figure 4 visualizes some qualitative results of the rollouts produced by our model. In this scenario, BehaviorGPT can generate multiple plausible futures given the same initial states of agents, which demonstrates its capability of simulating diverse yet realistic agent behavior. However, we also note that autoregressive models still suffer from accumulated errors in some cases. As shown in Figure 5, the vehicle in orange gradually goes out of the road as time goes by, which indicates the inherent limitations of autoregressive generation.

4.5 Ablation Studies

We conduct some ablation studies to gain a more in-depth understanding of our approach.

Impact of patch size. Table 2 presents the results of BehaviorGPT with varying patch sizes. According to the results, it is evident that using a patch size of 5, i.e., training and predicting with 2-Hz tokens, significantly outperforms the baseline without patching. Moreover, increasing the patch size to 10 further enhances the overall performance. These results demonstrate the benefits of incorporating the NP3 into agent simulation. However, changing the patch size also leads to a variation in replan frequency, which also has an influence on simulation. Next, we investigate the impact of replan frequency on the test set using the model submitted to the 2024 WOSAC.

Table 2: Impact of patch size on the validation set.

Patch Size	Replan Frequency	minADE (↓)	REALISM (↑)	LINEAR SPEED (↑)	LINEAR ACCEL (↑)	ANG SPEED (↑)	ANG ACCEL (↑)	DIST TO OBJ (↑)	COLLISION (↑)	TTC (↑)	DIST TO ROAD EDGE (↑)	OFFROAD (↑)
1	10 Hz	2.3752	0.6783	0.2559	0.2088	0.4022	0.5094	0.3201	0.9002	0.8015	0.6149	0.8432
5	2 Hz	1.5599	0.7273	**0.3543**	**0.3218**	0.4623	**0.5435**	0.3768	0.9181	**0.8339**	0.6564	0.9077
10	1 Hz	**1.5203**	**0.7335**	0.3517	0.3023	**0.4734**	0.5432	**0.3797**	**0.9358**	0.8329	**0.6645**	**0.9132**

Impact of replan frequency. During inference, we vary the replan frequency of the model with a patch size of 10 by discarding a portion of the predicted states at each simulation step. As shown in Table 3, increasing the replan frequency from 1 Hz to 2 Hz can even improve the overall performance, which may benefit from the enhanced reactivity. This phenomenon demonstrates that the performance gain is not merely due to the lower replan frequency, as the model with a patch size of 10 beats that with a patch size of 5 even harder if using the same replan frequency of 2 Hz. However, using an overly high replan frequency harms the performance, as indicated by the third row of Table 3. Overall, we conclude that using a larger patch indeed helps long-term reasoning, but a moderate replan frequency is important for temporal stability, which may be neglected by prior works.

Table 3: Impact of replan frequency on the test set.

Patch Size	Replan Frequency	minADE (↓)	REALISM (↑)	LINEAR SPEED (↑)	LINEAR ACCEL (↑)	ANG SPEED (↑)	ANG ACCEL (↑)	DIST TO OBJ (↑)	COLLISION (↑)	TTC (↑)	DIST TO ROAD EDGE (↑)	OFFROAD (↑)
10	1 Hz	1.5405	0.7414	0.3553	0.3153	0.4695	0.5303	0.3772	0.9520	0.8285	0.6664	0.9308
10	2 Hz	**1.4147**	**0.7473**	**0.3615**	0.3365	**0.4806**	0.5544	**0.3834**	**0.9537**	**0.8308**	**0.6702**	**0.9349**
10	5 Hz	1.5693	0.7342	0.3430	**0.3472**	0.4663	**0.5673**	0.3722	0.9429	0.8253	0.6534	0.9089

https://doi.org/10.52202/079017-2527

Impact of multi-agent interaction modeling. We remove all agent-agent self-attention layers in the first row of Table 4 to show that modeling the interactions among agents can boost minADE and REALISM. In particular, the realism in terms of collision is improved by 34.66% when employing agent-agent self-attention.

Table 4: Impact of agent-agent self-attention on the validation set.

Agent-Agent Self-Attention	minADE (↓)	REALISM (↑)	DIST TO OBJ (↑)	COLLISION (↑)	TTC (↑)
×	2.1489	0.6659	0.3539	0.6987	0.8070
✓	**1.6247**	**0.7349**	**0.3783**	**0.9409**	**0.8320**

Table 5: Effects of training data on the validation set.

Train Data	#Param	minADE (↓)	REALISM (↑)
20%	5M	1.4881	0.7396
50%	5M	1.4060	0.7427
100%	5M	**1.3804**	**0.7438**

Table 6: Effects of model depth on the validation set.

Model Depth	#Param	minADE (↓)	REALISM (↑)
2	3M	1.6247	0.7349
3	4M	1.5381	0.7387
4	5M	**1.4881**	**0.7396**

Table 7: Effects of model width on the validation set.

Model Width	#Param	minADE (↓)	REALISM (↑)
64	800K	1.9637	0.7251
128	3M	1.6247	0.7349
192	7M	**1.4993**	**0.7382**

Table 8: Extrapolation ability to generate longer sequences.

Training	Inference	minADE (↓)	REALISM (↑)	LINEAR SPEED (↑)	LINEAR ACCEL (↑)	ANG SPEED (↑)	ANG ACCEL (↑)	DIST TO OBJ (↑)	COLLISION (↑)	TTC (↑)	DIST TO ROAD EDGE (↑)	OFFROAD (↑)
9.1 sec	9.1 sec	**1.6247**	**0.7349**	0.3546	0.3105	**0.4689**	**0.5363**	0.3783	**0.9409**	**0.8320**	**0.6605**	**0.9163**
5.0 sec	9.1 sec	1.6294	0.7333	**0.3565**	**0.3471**	0.4613	0.5293	**0.3813**	0.9375	0.8273	0.6585	0.9100

Scaling with data. We train our models with different proportions of training data. All the models have 4 decoding blocks and a hidden size of 128, totaling 5M parameters. As shown in Table 5, BehaviorGPT is able to achieve remarkable performance with merely 20% of training data, which is attributed to the high data efficiency of our approach. Increasing the proportion of training data from 20% to 50% further improves the performance on minADE and REALISM, and training on 100% of the data continues to gain enhancement. Judging from the trend in Table 5, we believe that feeding more data for model training will continuously achieve better simulation performance.

Scaling with model size. We investigate the effects of scaling up the model size based on some preliminary experiments with 20% of training data. In Table 6, we vary the number of decoding blocks while fixing the hidden size as 128. On the other hand, we fix the number of decoding blocks as 2 and vary the hidden size, as depicted in Table 7. Based on the experimental results, we can summarize that enlarging the model consistently leads to more realistic simulation, which showcases the potential of BehaviorGPT for scaling up.

Extrapolation ability. We tried training a model on 5-second sequences and generating 9.1-second sequences during inference. The results in Table 8 show that this model achieves similar performance compared with the baseline trained with 9.1-second sequences, demonstrating our approach's extrapolation ability to generate longer sequences.

5 Conclusion

This work introduced BehaviorGPT, a fully autoregressive architecture designed to enhance smart agent simulation for autonomous driving. By applying homogeneous Transformer blocks to entire trajectory snippets and utilizing relative spacetime representations, BehaviorGPT simplifies the modeling process and maximizes data utilization. To enable high-level understanding and long-range interaction reasoning in space and time, we developed the Next-Patch Prediction Paradigm, which tasks models with generating trajectory patches instead of single-step states. Experimental results on the Waymo Open Sim Agents Challenge demonstrate that BehaviorGPT achieves outstanding performance with merely 3M model parameters, highlighting its potential to further improve the realism of agent simulation with more data and computation.

Limitations. First, BehaviorGPT is currently inferior in kinematics-related performance, which can be enhanced by incorporating a kinematic model, e.g., the bicycle model. Second, the current version of BehaviorGPT does not support controlling agent behavior with specific prompts such as language and goal points. However, achieving controllable generation should be trivial given a powerful base model. Finally, we have not verified whether BehaviorGPT will facilitate the development of motion planning, which we leave as future work.

https://doi.org/10.52202/079017-2527

Acknowledgement

This project is supported by a grant from Hong Kong Research Grant Council under GRF project 11216323 and CRF C1042-23G.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[2] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems*, 2020.

[3] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022.

[4] Luca Bergamini, Yawei Ye, Oliver Scheel, Long Chen, Chih Hu, Luca Del Pero, Błażej Osiński, Hugo Grimmett, and Peter Ondruska. Simnet: Learning reactive self-driving simulations from real-world observations. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5119–5125. IEEE, 2021.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, 2020.

[6] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8748–8757, 2019.

[7] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15590–15599, 2021.

[8] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on Robot Learning*, pages 66–75. PMLR, 2020.

[9] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[10] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4693–4700. IEEE, 2018.

[11] Pim De Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems*, 2019.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *North American Chapter of the Association for Computational Linguistics*, 2019.

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[14] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on Robot Learning*, pages 1–16. PMLR, 2017.

 https://doi.org/10.52202/079017-2527

[15] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

[16] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533, 2020.

[17] Fabian Gloeckle, Badr Youbi Idrissi, Baptiste Rozière, David Lopez-Paz, and Gabriel Synnaeve. Better & faster large language models via multi-token prediction. *arXiv preprint arXiv:2404.19737*, 2024.

[18] Junru Gu, Chen Sun, and Hang Zhao. Densetnt: End-to-end trajectory prediction from dense goal sets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15303–15312, 2021.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, 2020.

[21] Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. Hubert: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:3451–3460, 2021.

[22] Yihan Hu, Siqi Chai, Zhening Yang, Jingyu Qian, Kun Li, Wenxin Shao, Haichao Zhang, Wei Xu, and Qiang Liu. Solving motion planning tasks with a scalable generative model. In *European Conference on Computer Vision*, 2024.

[23] Zhiyu Huang, Zixu Zhang, Ameya Vaidya, Yuxiao Chen, Chen Lv, and Jaime Fernández Fisac. Versatile scene-consistent traffic scenario generation as optimization with diffusion. *arXiv preprint arXiv:2404.02524*, 2024.

[24] Maximilian Igl, Daewoo Kim, Alex Kuefler, Paul Mougin, Punit Shah, Kyriacos Shiarlis, Dragomir Anguelov, Mark Palatucci, Brandyn White, and Shimon Whiteson. Symphony: Learning realistic and diverse agents for autonomous driving simulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2445–2451. IEEE, 2022.

[25] Xiaosong Jia, Penghao Wu, Li Chen, Yu Liu, Hongyang Li, and Junchi Yan. Hdgt: Heterogeneous driving graph transformer for multi-agent trajectory prediction via scene encoding. *IEEE transactions on pattern analysis and machine intelligence*, 2023.

[26] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[27] Parth Kothari, Christian Perone, Luca Bergamini, Alexandre Alahi, and Peter Ondruska. Drivergym: Democratising reinforcement learning for autonomous driving. *arXiv preprint arXiv:2111.06889*, 2021.

[28] Daniel Krajzewicz, Georg Hertkorn, Christian Rössel, and Peter Wagner. Sumo (simulation of urban mobility)-an open-source traffic simulation. In *Proceedings of the 4th middle East Symposium on Simulation and Modelling (MESM20002)*, pages 183–187, 2002.

[29] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(3):3461–3475, 2022.

[30] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[32] Nico Montali, John Lambert, Paul Mougin, Alex Kuefler, Nicholas Rhinehart, Michelle Li, Cole Gulino, Tristan Emrich, Zoey Zeyu Yang, Shimon Whiteson, Brandyn White, and Dragomir Anguelov. The waymo open sim agents challenge. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2023.

[33] Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S Refaat, and Benjamin Sapp. Wayformer: Motion forecasting via simple & efficient attention networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2980–2987. IEEE, 2023.

[34] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *The Eleventh International Conference on Learning Representations*, 2023.

[35] Jonah Philion, Xue Bin Peng, and Sanja Fidler. Trajeglish: Traffic modeling as next-token prediction. In *The Twelfth International Conference on Learning Representations*, 2024.

[36] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems*, 1988.

[37] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. *OpenAI blog*, 2018.

[38] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[39] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 627–635, 2011.

[40] Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

[41] 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. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8579–8590, 2023.

[42] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Motion transformer with global intention localization and local movement refinement. In *Advances in Neural Information Processing Systems*, 2022.

[43] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[44] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10400–10409, 2021.

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

[46] Balakrishnan Varadarajan, Ahmed Hefny, Avikalp Srivastava, Khaled S Refaat, Nigamaa Nayakanti, Andre Cornman, Kan Chen, Bertrand Douillard, Chi Pang Lam, Dragomir Anguelov, et al. Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 7814–7821. IEEE, 2022.

 https://doi.org/10.52202/079017-2527

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

[48] Eugene Vinitsky, Nathan Lichtlé, Xiaomeng Yang, Brandon Amos, and Jakob Nicolaus Foerster. Nocturne: a scalable driving benchmark for bringing multi-agent learning one step closer to the real world. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2022.

[49] Yu Wang, Tiebiao Zhao, and Fan Yi. Multiverse transformer: 1st place solution for waymo open sim agents challenge 2023. *arXiv preprint arXiv:2306.11868*, 2023.

[50] 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. In *Advances in Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.

[51] Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021.

[52] Danfei Xu, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Bits: Bi-level imitation for traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2929–2936. IEEE, 2023.

[53] Ze Yang, Yun Chen, Jingkang Wang, Sivabalan Manivasagam, Wei-Chiu Ma, Anqi Joyce Yang, and Raquel Urtasun. Unisim: A neural closed-loop sensor simulator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1389–1399, 2023.

[54] Zhejun Zhang, Alexander Liniger, Christos Sakaridis, Fisher Yu, and Luc Van Gool. Real-time motion prediction via heterogeneous polyline transformer with relative pose encoding. In *Advances in Neural Information Processing Systems*, 2023.

[55] Zhejun Zhang, Christos Sakaridis, and Luc Van Gool. Trafficbots v1. 5: Traffic simulation via conditional vaes and transformers with relative pose encoding. *arXiv preprint arXiv:2406.10898*, 2024.

[56] Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. Guided conditional diffusion for controllable traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3560–3566. IEEE, 2023.

[57] Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, 2021.

[58] Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17863–17873, 2023.

[59] Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. Hivt: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8823–8833, 2022.

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: We have clearly outlined the scope and the targeted task of this work, and presented a concise introduction of our approach. The important contributions are described in detail, and the reported results obtained by our model are tested in the Waymo Open Sim Agents Challenge. This open challenge allows for fair comparisons with other baselines.

   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 have discussed the limitations of this work in 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?

https://doi.org/10.52202/079017-2527

Answer: [NA]

Justification: BehaviorGPT is an application-oriented approach for autonomous driving. No new theory has been proposed in this work.

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've described our approach in detail, including the model architecture, loss function, datasets, and metrics used for training and testing. We will also make our code publicly available.

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: The data used in this work is the Waymo Open Motion Dataset, which is publicly available. Our code will also be made public after the paper is published.

   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, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: As the data used in this work is the Waymo Open Motion Dataset, the training and testing details are available on the website of Waymo Open Dataset Challenge. We have also clarified other details in Section 3, Section 4, and the 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: [No]

   Justification: While not conducting significance tests over results, our experiments are conducted on the Waymo Open Motion Dataset, which has a large data scale. Thus, the experimental results are stable across multiple trials, and the reported results are reliable and authentic.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

https://doi.org/10.52202/079017-2527

- 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: The resources used for model training have been introduced clearly in the implementation details.

   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: There are no violations against the 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 proposed BehaviorGPT is particularly designed for smart agent simulation, which can benefit the verification of autonomous driving systems and is expected to promote the development of autonomous driving. While autonomous driving itself may be a double-sided sword, its negative side is out of the scope of this work and the developed BehaviorGPT will not directly bring negative societal impacts.

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: The paper poses no such risks.

    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: [Yes]

    Justification: The paper has cited the original paper that produced the Waymo Open Motion Dataset, and the information about the dataset is available online.

    Guidelines:

https://doi.org/10.52202/079017-2527

- 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` 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: The paper does not release new assets.

    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: The paper does not involve crowdsourcing nor 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: The paper does not involve crowdsourcing nor 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.

https://doi.org/10.52202/079017-2527