Bridging the Gap Between Multi-Step and One-Shot Trajectory Prediction via Self-Supervision

Faris Janjoš
Corporate Research
Robert Bosch GmbH
71272 Renningen, Germany
faris.janjos@de.bosch.com

Max Keller
Corporate Research
Robert Bosch GmbH
71272 Renningen, Germany
max.keller@de.bosch.com

Maxim Dolgov
Corporate Research
Robert Bosch GmbH
71272 Renningen, Germany
maxim.dolgov@de.bosch.com

J. Marius Zöllner
Research Center for
Information Technology (FZI)
76131 Karlsruhe, Germany
zoellner@fzi.de

arXiv:2306.03367v1 [cs.RO] 6 Jun 2023

Abstract—Accurate vehicle trajectory prediction is an unsolved problem in autonomous driving with various open research questions. State-of-the-art approaches regress trajectories either in a one-shot or step-wise manner. Although one-shot approaches are usually preferred for their simplicity, they relinquish powerful self-supervision schemes that can be constructed by chaining multiple time-steps. We address this issue by proposing a middle-ground where multiple trajectory segments are chained together. Our proposed Multi-Branch Self-Supervised Predictor receives additional training on new predictions starting at intermediate future segments. In addition, the model 'imagines' the latent context and 'predicts the past' while combining multi-modal trajectories in a tree-like manner. We deliberately keep aspects such as interaction and environment modeling simplistic and nevertheless achieve competitive results on the INTERACTION dataset. Furthermore, we investigate the sparsely explored uncertainty estimation of deterministic predictors. We find positive correlations between the prediction error and two proposed metrics, which might pave way for determining prediction confidence.

I. INTRODUCTION

The common separation of the Autonomous Driving (AD) stack into perception, prediction, and planning components drives a need for accurate forecasts of the future as a planner input. In prediction, different challenges exist such as but not limited to, representing the environment [1], modeling multi-agent interactions [2], capturing the multi-modality of the future motion distribution [3], adhering to kinematic constraints [4], as well as modeling a long prediction horizon [5]. In solving these tasks, various Deep Learning (DL) models usually generate predicted trajectories of the agents sharing a road situation with an autonomous vehicle (AV).

In terms of the approach to construct trajectories, prediction models can be categorized into one-shot approaches [4], [6]–[8], where a full trajectory is directly regressed, and step-wise, autoregressive approaches that generate a trajectory sequentially for each time step based on the previously predicted time steps [5], [9]–[13]. The main drawback of one-shot approaches is that long prediction horizons make it difficult to reason about comprehensive changes in scene dynamics. Since the models do not condition on future observations, the potential for errors and high uncertainty is much larger toward the end
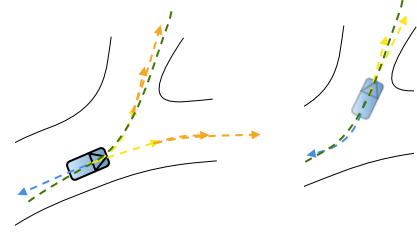
Fig. 1: Left: the Multi-Branch Self-Supervised Predictor splits the trajectory prediction problem into three equal time-segments, where the ground-truth history and future are shown in green. It builds a tree of multi-modal trajectories over two future segments (yellow and orange), with a multi-modal trajectory (two modes) chained at the output of each previous segment mode. Additionally, it 'predicts' a uni-modal trajectory in the past (blue). Right: knowing the ground-truth in training, the model builds another tree shifted by one segment into the future. By constructing multiple trees, reconstructing the past, as well as 'imagining' how the context will evolve in each segment (not depicted), the model receives additional (self-)supervision.

of a long prediction horizon. As opposed to one-shot models, step-wise, autoregressive approaches build each prediction based on the previous inference step. Similarly to one-shot approaches, they tend to accumulate errors for longer horizons, especially in interactive settings under distribution shift [14]. Furthermore, the autoregressive context makes it non-trivial to combine multi-modal predictions between successive time steps. In our work, we aim to bridge the two approaches and offer potential answers to the drawbacks at hand.

Our starting point in addressing the question of combining the different paradigms is the Self-Supervised Action-Space Predictor (SS-ASP) model in [4]. Its main characteristic is the ability to predict a latent representation of the driving context (e.g. map and motion of non-predicted agents) prior to predicting future trajectories for the vehicle of interest. In achieving this, it is trained with future context data in addition to future trajectories of the predicted agents, which separates it from the existing state-of-the-art models. Furthermore, it can 'go back in time' and check whether its reconstructions of the past are consistent with the observed ground-truth history through an auxiliary, inverse model task. These two aspects can be regarded as a form of self-supervision and they enable the model to interpret the trajectory prediction problem in a segment-wise manner, where each segment consists of multi-

ple time steps (for example, one second length) and multiple segments are chained together. With each new segment, future trajectories and context are predicted, and past trajectories of the previous segment are reconstructed. This is in contrast to pure one-shot or step-wise approaches, which do not employ such strategies. However, these abilities are sparsely explored in [4] and the proposed chaining of segments actually regresses the results compared to a simple one-shot-like setup.

In this work, we aim to incorporate novel autoregressive schemes from [15]–[17] to enable a multi-segment formulation of a one-shot trajectory prediction model. With this, we aim to answer the *research question (i): can the prediction performance of a one-shot model be improved by an autoregressive formulation that creates room for additional self-supervision?* Furthermore, in light of the multi-segment structure, we aim to investigate how the model uncertainty evolves with each segment, given a deterministic problem formulation. In this sense, we aim to answer the *research question (ii): can the confidence of the deterministic multi-segment model be accurately rated, in order to define the limitations of the model?* This might pave the way toward the prediction of a variable number of segments and thus a variable time-horizon.[1] The main contributions of our work can be summarized along:

- Segment-wise prediction: a novel paradigm where the resulting trajectory is obtained by predicting a number of 1-second time segments and chaining them together.
- Multi-Branch SS-ASP: a multi-branch and multi-segment extension of the approach from [4], trained by generating additional prediction paths (termed branches). The proposed model 'imagines' future context, reconstructs past trajectories, and combines segment-wise multi-modal predictions with various tree-search strategies. The model is deliberately simplistic in terms of interaction and multi-modality modeling but still competitive on the INTERACTION dataset [18].
- Uncertainty of deterministic predictors: we assess the given model's confidence along evolving time-segments with two novel approaches that exhibit a significant positive correlation with the measured prediction error.

II. RELATED WORK

The focus of our work is self-supervised autoregressive trajectory prediction and uncertainty estimation in the context of our proposed model. Therefore, we are interested in **autoregressive models** (both in trajectory prediction and other contexts), **self-supervision**, and **uncertainty estimation** in trajectory prediction; we outline the section accordingly.

A. Autoregressive prediction

Autoregressive trajectory prediction models [5], [9]–[13] incrementally model changes in scene dynamics compared to one-shot models [4], [6]–[8], which have larger requirements

---

[1] A use-case is stopping the prediction after a certain segment in case the uncertainty is too high. Providing only the 'confident' predictions to a planner might improve its performance since the output is not based on highly uncertain predictions (and thus potentially overconfident).

on model expressiveness within a single prediction. Among such models, [11] predicts the waypoints of the next step based on an agent's own state and the waypoints of the surrounding agents at the previous step. It handles multi-modality by predicting a fixed amount of latent intents, which condition the step-wise rolled-out trajectories. Similarly, [9] extends the problem by inferring time-varying discrete intents of the surrounding agents, which are incorporated into the step-wise discrete-continuous hybrid model. It uses a learned proposal function to 'traverse' the system and obtain multiple modes. In [19], step-wise environment prediction is done, similar to the latent context prediction in [4]. The environment prediction benefits from the autoregressive formulation; it is simpler to predict observations in a single time step than a long horizon.

Outside of vehicle trajectory prediction, autoregressive models are present in robotics and Imitation Learning (IL) [15]–[17], [20]. In [16], the so-called latent overshooting method is introduced that rolls-out new autoregressive predictions at each intermediate future step, in parallel to the first prediction sequence. This allows to increase the learning signal without additional data. An autoregressive formulation is useful in Model-Based Reinforcement Learning (RL) as well; [17] showed that a multi-step loss, based on the autoregressively predicted steps, increases the reward for deterministic planning modules compared to a single-step objective. Similarly, the Dreamer model in [21] learns behavior directly from autoregressive latent predictions ('imagination') instead of exploration. This significantly reduces the training time compared to an explorative RL agent. In summary, the potential for performance improvement as well as the larger design space compared to one-shot models motivate the usage of an autoregressive formulation in this work.

B. Self-supervision in trajectory prediction

Existing trajectory prediction models incorporate self-supervision either through (i) a separate training stage or (ii) through additional tasks. There are fewer approaches performing (ii) in the literature; [22] proposes a contrastive pre-training in which rasterized representations of intersecting trajectories are rotated or semantics are exchanged in order to learn an internal interaction representation. In contrast, [23] fine-tunes a pre-trained predictor in an online-setting to adapt to behaviors observed in inference. Among (ii), [24] enforces additional temporal and spatial consistency tasks for trajectory refinement that robustify the outputs in terms of pertubations. Additionally, [25] takes a graph-based approach where certain map and agent node features are masked out and presented as a completion task for the model.

Learning environment models through self-supervision and using the internal representations of scene dynamics for planning has received strong attention in RL [19], [26], [27]. It has shown to be a promising direction in IL-based AD as well; in [21] the model predicts an evolution of the scene in the latent space and reconstructs camera images, semantic maps, and actions taken by the AV. Similarly, the SS-ASP model [4] performs prediction in the latent space but

without full observation reconstruction. Instead, latent context predictions are compared to encoded future observations and an inverse model is learned as well [28], [29], which introduces another transition prior on the environment. Compared to [21], this is more efficient (due to lower dimensionality), however, reconstructing rich observations from the latent space induces stronger requirements on the expressiveness of the latent space.

C. Prediction uncertainty estimation

Despite the large number of deterministic trajectory predictors in literature and the importance of communicating the model uncertainty to a planner, the task has received limited attention in literature. In general, estimating the epistemic uncertainty of a prediction model is a challenging problem in DL. Several approaches in trajectory prediction use the computationally cumbersome deep ensembles [3], [30]. A more general approach is Bayesian inference, where uncertainty is directly estimated in conjunction with the prediction. However, significantly more effort is needed to design and train Bayesian networks compared to standard neural networks. In [31], a theoretical framework is described that "casts dropout training in deep neural networks as approximate Bayesian inference in deep Gaussian processes". In practice, the parameters of a Gaussian distribution are approximated by the mean and variance of multiple inference runs, each with different deactivated (dropped-out) neurons. Along these results, [32] provide a study of dropout-based Bayesian approximation in pedestrian trajectory prediction and find improved accuracy through in inference. In our work, we apply dropout-based techniques to estimate the prediction uncertainty of the developed model, due to the theoretical grounding and ease-of-use.

III. METHOD

In this section, we describe our method. In Sec. III-A we define the addressed problem of trajectory prediction and introduce notation. Sec. III-B describes the SS-ASP model from [4]. Sec. III-C extends the SS-ASP into an autoregressive formulation and offers strategies how to combine segment-wise multi-modal predictions. The proposed Multi-Branch SS-ASP model is given in Sec. III-D. Strategies how to determine the prediction uncertainty of the Multi-Branch SS-ASP are introduced in Sec. III-E.

A. Problem definition and notation

Vehicle trajectory prediction can be framed as non-interactive imitation learning [10], where given the observed information $\mathcal{D}$ and ground-truth future trajectories Y^*, we learn the conditional distribution $P(\hat{Y}|\mathcal{D})$ of future trajectories $\hat{Y}$. In practice, deterministic models represent the distribution by predicting K likely samples (modes) $\{Y_j\}_{j=1}^K$ as well as their associated pseudo-probabilities $\{p_j\}_{j=1}^K$. Furthermore, the prediction can be performed for a single vehicle or jointly for multiple vehicles in a scene. Even though joint prediction is a more sound approach to the problem [6], we limit the analysis to a single-agent setting for simplicity. Nevertheless, the proposed model has no methodological restrictions preventing an extension to joint prediction.
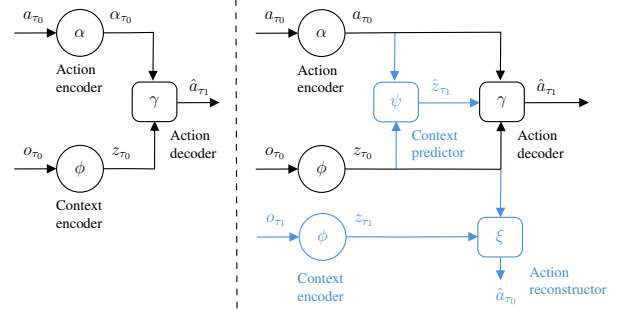

Fig. 2: Left: the (non-self-supervised) FF-ASP [4], conceptually very common in the literature (standard encoder-decoder structure) when action-spaces are excluded. Past actions a_{τ_0} and observations o_{τ_0} are encoded via an action encoder α and a context encoder ϕ into features α_{τ_0} (with a slight abuse of notation) and z_{τ_0}. Future actions $\hat{a}_{\tau_1}$ are predicted by the decoder γ. Right: self-supervised model with additional components in blue. The model additionally predicts future latent context $\hat{z}_{\tau_1}$ with the context predictor ψ and trains it against the encoding z_{τ_1} as pseudo-ground-truth. Furthermore, it reconstructs past actions $\hat{a}_{\tau_0}$ with an inverse model ξ and trains them against the past ground-truth a_{τ_0}. Multi-modal predicted actions and kinematic models converting actions to positions are omitted for clarity.

The proposed model uses a segment-wise prediction formulation where a time segment is a sequence of time steps and a full T time step prediction consists of N equal length segments[2]. Thus, we introduce supporting notation:

- i is the segment index, $i \in [1, ..., N]$
- t is the number of time steps in a segment, $T = N \cdot t$
- τ_i describes a future time segment i, $((i-1) \cdot t : it]$
- τ_0 describes a single past time segment, $(-t : 0]$

B. Self-Supervised Action-Space Predictor (SS-ASP)

The SS-ASP model [4] is the basis for developing the multi-segment model proposed in this work. It is an action-space prediction model, i.e. it predicts actions (accelerations and steering angles) and obtains positions via a kinematic model. At a high-level, it uses encoders for capturing past environment context information (e.g. a CNN encoding birds-eye-view grids or a GNN operating on graphs) into a latent (context) feature vector. Furthermore, it uses an action-based encoder for encoding past actions (e.g. RNN) and a multi-modal action-based decoder for regressing future actions (e.g. RNN). This does not separate it conceptually from a multitude of state-of-the-art approaches (irrespective of the action-space), since a vast majority uses a similar setup of encoding past information (context, trajectories) and predicting future trajectories. The described architecture is depicted in the left part of Fig. 2 (so-called Feed-Forward Action-Space Predictor (FF-ASP) [4]), where the context and action encoders, and action decoder are parameterized by ϕ, α, and γ, respectively.

The SS-ASP model stands out in the sense that, additionally to the aforementioned encoder and decoder components, it predicts a latent future context prior to predicting future actions. It trains this predicted future context against its own encoding of the future context. Furthermore, it reconstructs

---

[2]In this sense, a one-shot prediction is a single-segment prediction.

past actions via an inverse model taking in future actions. These two self-supervised tasks serve as additional regularization for the model. The SS-ASP model is depicted in the right part of Fig. 2, where the new context predictor component is parameterized by ψ, and the action reconstructor with ξ. For encoding the future context, the same past context encoder ϕ is reused, in this case receiving future information during training. The loss function of the model is

$$\mathcal{L}_{SS-ASP} = \mathcal{L}_{traj} + \mathcal{L}_{class} + \mathcal{L}_{context} + \mathcal{L}_{recon} , \quad (1)$$

with weights omitted for clarity. The trajectory regression loss is $\mathcal{L}_{traj} = \|\hat{Y}_{\tau_1}(\hat{z}_{\tau_1}) - Y^*_{\tau_1}\| + \|\hat{Y}_{\tau_1}(z_{\tau_1}) - Y^*_{\tau_1}\|$; its two terms reflect the fact that the action decoder γ in Fig. 2 is called with both predicted and encoded future context (Fig. 2 only shows the former) in training to promote consistency between components. The loss function (1) considers multi-modal outputs via the winner-takes-all [33] approach. The classification loss function $\mathcal{L}_{class}$ considers mode probabilities via cross-entropy. The context loss $\mathcal{L}_{context} = \|\hat{z}_{\tau_1} - z_{\tau_1}\|$ penalizes the mismatch between encoded and predicted future context, while the reconstruction $\mathcal{L}_{recon} = \|\hat{Y}_{\tau_0}(\hat{z}_{\tau_1}) - Y_{\tau_0}\| + \|\hat{Y}_{\tau_0}(z_{\tau_1}) - Y_{\tau_0}\|$ considers past 'predictions' (two terms promoting consistency similar to $\mathcal{L}_{traj}$). For more details, see [4].

C. Multi-Segment SS-ASP

The SS-ASP model can be extended into an autoregressive formulation with repeated calls of its components over successive time-segments. The components model the interplay between context and actions over a certain time-segment and chaining multiple calls is expected to perform reasonably well in inference. This extension is depicted in Fig. 3a. However, this naive formulation (partly presented in [4]) actually regresses the performance due to the induced distribution drift of chaining predictions on top of predictions [4], see Sec. IV-C.

In addition to the distribution drift, chaining multi-modal predictions along trajectory segments is non-trivial. If the action decoder γ generates k modes per segment, a decision has to be made on which modes to expand in the next segment. If the prediction is continued for a single mode in a segment, diversity is suppressed, while considering all permutations results in k^N trajectories. Therefore, different strategies for combining multi-modal predictions can be employed in order to traverse the tree and select K out of possible k^N modes.

In the following, six combination strategies are investigated, visualized in Fig. 4. In implementing different strategies, we make use of mode probabilities. These probabilities can be either generated by the action decoder in addition to each predicted mode, or by a separate learned classification component. Conceptually, all strategies can be placed between the All-Modes strategy, considering k^N modes, and the Single-Mode strategy that selects the highest probability mode and discards others[3]. Start-k and End-k strategies take k modes at the start or the end, and a single mode otherwise. In Best-m-of-all, the product of the probabilities of previous and subsequent modes

[3]Single-Mode can be viewed as an application of Best-first-search.
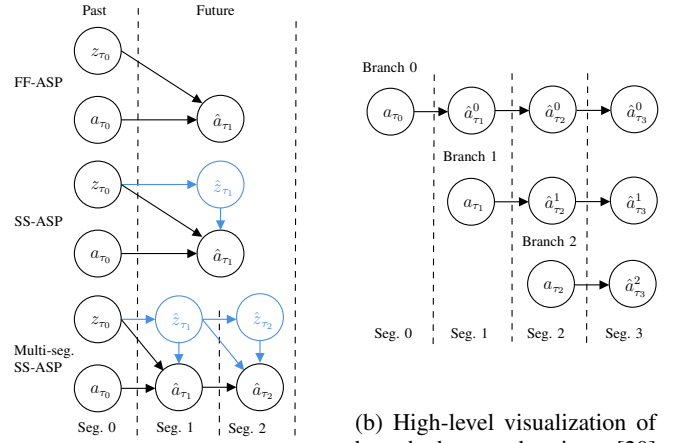
(a) High-level model comparison in terms of context and (uni-modal) action prediction. The FF-ASP uses only past features and actions to predict future actions. The SS-ASP predicts future features (blue) prior to future actions (action reconstruction not depicted). The Multi-Segment SS-ASP splits the future into segments and chains successive feature/action predictions. This induces a distribution shift however, since predictions are chained on top of predictions.

(b) High-level visualization of branched overshooting [20] over $N = 3$ future segments for predicted actions. At each intermediate future segment, a new prediction branch is started (denoted with superscript b, $\hat{a}^b_{\tau_i}$), with a shifted history and a shorter future prediction. In addition to predicting future actions in a branched manner, we branch context features as well as reconstructed actions (not visualized).

Fig. 3: Multi-segment (a) and multi-branch (b) SS-ASP depictions.

(a) All-Modes (b) Single-Mode (c) Start-k (d) End-k (e) Best-m-of-All (f) Best-m-of-Pred.
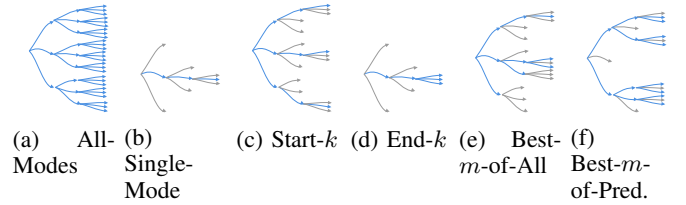
Fig. 4: Six investigated combination strategies for $N = 3$ segments. The selected modes are depicted in blue and non-selected in gray. The number of predicted modes in a segment is $k = 3$. Best-m-of-All uses $m = 3$ and Best-m-of-Prediction uses $m = 2$.

is calculated and the prediction is only continued for m most likely modes[4] of all modes within a segment. The last strategy is Best-m-of-Prediction, where the prediction is continued for $m \leq k$ most likely modes in a single multi-modal prediction, disregarding probabilities of earlier segments.

Each strategy has unique advantages and disadvantages. To quantify them, three properties are identified, see Tab. I. The property (i) is the maximum number of multi-modal prediction calls n for a sample, which serves as a proxy for the required computation time[5]. The property (ii) is the total number of obtained modes K over the entire prediction horizon. Property (iii) is qualitative and describes mode diversity via {*diverse*,

[4]Best-m-of-all corresponds to the Beam search heuristic.
[5]n can be larger than the number of segments N in the prediction horizon, e.g. $n = 13$ multi-modal prediction calls are required in the All-Modes visualization of Fig. 4.

TABLE I: Different strategies categorized by the maximum number of multi-modal prediction calls n for a sample, the total number of predicted modes K, and the mode diversity.

Strategy	K	n	Diversity
All-Modes	k^N	$\sum_{i=0}^{N-1} k^i$	*partially-diverse*
Single-Mode	1	N	*diverse*
Start-k	k	$1 + k \cdot (N-1)$	*diverse*
End-k	k	N	*partially-diverse*
Best-m-of-All	m	$1 + \sum_{i=1}^{N-1} m$	*unclear*
Best-m-of-Prediction	m^N	$\sum_{i=0}^{N-1} m^i$	*partially-diverse*

partialy-diverse, *unclear*} qualifiers. In *diverse* strategies the K resulting modes share no segment trajectory subsets, in *partially-diverse* strategies the modes share a least one segment trajectory subset, and for the *unclear* strategies the number of shared subsets varies per sample. The presented strategies of combining multi-modal prediction are general and can be integrated with different multi-modal decoders, i.e. do not depend on the specific action decoder used in this work.

D. Multi-Branch SS-ASP

The autoregressive formulation of the Multi-Segment SS-ASP opens room for advanced training methods able to reduce the distribution drift of chaining multiple predictions. In the following, branched overshooting, termed in [20], is used as well as a novel combination of context aggregation and prediction, designed to cope with the partial observability of the state through the autoregressive formulation. The resulting model aims to answer *research question (i)* from Sec. I; it is termed as the Multi-Branch SS-ASP.

1) Branched overshooting: It refers to a training method in which additional prediction branches starting from intermediate future time steps (segments) are trained in conjunction with the main branch [16], [20], visualized in Fig. 3b. A prediction branch is simply a prediction from a start segment to an end segment. For example, branch 0 refers to the full N segment main branch, as used in the Multi-Segment SS-ASP, while subsequent prediction branches start from shifted time segments. This allows the model to perform $N-1$ additional, shorter predictions (of lengths 1 to $N-1$) in addition to the N-segment prediction covering the entire prediction horizon. As a result, additional training is performed without adding training data. To the best of the authors' knowledge, this is the first work that applies such overshooting methods in trajectory prediction. We apply it for action prediction and action reconstruction, as well as latent context prediction.

Incorporating multiple segments as well as multiple branches into the self-supervised loss of Eq. (1) extends the loss function over time segments and branches. For example, the trajectory loss $\mathcal{L}_{traj}$ can be extended to a sum of losses per branch b, $\mathcal{L}_{traj} = \sum_{b=0}^{N-1} \mathcal{L}_{traj}^b$. A similar extension can be performed for the classification, context, and reconstruction components of Eq. (1). Thus, the overall loss in Eq. (1) can be extended over branches and λ_i-weighted segments

$$\mathcal{L}_{MB-SS} = \sum_{b=0}^{N-1} \sum_{i=1}^{N-b} \lambda_i \mathcal{L}_{SS-ASP}^{i,b} \ . \tag{2}$$

Compared to Eq. (1), the loss function above provides significant additional training of the model on the same data.

2) Combining context aggregation and prediction: In autoregressive models, information from a previous recurrence step is used to predict the next step. An agent's observable state (e.g. its dynamics) does not constitute sufficient statistics for its behavior – to address this limitation, autoregressive predictions can learn additional latent features. A natural framework for modeling such problems where partial observability occurs is the Partially Observable Markov Decision Process (POMDP). Furthermore, since historical behavior beyond the previous recurrence step is relevant to determining the prediction, an accumulation of 'intent' over multiple recurrence steps should be possible. In this way, the future development of the scene can be modeled following Markovian assumptions.

The partial observability of intent over the entire prediction horizon can be handled through the use of recurrence over the latent context, which motivates the usage of a context aggregator component. For a segment i, it accumulates the encoded as well as predicted contexts of previous segments into an aggregated latent context $\bar{z}_i$, $\{z_{\tau_0}, ..., \hat{z}_{\tau_{i-1}}, \hat{z}_{\tau_i}\} \xrightarrow{\chi} \bar{z}_i$ (parameterized by χ). The aggregated context $\bar{z}_i$ can be considered as a latent context state that merges scene information of multiple consecutive segments. Thus, we use it as a stand-in for wherever predicted latent context is used, either as input to an action predictor or reconstructor component (e.g. in Fig. 2 and Fig. 3a). A concept similar to incorporating recurrence in latent context prediction are the Recurrent State Space Models (RSSM) in [16], which include stochastic components as opposed to the fully deterministic Multi-Branch SS-ASP.

E. Prediction uncertainty estimation

In this section, we present novel prediction uncertainty estimation techniques for the deterministic Multi-Branch SS-ASP model, as well as a corresponding evaluation procedure. Since we perform prediction over time segments, we model the change in prediction error between successive segments i and $i-1$ to capture the possibility of early predictions being accurate and later ones deviating significantly from the ground-truth. Specifically, we model the relative change in the Minimum Average Displacement Error (minADE) for k modes

$$\Delta minADE_k(i, i-1) = minADE_k(i) - minADE_k(i-1) \ , \tag{3}$$

and find correlations to the deterministic model uncertainty. In this sense, we aim to capture the model's changing confidence over time, and answer the *research question (ii)* from Sec. I.

The first uncertainty estimation metric is the **reconstruction error**. This metric aims to capture disagreement between model components generating predictions and reconstructions (past 'predictions') in inference. If the trajectory prediction of a segment i significantly deviates from the reconstruction in the same segment, it indicates the epistemic uncertainty of the overall model. Specifically, we measure the deviation between reconstructed xy positions $\hat{Y}_{\tau_i, \xi}$ of a segment i, obtained through the action reconstruction of the inverse model ξ (see

Fig. 2), and the prediction $\hat{Y}_{\tau_i,\gamma}$, obtained through the action predictor γ (in case of segment 0 the ground-truth history Y_{τ_0})

$$\Delta_{recon}(i) = \begin{cases} \|\hat{Y}_{\tau_i,\xi} - \hat{Y}_{\tau_i,\gamma}\| & i \geq 1 , \\ \|\hat{Y}_{\tau_0,\xi} - Y_{\tau_0}\| & i = 0 . \end{cases} \quad (4)$$

In this implementation, the reconstruction error is coupled to the SS-ASP architecture due to a prerequisite for an inverse (reconstruction) model. However, adding the auxiliary task of inverse predictions to any prediction model can serve as a relatively straightforward-to-use additional regularization.

The second metric is the **mean of mode variances**. The metric is based on the application of Monte Carlo dropout in order to obtain an uncertainty estimate, naturally provided by Bayesian inference [31]. Here, we estimate the variances of predicted trajectories $\hat{Y}$ under the dropout parameter distribution $q(w|\mathcal{D}_{train})$ of weights w given the training data $\mathcal{D}_{train}$. For a single xy position in a mode j of segment i, it is

$$\sigma^2_{\hat{Y}_{t,j,i}} = \frac{1}{2}\left(\text{Var}\left(\hat{x}_{t,j,i}\right) + \text{Var}\left(\hat{y}_{t,j,i}\right)\right) . \quad (5)$$

In practice, the variances are Monte-Carlo approximated by drawing multiple samples $w \sim q(w|\mathcal{D}_{train})$ with different weights dropped out [31]. In Eq. (5), we match same modes between different inference runs (by output order). We assume that variance between modes in a single run does not change significantly by applying dropout due to the inherent model determinism. The overall metric is obtained by averaging over modes and time steps (assuming independence)

$$\Delta_{mode-var}(i) = \sum_j^k \sum_t^T \frac{\sigma^2_{\hat{Y}_{t,j,i}}}{kT} . \quad (6)$$

The metric in Eq. (6) serves as an estimate of the covariance within a Bayesian network model whose weights are Gaussian distributed, which is theoretically grounded in [31]. Therefore, it serves as an indicator of the epistemic model uncertainty.

IV. RESULTS

A. Implementation

In implementing the Multi-Branch SS-ASP model, we use various network types to implement the components in Fig. 2. The context encoder ϕ embeds semantic images containing minimal driving context information (see Fig. 5) via a ResNet18 CNN with output feature dimension 256. The action encoder α is a 1D-CNN ActorNet model adapted from [34] generating 128-dim. output features. The action and context predictors γ and ψ, as well as the reconstructor ξ, are realized by three linear layers of dimensions {512, 256, 256} (with tanh activation). The predictors γ and ψ have an additional two-layer Gated Recurrent Unit (GRU) with hidden state dimension 256, called iteratively three times. At the output of the action predictor γ, we use an additional linear layer and a softmax operation to map the feature vector to pseudo-probabilities of predicted modes. We use the same kinematic bicycle model setup as in [4] to obtain positions.

In training the model on the loss function in Eq. (2), we used the Huber loss function for all loss components in Eq. (1) (equal weights). For the multi-segment formulation, we used the segment length of 1s since we found it strikes a balance in capturing rich information on a short time interval.
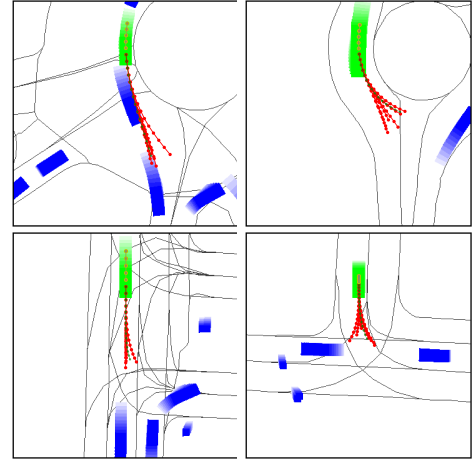

Fig. 5: Multi-Branch SS-ASP predictions on the INTERACTION dataset. The model uses a simplistic image-based representation of the driving context as input, representing past agent tracks by faded bounding boxes (prediction-ego in green, other agents in blue). The 6 predicted modes are shown in red and the ground-truth in green. The past reconstruction is shown within the prediction-ego depiction.

The loss values of different tasks (trajectory prediction and reconstruction, and context prediction) within a segment are averaged. This ensures that a segment is not over-represented in the overall loss since more terms can be present in later segments depending on the combination strategy. For efficiency, we batch different component calls over modes and among different prediction branches; we observed an approximately 1.75-times increase in training time over SS-ASP.

B. Datasets and training setup

The models are trained on the INTERACTION [18] and inD [35] datasets (using the same partition as in [6]), with 3s predictions based on 1s history ($N = 3$ in the multi-segment model). Implementation is done in PyTorch [36] with Adam optimizer [37] in training over 20 epochs and batch size 32, lasting two days for INTERACTION on a single Nvidia V100 GPU. The learning rate is set to 10^{-4} and multiplied with 0.5 if no improvement is observed in two consecutive epochs.

C. Prediction performance

We ablate the different multi-modal trajectory combination strategies in Tab. II. We see that Start-k outperforms others; this is expected due to its largest diversity in the first segment. Additionally, we offer an ablation study of the proposed approaches in Tab. III. It can be seen that (i) multi-branching and (ii) context aggregation bring boosts in metrics while multi-segmenting regresses the performance unless augmented with (i) and (ii). This is consistent with the results in [4], where a naive formulation with an End-k-like strategy is proposed. Overall, the Multi-Branch SS-ASP model brings a significant improvement of almost 25 % over the basic model. It shows that introducing additional training tasks without modifying more problem-relevant aspects (e.g. interaction modeling) can greatly improve prediction performance.

We compare the Multi-Branch-SS-ASP prediction results to reported results of other state-of-the-art models on the

TABLE II: Comparison of combination strategies for multi-segment multi-modal prediction. The parameters k and m are chosen such that the same number of resulting modes is obtained, $K = 8$.

Method	Configuration	inD [35]	
		$minADE_6$	$minFDE_6$
All-Modes	$k = 2$	0.25	0.61
Start-k	$k = 8$	**0.20**	**0.51**
End-k	$k = 8$	**0.20**	0.53
Best-m-of-All	$m = 8$	0.24	0.59
Best-m-of-Pred.	$m = 2$	0.22	0.54

TABLE III: Ablation study of proposed approaches: self-supervision, multi-segment chaining (with Start-k strategy, $k = 9$), branched overshooting, and context aggregation. Multi-branch* denotes branched overshooting (Sec. III-D1) without context aggregation (Sec. III-D2).

Model	Self-Sup.	Multi-seg.	Multi-branch	Context agg.	inD [35]	
					$minADE_9$	$minFDE_9$
FF-ASP [4]	✗	✗	✗	✗	0.22	0.56
SS-ASP [4]	✓	✗	✗	✗	0.19	0.50
Multi-seg. SS-ASP	✓	✓	✗	✗	0.20	0.52
Multi-branch* SS-ASP	✓	✓	✓	✗	0.17	0.45
Multi-branch SS-ASP	✓	✓	✓	✓	**0.17**	**0.43**

TABLE IV: Minimal displacement metrics on the INTERACTION validation dataset. All methods predict $K = 6$ modes. We do not include [6], [41] due to a different number of predicted modes.

	INTERACTION [18]	
	$minADE_6$	$minFDE_6$
TNT [42]	0.21	0.67
STG-DAT [43]	0.29	0.54
ITRA [13]	0.17	0.49
GOHOME [7]	-	0.45
FF-ASP [4]	0.12	0.35
DIPA [44]	0.11	0.34
SS-ASP [4]	0.11	0.33
Multi-Branch SS-ASP	**0.10**	**0.30**

INTERACTION validation dataset in Tab. IV. Furthermore, we evaluate the model on the INTERACTION test set online leaderboard[6], where it achieves a competitive *3rd* place in $minADE_6$ and $minFDE_6$. However, it scores *9th* in Miss Rate (MR); this is understandable since the model components are inherently ill-equipped to handle interaction modeling due to the very low-information-density environment representation and simplistic CNN encoding. For such purposes, many state-of-the-art approaches use graph- or Transformer-based [38] architectures in their encoders [6], [39] as well as target selection heuristics in their decoders [3], [40]. Such approaches could be easily integrated into the overall architecture. The generality of the self-supervision, segment-wise prediction, and branched training does not preclude component-level improvements.

D. Prediction uncertainty estimation

We evaluate the uncertainty quantification strategies from Sec. III-E by observing whether they correlate with the change in prediction error over successive segments. We quantify the prediction error by the $\Delta minADE$ in Eq. (3). In this way, a high value of the uncertainty metric could indicate that the model's predictions will deteriorate over time.

We calculate the metrics from Sec. III-E for each predicted segment on a randomly chosen 10% inD subset. The results are visualized in Fig. 6. To ensure comparability between the two methods, we group the (sorted) obtained values into four quarters, where each quarter contains 25% of the overall values (the first quarter is equivalent to the first quantile). Then, within each quarter we approximate the $\Delta minADE_k$ error distribution by a four bin histogram (lightest to darkest blue in Fig. 6). The bin intervals are determined by the quarters of the $\Delta minADE_k$ error distribution on the validation set.

Interpreting Fig. 6, we see that the change in the error distribution between quarters is evident. For example, quartile 1 of segment 1 (Fig. 6a) contains the lowest-reconstruction-error and more than 60% of its values lie in the low $\Delta minADE_k$ range (lightest blue). Similarly, in quarter 4 of Fig. 6a (containing the highest metric values) the histogram distribution is biased towards high $\Delta minADE_k$ samples (darkest blue). Therefore, a correlation between the metric and the actual change in prediction error over segments can be confirmed. Similar relationships can be found for the dropout-based mean-of-mode-variances, where 20 Monte Carlo runs are performed (we dropped-out the two linear layers before the action predictor with $p = 0.5$). Furthermore, we observe that in both metrics the histogram distribution favors higher $\Delta minADE_k$ at later segments, which is reasonable.

V. CONCLUSION

In this paper, we investigated connections between one-shot and autoregressive trajectory prediction models. We deliberately focused on the structure of output representations and the training approach, as opposed to more problem-relevant aspects such as driving context and interaction modeling, in order to better see the effects of the proposed approach. We found significant gains by converting an existing one-shot predictor into a novel, segment-wise prediction trained with self-supervision and overshooting. Furthermore, we proposed two epistemic uncertainty measures for deterministic predictors. In combination with the segment-wise output structure, they pave way for prediction of a variable time horizon with the goal of providing only confident predictions to a downstream planner.

REFERENCES

[1] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[2] Y. Yuan, X. Weng, Y. Ou, and K. Kitani, "Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting," *CoRR*, vol. abs/2103.14023, 2021. [Online]. Available: https://arxiv.org/abs/2103.14023

[3] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov *et al.*, "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction," *arXiv preprint arXiv:2111.14973*, 2021.

[4] F. Janjoš, M. Dolgov, and M. J. Zöllner, "Self-Supervised Action-Space Prediction for Automated Driving," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, 2021.

[5] Q. Lu, W. Han, J. Ling, M. Wang, H. Chen, B. Varadarajan, and P. Covington, "Kemp: Keyframe-based hierarchical end-to-end deep model for long- term trajectory prediction," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

(a) Rec. error seg. 1 (b) Rec. error seg. 2 (c) Rec. error seg. 3 (d) Mode var. seg. 1 (e) Mode var. seg. 2 (f) Mode var. seg. 3
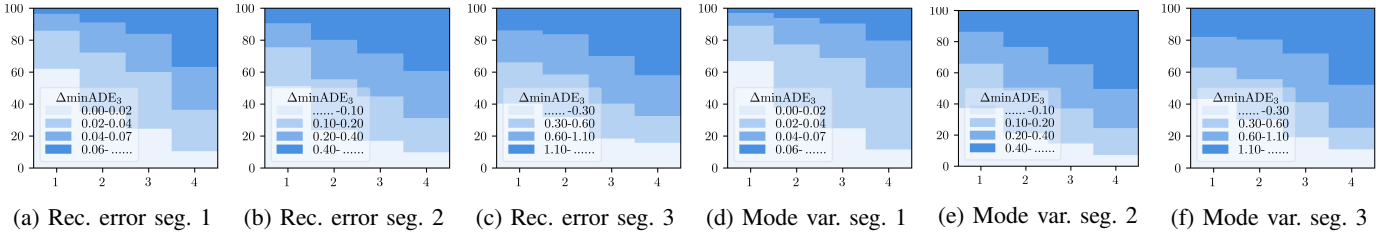
Fig. 6: Evaluation of reconstruction error and mean-of-mode-variances uncertainty estimation metrics over three predicted segments. The metric values are split into four quarters (horizontal axes) and the distribution of each quarter w.r.t. ΔminADE_k (Eq. (3)) is shown in different shades of blue (vertical axes indicate %). The metrics correlate with prediction error change – it can be seen that lower metric quarters contain mostly lower ΔminADE_k values (lighter shades of blue) and higher metrics contain higher ΔminADE_k (darker shades).

[6] F. Janjoš, M. Dolgov, and M. J. Zöllner, "StarNet: Joint Action-Space Prediction with Star Graphs and Implicit Global-Frame Self-Attention," *arXiv preprint arXiv:2111.13566*, 2021.

[7] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "Gohome: Graph-oriented heatmap output for future motion estimation," *arXiv preprint arXiv:2109.01827*, 2021.

[8] S. Casas, C. Gulino, S. Suo, K. Luo, R. Liao, and R. Urtasun, "Implicit Latent Variable Model for Scene-Consistent Motion Forecasting," *arXiv preprint arXiv:2007.12036*, 2020.

[9] X. Huang, G. Rosman, I. Gilitschenski, A. Jasour, S. G. McGill, J. J. Leonard, and B. C. Williams, "Hyper: Learned hybrid trajectory prediction via factored inference and adaptive sampling," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022.

[10] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction Conditioned on Goals in Visual Multi-Agent Settings," in *Proceedings of the IEEE Int. Conf. on Computer Vision*, 2019.

[11] C. Tang and R. R. Salakhutdinov, "Multiple Futures Prediction," in *Advances in Neural Information Processing Systems*, 2019.

[12] R. Mahjourian, J. Kim, Y. Chai, M. Tan, B. Sapp, and D. Anguelov, "Occupancy flow fields for motion forecasting in autonomous driving," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, apr 2022. [Online]. Available: https://doi.org/10.1109%2Flra.2022.3151613

[13] A. Scibior, V. Lioutas, D. Reda, P. Bateni, and F. Wood, "Imagining The Road Ahead: Multi-Agent Trajectory Prediction via Differentiable Simulation," *arXiv preprint arXiv:2104.11212*, 2021.

[14] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proc. of the 14th international conf. on artificial intelligence and statistics*, 2011.

[15] A. Venkatraman, M. Hebert, and J. A. Bagnell, "Improving multi-step prediction of learned time series models," in *AAAI*, 2015.

[16] D. Hafner, T. P. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," *CoRR*, vol. abs/1811.04551, 2018. [Online]. Available: http://arxiv.org/abs/1811.04551

[17] M. Lutter, L. Hasenclever, A. Byravan, G. Dulac-Arnold, P. Trochim, N. Heess, J. Merel, and Y. Tassa, "Learning dynamics models for model predictive agents," *CoRR*, vol. abs/2109.14311, 2021. [Online]. Available: https://arxiv.org/abs/2109.14311

[18] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle *et al.*, "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative MoTION Dataset in Interactive Driving Scenarios with Semantic Maps," *arXiv preprint arXiv:1910.03088*, 2019.

[19] A. Hu, G. Corrado, N. Griffiths, Z. Murez, C. Gurau, H. Yeo, A. Kendall, R. Cipolla, and J. Shotton, "Model-based imitation learning for urban driving," *arXiv preprint arXiv:2210.07729*, 2022.

[20] A. H. Li, P. Wu, and M. Kennedy, "Replay overshooting: Learning stochastic latent dynamics with the extended kalman filter," in *2021 IEEE International Conf. on Robotics and Automation (ICRA)*, 2021.

[21] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv:1912.01603*, 2019.

[22] H. Ma, Y. Sun, J. Li, and M. Tomizuka, "Multi-agent driving behavior prediction across different scenarios with self-supervised domain knowledge," in *2021 IEEE Intelligent Transportation Systems (ITSC)*.

[23] M. Geisslinger, P. Karle, J. Betz, and M. Lienkamp, "Watch-and-learn-net: Self-supervised online learning for probabilistic vehicle trajectory prediction," in *2021 IEEE international conference on systems, man, and cybernetics (SMC)*, 2021.

[24] M. Ye, J. Xu, X. Xu, T. Cao, and Q. Chen, "Dcms: Motion forecasting with dual consistency and multi-pseudo-target supervision," *arXiv preprint arXiv:2204.05859*, 2022.

[25] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, and C. Schmid, "Vectornet: Encoding HD maps and agent dynamics from vectorized representation," *CoRR*, vol. abs/2005.04259, 2020. [Online]. Available: https://arxiv.org/abs/2005.04259

[26] P. Wu, A. Escontrela, D. Hafner, K. Goldberg, and P. Abbeel, "Daydreamer: World models for physical robot learning," *arXiv preprint arXiv:2206.14176*, 2022.

[27] J. Y. Koh, H. Lee, Y. Yang, J. Baldridge, and P. Anderson, "Pathdreamer: A world model for indoor navigation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[28] P. Agrawal, A. V. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to Poke by Poking: Experiential Learning of Intuitive Physics," in *Advances in Neural Information Processing Systems*, 2016.

[29] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International conference on machine learning*, 2017.

[30] A. Filos, P. Tigkas, R. McAllister, N. Rhinehart, S. Levine, and Y. Gal, "Can autonomous vehicles identify, recover from, and adapt to distribution shifts?" in *Int. Conf. on Machine Learning (ICML)*, 2020.

[31] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016.

[32] A. Nayak, A. Eskandarian, and Z. Doerzaph, "Uncertainty estimation of pedestrian future trajectory using bayesian approximation," *IEEE Open Journal of Intelligent Transportation Systems*, 2022.

[33] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-If Motion Prediction for Autonomous Driving," *arXiv preprint arXiv:2008.10587*, 2020.

[34] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, and R. Urtasun, "Learning Lane Graph Representations for Motion Forecasting," in *European Conference on Computer Vision*, 2020.

[35] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," *CoRR*, vol. abs/1911.07602, 2019. [Online]. Available: http://arxiv.org/abs/1911.07602

[36] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *arXiv preprint arXiv:1912.01703*, 2019.

[37] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[38] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems*, 2017.

[39] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple & efficient attention networks," *arXiv preprint arXiv:2207.05844*, 2022.

[40] J. Gu, C. Sun, and H. Zhao, "Densetnt: End-to-end trajectory prediction from dense goal sets," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021.

[41] F. Janjoš, M. Dolgov, M. Kurić, Y. Shen, and J. M. Zöllner, "San: Scene anchor networks for joint action-space prediction," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, 2022.

[42] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid *et al.*, "TNT: Target-driveN Trajectory Prediction," *arXiv preprint arXiv:2008.08294*, 2020.

[43] J. Li, H. Ma, Z. Zhang, J. Li, and M. Tomizuka, "Spatio-Temporal Graph Dual-Attention Network for Multi-Agent Prediction and Tracking," *arXiv preprint arXiv:2102.09117*, 2021.

[44] A. Knittel, M. Hawasly, S. V. Albrecht, J. Redford, and S. Ramamoorthy, "Dipa: Diverse and probabilistically accurate interactive prediction," *arXiv preprint arXiv:2210.06106*, 2022.