Tracing the Influence of Predecessors on Trajectory Prediction

Mengmeng Liu, Hao Cheng, Michael Ying Yang
University of Twente, The Netherlands
mengmeng.liu1998@gmail.com; {h.cheng-2, michael.yang}@utwente.nl

Abstract

In real-world traffic scenarios, agents such as pedestrians and car drivers often observe neighboring agents who exhibit similar behavior as examples and then mimic their actions to some extent in their own behavior. This information can serve as prior knowledge for trajectory prediction, which is unfortunately largely overlooked in current trajectory prediction models. This paper introduces a novel Predecessor-and-Successor (PnS) method that incorporates a predecessor tracing module to model the influence of predecessors (identified from concurrent neighboring agents) on the successor (target agent) within the same scene. The method utilizes the moving patterns of these predecessors to guide the predictor in trajectory prediction. PnS effectively aligns the motion encodings of the successor with multiple potential predecessors in a probabilistic manner, facilitating the decoding process. We demonstrate the effectiveness of PnS by integrating it into a graph-based predictor for pedestrian trajectory prediction on the ETH/UCY datasets, resulting in a new state-of-the-art performance. Furthermore, we replace the HD map-based scene-context module with our PnS method in a transformer-based predictor for vehicle trajectory prediction on the nuScenes dataset, showing that the predictor maintains good prediction performance even without relying on any map information.

1. Introduction

Trajectory prediction is a fundamental problem in various fields, such as transportation, robotics, and surveillance. Trajectory predictors, typically, rely on observed trajectories of the target and neighboring agents to model their motion dynamics and interactions [1, 22, 16]. In addition, state-of-the-art trajectory prediction methods, especially for autonomous driving, often make use of other prior information, such as semantic maps, e.g., [17, 37, 36, 51], high-definition (HD) maps, e.g., [11, 19, 32, 57, 18, 44, 42, 41, 10, 34], or goal information, e.g., [13, 12, 55, 14, 8]. Besides that, in real-world traffic scenarios, we often observe that pedestrians and car drivers follow the trace of neighbors who have conducted a similar behavior in the same scene, assuming that all the agents behave rationally, e.g., complaint to scene constraints and avoid collisions. As shown in Fig. 1(b), given previously-captured frames in this toy scenario, the future trajectory of the target agent (red box) becomes more predictable if we can identify similar moving patterns in the same space that have demonstrated by other agents (dark green). This implies that trajectories observed from other agents are not only useful for interaction modeling, but to some extent also can be treated as an informative prior to facilitate the prediction.

However, valuable dynamic information from the neighboring agents within the same scene to guide the target agent is often overlooked by mainstream trajectory prediction methods that heavily rely on static scene information, as shown in Fig. 1(a). Although existing memory-based search techniques like [30, 47, 24] and clustering methods such as [38, 49, 31] use historical trajectories to iden-
tify similar moving patterns, they require extra efforts to accumulate the historical trajectories before carrying out the searching step. Also, there could be a huge time lag between the current target agent and the historical agents, which can seriously hinder the match of spatial and temporal relationships between the current and past traffic scenes which are associated with many changing factors.

Therefore, the research gap mentioned above motivates us to propose a novel Predecessor-and-Successor (PnS) method to explore the influence of neighboring agents (potential predecessors) on the guidance of the target agent (successor). Predecessors are defined as agents that have been concurrent in the same scene as the target agent and could potentially be part of its guide. In other words, in addition to modeling the interactions between the target and neighboring agents for collision avoidance, we also explicitly model the potential guidance of the predecessors identified from these neighboring agents on the target agent. As shown in Fig. 1 (b) and (c), PnS employs an attention-based probabilistic approach to identify predecessors and learn their influence on the target agent. For example, neighboring agents \(\{n_3, n_5, n_1\}\) are identified as potential predecessors \(\{p_1, p_2, p_3\}\) indexed by the rank of probabilities based on the alignment between their spatial and temporal relationships. Compared to the memory- and cluster-based methods, no extra efforts are needed to collect all the historical trajectories beforehand, and the time lag is much smaller from observing the predecessors to the successor following the predecessors’ trace only after a few seconds.

To demonstrate the effectiveness of the PnS method for both pedestrian and vehicle trajectory predictions, we integrate PnS into a state-of-the-art graph-based predictor GATraj [7] for pedestrian trajectory prediction on the ETH/UCY [35, 23] datasets, and we replace the HD map-based scene-context module with PnS in another transformer-based predictor LAformer [26] that holds a high rank for vehicle trajectory prediction on the nuScenes [3] benchmark. The main contributions of this work are summarized as follows:

- **Our work proposes a novel Predecessor-and- Successor (PnS) method to learn the probabilistic influence from neighboring agents on the target agent. It effectively explores the predecessors motion as prior information to guide the prediction of the successor’s trajectory.**

- **We demonstrate that leveraging predecessor information further pushes the state-of-the-art performance for the pedestrian trajectory prediction on the ETH/UCY datasets. In the mapless-based setting for the vehicle trajectory prediction on the nuScenes dataset, the PnS method largely mitigates the performance degradation when the HD-map information is removed.**

### 2. Related Work

**Multimodal trajectory prediction.** Due to the stochastic nature of road agents’ future trajectories, arising from e.g., their randomness, subjectivity of intent, mutual influence, and scene constraints, predicting multimodal trajectories has become a prevalent approach in trajectory prediction. Here, multimodal prediction refers to generate multiple plausible trajectories of the target agent. This trend is further motivated by large-scale benchmarks featuring real-world traffic scenarios [35, 23, 3, 5, 53, 46, 21]. Various methods address behavior uncertainty, including using Gaussian or Laplacian mixture models trained with Mixture Density Networks (MDNs) to estimate the likelihood of each mode [57, 37, 11, 10]. Another approach involves modeling multimodality implicitly via latent variables sampled from a prior distribution to generate diverse futures, including Variational Auto-Encoders (VAEs) [37, 51, 6], Generative Adversarial Networks (GANs) [16, 22], and Diffusion models [15, 29]. In this paper, we follow the mainstream approaches to demonstrate our PnS by integrating it into two representative multimodal trajectory predictors [7, 26]. To account for the multimodality of predictions, PnS employs a predecessor tracing module to identify multiple potential predecessors and learn a probabilistic guidance from each of the predecessors on the target agent to generate multiple predictions.

**Interaction modeling.** Effectively modeling the interactions among agents is crucial to account for their mutual influences. Many of the approaches only focus on the interactions between the target and neighboring agents in the observation time for interaction modeling [40]. Their motion encodings are aggregated by e.g., pooling [1, 16, 48], message passing [54, 4, 24] via a multi-scale graph using Graph Convolutional Networks (GCNs) [45], and attention mechanisms [42] that focus on the salient spatial and temporal features of the encodings [51, 27, 6, 26, 18]. Our work goes beyond by modeling the interactions between the target and neighboring agents in the observation time using the attention mechanisms. We also condition the target agent’s future trajectory on the historical trajectories by learning a probabilistic interactive influence of the predecessor agents. As we assume that these predecessors conducted their behavior rationally following the speed profiles complaint to scene constraints; The moving patterns derived from the predecessors in the same scene as the target agent is treated as a prior to better guide the prediction.

**Prior information from historical trajectories.** Memory-based methods [30, 47, 24] utilize historical trajectories to improve prediction performance. For example, [30] reads trajectories that are most likely to occur in future from stored observations in memory and uses them to augment the encodings of the target agent. Similarly, [47] employs a
pair of memory banks to store representative instances from the training set, acting as the prefrontal cortex in the neural system. It also employs a trainable memory addresser to adaptively search for situations similar to current ones in the memory bank, acting as the “basal ganglia”. [24] forecasts multiple paths based on historical trajectories by modeling multi-scale graph-based spatial transformers combined with a trajectory smoothing algorithm named “Memory Replay” using a memory graph. Moreover, cluster-based methods aggregate moving patterns, such as grouping target agent-neighbors [38, 49] and focusing on historical group trajectories [31], for trajectory prediction. Also, [39] clusters the trajectories into a fixed number of categories for trajectory classification. However, there could be a huge time lag between the current and past traffic scenes that are associated with many changing factors, like traffic density and scene contexts. This makes it difficult for the existing memory-based search techniques and clustering methods to distinguish the impact of historical trajectories on the target agent, taking into account its current traffic situation and motion dynamics. Moreover, these methods require to accumulate historical trajectories to build up the memory base beforehand, which causes extra burden and may limit their application in new scenes. In contrast, we propose a probabilistic Predecessor-and-Successor method that leverages trajectories of the neighboring agents to identify representative predecessors as reference trajectories for the target agent. Because the target and neighboring agents appear at the same time and share the same scene, their spatial and temporal relationships can be better modeled.

3. Method

3.1. Problem formulation

Following mainstream methods [11, 14, 57, 41], we define the motion forecasting problem as predicting the subsequent trajectory $Y_{t+1:tf}$ of a target agent $i$ given the set of observed trajectories $X$ of a total of $N$ agents in the same scenario, including both the target and neighboring agents. To differentiate the target and neighboring agents, we use $X_{t_n-1:0}$ to denote the observed trajectory of target agent $i$ and $X = X \setminus X^i$ the remaining set of the observed trajectories of neighboring agents. The observation time horizon is $\{t_n - 1, \ldots, 0\}$ and the future time horizon is $\{1, \ldots, tf\}$. To simplify the notation, time steps are omitted if they are not otherwise noted, and we call the target agent successor in the remaining of this paper.

To fully exploit the observed trajectories of the neighboring agents $X$, a predecessor tracing function $g(X, X^i)$ learns a probabilistic guidance from the predecessors on the successor $i$. Consequently, the prediction $\hat{Y}^i = f(X, g(\cdot, \cdot))$ conditioned $Y$ on $X$ and $g(\cdot, \cdot)$. Compared to many previous trajectory predictors formulated as $\hat{Y}^i = f(X)$, the only added component is $g(\cdot, \cdot)$, which does not require any extra input data. Hence, this predecessor tracing can be easily incorporated into these predictors.

3.2. Predecessor tracing

To effectively trace the impact of predecessors on the successor, we propose a time step-wise relationship predictor to identify their spatial relationship and use attention
In order to model the influence from predecessors, we propose to use a cross-attention mechanism to design the interaction mapping function \( \pi(\cdot, \cdot) \). Specifically, we use linear projections to transform predecessor \( p \)'s motion encoding \( h^p_t \) into a query vector \( Q \), and the successor's motion encoding \( h^i_t \) into key \((K)\) and value \((V)\) vectors. These vectors are then used as inputs to a scaled dot-product attention block to calculate the attention score \( s^{i,p} \).

\[
s^{i,p} = \text{Softmax}(\frac{QR^T}{\sqrt{d_k}})V,
\]

where \( d_k \) is the dimensionality of the key vectors. For more details on the encodings, refer to Sec. 3.3.

The influence from the predecessor \( p \) to successor \( i \) is then mapped by taking as inputs the concatenation of \((h^i_t, h^p, s^{i,p})\). Here, we implement a Multi-Layer Perceptron (MLP) for the mapping. The output of the MLP is denoted by \( \pi(h^i_t, h^p) \).

\[
\pi(h^i_t, h^p) = \text{MLP}(h^i_t, h^p, s^{i,p}).
\]

Rather than using a deterministic approach to model the influence of predecessors, we propose a probabilistic approach that better considers the stochasticity of the successor's behavior. Specifically, we use a Softmax function to calculate the probability of predecessor \( p \) influencing the successor \( i \) at time step \( t \) among all the potential \( N \setminus i \) predecessors.

\[
\hat{X}^p_i = g(X^p_t, X^i_t) = \frac{\exp(\pi(h^i_t, h^p))}{\sum_{n=1}^{N \setminus i} \exp(\pi(h^i_t, h^n))}.
\]

### 3.3. Trajectory encoding and decoding

In this section, we describe the trajectory encoding and decoding process of our proposed model. Similar to previous attention-based models [51, 27, 6, 26, 18], we utilize the self-attention mechanism to encode the motion dynamics of the successor, neighboring, and predecessor agents, and decode the future trajectories of the successor, as shown in Fig. 2.

**Encoding.** We start by extracting the motion encodings of the successor, neighboring, and predecessor agents using a MLP followed by a Gated Recurrent Unit (GRU).

\[
h^o_t = \text{GRU}(\text{MLP}(X^o_t)),
\]

where \( o = \{i, j, p\} \) denotes the index of the successor, neighboring, or predecessor agent, and \( j, p \in N \setminus i \).

Next, we aggregate the interaction information among these three types of agents using the self-attention mechanism, as shown in Eq. (5):

\[
h^o_t = h^o_t + \text{SelfAtt}(h^o_t),
\]

where SelfAtt represents the self-attention function. In this step, the pair-wise interaction information among them is learned by the attention and added to their original encodings via the skip connection. In this way, their interconnections are aggregated.

**Decoding.** To generate multimodal predictions for the successor, we implement a Laplacian Mixture Density Network (MDN) decoder following [57, 26].

First, instead of focusing on the impact of a single predecessor, we include multiple potential predecessors to enable the decoder to mimic the successor’s stochastic behavior. For example, the predecessors may turn into different directions at an intersection or drive at different speeds. To achieve this goal, we select \( K \) predecessors based on their probability score \( \hat{X}^p_i \) in a descending order and aggregate these predecessors’ motion encodings, as shown in Eq. (6):

\[
\hat{X}^K_i = \text{ConCat}[(\hat{X}^p_i, \hat{X}^p_i)_{i=1}^{K}].
\]

After that, the decoder takes as input the motion encoding \( h^o_t \) of the successor and \( \hat{X}^K_i \) of the \( K \) most influencing predecessors and outputs the distributions of the successor’s future positions. More specifically, The outputs are parameterized by the location \( \mu^m \) and scale \( b^m \) parameters of a total of \( M \) components, where \( m \in M \) and \( M \) corresponds to the different modalities of the predictions.

Similar to the encoding process, we use a MLP layer followed by a GRU layer to implement the decoder. It outputs the location and scale parameters of each component of the MDN, as well as the associated likelihood \( \pi_m \).

### 3.4. Incorporating with the existing models

To analyze the effectiveness of the proposed PnS method, we incorporated it into two trajectory predictors – GATraj [7] and LAformer [26]. These two models are chosen based on the following reasons: First, both GATraj and LAformer are one of the latest models and have shown excellent performances on the ETH/UCY datasets for pedestrian trajectory prediction and nuScenes dataset for vehicle trajectory prediction, respectively; Second, they apply different mainstream frameworks for trajectory prediction. Namely, GATraj exploits a graph-based framework with message passing, while LAformer employs the transformer framework with attention mechanisms, to learn spatial and temporal information for trajectory prediction. These two different models are representative examples to demonstrate the compatibility of the PnS method. Third, both models use a MDN decoder, the PnS method can be smoothly incorporated into the backbones and jointly trained without drastic modification. It should be noted that because the original LAformer heavily relies on lane segments from an HD map as a strong prior to guide the prediction. Hence, we substitute the lane alignment module in LAformer with
our PnS method to guide the prediction.

3.5. Training

The predecessor tracing module is optimized using the binary cross-entropy loss $\mathcal{L}_{\text{PnS}}$ to improve the probability estimation. The loss function is defined as follows:

$$\mathcal{L}_{\text{PnS}} = \sum_{t=1}^{T} \mathcal{L}_{\text{CE}}(\mathbf{X}_t^p, \hat{\mathbf{X}}_t^p),$$

where $\mathbf{X}_t^p$ and $\hat{\mathbf{X}}_t^p$ denote the ground truth and predicted probability of predecessor $p$ influencing the successor $i$ at time step $t$, respectively.

In the training phase, we identify the predecessor that has the closest spatial relationship and moving patterns with the successor at each time step. This is achieved by identifying predecessor $p$ from $\hat{\mathbf{X}}$ using a distance metric $\phi$.

$$\mathbf{X}_t^p = \begin{cases} 1 & \text{if } p = \arg \min_{p \in \mathcal{N}\setminus i} \phi(Y_i^t, \hat{\mathbf{X}}), \\ 0 & \text{otherwise.} \end{cases}$$

Specifically, we use the $L_2$ distance metric to find the predecessor that is closest to the successor at time step $t$. The identified predecessor is labeled as the true predecessor, while all other agents from $\hat{\mathbf{X}}$ are labeled as false predecessors.

Following previous works [28, 57, 10], we optimize the decoder in both GATraj and LAformer using the negative log-likelihood (NLL) of the best predicted mode $m^*$ of the Laplacian MDN.

$$\mathcal{L}_{\text{NLL}} = \frac{1}{t_f} \sum_{t=1}^{t_f} -\log P(Y_t^i|\mu_i^{m^*}, b_i^{m^*}),$$

where $\mu_i^{m^*}$ and $b_i^{m^*}$ are the location and scale parameters of the component, respectively, and $m^*$ represents the mode with the minimum $L_2$ error of the predicted and ground truth trajectories among the total $M$ components. We utilize the cross-entropy to optimize the mode classification.

$$\mathcal{L}_{\text{cls}} = \sum_{m=1}^{M} -\pi_m \log(\hat{\pi}_m),$$

where $\pi_m$ denotes the target probability of the mode. This target probability is defined by a soft displacement function, using the same method proposed in [57].

The overall objective function is formulated as:

$$\mathcal{L} = \lambda \mathcal{L}_{\text{PnS}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{NLL}},$$

where $\lambda$ is a hyperparameter that controls the weights of loss terms in the objective function, allowing us to balance their respective contributions.

4. Experiments

4.1. Experimental setup

**Dataset.** We utilize the ETH/UCY datasets [35, 23] to train and test our PnS method using GATraj [7] for pedestrian trajectory prediction. These datasets consist of multiple subsets captured at different locations, each with varying pedestrian densities, including Eth, Hotel, Uni, Zara1, and Zara2. Following the most common setting [1, 16, 37, 51, 31], the trajectories are down-sampled to a frequency of 2.5 Hz, with an observation time horizon of 3.2 s and a prediction time horizon of 4.8 s. For training and testing, we follow the standard leave-one-out data partitioning to train the models on four out of these five subsets and test them on the remaining one. This procedure is repeated for each subset. Moreover, we employ the nuScenes dataset [3] for training and testing our PnS method using LAformer [26] for vehicle trajectory prediction. This dataset encompasses various driving scenarios involving complex intersections and interactions with pedestrians, cyclists, and other elements. It comprises a total of 245,414 trajectory instances across 1,000 driving scenes, each lasting 20 seconds and sampled at 2Hz. The observation time horizon is set to tw seconds, while the prediction time horizon is set to six seconds. To facilitate offline training and validation, 850 scenes are provided with ground truth information, while the remaining 150 scenes are reserved for online testing.

**Evaluation metrics.** We adhere to standard evaluation metrics to assess the prediction performance [1, 3, 57]. Specifically, we employ displacement errors (DE) to evaluate the performance on the ETH/UCY and nuScenes datasets. In particular, we utilize two metrics: mFDE and mADE, which measure the minimum $L_2$ error in meters at the Final step and the Average of each step, respectively, for predicting $K$ modes. The letter “m” signifies the minimum error among the $K$ modes. In the case of nuScenes, we set $K$ to five or ten, while for ETH/UCY, it is set to 20. It is important to note that for all evaluation metrics, a lower value indicates better performance.

**Implementation details.** To ensure compatibility between PnS and GATraj [7] and LAformer [27], we set the hidden dimensions of all feature vectors to match the original configuration. We set the predecessor number $K$ to 2, aiming to strike a balance between the number of predecessors and the diversity of predictions. The value of $\lambda$ is set to 0.5, which helps maintain a balance among the loss terms in the objective function indicated by Eq. (10). For the activation function of the intermediate layers, we use the Rectified Linear Unit (ReLU). In the decoder, we employ the Exponential Linear Unit (ELU), specifically ELU(.) + 1 + $\epsilon$, as the activation function to generate positive probability estimations. Here, $\epsilon$ is set to $1e^{-3}$. All the models were trained on eight RTX 3090 GPUs using the Adam optimizer [20].
4.2. Quantitative results and comparison

To effectively demonstrate the performance enhancement achieved by incorporating the PnS method into GATraj and LAformer, we present a comparison of their performance with and without PnS in Tables 1 and 2.

Table 1 reveals that GATraj+PnS exhibits improved performance on Eth, Uni, and Zara2, as measured by FDE20. Despite GATraj only yielding minimal errors in each subset, the addition of our PnS method further reduces prediction errors, particularly in datasets with high pedestrian density such as Uni by 2.6% in mFDE20 and Zara2 by 8.3% and 4.8% in mADE20 and mFDE20, respectively. This is because the predecessor tracing module can derive information cues from more neighboring agents.

Table 1. Quantitative results on ETH/UCY [35, 23] measured by mADE20/mFDE20.

<table>
<thead>
<tr>
<th>Models</th>
<th>Eth</th>
<th>Hotel</th>
<th>Uni</th>
<th>Zara1</th>
<th>Zara2</th>
</tr>
</thead>
<tbody>
<tr>
<td>GATraj [7]</td>
<td>0.15/0.42</td>
<td>0.10/0.15</td>
<td>0.21/0.38</td>
<td>0.16/0.28</td>
<td>0.12/0.21</td>
</tr>
<tr>
<td>GATraj [7]+PnS</td>
<td>0.26/0.40</td>
<td>0.10/0.15</td>
<td>0.21/0.37</td>
<td>0.16/0.28</td>
<td>0.11/0.20</td>
</tr>
<tr>
<td>Improvement</td>
<td>-4.7%</td>
<td>-/-</td>
<td>-2.6%</td>
<td>-/-</td>
<td>8.3%/4.8%</td>
</tr>
</tbody>
</table>

Table 2 showcases the performance of PnS integrated into LAformer on the nuScenes dataset. The comparison between LAformer and LAformer+PnS reveals that LAformer’s performance is significantly compromised when it relies solely on observed trajectories without any map information. However, the inclusion of the PnS method greatly mitigates this issue, resulting in a reduction of approximately 5.1% in mADE5 and 9.8% in mADE10. This advantage indicates that the PnS method offers a practical alternative for vehicle trajectory prediction in scenarios where map information is unavailable, such as when a vehicle enters a new location without access to HD map data.

Interestingly, we also observe that providing both HD map and predecessor information to LAformer does not yield a combined improvement. Our conjecture is that the HD map already contains rich contextual details, including lane segments, road geometry, and traffic rules. Consequently, when we simply concatenate these two types of information, the PnS method may not exert a strong influence on the successors in this specific setup.

Table 2. The results on the nuScenes [3] test set.

<table>
<thead>
<tr>
<th>Model</th>
<th>mADE5</th>
<th>mADE10</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAformer [27]</td>
<td>1.57</td>
<td>1.32</td>
</tr>
<tr>
<td>LAformer [27]+PnS</td>
<td>1.49</td>
<td>1.19</td>
</tr>
<tr>
<td>Improvement</td>
<td>5.1%</td>
<td>9.8%</td>
</tr>
<tr>
<td>LAformer [27]+HD</td>
<td>1.19</td>
<td>0.93</td>
</tr>
<tr>
<td>LAformer [27]+HD+PnS</td>
<td>1.20</td>
<td>0.93</td>
</tr>
<tr>
<td>Improvement</td>
<td>-0.8%</td>
<td>-</td>
</tr>
</tbody>
</table>

Furthermore, we conduct a benchmark comparison of GATraj+PnS and LAformer+PnS with current models in Tables 3 and 4 for pedestrian and vehicle trajectory prediction, respectively.

In Table 3, we compare GATraj+PnS with the most recent top models. Among the other models, particularly the Retrospective-Memory-based MemoNet [47] and the scene history-based SHENet [31] share similarities with the PnS concept to use existing trajectories as prior information. We also include the latest diffusion-based LED [29] that holds the top performance for this task. GATraj+PnS achieves the best performance across all subsets, except for Zara1 in terms of FDE20 compared to LED. This sets a new state-of-the-art performance on the ETH/UCY datasets for pedestrian trajectory prediction, even without the effort to accumulate historical trajectories for building up a memory base.

Table 3. The comparison on the ETH/UCY datasets measured by ADE20/FDE20. The best performance is in boldface.

<table>
<thead>
<tr>
<th>Models</th>
<th>Eth</th>
<th>Hotel</th>
<th>Uni</th>
<th>Zara1</th>
<th>Zara2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Social-GAN [16]</td>
<td>0.81/1.52</td>
<td>0.72/1.61</td>
<td>0.60/1.26</td>
<td>0.34/0.69</td>
<td>0.42/0.84</td>
</tr>
<tr>
<td>Trajectron++[37]</td>
<td>0.67/1.18</td>
<td>0.18/0.28</td>
<td>0.30/0.54</td>
<td>0.25/0.41</td>
<td>0.18/0.32</td>
</tr>
<tr>
<td>STAR [50]</td>
<td>0.36/0.65</td>
<td>0.17/0.36</td>
<td>0.31/0.62</td>
<td>0.26/0.55</td>
<td>0.22/0.46</td>
</tr>
<tr>
<td>AgentFormer [51]</td>
<td>0.46/0.75</td>
<td>0.14/0.22</td>
<td>0.25/0.45</td>
<td>0.18/0.30</td>
<td>0.14/0.24</td>
</tr>
<tr>
<td>MID [15]</td>
<td>0.39/0.66</td>
<td>0.13/0.22</td>
<td>0.22/0.45</td>
<td>0.17/0.30</td>
<td>0.13/0.27</td>
</tr>
<tr>
<td>LB-EBM [33]</td>
<td>0.30/0.52</td>
<td>0.13/0.20</td>
<td>0.27/0.52</td>
<td>0.20/0.37</td>
<td>0.15/0.29</td>
</tr>
<tr>
<td>PCCSNet [39]</td>
<td>0.28/0.54</td>
<td>0.11/0.19</td>
<td>0.29/0.60</td>
<td>0.21/0.44</td>
<td>0.15/0.34</td>
</tr>
<tr>
<td>GP-Graph [2]</td>
<td>0.43/0.63</td>
<td>0.18/0.30</td>
<td>0.24/0.42</td>
<td>0.17/0.31</td>
<td>0.15/0.29</td>
</tr>
<tr>
<td>MemoNet [47]</td>
<td>0.40/0.61</td>
<td>0.11/0.17</td>
<td>0.24/0.43</td>
<td>0.18/0.32</td>
<td>0.14/0.24</td>
</tr>
<tr>
<td>SHENet [31]</td>
<td>0.41/0.61</td>
<td>0.13/0.20</td>
<td>0.25/0.43</td>
<td>0.21/0.32</td>
<td>0.15/0.26</td>
</tr>
<tr>
<td>LED [29]</td>
<td>0.39/0.58</td>
<td>0.11/0.17</td>
<td>0.26/0.43</td>
<td>0.18/0.26</td>
<td>0.13/0.22</td>
</tr>
<tr>
<td>GATraj [7]+PnS</td>
<td>0.26/0.40</td>
<td>0.10/0.15</td>
<td>0.21/0.37</td>
<td>0.16/0.28</td>
<td>0.11/0.20</td>
</tr>
</tbody>
</table>

Table 4. The comparison of Models on the nuScenes test set.

<table>
<thead>
<tr>
<th>Models</th>
<th>mADE5</th>
<th>mADE10</th>
</tr>
</thead>
<tbody>
<tr>
<td>LAformer+PnS</td>
<td>0.67</td>
<td>0.49</td>
</tr>
<tr>
<td>LAformer+PnS+HD</td>
<td>0.63</td>
<td>0.46</td>
</tr>
<tr>
<td>LAformer+PnS+HD+LBA</td>
<td>0.58</td>
<td>0.41</td>
</tr>
</tbody>
</table>

1We were unable to implement LBA with LAformer and directly compare it with LAformer+PnS+HD due to the broken link to the code of [56].
P2T+LBA+HD directly traces predecessors from concurrent neighboring agents within the same scene without extra input data in a more challenging yet practical setting.

<table>
<thead>
<tr>
<th>Model</th>
<th>mADE5</th>
<th>mADE10</th>
<th>mFDE5</th>
<th>mFDE10</th>
</tr>
</thead>
<tbody>
<tr>
<td>AgentFormer[51]</td>
<td>1.97</td>
<td>1.58</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SG-Net[43]</td>
<td>1.86</td>
<td>1.40</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>LAformer[27]+PnS</td>
<td>1.49</td>
<td>1.19</td>
<td>8.08</td>
<td>6.99</td>
</tr>
</tbody>
</table>

Table 4. The prediction comparison on the nuScenes [3] test set. The best values are highlighted in boldface.

4.3. Ablation study

To analyze the effectiveness of the proposed components in the PnS method, we conduct an ablation study using the train-val split provided by the nuScenes dataset.

Firstly, we ablate the predecessor tracing (PT) module to evaluate its efficacy in our proposed model, denoted as PnS-Baseline. In this setting, the prediction model can only condition future trajectories on the observed trajectories. As shown in Table 5, removing the PT module results in a significant drop in performance, about 5% in both mADE5 and mFDE5 for predicting five trajectories, and approximately 11% in mADE10 and 17% in mFDE10 for predicting ten trajectories.

<table>
<thead>
<tr>
<th>PT</th>
<th>mADE5</th>
<th>mADE10</th>
<th>mFDE5</th>
<th>mFDE10</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓</td>
<td>1.49</td>
<td>1.19</td>
<td>2.24</td>
<td>6.69</td>
</tr>
<tr>
<td>✗</td>
<td>1.57</td>
<td>3.28</td>
<td>1.32</td>
<td>2.63</td>
</tr>
</tbody>
</table>

Table 5. Ablation study on PnS.

Secondly, we analyze the number of predecessors aggregated via Eq. (6) for the predecessor tracing module, as shown in Table 6. With a single predecessor, the model performs slightly worse than with two predecessors because the successor can be influenced by other predecessors as well. However, increasing the number of potential predecessors to three leads to a clear performance drop, as measured by mFDE10. This is because the chances of including less influential or even irrelevant predecessors also increase.

<table>
<thead>
<tr>
<th>k</th>
<th>mADE10</th>
<th>mFDE10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.19</td>
<td>2.27</td>
</tr>
<tr>
<td>2</td>
<td>1.19</td>
<td>2.24</td>
</tr>
<tr>
<td>3</td>
<td>1.21</td>
<td>2.32</td>
</tr>
</tbody>
</table>

Table 6. The number of top predecessors.

Thirdly, we explore another distance metric in φ for predecessor identification. Namely, we substitute L2 by L1 in φ. As shown in Table 7, L1 does not work as well as L2. It increases, e.g. mFDE10, by about 3%.

Finally, we investigate the sensitivity of the weight parameter λ in the loss function presented in Eq. (10). We vary the value of λ while keeping other settings unchanged to analyze its effect on the prediction performance. We observe that decreasing or increasing this value to 0.1 or 1 leads to slightly worse performance in terms of mADE10 and mFDE10, as shown in Table 8.

<table>
<thead>
<tr>
<th>λ</th>
<th>mADE10</th>
<th>mFDE10</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>1.25</td>
<td>2.45</td>
</tr>
<tr>
<td>0.5</td>
<td>1.22</td>
<td>2.36</td>
</tr>
<tr>
<td>1</td>
<td>1.28</td>
<td>2.53</td>
</tr>
</tbody>
</table>

Table 7. Distance metric for predecessor identification.

Table 8. Sensitivity analysis of λ in the objective function.

4.4. Qualitative results

Fig. 3 illustrates the qualitative results obtained by LAformer, LAformer+PnS, and LAformer+HD. In the first row, a vehicle is seen traversing an eight-arm intersection. LAformer, without any scene cues, generates a right turn that is incompatible with lane connections, and drives through areas outside lane boundaries. In contrast, LAformer+PnS follows traces from predecessors and produces more accurate predictions. With the aid of lane centers, LAformer+HD accurately captures the vehicle’s driving intent. The differences among these three models become more apparent in the second row, as the vehicle drives along on the road. LAformer generates several predictions that are incompatible with the scene, while the predictions of LAformer+PnS and LAformer+HD are more compliant with the scene. We also observe that LAformer+PnS generates more divergent predictions in lateral directions in the third row, as it may follow the traces of predecessors turning into different directions in this traffic situation. Interestingly, in the fourth row, LAformer+PnS predicts a turning modality that is well overlapped with the ground truth trajectory, while LAformer fails to predict the vehicle’s intent, and the predictions of LAformer+HD have an offset from the ground truth trajectory. However, we also note that some modalities predicted by LAformer+PnS are not perfectly aligned with the road geometry due to the lack of lane information.

4.5. Discussion

To avoid false positive predecessor identification, for example, predecessors are too far away from the successor or the angles from the successor to the predecessors are too sharp to make a maneuver, we have tested setting a maximum distance threshold and angle difference. This ensures that the identified predecessor agent is spatially close
to the successor and driving in a feasible direction. However, as a limitation in some cases, the successor may not find any predecessor that meets the criteria, resulting in a lack of observed predecessor information. In such cases, the prediction rolls back to conditioning on the past trajectories with an empty predecessor. The empirical studies on the nuScenes dataset indicate that even with a few those cases, this did not lead to a significant performance difference (more details in supplementary material). Overall, the prediction model achieves evidently better performance by including the PnS with the predecessor tracing module. We leave exploration of more sophisticated solutions, such as knowledge distillation [56], to address the cases of no HD map and no predecessors for future work.

5. Conclusion

In this paper, we propose a probabilistic approach named Predecessor-and-Successor (PnS) to trace the influence of agents (predecessors) identified from the neighboring agents within the same scene on the target agent (successor). The moving patterns derived from the traced predecessors serve as informative priors to guide the prediction of the successor’s movement. We use a probabilistic predecessor tracing module to select several highly influential predecessors to account for the stochasticity of the successor’s future behavior. Compared to memory-based models, our method requires no extra effort to collect all the trajectory data beforehand. Our simple but effective PnS is integrated into a graph-based predictor for pedestrian trajectory prediction on the ETH/UCY datasets, which achieves a new state-of-the-art performance. Furthermore, we demonstrate the effectiveness of PnS by replacing the HD-map extraction module with PnS in a transformer-based predictor for vehicle trajectory prediction on the nuScenes dataset. Our method largely mitigates the performance degradation when the map information is removed. It also achieves comparable performance when it is compared with a Local Aware module that exploits the historical trajectories traversed in the same location.

Acknowledgments: This work is partially funded by MSCA European Postdoctoral Fellowships under the 101062870 - VeVuSafety project.
References


