# Smooth-Trajectron++

## Augmenting the Trajectron++ behaviour prediction model with smooth attention

MSc Thesis Robotics

Frederik Westerhout

Delft University of Technology

TU Delft

# Smooth-Trajectron++

## Augmenting the Trajectron++ behaviour prediction model with smooth attention

by

Frederik Westerhout

| Student Name | Student Number |
| --- | --- |
| F.S.B. Westerhout | 4293940 |


| | |
| --- | --- |
| Primary supervisor: | Arkady Zgonnikov |
| Daily supervisor: | Julian Schumann |
| Institution: | Delft University of Technology |
| Place: | Faculty of Cognitive Robotics, Delft |
| Project Duration: | August, 2022 - May, 2023 |

Cover Image: This image was created with the assistance of DALL·E 2 from https://labs.openai.com/

TU Delft

## Contents

## List of Figures

## List of Tables

# Smooth-Trajectron++: Augmenting the Trajectron++ behaviour prediction model with smooth attention

F.S.B. Westerhout
*Department of Cognitive Robotics*
*Delft University of Technology*
*Delft, Netherlands*

***Abstract*—Understanding traffic participants' behaviour is crucial for predicting their future trajectories, enabling autonomous vehicles to better assess the environment and consequently anticipate possible dangerous situations at an early stage. While the integration of cognitive processes and machine learning models has demonstrated promise in various domains, its application in trajectory forecasting of multiple traffic agents in large-scale autonomous driving datasets remains lacking. This work investigates the state-of-the-art trajectory forecasting model Trajectron++ which we enhance by incorporating a smoothing term in its attention module. This attention mechanism mimics human attention inspired by cognitive science research indicating limits to attention switching. We evaluate the performance of the resulting Smooth-Trajectron++ model and compare it to the original model on various benchmarks. Our results show improved performance on the large-scale nuScenes dataset, revealing the potential of incorporating insights from human cognition into trajectory prediction models.**

## 1. Introduction

In a world where the demand for intelligent vehicles is increasing rapidly [1], there is a growing concern about the safety of passengers and other road users. According to the World Health Organization, approximately 1.3 million people die each year due to road traffic accidents, and this number is expected to increase if proper measures are not taken [2]. Therefore, ensuring a safe environment for human-robot interaction in traffic is of paramount importance.

One of the most important factors for ensuring a safe environment around intelligent vehicles is accurately predicting the future movements of surrounding traffic participants [3]. These predictions help to better assess the environment and anticipate a dangerous situation at an early stage, which could prevent accidents. For those predictions, understanding how traffic agents interact with each other is essential for predicting their trajectories.

There are numerous methods that have been used to tackle the challenge of trajectory prediction [4, 3, 5]. They contain examples ranging from reason-based methods to data-driven techniques. Over the last few years, data-driven methods have shown great potential [6, 8, 9, 10, 11, 12, 13, 14, 15, 7]. These methods use machine learning algorithms to learn from large amounts of data to predict the trajectories of surrounding traffic participants. One of these data-driven models is *Trajectron++* (T++) [10], which stands out due to its public code availability, the general applicability and the results it achieved on multiple challenges [10, 16] containing datasets ETH [17], UCY [18], nuScenes [19] and highD [20].

Other methods to predict future trajectories are based on research from cognitive science. Instead of learning merely from data, the functioning of small parts of the human brain is taken as a starting point for constructing components of the model. An example of such a brain-based component is evidence accumulation, which is a cognitive process wherein information from multiple sources is gathered and integrated over time to form a decision or judgment. It is believed to be the primary mechanism that guides human decision-making [21]. The theory of evidence accumulation is already used to predict whether car drivers [22, 23] or pedestrians [24] are merging on a road with traffic. Another example of using insights from cognitive science for behaviour prediction in traffic is the use of a quantum-like Bayesian model. This is a mathematical framework that combines elements of quantum theory and Bayesian probability theory to describe decision-making and information processing in complex and uncertain environments [25]. It is used in [26] to more accurately predict human street crossing behaviour, compared to the more data-driven model *Social-LSTM* [6]. Yet another insight from cognitive science suggests that the brain has a limited capacity for shifting attention rapidly between different tasks [27]. This is used in [28], where the application of the smoothing term to the attention module of a machine-learning prediction model – referred to as *Smooth-Attention* – which mimics human cognition, allows for better predictive performance.

As seen in these examples, recent research has shown promising results in enhancing the performance of trajectory prediction models by incorporating principles from cognitive science. However, there is a need to further investigate and explore the potential of these cognitively inspired models in a more extensive manner. In particular, Cao et al. [28]

highlight the importance of integrating smooth attention mechanisms with advanced network architectures for modelling interactions to improve trajectory prediction.

In this study, our objective is to tackle this challenge by incorporating smooth attention into a state-of-the-art behaviour prediction model. Specifically, we seek to enhance the performance of the *Trajectron++ (T++)* [10] by leveraging the technique of smooth attention proposed in [28]. By imposing a smoothness constraint on the attention module, we significantly reduce abrupt changes in attention, thus enabling the module's output to closely resemble natural human cognitive processing. To denote the combined model resulting from this integration, we refer to it as *Smooth-Trajectron++*. To evaluate the efficacy of this novel model, we conduct experiments on multiple datasets, including *nuScenes* [19], *highD* [20], *rounD* [29] and *L-GAP* [22].

## 2. Methods

This chapter provides an overview that explains the various elements of the *Trajectron++* (T++) model (2.1) and the functioning of the smooth attention module (2.2). Additionally, the used dataset, nuScenes, is discussed (2.3) and a benchmarking framework, including three gap acceptance datasets, is introduced to validate the model's results (2.4). Finally, we list the used metrics for the evaluation of the data (2.5).

### 2.1. Trajectron++

We use *Trajectron++ (T++)* [10] as the baseline model, for several reasons. Firstly, the model showed state-of-the-art performance on various public datasets [10] while including an attention module. Secondly, the authors have made the source code publicly available, including proper documentation regarding its application. This offers the opportunity to potentially reproduce the originally reported results while minimizing deviations from the original setup used by the authors. Lastly, the model has been tested on a large-scale public autonomous driving dataset, *nuScenes* [19], described in Section 2.3. This provides evidence of the applicability of the model to real-world scenarios concerning interactions between multiple traffic participants.

*T++* is based on the Trajectron model [30] and consists of a combination of different elements, including conditional variational auto-encoders (CVAEs) [31], long short-term memory networks (LSTMs) [32] and graph neural networks (GNNs) [33]. For these components we provide an explanation in the coming paragraphs.

**CVAE** The backbone of the model is a CVAE, proposed by Sohn et al. in [31]. This model uses the concept of a Variational Autoencoder (VAE) [34], which is a generative model that consists of an encoder and a decoder. The main goal of a VAE is to learn a compressed, continuous representation of input data in an unsupervised manner to generate new data samples that resemble the training data. Specifically, VAEs aim to encode input data into a low-dimensional latent space using the autoencoder (encoder), where each dimension of the latent space corresponds to a different feature of the input data. This compressed representation can then be decoded back into the original input data (decoder). The main difference between a CVAE and a VAE is that a CVAE can learn to generate data samples conditioned on a given set of attributes or labels (like cars and pedestrians), whereas a VAE can only generate samples from the learned latent space distribution, i.e., without any control over the data generated by the model Regarding the case of trajectory forecasting, control over the generated output is necessary to match the prediction with a specific class. This means that by using a CVAE it is possible to differentiate clearly between future behaviour of a pedestrian on the one hand, and a car on the other hand.

**LSTM** LSTM networks are a type of recurrent neural network (RNN) [35] architecture capable of learning long-term dependencies in sequential data. In abstract words: the key feature of LSTMs is the incorporation of memory cells, which enable the network to selectively store, modify, and retrieve information over extended periods of time. This is achieved through a series of gates, including the input gate, output gate, and forget gate, that regulate the flow of information into and out of the memory cells. The LSTM architecture has demonstrated significant success in a variety of applications, such as natural language processing, speech recognition, and time series prediction.

**GNN** GNNs are a class of deep learning models that can operate on graph-structured data to learn node and edge representations that capture structural properties of the graph [36]. GNNs have gained increasing popularity in recent years due to their ability to effectively model complex relationships in a variety of domains, including social networks, bioinformatics, and computer vision. In *T++* a specific type of GNNs is used, called directed spatiotemporal graphs. Spatiotemporal refers to the combination of both spatial (related to space or location) and temporal (related to time) dimensions. It refers to how things change and move over time and in different places or locations. Directed spatiotemporal graphs are a type of graph data structure that capture the temporal and spatial relationships between entities over time. By incorporating a directed spatiotemporal graph as input to a GNN, the model can learn to effectively capture the spatiotemporal dependencies between entities, and exploit the temporal evolution of the graph to predict future states of the system. Specifically, the GNN can be used to learn node and edge representations that capture the temporal and spatial dependencies between nodes and make predictions based on these learned representations. An example of a graph-like situation can be seen in Figure 1, wherein the nodes are shown as the outer pedestrians and cars ($ped_1, ped_2, car_1, car_2$) and the edges as the arrows pointing towards the target pedestrian ($ped_{tar}$). The target pedestrian is the agent in the scene whose future locations are being predicted.

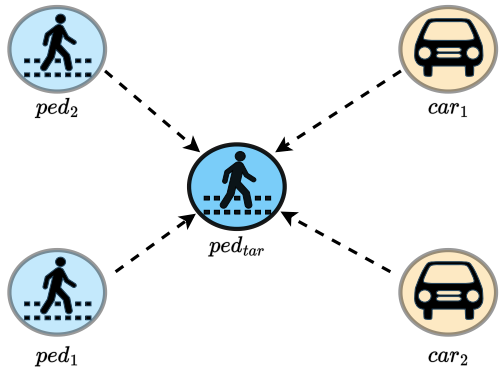


Figure 1. Example of graph-like representation of traffic scene containing two pedestrians ($ped_1$,$ped_2$) with two cars ($car_1$,$car_2$) influencing the pedestrian being predicted ($ped_{tar}$)

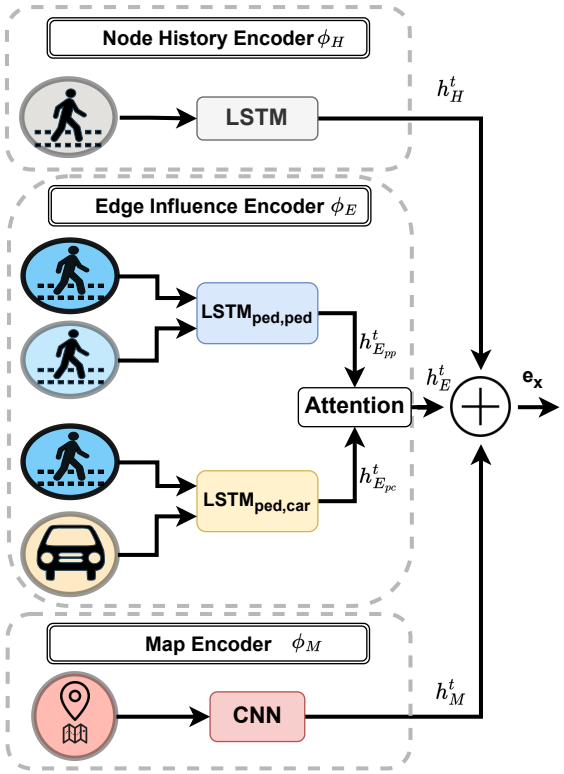


Figure 2. Encoder part of CVAE at $x^t$ with representation vector $e_x$

**2.1.1. Encoder Architecture.** The graph-structured model consists of various modules that each represent a different influence on the trajectory forecast, see Figure 2.

First, the past location and speed of the chosen traffic agent with multiple input time steps $(x_1^0, x_1^1, ...)$ are fed into the Node History Encoder $\phi_H$, whose main component is a long-short-term memory cell (LSTM). This ensures that the past positional data is used for future predictions. The output of the LSTM is the hidden state $h_H^t$. Secondly, the *T++* model makes use of road map data to make its predictions more feasible and realistic. The Map Encoder module $\phi_M$ takes relevant environmental information, which is fed to a convolutional neural network (CNN), which has the hidden state $h_M^t$ as output. Thirdly, the Edge Influence Encoder $\phi_E$ is used for taking into account to what extent other traffic agents in the scene are influencing the prediction. This module contributes to the "social awareness", which means that the behaviour of the agent gets influenced by other traffic participants. For example, when a car is planning to cross an intersection, but the driver sees a pedestrian that is about to cross the road, the car needs to change its future plan to avoid a potential collision.

More specifically, the Edge Influence Encoder follows a graph-based approach to model the influence of neighbouring agents on a target agent (in this case $ped_{tar}$) from $x_{t-T}$ to $x_t$, with $T$ being the maximum of past input time steps. To encode graph edges, the method follows a two-step process. Firstly, edge information is collected from neighbouring agents belonging to the same semantic class. This can be seen in Figure 2, where the pedestrian-pedestrian and pedestrian-car semantic classes are collected independently. A summation operation is used for feature aggregation, as opposed to concatenation or averaging, to handle varying numbers of neighbouring nodes while preserving count information, following the method performed by Jain et al. in [37]. Subsequently, the aggregated edge information is input to an LSTM for all edge instances of the same type. The encodings of all the connections between the modelled node and its neighbouring nodes are combined to create an "influence" representation vector, which represents the overall impact of the neighbouring nodes. In Figure 2, this is visible as ${h_{Epp}}^t$ for the pedestrian-pedestrian and ${h_{Epc}}^t$ for the pedestrian-car interactions. An additive attention module is then used for obtaining a single vector representation of the different edge influence encodings $h_E^t$ [38]. The final step involves concatenating the node's history, the map encoder and the edge influences to produce a unified node representation vector $e_x$.

## 2.2. Smooth Attention

The *smooth attention* approach [28] presents a novel perspective on attention modules. Unlike traditional methods, it applies attention at each time step, following [12]. By emulating human attention during deliberate tasks, it incorporates a smoothness constraint based on the hypothesis that attention does not frequently change over time. Previous research [27] shows that deliberate attention movements are slower due to internal limitations. This implies that attention does not frequently fluctuate during driving, as it falls under intentional movement. By incorporating the smoothness constraint, the *smooth attention* approach enhances the attention mechanism, improving the selection of important information while disregarding less relevant input variables and aligning better with the characteristics of human attention.

## 2.3. nuScenes

The nuScenes dataset is developed for accelerating research in the area of autonomous driving [19]. This dataset consists of 1000 driving scenes in two major cities, Boston and Singapore, characterized by their high traffic volumes and challenging driving situations. The data is captured from a driving car, equipped with multiple sensors. Its multi-sensor capabilities make it remarkable; in addition to cameras, it is equipped with LIDAR, RADAR, GPS, and IMU sensors. The driving scenes are each 20 seconds in length and are annotated at 2 Hz. Next to 23 annotated object classes with 3D bounding boxes, mostly pedestrians and cars, it also includes semantic map information like crosswalks and sidewalks.

## 2.4. Benchmarking Framework

To prevent evaluating the trajectory prediction model only on the nuScenes dataset, we make use of a novel benchmarking framework proposed by Schumann et al. in [16]. Initially, this framework is meant for benchmarking behaviour models in gap acceptance scenarios on three publicly available datasets: highD [20], rounD [29] and L-GAP [22], but it can also be used for trajectory prediction models. This makes it possible to make a fair comparison between different models, wherein multiple metrics are taken into consideration. Also, there is a special focus on safety-critical scenarios in the benchmarking framework, which are especially important when developing safe and reliable prediction models.

## 2.5. Metrics

In order to ensure equitable comparisons between datasets, the use of multiple metrics is imperative. In the present study, we employ a diverse set of evaluation criteria to assess the datasets under investigation. Specifically, for the nuScenes dataset [19], we employ the metrics of Final Displacement Error (FDE), Average Displacement Error (ADE), and Kernel Density Estimation Negative Log-Likelihood (KDE-NLL). Furthermore, for the datasets associated with the benchmarking framework [16], next to FDE and ADE, we use Area Under the Curve (AUC), and True Negative Rate Precision-Recall (TNR-PR). Whereas the FDE and ADE are used as trajectory metrics, the AUC and TNR-PR are used as classification metrics. For these metrics, a short explanation is given below.

**FDE**: The Final Displacement Error (FDE) is a metric used to evaluate the performance of trajectory prediction models. It measures the Euclidean distance between the predicted final location of an object and its true final location. In other words, FDE indicates how far off the model's predicted location is from the actual location at the end of a predicted trajectory.

**ADE**: The Average Displacement Error (ADE) is another metric used to evaluate the performance of trajectory prediction models. Unlike FDE, which measures the error of the final predicted location, ADE measures the average Euclidean distance between the predicted location of an object at each time step and its true location. ADE is particularly useful for evaluating the overall accuracy of a model's trajectory predictions, as it takes into account the entire predicted trajectory rather than just the final location.

**KDE-NLL**: Kernel Density Estimation of Negative Log Likelihood (KDE-NLL) is a measure of the model's ability to accurately estimate the probability distribution of the true trajectory given the observed data. A lower negative log-likelihood score indicates a better fit between the predicted and true trajectories. In other words; lower is better, which is considered to be the most optimal and representative of the underlying distribution. KDE-NLL is a useful metric for evaluating the uncertainty of a model's predictions, as it provides a measure of how well the model is able to capture the true distribution of the data.

**AUC**: Area under Curve (AUC) is a metric used to evaluate the performance of binary classification models. AUC measures the overall performance of a model in distinguishing between positive and negative classes by calculating the area under the receiver operating characteristic (ROC) curve.

**TNR-PR**: True Negative Rate - Positive Rate (TNR-PR) is a useful metric for evaluating the performance of binary classification models when the dataset is imbalanced and the cost of false positives and false negatives is not equal. A high TNR-PR indicates that the model is performing well in correctly identifying negative instances while also identifying positive instances with high accuracy.

# 3. Smooth-Trajectron++

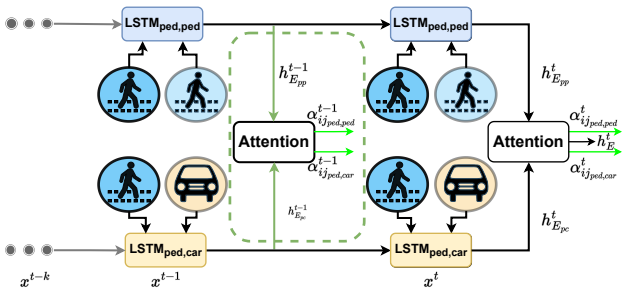


Figure 3. Edge Influence Encoder including Smooth-Attention (in green)

In this chapter, we propose a way to apply the smooth attention module [28] specifically to the *Trajectron++* model. To do this, we alter the Edge Influence Encoder module of *Trajectron++*, see Figure 2, as this is the part

where the social interactions are modelled, and the attention is applied.

Our approach[1], which we call *Smooth-Trajectron++*, is illustrated in Figure 3. At a high level, the original Edge Influence Encoder is expanded by applying the attention module at each time step in a similar fashion as in the *smooth attention* model (the green highlighted box in Figure 3). Here, the outputs $\alpha^{\tau}_{i,j_{ab}}$ are the attention weights that are used to rank the importance the human agent $i$ assigns to the semantic class $j_{ab}$ for neighbouring agents of types $a$ and $b$ ($a$ and $b$ can stand for agent types such as cars or pedestrians) at the time $\tau$. To represent the number of timesteps and agents for which we predict the trajectory, we use $T$ and $N$ respectively. All these attention weights from every time step are then used as an input for the added smoothing term in the loss function to incorporate the regularising of the attention by imposing a vectorial total variation penalty $\mathcal{L}_{\text{smooth}}$ :

$$\mathcal{L}_{\text{smooth}}(\boldsymbol{\alpha}) = \sum_{\tau=t-T+1}^{t} \sum_{i=1}^{N} \sqrt{\sum_{j} \left(\alpha_{ij}^{t} - \alpha_{ij}^{t-1}\right)^2}. \quad (1)$$

To ensure that the attention weights are utilised during the model training process, we incorporate $\mathcal{L}_{\text{smooth}}$ into the original loss function $\mathcal{L}_0$ [10]:

$$\mathcal{L}_{new} = \mathcal{L}_0 + \beta \mathcal{L}_{smooth}. \quad (2)$$

The scaling factor $\beta$ is introduced to fine-tune the influence of $\mathcal{L}_{smooth}$. By adjusting $\beta$, the model can be trained to effectively balance the contribution of the attention weights with the original loss function.

The extra loss term $\mathcal{L}_{smooth}$ and associated additional calls to the attention module increase the number of computations and therefore have an effect on the training time, which is approximately 1.5 times slower compared to the original version of *Trajectron++*.

# 4. Results

We evaluate our method on four publicly available datasets, as introduced in subsection 2.3-2.4: nuScenes [19], highD [20], rounD [29] and L-GAP [22]. Firstly, the original *Trajectron++* model is trained and evaluated on the nuScenes dataset. Secondly, we trained the expanded model and evaluated it on the nuScenes data, with $\beta$-values ranging from $0.01$ to $10$. The steps between these scaling factors ($\beta$) are logarithmic to study a wide range for a possible optimal solution. Lastly, *Smooth-Trajectron++* is implemented in the benchmarking framework proposed by Schumann et al. [16] on the remaining datasets highD, rounD and L-GAP. As this benchmarking framework focuses on gap acceptance, only subsets of the data including gap acceptance scenarios are used. The experiments were performed on DelftBlue, the high-performance computer from the Technical University of Delft [39].

1. The source code is available at GitHub/Smooth-Trajectron++

## 4.1. nuScenes

Both the original Trajectron++ (subsection 2.1) and the Smooth-Trajectron++ model (section 3) were trained and evaluated on three metrics: Final Displacement Error (FDE), Average Displacement Error (ADE) and Kernel Density Estimation of Negative Log Likelihood (KDE-NLL), as explained in subsection 2.5. These metrics are chosen as they were used in the original paper [10], therefore needed for a fair comparison. Using these metrics, the results of the reported values mentioned in the *Trajectron++* paper are compared to the reproduced and the various *Smooth-Trajectron++* versions. For the training of the model, we used the same parameters as mentioned in [10], e.g. 12 training epochs on the nuScenes dataset.

**4.1.1. Reproduced vs. reported results.** Looking at Table 1-Table 3 and comparing the "*T++*" rows with the reproduced rows *(T++(rep.))*, the FDE at the prediction horizon of 1 second (@1s) is at most more than ten times smaller in the reported values than in the reproduced results. At later prediction horizons, the relative difference of the errors becomes smaller, but the difference is still significant. For the pedestrian-only predictions, the difference is bigger than for the vehicle-only predictions. The ADE follows a similar pattern; at the smallest prediction horizon and in the pedestrian-only case, the difference is the greatest. Also when comparing the KDE-NLL values, the reproduced values do not come close to the ones reported in the paper [10]. The discrepancy between these results and our achieved results is significant but does not entail the focus of our study. Therefore, we proceed with comparing the performance of the modified models to our own achieved results, which serves as the new benchmark.

**4.1.2. Pedestrian-only predictions.** There are two main prediction classes in the nuScenes dataset, pedestrians and vehicles. As their behaviour is significantly different, the model predicts these classes separately. In this subsection, the pedestrian-only results are shown of the *Trajectron++* and the *Smooth-Trajectron++*.

In Table 1-Table 3, the results are shown for the predicted pedestrian trajectories. The pedestrian-only results are visible on the left side of the slash in the columns. The numbers in bold represent the lowest score per prediction horizon, compared to the "T++(rep.)" row, which serves as a reference for comparative analysis and is equivalent to a $\beta$-value of 0. The FDE and the ADE outputs consist of the most likely single trajectory prediction within the distribution of trajectory predictions, using the "Most Likely" output configuration as in [10]. The *Smooth-Trajectron++* versions, depicted as "Smooth" in the table, consistently outperform the reproduced version of *Trajectron++*. Tuning the scaling factor $\beta$ does have an effect on the error. Regarding the FDE,

the parameter $\beta = 0.1$ has the lowest error in all cases, except for the shared lowest error at the first prediction horizon (@1s) of $\beta = 1.0$. Looking at the ADE, either $\beta = 0.1$ or $\beta = 1.0$ both have the lowest or equally lowest error. It seems that the higher the $\beta$-value, the more it resembles the "Reproduced" reference values. In Table 3 the opposite seems to be happening; the "Reproduced" row shows the lowest values for almost all cases. An exception is the smooth version with $\beta = 0.01$ at the prediction horizon of 4 seconds (@4s), where only a marginal performance increase is seen. However, it can be said that in general in this pedestrian-only case, the Smooth Attention does not improve this metric, although the decline for the smooth versions is minimal.

TABLE 1. RESULTS *nuScenes* T++ PEDESTRIAN-ONLY/VEHICLE-ONLY: FDE (M)

| *Model* | @1s | @2s | @3s | @4s |
|---|---|---|---|---|
| T++ [10] | 0.014/0.07 | 0.17/0.45 | 0.37/1.14 | 0.62/2.20 |
| T++ (rep.) | 0.168/0.430 | 0.369/1.168 | 0.608/2.323 | 0.886/3.868 |
| $\beta = 0.01$ | 0.157/**0.413** | 0.353/1.102 | 0.586/2.141 | 0.855/3.546 |
| $\beta = 0.1$ | **0.155**/0.419 | **0.350/1.081** | **0.580/2.122** | **0.842/3.496** |
| $\beta = 0.5$ | 0.159/0.421 | 0.354/1.123 | 0.588/2.181 | 0.857/3.560 |
| $\beta = 1.0$ | **0.155**/0.448 | 0.351/1.128 | 0.582/2.165 | 0.845/3.507 |
| $\beta = 10$ | 0.160/0.425 | 0.366/1.149 | 0.607/2.190 | 0.876/3.539 |

TABLE 2. RESULTS *nuScenes* T++ PEDESTRIAN-ONLY/VEHICLE-ONLY: ADE (M)

| *Model* | @1s | @2s | @3s | @4s |
|---|---|---|---|---|
| T++ [10] | 0.021/- | 0.073/- | 0.15/- | 0.25/- |
| T++ (rep) | 0.126/0.307 | 0.221/0.632 | 0.329/1.092 | 0.450/1.689 |
| $\beta = 0.01$ | 0.116/**0.296** | 0.208/0.602 | 0.314/1.021 | 0.432/1.559 |
| $\beta = 0.1$ | **0.114/0.302** | **0.206/0.597** | **0.311/1.012** | **0.427/0.543** |
| $\beta = 0.5$ | 0.118/0.301 | 0.210/0.613 | 0.316/1.041 | 0.434/1.580 |
| $\beta = 1.0$ | **0.114**/0.319 | 0.207/0.630 | 0.312/1.048 | 0.428/1.575 |
| $\beta = 10$ | 0.118/0.303 | 0.215/0.628 | 0.325/1.055 | 0.445/1.586 |

TABLE 3. RESULTS *nuScenes* T++ PEDESTRIAN-ONLY/VEHICLE-ONLY: KDE NLL

| *Model* | @1s | @2s | @3s | @4s |
|---|---|---|---|---|
| T++ [10] | -5.58/-4.17 | -3.96/-2.74 | -2.77/-1.62 | -1.89/-0.71 |
| T++ (rep) | **-2.575**/-1.760 | **-1.530**/-0.604 | **-0.797**/0.235 | -0.230/0.927 |
| $\beta = 0.01$ | -2.560/-1.861 | -1.519/**-0.726** | -0.795/**0.108** | **-0.240/0.801** |
| $\beta = 0.1$ | -2.541/-1.856 | -1.502/-0.679 | -0.776/0.176 | -0.216/0.875 |
| $\beta = 0.5$ | -2.542/**-1.885** | -1.505/-0.690 | -0.779/0.150 | -0.211/0.818 |
| $\beta = 1.0$ | -2.549/-1.880 | -1.507/-0.679 | -0.785/0.173 | -0.226/0.868 |
| $\beta = 10$ | -2.480/-1.861 | -1.471/-0.661 | -0.759/0.173 | -0.205/0.845 |

**4.1.3. Vehicle-only predictions.** In this subsection, the evaluations on the vehicle-only predictions between the various models are shown.

Rightmost numbers in the columns of Table 1 - Table 3 show the outcomes of the vehicle-only predicted trajectories. Similarly to the previous pedestrian-only case, again a general FDE and ADE decline is seen along the $\beta$-versions of the *Smooth-Trajectron++*. The ADE values of the *Trajectron++* paper are missing in Table 2, as they are not reported by the authors in the original paper. The version with $\beta = 0.01$ holds the lowest value for the prediction horizon of 1 second (@1s), while $\beta = 0.1$ has the smallest error for the remaining prediction horizons. In contrary to the pedestrian-only predictions in Table 3, the *Smooth-Trajectron++* on the vehicle-only forecasts has better KDE-NLL numbers than the reproduced model, which indicates that the model is better able to match the original distribution of predicted trajectories with the inclusion of the Smooth-term in the loss function. Furthermore, this can be said for all $\beta$-factors, while in this case the $\beta = 0.01$ has the most optimal values.

## 4.2. Benchmark for Gap Acceptance

For the implementation of this second section of the results, we follow the training and evaluation procedure of [16]. Instead of studying the full range of scenarios as we did in the previous section, now we focus on gap-acceptance scenarios. Gap-acceptance scenarios are situations where drivers decide whether to enter or wait for a gap in traffic, such as when a car approaches an intersection and decides whether to turn left immediately or wait for a break in oncoming traffic. The datasets that are included in the benchmark contain information about lane changes (highD), roundabouts (rounD) and left turns (L-GAP). In each of these datasets, the gap-acceptance scenarios are divided into two splitting methods; the random split and the critical split. The first method randomly splits the data for testing, while the second method deliberately selects the most unusual behaviour of the target vehicle for testing, such as accepting a very small gap or rejecting a very large gap. Extrapolation is required for testing the model's performance on situations that lie outside the training distribution, making it more challenging to consider these less intuitive samples [40]. Furthermore, a varying amount of input time steps is used in the benchmark. In the coming results, we use two input time steps to study the input-dependability of the model: $n_I = 2$ and $n_I = 10$.

The benchmarking framework includes several metrics next to the FDE and ADE that we have seen in Section 2.3. In addition to those, the results are evaluated on Area Under Curve (AUC) and True Negative Rate under Perfect Recall (TNR-PR).

In Figure 4 and Figure 5 an overview can be seen of the results of the various models on the benchmark. In

the coming subsections, this figure is analysed per column (dataset) and row (metric).

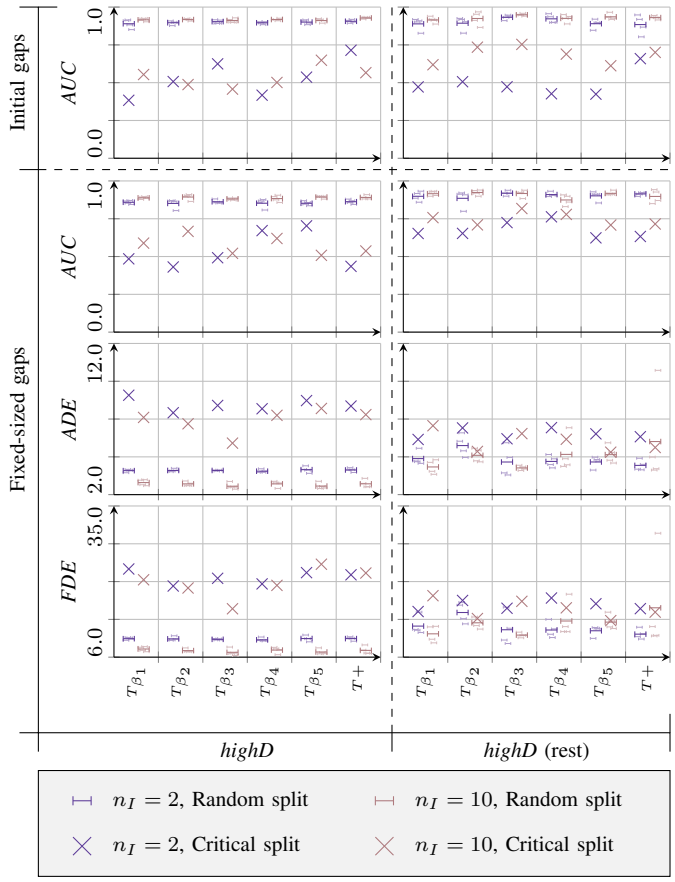


Figure 4. The performance of *Smooth-Trajectron++* and *T++* on the *highD* and *highD(rest)* dataset: $\beta_{1-5}$ refer to the $\beta$-values of 0.01, 0.1, 0.5, 1.0 and 10 respectively

**4.2.1. highD.** First, we analyse the performance of the various models at initial gaps. In the context of gap acceptance, an initial gap refers to the space or distance between two vehicles that a driver must perceive and judge in order to determine whether they can safely enter or cross a traffic stream. It represents the initial opportunity for a driver to make a decision about whether there is sufficient time and space to merge into or cross the traffic flow. This situation requires a binary decision, so this is evaluated using the metric Area under Curve (AUC). Looking at the random splits first, the values seem very similar between both the $\beta$-versions and the base model T++ ($T+$). However, when studying the data of Table 4-Table 5 in section A, one can see that the non-smooth version ($T+$) has slightly higher AUC values. Contrary to the random split, the critical split deviates a lot more between the $\beta$-versions compared to the $T+$-version. Again, the original $T+$-model has the highest value for $n_I = 2$, but for $n_I = 10$ the $\beta = 10$ version outperforms the original model.

Secondly, predictions at fixed-sized gaps are examined. A fixed-sized gap refers to predictions regarding gap acceptance scenarios when the provided gap has a consistent duration, it ensures that each prediction poses an equal level of difficulty by offering a comparable prediction horizon [16]. In a similar fashion as in the initial gap scenario, the AUC values for the random split are again close together. By a small margin, *T++* ($T+$) has the highest value for $n_I = 2$ and $\beta = 0.1$ has the highest value for $n_I = 10$. However, when taking the standard deviation into account these differences cannot be called sufficiently significant. At the critical splits, we see more variation between the model versions. In this case, almost every $\beta$-version is outperforming the $T+$ for both the $n_I = 2$ and $n_I = 10$, except for $\beta = 0.1$ at the $n_I = 2$. This could indicate that an added smoothing term has a positive effect on these critical-split cases. Nevertheless, these results show a wide-varying nature and also entail non-logical results. For example, the values for $n_I = 10$ are incidentally lower than for $n_I = 2$, which is not expected as more input time steps should give the model more information to fine-tune its predictions. This issue is addressed in Section 4.2.4. Furthermore, for both the ADE and the FDE metric, on the random split, $\beta$-values of $0.1, 0.5$ and $1.0$ outperform the $T+$-version by a small amount on both input time-steps. Regarding the critical split, the $n_I = 10$ is generally lower at the different $\beta$-values than the $T+$-version. Yet, there is not a clear trend visible that would imply that some $\beta$-value is consistently outperforming the base model.

Lastly, the critical gaps are assessed, which involves making predictions in scenarios where it is still meaningful to accept a gap at the last available moment in time, as defined in [16]. However, for this dataset, there is no data regarding critical gaps. This will be shown for rounD and L-GAP in section 4.2.3.

**4.2.2. highD(rest).** In the highD(rest) column of Figure 4, the restricted (rest) version of the full highD dataset is used for evaluation, which only includes samples for which it is known that a vehicle considered a lane-change. This is done to decrease the existing bias in the full dataset towards trajectories without an intended lane change, which makes the dataset more balanced [16].

**Initial gaps**: only in the case of $\beta = 0.5$ there is a notable increase in AUC for the random split compared to $T+$ in both $n_I$-instances. For the critical split, almost all AUC-values are lower than $T+$, except for $\beta = 0.1$ and $\beta = 0.5$ at $n_I = 10$ where it is slightly higher. Generally, the $\beta$-term does not seem to increase the performance of the base model.

**Fixed-sized gaps**: looking at the random splits, again for $\beta = 0.5$ there is an increase in AUC for both $n_I$-situations. The changes for the other $\beta$-versions are not consistently significant when compared to $T+$, having minor fluctuations to perform slightly better or slightly worse. Concerning the

critical split, all $\beta$-versions but $\beta = 10$ perform better than the base model, where $\beta = 10$ performs very similarly to $T+$. Also, the difference between the $n_I = 2$ and $n_I = 10$ is more logical this time, as the latter consistently has a higher AUC than the first. For both the FDE and ADE, all the $\beta$-versions are outperforming the $T+$-model at $n_I = 10$ on the random split. However, this seems due to one extremely high value of one of the random splits of $T+$ (each random split consists of three sub-splits, which are averaged to minimize the effect of randomness). This could be an outlier, caused by some error in the training process. At $n_I = 2$, the $\beta$-values under-perform compared to $T+$ for the random split, indicating no significant improvement. At the critical split, only at $\beta = 0.1$ and $\beta = 10$ both ADE and FDE values are lower at $n_I = 10$. In general, there is no clear improvement regarding these metrics across the various $\beta$-values. Furthermore, when looking at the random splits in general, again some non-logical behaviour is happening concerning the lack of improvement regarding input time steps. For example at $\beta = 1.0$, an increase in both ADE and FDE occurs, where the expectation would be that the value lowers as more input time steps are offered. A potential solution to this issue is proposed in Section 4.2.4.

**4.2.3. rounD and L-GAP.** At both the rounD and L-GAP dataset columns in Figure 5, we notice that in general, the values do not change with regard to every metric. Only at the critical splits at $n_I = 10$, there are some minor differences at FDE and ADE, but this cannot be classified as a significant improvement, as the difference is marginal and not consistent between the different splits and input time steps. This means that the addition of a $\beta$-term to the loss function does not have any measurable effect on the performance of the model when compared to the original *Trajectron++* model.

**4.2.4. Modification of loss function.** To address the issue regarding the decrease in performance after more input time steps at several instances, as we observed in Figure 4, we propose a modification in the loss function of Equation 1:

$$\mathcal{L}_{\text{smooth,m}}(\boldsymbol{\alpha}) = 8 \cdot \frac{1}{T} \sum_{\tau=t-T+1}^{t} \sum_{i=1}^{N} \sqrt{\sum_{j} \left(\alpha_{ij}^{t} - \alpha_{ij}^{t-1}\right)^2}. \quad (3)$$

Our hypothesis is that adding a mean (1/T) to the loss function decreases a potential accumulation of errors, which seems to be happening sometimes while increasing the number of input time steps. This added mean is multiplied by 8, as this corresponds to the length of the original sum so that the $\beta$ values keep the same impact between different loss versions (Equation 1 and Equation 3).

The performance of this modified loss function for $\beta = 1.0$ on the *highD* and *highD(rest)* dataset is visualised in Figure 6. We chose this $\beta$ value as unintuitive behaviour was observed both on the critical split on *highD* as on the random split on *highD(rest*. Looking at initial gaps in Figure 6, the

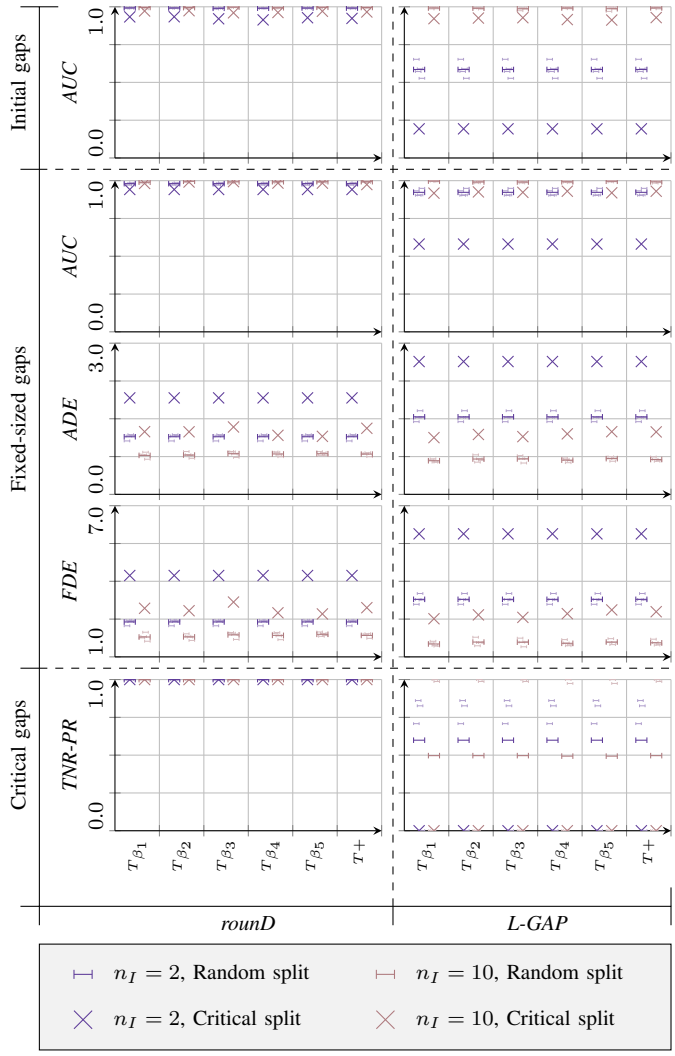


Figure 5. The performance of *Smooth-Trajectron++* and *T++* on the *rounD* and *L-GAP* dataset: $\beta_{1-5}$ refer to the $\beta$-values of 0.01, 0.1, 0.5, 1.0 and 10 respectively

random split is similar and the critical split has become less logical for *highD*, as the AUC value decreases going from $n_2$ to $n_10$. However, when studying both the random split and critical split at *highD* and *highD(rest)* for fixed-sized gaps, all results are as expected when transitioning from $n_2$ to $n_{10}$: AUC becomes higher and FDE and ADE lower. Furthermore, most values are similar when comparing the original loss function ($T_s$) with the modified loss function ($T_m$), but the critical split differs quite significantly on several occasions. Especially at the FDE and ADE at fixed-sized gaps, but also regarding the AUC at initial gaps.

## 5. Discussion

The first thing that stands out from the results, is the significant difference between the reported performance values in the original *Trajectron++* paper and the

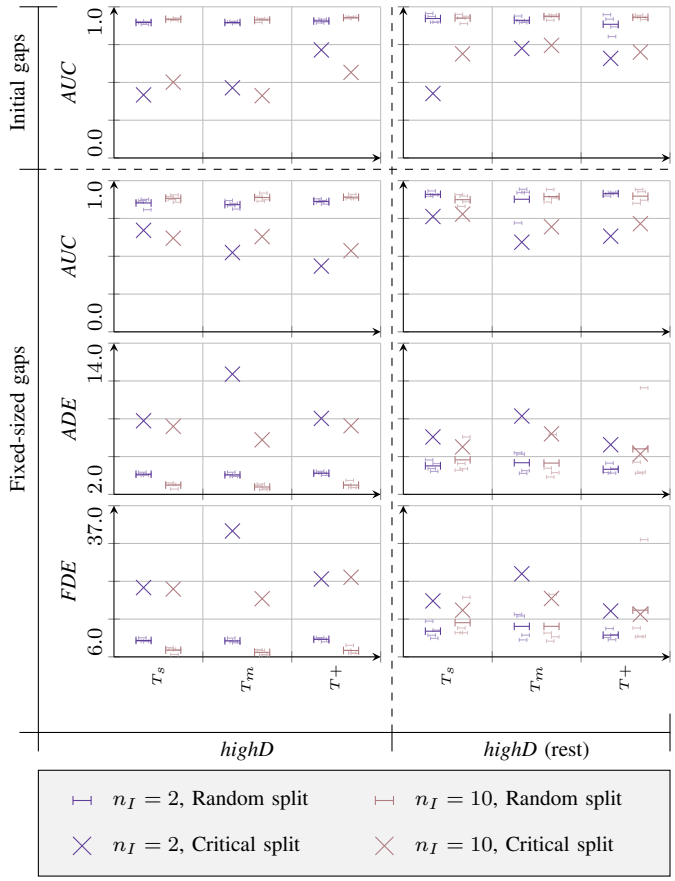


Figure 6. The performance of *Smooth-Trajectron++* and *Trajectron++* (T+) on the *highD* and *highD(rest)* dataset with original loss function from Equation 1 ($T_s$) and modified loss function from Equation 3 ($T_m$) for $\beta = 1.0$

reproduced results of this paper. There could be many reasons for this huge offset in the evaluation metrics. First, the nuScenes dataset could have been updated or changed in the meantime, which would mean other input data and therefore cause a different output. Secondly, the authors might have used hyperparameters and model settings for the results of their paper that are not updated on their GitHub page or in the text of the online version of the paper. In this case, it is impossible for an external researcher to replicate the results from the paper. Another possibility is that something in the paper reproduction process has gone wrong, either in downloading the data, installing required packages or a different random seed from our computer when training the model. Although the process was handled conscientiously, following the steps provided by the authors, an error could have occurred somewhere in this process. Due to the multitude of potential causes, and this not being the main focus of our study, we decided not to investigate this any further. However, it could be interesting for future research to reproduce the *Trajectron++* paper on the nuScenes dataset and compare their results both to those reported by the original paper and to the ones in this work.

Another interesting finding from the evaluation of the nuScenes dataset is that the *Smooth-Trajectron++* consistently outperforms the original *Trajectron++* regarding the FDE and ADE. The scores on these two metrics are lower, regardless of the $\beta$-values that are applied to the loss function. The scaling factor $\beta = 0.1$ has for almost every case the lowest value, or shared lowest value. The only deviating case is the FDE at the vehicle-only prediction, there the $\beta = 0.01$ has a slightly better score. This implies that the addition of a smoothing term to the loss function actually decreases the Final Displacement and Average Displacement errors both for the pedestrian-only and the vehicle-only predictions. When looking at the remaining metric, KDE-NLL, the same holds for the vehicle-only prediction case. In Table 3, it is evident that for all of the $\beta$-values the KDE-NLL is lower than the reproduced version. This indicates that the smooth model produces more accurate and reliable predictions with respect to the ground truth trajectories. However, when looking at Table 3 there is not much difference between the Smooth-versions ($\beta$) and the "Reproduced" row (T++(rep.)). In fact, the KDE-NLL scores for the *Smooth-Trajectron++* model are slightly worse than those of the original model, indicating a slight decrease in performance. This means that the distribution of the predicted trajectories for pedestrians is slightly less similar than the ground truth distribution. One explanation for this could be that the future behaviour of pedestrians is more difficult to predict than future vehicle trajectories. The smoothing term might decrease the variety of the distribution predictions in such a way, that the average of the predicted trajectories is eventually affected and therefore less similar to the ground truth trajectories. It would be interesting to perform more research to determine the generalizability of this hypothesis to other scenarios that involve pedestrian-only predictions.

In contrast to the perceived results from the nuScenes dataset, the performed experiments on the benchmark offer a less straightforward answer to the question of whether the general model's performance increases when adding a $\beta$ term to the loss function. When referring back to Figure 4 and focusing on $\beta = 0.1$ for example ($\beta_2$ in the figure), as this was mostly the optimal value in the nuScenes dataset for FDE and ADE, sometimes there is a performance increase and at other instances a performance decrease. One of the causes of this varying behaviour could originate from the used datasets in the benchmark. While the nuScenes dataset consists of a large-scale comprehensive sensor data collection with mostly cars and pedestrians, the datasets in the benchmark have a focus on only cars (highD), mostly cars with only 0.2% pedestrians (rounD) and again only cars in a driving simulator (L-GAP). As the smoothing term $\beta$ in *Smooth-Trajectron++* is applied to the attention module comparing different classes of traffic participants, it might not work as intended as there

is predominantly one major class in the dataset. Especially, when there is only 1 other traffic participant in the scene, it has nothing to be compared with, which results in the same output as the base model. This is exactly what happens in the L-GAP dataset column of Figure 5, for L-GAP is a dataset consisting of only two agents. Moreover, the exact same results on the L-GAP dataset between all and the base model show that the implementation of the Smooth-Attention is valid, as we would expect these results. Considering these results, one could imply that the *Smooth-Trajectron++* does not consistently outperform the base model ($T+$). However, we argue that the overall performance is mostly similar, sometimes better, and therefore not detrimental to the general functioning of such a model. Also, further research using *Smooth-Trajectron++* on other datasets that consist of a more balanced variety of traffic classes could indicate improvements in performance similar to nuScenes and is advised.

Furthermore, the unintuitive behaviour regarding performance loss with more input data we observed in Figure 4 might have been caused by an accumulation of errors by the summation operation as pointed out in Section 4.2.1. For testing this hypothesis, we applied an alternative loss function, Equation 3, including a mean over the time steps. Looking back at Figure 6, we notice a more logical increase in performance with more input data at all but one of the illogical cases comparing summation ($T_s$) versus mean ($T_m$). Logical would be higher AUC and lower FDE and ADE after more input time steps. Whereas the unintuitive behaviour happened 4 times when we applied $T_s$, it happened only 1 time when we used the averaged $T_m$ (for the critical split regarding initial gaps at *highD*). Generally, we could say that the results became more intuitive, but to verify the hypothesis of using $T_m$ over $T_s$ we advise more testing, starting with validating the hypothesis on more $\beta$ values.

Considering the wider implications of examining whether it is beneficial to incorporate insights from cognitive science into an existing machine-learning framework regarding trajectory prediction, we could say several things. First, we showed that it is possible to reproduce a state-of-the-art trajectory prediction model on a large-scale dataset with a practical approach in a time span of several months. Secondly, we successfully implemented a cognitive-based novel idea, Smooth-Attention, without public-code availability and integrated it as a modification to *Trajectron++*, which had its public code available. Thirdly, we tested the model on various datasets to validate its performance and examine its generalisability. Broadly speaking, we show that it is possible to integrate an idea from cognitive science into a machine-learning framework and achieve increased performance on a large-scale autonomous driving dataset. Smooth-Attention seems to work better on such large-scale multi-agent datasets than on unbalanced datasets with few traffic agents. Also, the implementation of *Smooth-Trajectron++* does not give convincingly consistent improvements in results on gap-acceptance scenarios. However, the authors of the Smooth-Attention paper [28] show an increase in FDE and ADE performance regarding the INTERACTION dataset [41], which entails merging and roundabout scenarios. We advise to research the validity of these claims, no public code available, and to perform experiments with *Smooth-Trajectron* on the INTERACTION dataset to compare these results. Moreover, one could apply the attention over individual agents instead of to agent classes, which is the case in *Trajectron++*. This could address the issue of datasets containing only one or two classes having poor results, which happened in Section 4.2.3. In this way, the applicability of the model becomes even more general, including more potential datasets to test *Smooth-Trajectron++* on. However, a potential downfall of this method could be that the processing time gets larger when the number of agents increases, so this would probably only be feasible when the number of traffic participants is limited.

## 6. Conclusion

In this study, we propose *Smooth-Trajectron++*, a trajectory prediction model that builds upon the existing state-of-the-art model, *Trajectron++* [10]. Our approach involves the incorporation of a cognitively-inspired smooth attention module [28]. We demonstrate that our smooth attention version of Trajectron++ achieves improved performance on the large-scale nuScenes dataset. However, we observe that this smooth attention approach does not yield significant improvements on the datasets included in the benchmarking framework for gap acceptance [16]. These findings suggest that the smooth attention technique appears to be more suitable for large-scale multi-agent datasets that involve multiple agent types, as opposed to datasets comprising a small number of traffic agents of mostly the same type. Consequently, the concept of *smooth attention* may be better applied in models where the attention module operates over individual agents rather than semantic classes. Nonetheless, our results provide further support to previous research [23, 26, 28], highlighting the potential of incorporating cognitive insights to enhance predictions of human behaviour in traffic.

## Acknowledgments

I would like to express my sincere gratitude to Julian Schumann for his invaluable role as my daily supervisor throughout this project. Julian has consistently been available and willing to assist me, demonstrating a deep understanding of the subject matter. I am particularly grateful for his guidance in using the benchmarking framework and his ability to make complex concepts more comprehensible. I would also like to extend my appreciation to Arkady Zgonnikov for his role as a supervisor. Arkady's critical and clear insights into the technical aspects of the project have been invaluable. Furthermore, I am grateful for his compassionate mentorship during challenging moments.

Additionally, I acknowledge the contribution of DelftBlue, which provided me with the necessary resources to conduct all the experiments within the given time frame. Without their support, this research endeavor would not have been possible.

## References


[1] Abhay Singh and Sonia Mutreja. *Autonomous vehicle market size, share, value, report, growth*. URL: https://www.alliedmarketresearch.com/autonomous-vehicle-market.

[2] World Health Organization. *Road Traffic Injuries*. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries. Accessed on May 12, 2023. 2021.

[3] Ariyan Bighashdel and Gijs Dubbelman. "A Survey on Path Prediction Techniques for Vulnerable Road Users: From Traditional to Deep-Learning Approaches". In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Oct. 2019, pp. 1039–1046. DOI: 10.1109/ITSC.2019.8917053.

[4] Fanta Camara et al. "Pedestrian Models for Autonomous Driving Part II: High-Level Models of Human Behavior". In: *IEEE Transactions on Intelligent Transportation Systems* 22.9 (Sept. 2021). Conference Name: IEEE Transactions on Intelligent Transportation Systems, pp. 5453–5472. ISSN: 1558-0016. DOI: 10.1109/TITS.2020.3006767.

[5] Andrey Rudenko et al. "Human Motion Trajectory Prediction: A Survey". In: (May 15, 2019). DOI: 10.1177/0278364920917446. URL: https://arxiv.org/abs/1905.06113v3 (visited on 02/22/2022).

[6] Alexandre Alahi et al. "Social lstm: Human trajectory prediction in crowded spaces". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 961–971.

[7] Amir Rasouli, Iuliia Kotseruba, and John K. Tsotsos. "Are They Going to Cross? A Benchmark Dataset and Baseline for Pedestrian Crosswalk Behavior". In: *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*. 2017 IEEE International Conference on Computer Vision Workshops (ICCVW). ISSN: 2473-9944. Oct. 2017, pp. 206–213. DOI: 10.1109/ICCVW.2017.33.

[8] Abduallah Mohamed et al. *Social-stgcnn: A social spatio-temporal graph convolutional neural network for human trajectory prediction*. Tech. rep. 2020, pp. 14424–14432.

[9] Francesco Giuliari et al. "Transformer Networks for Trajectory Forecasting". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. 2020 25th International Conference on Pattern Recognition (ICPR). ISSN: 1051-4651. Jan. 2021, pp. 10335–10342. DOI: 10.1109/ICPR48806.2021.9412190.

[10] Tim Salzmann et al. "Trajectron++: Dynamically-Feasible Trajectory Forecasting with Heterogeneous Data". en. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 683–700. ISBN: 978-3-030-58523-5. DOI: 10.1007/978-3-030-58523-5_40.

[11] Agrim Gupta et al. "Social gan: Socially acceptable trajectories with generative adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 2255–2264.

[12] Anirudh Vemula, Katharina Muelling, and Jean Oh. "Social attention: Modeling attention in human crowds". In: *2018 IEEE international Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 4601–4607.

[13] Ye Yuan et al. "Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 9813–9823.

[14] Suresh Kumaar Jayaraman et al. "Analysis and Prediction of Pedestrian Crosswalk Behavior during Automated Vehicle Interactions". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020 IEEE International Conference on Robotics and Automation (ICRA). ISSN: 2577-087X. May 2020, pp. 6426–6432. DOI: 10.1109/ICRA40945.2020.9197347.

[15] Arash Kalatian and Bilal Farooq. "A context-aware pedestrian trajectory prediction framework for automated vehicles". In: *Transportation Research Part C: Emerging Technologies* 134 (Jan. 1, 2022), p. 103453. ISSN: 0968-090X. DOI: 10.1016/j.trc.2021.103453. URL: https://www.sciencedirect.com/science/article/pii/S0968090X21004423 (visited on 04/15/2022).

[16] Julian F. Schumann, Jens Kober, and Arkady Zgonnikov. "Benchmarking Behavior Prediction Models in Gap Acceptance Scenarios". In: *IEEE Transactions on Intelligent Vehicles* (2023), pp. 1–12. DOI: 10.1109/TIV.2023.3244280.

[17] S. Pellegrini et al. "You'll never walk alone: Modeling social behavior for multi-target tracking". In: *2009 IEEE 12th International Conference on Computer Vision*. ISSN: 2380-7504. Sept. 2009, pp. 261–268. DOI: 10.1109/ICCV.2009.5459260.

[18] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. "Crowds by example". In: 26.3 (2007), pp. 655–664.

[19] Holger Caesar et al. "nuScenes: A Multimodal Dataset for Autonomous Driving". In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. ISSN: 2575-7075. June 2020, pp. 11618–11628. DOI: 10.1109/CVPR42600.2020.01164.

[20] Robert Krajewski et al. "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems". In: *2018 21st International Conference*

*on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 2118–2125.

[21] Joshua I Gold and Michael N Shadlen. "The neural basis of decision making". In: *Annu. Rev. Neurosci.* 30 (2007), pp. 535–574.

[22] Arkady Zgonnikov, David Abbink, and Gustav Markkula. "Should I stay or should I go? Cognitive modeling of left-turn gap acceptance decisions in human drivers". In: *Human factors* (2022), p. 00187208221144561.

[23] Julian F. Schumann et al. *Using Models Based on Cognitive Theory to Predict Human Behavior in Traffic: A Case Study*. arXiv:2305.15187 [cs]. May 2023. DOI: 10.48550/arXiv.2305.15187. URL: http://arxiv.org/abs/2305.15187 (visited on 05/26/2023).

[24] Jami Pekkanen et al. "Variable-drift diffusion models of pedestrian road-crossing decisions". In: *Computational Brain & Behavior* (2021), pp. 1–21.

[25] Catarina Alexandra Pinto Moreira. "Quantum Probabilistic Graphical Models for Cognition and Decision". In: *D. Universidade de Lisboa* (2017).

[26] Qingyuan Song et al. "Research on quantum cognition in autonomous driving". In: *Scientific Reports* 12.1 (Jan. 7, 2022). Number: 1 Publisher: Nature Publishing Group, p. 300. ISSN: 2045-2322. DOI: 10.1038/s41598-021-04239-y. URL: http://www.nature.com/articles/s41598-021-04239-y (visited on 04/05/2022).

[27] Jeremy M Wolfe, George A Alvarez, and Todd S Horowitz. "Attention is fast but volition is slow". In: *Nature* 406.6797 (2000), pp. 691–691.

[28] Zhangjie Cao et al. *Leveraging Smooth Attention Prior for Multi-Agent Trajectory Prediction*. IEEE, 2022.

[29] Robert Krajewski et al. "The round dataset: A drone dataset of road user trajectories at roundabouts in germany". In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2020, pp. 1–6.

[30] Boris Ivanovic and Marco Pavone. "The Trajectron: Probabilistic Multi-Agent Trajectory Modeling With Dynamic Spatiotemporal Graphs". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). ISSN: 2380-7504. Oct. 2019, pp. 2375–2384. DOI: 10.1109/ICCV.2019.00246.

[31] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. "Learning structured output representation using deep conditional generative models". In: *Advances in neural information processing systems* 28 (2015).

[32] Sepp Hochreiter and Jürgen Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[33] Peter W Battaglia et al. "Relational inductive biases, deep learning, and graph networks". In: *arXiv preprint arXiv:1806.01261* (2018).

[34] Diederik P Kingma and Max Welling. "Auto-encoding variational bayes". In: *arXiv preprint arXiv:1312.6114* (2013).

[35] Quoc V Le, Navdeep Jaitly, and Geoffrey E Hinton. "A simple way to initialize recurrent networks of rectified linear units". In: *arXiv preprint arXiv:1504.00941* (2015).

[36] Thomas N Kipf and Max Welling. "Semi-supervised classification with graph convolutional networks". In: *arXiv preprint arXiv:1609.02907* (2016).

[37] Ashesh Jain et al. "Structural-rnn: Deep learning on spatio-temporal graphs". In: *Proceedings of the ieee conference on computer vision and pattern recognition*. 2016, pp. 5308–5317.

[38] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *arXiv preprint arXiv:1409.0473* (2014).

[39] Delft High Performance Computing Centre (DHPC). *DelftBlue Supercomputer (Phase 1)*. https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1. 2022.

[40] Etienne Barnard and LFA Wessels. "Extrapolation and interpolation in neural network classifiers". In: *IEEE Control Systems Magazine* 12.5 (1992), pp. 50–53.

[41] Wei Zhan et al. "Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps". In: *arXiv preprint arXiv:1910.03088* (2019).

# Appendix A.
# Tables from Benchmarking Framework Figures

TABLE 4. AUC INITIAL GAP 1/2

| Dataset | Models | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $T_{\beta_1}$ | | $T_{\beta_2}$ | | $T_{\beta_3}$ | | $T+$ | |
| *highD* | $0.890^{\pm 0.026}$ | $0.916^{\pm 0.009}$ | $0.896^{\pm 0.012}$ | $0.917^{\pm 0.006}$ | $0.904^{\pm 0.011}$ | $0.912^{\pm 0.012}$ | $\underline{0.906}^{\pm 0.010}$ | $\underline{0.928}^{\pm 0.005}$ |
| | 0.384 | 0.554 | 0.507 | 0.487 | 0.624 | 0.457 | $\underline{0.715}$ | $\underline{0.566}$ |
| *highD* (rest) | $0.889^{\pm 0.036}$ | $0.915^{\pm 0.018}$ | $0.893^{\pm 0.039}$ | $0.925^{\pm 0.040}$ | $\underline{0.933}^{\pm 0.014}$ | $\underline{0.949}^{\pm 0.007}$ | $0.884^{\pm 0.055}$ | $0.930^{\pm 0.011}$ |
| | 0.472 | 0.619 | 0.506 | 0.735 | 0.473 | $\underline{0.753}$ | $\underline{0.659}$ | 0.700 |
| *rounD* | $\underline{0.993}^{\pm 0.002}$ | $0.996^{\pm 0.005}$ | $0.992^{\pm 0.002}$ | $0.995^{\pm 0.005}$ | $0.990^{\pm 0.003}$ | $\underline{0.998}^{\pm 0.002}$ | $0.992^{\pm 0.002}$ | $0.996^{\pm 0.003}$ |
| | 0.933 | 0.972 | $\underline{0.934}$ | $\underline{0.974}$ | 0.921 | 0.960 | 0.923 | 0.966 |
| *L-GAP* | $\underline{0.585}^{\pm 0.045}$ | $0.992^{\pm 0.006}$ | $\underline{0.585}^{\pm 0.045}$ | $0.991^{\pm 0.003}$ | $\underline{0.585}^{\pm 0.045}$ | $0.987^{\pm 0.008}$ | $\underline{0.585}^{\pm 0.045}$ | $\underline{0.993}^{\pm 0.005}$ |
| | $\underline{0.193}$ | 0.922 | $\underline{0.193}$ | 0.926 | $\underline{0.193}$ | 0.928 | $\underline{0.193}$ | $\underline{0.929}$ |

TABLE 5. AUC INITIAL GAP 2/2

| Dataset | Models | | | | | |
|---|---|---|---|---|---|---|
| | $T_{\beta_4}$ | | $T_{\beta_5}$ | | $T+$ | |
| *highD* | $0.896^{\pm 0.008}$ | $0.918^{\pm 0.006}$ | $0.901^{\pm 0.012}$ | $0.911^{\pm 0.011}$ | $\underline{0.906}^{\pm 0.010}$ | $\underline{0.928}^{\pm 0.005}$ |
| | 0.417 | 0.501 | 0.537 | $\underline{0.648}$ | $\underline{0.715}$ | 0.566 |
| *highD* (rest) | $\underline{0.922}^{\pm 0.025}$ | $0.925^{\pm 0.023}$ | $0.890^{\pm 0.027}$ | $\underline{0.936}^{\pm 0.018}$ | $0.884^{\pm 0.055}$ | $0.930^{\pm 0.011}$ |
| | 0.427 | 0.689 | 0.424 | 0.612 | $\underline{0.659}$ | $\underline{0.700}$ |
| *rounD* | $0.991^{\pm 0.001}$ | $0.994^{\pm 0.003}$ | $\underline{0.992}^{\pm 0.003}$ | $\underline{0.997}^{\pm 0.003}$ | $0.992^{\pm 0.002}$ | $0.996^{\pm 0.003}$ |
| | 0.914 | 0.962 | $\underline{0.928}$ | $\underline{0.971}$ | 0.923 | 0.966 |
| *L-GAP* | $\underline{0.585}^{\pm 0.045}$ | $0.992^{\pm 0.005}$ | $\underline{0.585}^{\pm 0.045}$ | $0.987^{\pm 0.011}$ | $\underline{0.585}^{\pm 0.045}$ | $\underline{0.993}^{\pm 0.005}$ |
| | $\underline{0.193}$ | 0.916 | $\underline{0.193}$ | 0.912 | $\underline{0.193}$ | $\underline{0.929}$ |

TABLE 6. AUC FIXED-SIZED GAP (AUC) 1/2

| Dataset | Models | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $T_{\beta_1}$ | | $T_{\beta_2}$ | | $T_{\beta_3}$ | | $T+$ | |
| *highD* | $0.859^{\pm 0.011}$ | $0.888^{\pm 0.007}$ | $0.851^{\pm 0.027}$ | $\underline{0.894}^{\pm 0.017}$ | $0.862^{\pm 0.011}$ | $0.881^{\pm 0.006}$ | $\underline{0.863}^{\pm 0.013}$ | $0.891^{\pm 0.012}$ |
| | 0.485 | 0.589 | 0.431 | $\underline{0.666}$ | $\underline{0.492}$ | 0.521 | 0.435 | 0.537 |
| *highD* (rest) | $0.899^{\pm 0.028}$ | $0.916^{\pm 0.015}$ | $0.885^{\pm 0.050}$ | $\underline{0.924}^{\pm 0.016}$ | $\underline{0.919}^{\pm 0.014}$ | $0.917^{\pm 0.020}$ | $0.914^{\pm 0.010}$ | $0.898^{\pm 0.038}$ |
| | 0.653 | 0.757 | 0.653 | 0.710 | $\underline{0.725}$ | $\underline{0.817}$ | 0.633 | 0.715 |
| *rounD* | $\underline{0.980}^{\pm 0.006}$ | $0.990^{\pm 0.005}$ | $\underline{0.980}^{\pm 0.006}$ | $\underline{0.996}^{\pm 0.005}$ | $\underline{0.980}^{\pm 0.006}$ | $0.993^{\pm 0.006}$ | $\underline{0.980}^{\pm 0.006}$ | $0.993^{\pm 0.005}$ |
| | $\underline{0.942}$ | 0.982 | $\underline{0.942}$ | 0.991 | $\underline{0.942}$ | $\underline{0.994}$ | $\underline{0.942}$ | 0.974 |
| *L-GAP* | $\underline{0.923}^{\pm 0.019}$ | $\underline{0.997}^{\pm 0.003}$ | $\underline{0.923}^{\pm 0.019}$ | $0.991^{\pm 0.006}$ | $\underline{0.923}^{\pm 0.019}$ | $0.991^{\pm 0.006}$ | $\underline{0.923}^{\pm 0.019}$ | $0.992^{\pm 0.007}$ |
| | $\underline{0.581}$ | 0.917 | $\underline{0.581}$ | 0.926 | $\underline{0.581}$ | 0.923 | $\underline{0.581}$ | $\underline{0.930}$ |

TABLE 7. AUC FIXED-SIZED GAP 2/2

| **Dataset** | **Models** | | | | | |
|---|---|---|---|---|---|---|
| | $T_{\beta_4}$ | | $T_{\beta_5}$ | | $T+$ | |
| *highD* | $0.853^{\pm 0.027}$ | $0.884^{\pm 0.019}$ | $0.852^{\pm 0.012}$ | $\underline{0.893}^{\pm 0.008}$ | $\underline{0.863}^{\pm 0.013}$ | $0.891^{\pm 0.012}$ |
| | 0.672 | <u>0.620</u> | <u>0.703</u> | 0.508 | 0.435 | 0.537 |
| *highD* (rest) | $0.910^{\pm 0.013}$ | $0.874^{\pm 0.031}$ | $0.901^{\pm 0.028}$ | $\underline{0.918}^{\pm 0.012}$ | $\underline{0.914}^{\pm 0.010}$ | $0.898^{\pm 0.038}$ |
| | <u>0.763</u> | <u>0.779</u> | 0.624 | 0.708 | 0.633 | 0.715 |
| *rounD* | $\underline{0.980}^{\pm 0.006}$ | $0.993^{\pm 0.006}$ | $\underline{0.980}^{\pm 0.006}$ | $0.993^{\pm 0.006}$ | $\underline{0.980}^{\pm 0.006}$ | $\underline{0.993}^{\pm 0.005}$ |
| | <u>0.942</u> | <u>0.984</u> | <u>0.942</u> | 0.984 | <u>0.942</u> | 0.974 |
| *L-GAP* | $\underline{0.923}^{\pm 0.019}$ | $0.993^{\pm 0.004}$ | $\underline{0.923}^{\pm 0.019}$ | $\underline{0.996}^{\pm 0.004}$ | $\underline{0.923}^{\pm 0.019}$ | $0.992^{\pm 0.007}$ |
| | <u>0.581</u> | <u>0.930</u> | <u>0.581</u> | 0.920 | <u>0.581</u> | 0.930 |

TABLE 8. ADE FIXED-SIZED GAP 1/2

| **Dataset** | **Models** | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $T_{\beta_1}$ | | $T_{\beta_2}$ | | $T_{\beta_3}$ | | $T+$ | |
| *highD* | $\underline{3.905}^{\pm 0.098}$ | $2.957^{\pm 0.212}$ | $3.919^{\pm 0.104}$ | $2.855^{\pm 0.154}$ | $3.920^{\pm 0.029}$ | $\underline{2.669}^{\pm 0.228}$ | $3.958^{\pm 0.135}$ | $2.858^{\pm 0.273}$ |
| | 9.893 | 8.130 | <u>8.499</u> | 7.624 | 9.080 | <u>6.088</u> | 9.032 | 8.352 |
| *highD* (rest) | $4.861^{\pm 0.477}$ | $4.198^{\pm 0.479}$ | $5.895^{\pm 0.742}$ | $5.121^{\pm 0.480}$ | $4.590^{\pm 1.034}$ | $\underline{4.120}^{\pm 0.159}$ | $\underline{4.321}^{\pm 0.367}$ | $6.214^{\pm 3.288}$ |
| | <u>6.381</u> | 7.473 | 7.295 | <u>5.431</u> | 6.434 | 6.844 | 6.598 | 5.743 |
| *rounD* | $\underline{1.143}^{\pm 0.049}$ | $\underline{0.773}^{\pm 0.050}$ | $\underline{1.143}^{\pm 0.049}$ | $0.780^{\pm 0.048}$ | $\underline{1.143}^{\pm 0.049}$ | $0.807^{\pm 0.047}$ | $\underline{1.143}^{\pm 0.049}$ | $0.799^{\pm 0.030}$ |
| | <u>1.915</u> | 1.243 | <u>1.915</u> | <u>1.242</u> | <u>1.915</u> | 1.337 | <u>1.915</u> | 1.311 |
| *L-GAP* | $\underline{1.539}^{\pm 0.078}$ | $\underline{0.668}^{\pm 0.028}$ | $\underline{1.539}^{\pm 0.078}$ | $0.701^{\pm 0.054}$ | $\underline{1.539}^{\pm 0.078}$ | $0.703^{\pm 0.053}$ | $\underline{1.539}^{\pm 0.078}$ | $0.691^{\pm 0.033}$ |
| | <u>2.636</u> | <u>1.126</u> | <u>2.636</u> | 1.188 | <u>2.636</u> | 1.145 | <u>2.636</u> | 1.240 |

TABLE 9. ADE FIXED-SIZED GAP 2/2

| **Dataset** | **Models** | | | | | |
|---|---|---|---|---|---|---|
| | $T_{\beta_4}$ | | $T_{\beta_5}$ | | $T+$ | |
| *highD* | $\underline{3.861}^{\pm 0.117}$ | $2.853^{\pm 0.215}$ | $3.983^{\pm 0.213}$ | $\underline{2.681}^{\pm 0.227}$ | $3.958^{\pm 0.135}$ | $2.858^{\pm 0.273}$ |
| | <u>8.817</u> | <u>8.296</u> | 9.466 | 8.841 | 9.032 | 8.352 |
| *highD* (rest) | $4.635^{\pm 0.409}$ | $5.194^{\pm 1.244}$ | $4.593^{\pm 0.387}$ | $\underline{5.176}^{\pm 0.624}$ | $\underline{4.321}^{\pm 0.367}$ | $6.214^{\pm 3.288}$ |
| | 7.319 | 6.396 | 6.825 | <u>5.371</u> | <u>6.598</u> | 5.743 |
| *rounD* | $\underline{1.143}^{\pm 0.049}$ | $0.802^{\pm 0.039}$ | $\underline{1.143}^{\pm 0.049}$ | $0.806^{\pm 0.032}$ | $\underline{1.143}^{\pm 0.049}$ | $\underline{0.799}^{\pm 0.030}$ |
| | <u>1.915</u> | 1.172 | <u>1.915</u> | <u>1.151</u> | <u>1.915</u> | 1.311 |
| *L-GAP* | $\underline{1.539}^{\pm 0.078}$ | $\underline{0.678}^{\pm 0.034}$ | $\underline{1.539}^{\pm 0.078}$ | $0.709^{\pm 0.032}$ | $\underline{1.539}^{\pm 0.078}$ | $0.691^{\pm 0.033}$ |
| | <u>2.636</u> | <u>1.200</u> | <u>2.636</u> | 1.244 | <u>2.636</u> | 1.240 |

TABLE 10. FDE FIXED-SIZED GAP 1/2

| Dataset | Models | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $T_{\beta_1}$ | | $T_{\beta_2}$ | | $T_{\beta_3}$ | | $T+$ | |
| *highD* | $10.276^{\pm 0.254}$ | $7.888^{\pm 0.490}$ | $10.237^{\pm 0.420}$ | $7.518^{\pm 0.363}$ | $10.127^{\pm 0.144}$ | $7.123^{\pm 0.729}$ | $10.293^{\pm 0.373}$ | $7.566^{\pm 0.789}$ |
| | 26.416 | 23.914 | 22.473 | 22.010 | 24.259 | 17.203 | 25.068 | 25.481 |
| *highD* (rest) | $13.192^{\pm 1.884}$ | $11.402^{\pm 1.697}$ | $16.337^{\pm 2.082}$ | $13.954^{\pm 1.107}$ | $12.346^{\pm 3.286}$ | $11.107^{\pm 0.464}$ | $11.325^{\pm 1.023}$ | $17.431^{\pm 10.003}$ |
| | 16.562 | 20.230 | 19.131 | 14.943 | 17.325 | 18.929 | 17.230 | 16.427 |
| *rounD* | $2.617^{\pm 0.105}$ | $1.921^{\pm 0.150}$ | $2.617^{\pm 0.105}$ | $1.926^{\pm 0.110}$ | $2.617^{\pm 0.105}$ | $2.017^{\pm 0.125}$ | $2.617^{\pm 0.105}$ | $1.994^{\pm 0.083}$ |
| | 4.768 | 3.251 | 4.768 | 3.137 | 4.768 | 3.538 | 4.768 | 3.278 |
| *L-GAP* | $3.665^{\pm 0.178}$ | $1.598^{\pm 0.089}$ | $3.665^{\pm 0.178}$ | $1.686^{\pm 0.142}$ | $3.665^{\pm 0.178}$ | $1.680^{\pm 0.142}$ | $3.665^{\pm 0.178}$ | $1.652^{\pm 0.106}$ |
| | 6.697 | 2.766 | 6.697 | 2.944 | 6.697 | 2.829 | 6.697 | 3.092 |

TABLE 11. FDE FIXED-SIZED GAP 2/2

| Dataset | Models | | | | | |
|---|---|---|---|---|---|---|
| | $T_{\beta_4}$ | | $T_{\beta_5}$ | | $T+$ | |
| *highD* | $10.019^{\pm 0.374}$ | $7.636^{\pm 0.640}$ | $10.314^{\pm 0.557}$ | $7.197^{\pm 0.563}$ | $10.293^{\pm 0.373}$ | $7.566^{\pm 0.789}$ |
| | 22.964 | 22.622 | 25.566 | 27.518 | 25.068 | 25.481 |
| *highD* (rest) | $12.285^{\pm 1.593}$ | $14.370^{\pm 3.614}$ | $12.109^{\pm 1.024}$ | $14.110^{\pm 1.482}$ | $11.325^{\pm 1.023}$ | $17.431^{\pm 10.003}$ |
| | 19.694 | 17.417 | 18.364 | 14.455 | 17.230 | 16.427 |
| *rounD* | $2.617^{\pm 0.105}$ | $1.996^{\pm 0.129}$ | $2.617^{\pm 0.105}$ | $2.045^{\pm 0.079}$ | $2.617^{\pm 0.105}$ | $1.994^{\pm 0.083}$ |
| | 4.768 | 3.040 | 4.768 | 2.989 | 4.768 | 3.278 |
| *L-GAP* | $3.665^{\pm 0.178}$ | $1.629^{\pm 0.109}$ | $3.665^{\pm 0.178}$ | $1.688^{\pm 0.099}$ | $3.665^{\pm 0.178}$ | $1.652^{\pm 0.106}$ |
| | 6.697 | 3.001 | 6.697 | 3.174 | 6.697 | 3.092 |

TABLE 12. TNR-PR CRITICAL GAP 1/2

| Dataset | Models | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | $T_{\beta_1}$ | | $T_{\beta_2}$ | | $T_{\beta_3}$ | | $T+$ | |
| *rounD* | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| *L-GAP* | $0.599^{\pm 0.350}$ | $0.497^{\pm 0.497}$ | $0.599^{\pm 0.350}$ | $0.497^{\pm 0.497}$ | $0.599^{\pm 0.350}$ | $0.497^{\pm 0.497}$ | $0.599^{\pm 0.350}$ | $0.497^{\pm 0.497}$ |
| | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |

TABLE 13. TNR-PR CRITICAL GAP 2/2

| Dataset | Models | | | | | |
|---|---|---|---|---|---|---|
| | $T_{\beta_4}$ | | $T_{\beta_5}$ | | $T+$ | |
| *rounD* | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 |
| *L-GAP* | $0.599^{\pm 0.350}$ | $0.494^{\pm 0.494}$ | $0.599^{\pm 0.350}$ | $0.494^{\pm 0.494}$ | $0.599^{\pm 0.350}$ | $0.497^{\pm 0.497}$ |
| | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 | 0.000 |