Human Trajectory Forecasting in Crowds:
A Deep Learning Perspective

Parth Kothari, Sven Kreiss, Alexandre Alahi
Visual Intelligence for Transportation (VITA) Lab, EPFL

Abstract.
Since the past few decades, human trajectory forecasting has been a field of active research owing to its numerous real-world applications: evacuation situation analysis, traffic operations, deployment of social robots in crowded environments, to name a few. In this work, we cast the problem of human trajectory forecasting as learning a representation of human social interactions. Early works handcrafted this representation based on domain knowledge. However, social interactions in crowded environments are not only diverse but often subtle. Recently, deep learning methods have outperformed their handcrafted counterparts, as they learned about human-human interactions in a more generic data-driven fashion. In this work, we present an in-depth analysis of existing deep learning based methods for modelling social interactions. Based on our analysis, we propose a simple yet powerful method for effectively capturing these social interactions. To objectively compare the performance of these interaction-based forecasting models, we develop a large scale interaction-centric benchmark TrajNet++, a significant yet missing component in the field of human trajectory forecasting. We propose novel performance metrics that evaluate the ability of a model to output socially acceptable trajectories. Experiments on TrajNet++ validate the need for our proposed metrics, and our method outperforms competitive baselines on both real-world and synthetic datasets.

1 Introduction
Humans possess the natural ability to navigate in social environments. In other words, we have understood the social etiquette of human motion from respecting personal space and yielding right-of-way to avoid walking through people belonging to the same group. Our social interactions lead to various complex pattern-formation phenomena in crowds, for instance, the emergence of lanes of pedestrians with uniform walking direction, oscillations of the pedestrian flow at bottlenecks. The ability to model social interactions and thereby forecast crowd dynamics in real world environments is extremely valuable for a wide range of applications: infrastructure design [1, 2, 3], traffic operations [4], crowd abnormality detection systems [5], evacuation situation analysis [6, 7, 8, 9], deployment of autonomous vehicles [10, 11], deployment of social robots in pedestrian-only environments [12] and recently helping in the broad quest of building a digital twin of our built environment. However, modelling social interactions is an extremely challenging task as there exists no fixed set of rules which govern human motion. A task closely related to learning human social interactions is forecasting the
Figure 1: Human trajectory forecasting refers to the task of forecasting the future trajectories of all humans which conform to the social norms, given the past observed scene. It is the presence of social interactions that distinguish human trajectory forecasting from other sequence modelling tasks: the primary pedestrian (X1) deviates from his direction of motion to avoid a collision, by forecasting the trajectory of the child (X2) in front of him. Moreover, one needs to have the ability to measure the performance of the model with respect to predicting socially acceptable outputs: instead of deviating, the model can predict that the primary pedestrian slows down. This prediction can lead to high errors in distance-based metrics, even though it is socially acceptable.

Before formally defining human trajectory forecasting, we introduce the notion of Trajectory and Scene. We define a Trajectory as the time-profile of pedestrian motion states. Generally, these states are the position and velocity of a human. However, we can consider more complex states like body pose, to glean more information about a person’s movement. We define a Scene as a collection of trajectories of multiple humans interacting in a social setting. A scene may also comprise physical objects and non-navigable areas that affect the human trajectories, e.g., walls, doors and elevators.

We define human trajectory forecasting as follows:

Given the past trajectories of all humans in a scene, forecast the future trajectories which conform to the social norms.

Human trajectory forecasting is primarily a sequence modelling task. The typical challenges for a sequence modelling task are (1) encoding observation sequence: we need to learn to model the long-term dependency in the past trajectory effectively, (2) multimodality: given the history of a scene, multiple futures (predictions) are plausible. In addition to this, for human trajectory forecasting, there exist two crucial challenges that differentiate it from other sequence prediction tasks such as language modelling, weather forecasting, and stock market forecasting (see Fig 1):

- Presence of social interactions: the trajectory of a person is affected by the motion of the other people in his/her surroundings. Modelling how the observation of one sequence affects the forecast of another sequence is an essential prerequisite for a good human trajectory forecasting model.
- Physically acceptable outputs: a good human trajectory forecasting model should provide physically acceptable outputs, for instance, the model prediction should not undergo collisions. Quantifying the physical feasibility of a model prediction is crucial for safety-critical applications.

In this work, we cast the problem of human trajectory forecasting as learning a representation of human social interactions. In other words, our objective is to encode the observed scene into a representation that captures all information necessary to forecast human motion. For instance, a
representation can be the history of the velocities for each human. Such a representation considers human motion in isolation. However, as mentioned above, an ideal representation has to encode not only the past motion but also the social interactions that a human undergoes with the surrounding humans. We term this representation as **Social Representation**. To ensure that this representation encodes only the social interactions, we assume that there do not exist any physical constraints in our scenes. Moreover, the representation can also be affected by the long-term goal of the human, which cannot always be observed or inferred. To find a representation that captures dominantly the social interactions, our focus in this work is on **short-term** human trajectory forecasting (next 5 seconds).

Early works [13, 14, 15] handcrafted the social representation of the scene based on the domain knowledge of human motion. Social Force [13], one of the first seminal work on human trajectory forecasting, defined attractive forces (towards the goal of a person and towards his/her group) and repulsive forces (away from people not belonging to a person’s group and physical obstacles) to forecast human motion. Antonini et al. [14] utilized the discrete choice framework for modelling pedestrian dynamics in a crowd by proposing a dynamic and individual-based discretization of space around the pedestrian. Pellegrini et al. [15] defined an energy functional to capture interactions between two people based on their distance of closest approach (assuming constant-velocity movement). However, such handcrafted representations cannot capture all the diverse, higher-order, and often subtle interactions of human motion. To overcome this limitation, Alahi et al. [16] proposed the first neural network based model ‘Social LSTM’, paving the way for new deep learning methods for human trajectory forecasting. Neural networks (NNs) are powerful function approximators that can learn useful representations given large amounts of real data, without any prior assumptions.

Following the success of Social LSTM, a variety of NN-based interaction modules have been proposed in literature to model the social interactions. In this work, we explicitly focus on the design of these interaction modules and not the entire forecasting model. The challenge in designing these modules lies in handling a variable number of neighbours and modelling how they collectively influence one’s future trajectory. We present a broad umbrella encompassing the existing designs of interaction modules based on the encoding architecture and the process by which the individual information of each neighbour is aggregated. Based on our taxonomy, we propose a simple yet novel module that improves the social interaction due to its ability to preserve the uniqueness of the neighbours and model higher-order interactions in the temporal domain.

To demonstrate the efficacy of a trajectory forecasting model, one needs to have the means to objectively compare with other forecasting baselines on good quality datasets. However, current methods have been evaluated on different subsets of available data without proper sampling of scenes in which social interactions occur. As our final contribution, we introduce TrajNet++, a large scale interaction-centric trajectory forecasting benchmark comprising explicit agent-agent scenarios. Our benchmark provides proper indexing of trajectories by defining a hierarchy of trajectory categorization. In addition, we provide an extensive evaluation system to test the gathered methods for a fair comparison. In our evaluation, we go beyond the standard distance-based metrics and introduce novel metrics that measure the capability of a model to emulate pedestrian behavior in crowds. Finally, we demonstrate the efficacy of our proposed baseline on TrajNet++, in comparison to existing works. We rely on the spirit of crowdsourcing and encourage researchers to submit their models to our benchmark, so the quality of trajectory forecasting models can keep increasing in tackling more challenging scenarios.

To summarize, our main contributions are as follows:

1. We provide an in-depth analysis of existing designs of interaction encoders along with their source code.
2. We propose a simple yet novel method for capturing social interactions, preserving the unique identity of surrounding pedestrians and providing an improved representation with time.

3. We present a large scale interaction-centric trajectory forecasting benchmark with novel evaluation metrics that quantify the physical feasibility of a model.

2 Related Work

Finding the ideal representation to encode human social interactions in crowded environments is an extremely challenging task. Social interactions are not only diverse but often subtle. In this work, we consider agent-based or microscopic models of pedestrian crowds, where collective phenomena emerge from the complex interactions between many individuals (self-organizing effects). Current human trajectory forecasting works can be categorized into learning human-human (social) interactions or human-space (physical) interactions or both. Our work is focused on deep learning based models that capture social interactions. In this section, we review the work done for modelling the agent-agent interactions to obtain the social representation.

Among the simplest models for representing human motion are the kinematic models such as constant velocity models and constant acceleration models. Mogelmose et al. [17] used a linear motion predictor to infer critical situations near roadside. Classical path prediction algorithms like Kalman filters [18], autoregressive models [19, 20] have also been explored to represent human motion.

With a specific focus on pedestrian path forecasting problem, Helbing and Molnar [13] presented a motion model with attractive forces (towards the goal) and repulsive forces (away from obstacles), called Social Force model, which captures the social and physical interactions. Their seminal work displays competitive results even on modern pedestrian datasets and has been extended for improved trajectory forecasting [21, 22, 23], tracking [24, 25, 26] and activity forecasting [27, 28]. Another prominent model for human motion is Reciprocal Velocity Obstacles (RVO) [29], which guarantees safe and oscillation-free motion, assuming that each agent follows identical collision avoidance reasoning. Social interaction modelling has been approached from different perspectives such as Discrete Choice framework [30], continuum dynamics [31] and Gaussian processes [32, 33]. Scholler et al. [34] proposed an effective constant velocity baseline for motion prediction. Robicquet et al. [35] defined social sensitivity to characterize human motion into different navigation styles. Alahi et al. [36] defined Social Affinity Maps to link broken or unobserved trajectories to forecast pedestrian destinations. Yi et al. [37] exploited crowd grouping as a cue to better forecast trajectories.

However, all these methods use handcrafted functions based on relative distances and specific rules to model interactions. These functions impose not only strong priors but also have limited capacity when modelling complex interactions. In recent times, methods based on neural networks (NNs) that infer interactions in a data-driven fashion have been shown to outperform the works mentioned above.

Inspired by the application of recurrent neural networks (RNNs) in diverse sequence prediction tasks [38, 39, 40, 41], Alahi et al. [16] proposed Social LSTM, the first NN-based model for human trajectory forecasting. Social LSTM is an LSTM [42] network with a novel social pooling layer to capture social interactions of nearby pedestrians. RNNs incorporating social interactions allow anticipating interactions that can occur in a more distant future. The social pooling module has been extended to incorporate physical space context [43, 44, 45, 46, 47, 48] and various other designs of NN-based interaction module have been proposed [49, 50, 51, 52, 53, 54, 55, 56]. Pfeiffer et al. [49] proposed an angular pooling grid for efficient computation. Shi et al. [50] proposed an elliptical pooling grid placed along the direction of movement of the pedestrian with more focus on the
pedestrians in the front. Bisagno et al. [51] proposed to consider only pedestrians not belonging to the same group during social pooling. Gupta et al. [52] propose to encode neighbourhood information through the use of a permutation-invariant (symmetric) max-pooling function. Zhang et al. [53] proposed to refine the state of the LSTM cell using message passing algorithms. Zhu et al. [54] proposed a novel star topology to model interactions. The center hub maintains information of the entire scene which each pedestrian can query. Ivanovic et al. [55] proposed to sum-pool the neighbour states and pass it through an LSTM-based encoder to obtain the interaction vector. Liang et al. [56] proposed to utilize geometric relations obtained from the spatial distance between pedestrians, to derive the interaction representation. Many works [57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67] propose interaction module designs based on attention mechanisms [68, 69] to identify the neighbours which affect the trajectory of the person of interest. The attention weights are either learned or handcrafted based on domain knowledge (e.g., euclidean distance).

Few works augment the position or velocity of a person with additional social cues to represent human motion and interaction better. Hasan et al. [70] uses head direction along with velocity as input to their architecture. While modelling social interactions, the authors only consider the pedestrians in the visual frustum of attention [71]. Sun et al. [72] augments body direction with time and date. Ma et al. [73] extract visual attributes from the image, and utilize them to predict future trajectories using concept of fictitious play.

Different from the general notion of encoding motion using RNNs, Radwan et al. [74] proposed temporal convolutional neural networks (CNNs). Recently, Guiliari et al. [75] proposed Transformer architecture for the task of trajectory forecasting, but they do not take social interactions into account. For an extensive survey of all human forecasting methods capturing both social and physical interactions, one can refer to Rudenko et al. [76].

3 Method

A global data-driven pipeline for forecasting human motion is illustrated in Fig 2. It comprises of the motion encoding module, the interaction module and the decoder module. On a high level, the motion encoding module is responsible for encoding the past motion of pedestrians. The interaction module learns to capture the social interactions between pedestrians. The motion encoding module and the interaction module are not necessarily mutually exclusive. The output of the interaction module is the social representation of the scene. The social representation is passed to the decoder module to predict a single trajectory or a trajectory distribution depending on the decoder architecture. In this work, we focus on investigating the design choices for the interaction module.

3.1 Problem Statement

Our objective is to forecast the future trajectories of all the pedestrians present in a scene. The network takes as input the trajectories of all the people in a scene denoted by \( X = X_1, X_2, \ldots, X_n \) and our task is to forecast the corresponding future trajectories \( Y = Y_1, Y_2, \ldots, Y_n \). The position of pedestrian \( i \) at time-step \( t \) is denoted by \( x^t_i = (x^t_i, y^t_i) \). We receive the positions of all pedestrians at time-steps \( t = 1, \ldots, T_{obs} \) and want to forecast the future (ground truth) positions \( y^t_i = (x^t_i, y^t_i) \) from time-steps \( t = T_{obs} + 1 \) to \( T_{pred} \). We denote our predictions using \( \hat{Y} \). The velocity of a pedestrian \( i \) at time-step \( t \) is denoted by \( v^t_i \). We denote the state of pedestrian \( i \) at time-step \( t \) by \( s^t_i \). The state can refer to different attributes of the person, e.g., the position as well as velocity of the person (\( s^t_i = [x^t_i, v^t_i] \)). The problem statement can be extended to take as input more attributes at each time-step, e.g., the body pose, as well as predicting \( k \) most-likely future trajectories.
Figure 2: The global data-driven pipeline for human trajectory forecasting. The motion encoding module is responsible for encoding the past motion of pedestrians. The interaction module learns to capture the social interactions between pedestrians. The output of the interaction module is the social representation of the scene. The social representation is passed to the decoder module to forecast a single trajectory or a trajectory distribution.

3.2 Interaction Module

Humans have the capability to navigate with ease in complex, crowded environments by following unspoken social rules. The social interactions arising as a result of these unspoken rules are captured by designing novel interaction modules. We now discuss in detail the different components of data-driven interaction encoders. We present a broad overview of the interaction model designs proposed in the literature. The existing interaction models can be categorized into two categories, based on the input representation:

- Grid based
- Non-Grid based

3.2.1 Grid Based Interaction Models

In grid-based models [16, 49, 46, 47, 51, 43, 48, 44], the interaction module takes as input a local grid constructed around the pedestrian of interest. Each cell within the grid represents a particular spatial position relative to the pedestrian of interest. Each cell contains information about neighbours located in that corresponding position. The information of the neighbours can be provided in various forms:

1. **Occupancy Pooling** (Fig 3a): Each cell indicates the presence of a neighbour [16, 45]
2. **Social Pooling** (Fig 3c): Each cell contains the entire past history of the neighbour, represented, e.g., the LSTM hidden state of the neighbours [16, 45, 47, 51, 43, 48, 44].

Grid-based modules provide the advantage of implicitly modelling the spatial context around the primary pedestrian. However, these methods can suffer from (1) loss of resolution arising from a defined size of each cell, and (2) the inability to model far-away pedestrians due to fixed grid size. These issues can be resolved by increasing the resolution and the grid size, but the solution significantly hampers the computational capability.

Mathematically, for occupancy pooling, at time-step $t$, we denote the neighbourhood of pedestrian $i$ as $O_t^i$, which is a $N_o \times N_o$ matrix, where $N_o$ is the size of the grid. The $(m, n)$ element of this grid is

$$O_t^i(m, n) = \sum_{j \in N_i} I_{mn}[x_j^t - x_i^t, y_j^t - y_i^t],$$

where $I_{mn}[x, y]$ is an indicator function to check if $(x, y)$ lies in the $(m, n)^{th}$ cell of the grid, and $N_i$
is the set of neighbors corresponding to person $i$. We denote this architecture by $\text{O-Grid}$.

For social pooling, given $h_i^t$ denotes the $D$-dimensional hidden-state of the LSTM of person $i$ at time-step $t$, the ‘social’ tensor $H_i^t$ of size $N_o \times N_o \times D$ is constructed as:

$$H_i^t(m,n,:) = \sum_{j \in N_i} \mathbb{1}_{mn}[x_j^t - x_i^t, y_j^t - y_i^t]h_j^{t-1}. \quad (2)$$

We denote this architecture by $\text{S-Grid}$.

The resulting grid is embedded to get the interaction vector $p_i^t$ (Fig 3d):

$$p_i^t = \phi(H_i^t; W_p), \quad (3)$$

where $\phi$ is an MLP and the weights $W_p$ are learned. We would like to note that the input grid can also be represented in polar coordinates [49, 46].

Despite being more expressive than occupancy pooling, social pooling is known to suffer from high complexity, especially in cases of high-resolution grids. To reduce complexity, we propose Directional Pooling (see Fig 3b), wherein each cell comprises of the relative velocity of the neighbour with respect to the primary pedestrian. Let $v_{ji}^t$ denote the relative velocity of person $j$ with respect to person $i$, i.e., $v_{ji}^t = v_i^t - v_j^t$. The directional pooling tensor $D_i^t$ is constructed as:

$$D_i^t(m,n,:) = \sum_{j \in N_i} \mathbb{1}_{mn}[x_j^t - x_i^t, y_j^t - y_i^t]v_{ji}^t. \quad (4)$$

We denote this architecture by $\text{D-Grid}$. We will demonstrate in the experimental section that directional pooling, in addition to its computational advantages, performs at par with social pooling in controlled synthetic scenarios. Moreover, in real-world settings, directional pooling provides superior performance with respect to the physical acceptability of predicted trajectories.

### 3.2.2 Non-Grid Based Interaction Models

Non-grid based modules [58, 77, 57, 52, 50, 53, 54, 55, 62, 60], as the name suggests, capture the social interactions in a grid-free manner. The challenge in designing non-grid based models lies in (1) handling a variable number of neighbours and (2) aggregating the state information of multiple neighbours to obtain the interaction vector $p_i^t$. To achieve this, these models utilize the concepts of social attention [58, 57, 53, 62, 59, 65, 60], or application of a learned symmetric function [52]. Fig 4 illustrates the different neighbouring information aggregation strategies.

Recent works [58, 57, 53, 62, 59] propose to provide different weights to the neighbouring hidden-states to make the interaction vector $p_i^t$:

$$H_i^t = \sum_j a_{ij}^t \ast h_j^t, \quad (5)$$

$$p_i^t = \phi(H_i^t; W_p), \quad (6)$$

where $h_j^t$ denotes the hidden-state vector of pedestrian $j$, $a_{ij}^t$ is the weight indicating the influence of pedestrian $j$ on the trajectory of pedestrian $i$ at time-step $t$. $\phi$ is an MLP and the weights $W_p$ are learned. We denote this attention-based design by $\text{Att-MLP}$.

Fernando et al.[58] propose hardwired attention weights based on the distance of the neighbour from the pedestrian of interest. Amirian et al.[62] obtain the attention weights using hidden-state of primary pedestrian and interaction feature vectors of the neighbours learnt from pre-defined geometric features. On top of the attention mechanism, Zhang et al. [53] proposes a gating mechanism
Figure 3: Illustration of the grid-based interaction encoding modules. (a) Occupancy Pooling: each cell indicates the presence of a neighbour (b) Our proposed Directional Pooling: each cell contains the relative velocity of the neighbour with respect to the primary pedestrian. (c) Social Pooling: each cell contains the LSTM hidden-state of the neighbour (d) The constructed grid tensors are passed through an MLP-based neural network to obtain the interaction vector to select the hidden-state features to consider for interaction adaptively. The attention mechanism can be applied multiple times to model higher-order spatial interactions.

Gupta et al. [52] proposed to aggregate the interaction information by applying a symmetric Max-Pooling function on the LSTM hidden-states of the neighbouring pedestrians. Non-grid based methods do not contain an implicit notion of the spatial position of neighbours with respect to the primary pedestrian. This problem is usually tackled by additionally concatenating an embedded representation of the relative positions of the surrounding pedestrians like in [52]:

\[
\begin{align*}
    \mathbf{r}_{ji}^t &= \phi_1(x_j^t - x_i^t; \mathbf{W}_r), \\
    \mathbf{h}_{emb_j}^t &= \phi_2(h_j^t; \mathbf{W}_h), \\
    \mathbf{hr}_{ji}^t &= \phi_3([\mathbf{r}_{ji}^t; \mathbf{h}_{emb_j}^t]; \mathbf{W}_{rh}), \\
    \mathbf{p}_i^t &= \text{MaxPool}(\mathbf{hr}_{1i}^t, \mathbf{hr}_{2i}^t, \ldots, \mathbf{hr}_{ni}^t),
\end{align*}
\]

where \(\phi_1, \phi_2, \phi_3\) are MLP and the embedding weights \(\mathbf{W}_r, \mathbf{W}_h, \mathbf{W}_{rh}\) are learned. We denote this architecture by [MaxPool-MLP].

The aggregating mechanisms mentioned above, namely attention and max-pooling, merge the
information of the neighbours resulting in the loss of their identity. We propose an additional design to compare against the above-discussed methods, maintaining the uniqueness of each pedestrian: we concatenate the embeddings of the relative position of the neighbours.

\[
r_{ji}^t = \phi_1(x_j^t - x_i^t; W_r),
\]

\[
p_t^i = \text{Concat}(r_{i1}^t, r_{i2}^t, \ldots, r_{in}^t),
\]

where \(\phi_1\) is an MLP and the embedding weights \(W_r, W_p\) are learned. We denote this architecture by [Concat-MLP]. The architecture is illustrated in Fig ???. The issue with our proposed baseline is handling a variable number of pedestrians in a scene, i.e., the concatenated vector is required to have a fixed length. To tackle this, we investigate the performance of this scheme by filtering \(n\) neighbours based on a defined criterion, (e.g., euclidean distance). The different aggregation strategies are illustrated visually in Fig 4.

Figure 4: Illustration of the non-grid based encoding modules to obtain the interaction vector (pooled). The challenge lies in handling a variable number of neighbours and aggregating their state information to construct the interaction vector (a) Neighbour information is aggregated via attention mechanism (b) Neighbour information is aggregated utilizing a symmetric function (c) Neighbour information is aggregated via concatenation.

In the interaction modules described till now, each neighbouring state has been encoded and merged using an MLP. [55, 77, 64] propose to utilize an RNN-based modules for these tasks as shown in Fig 5. [77, 64] use an RNN architecture to update the neighbouring state information recurrently.
The authors define ‘spatial edgeRNNs’ to model the dynamics of human-human interactions. Each connection of a primary pedestrian to his neighbour is a different spatial edgeRNN. The relative position of each neighbour is passed to the RNN at each time-step. An attention mechanism is then implemented to weigh the different spatial edges at each time-step as follows:

\[ r_{ji}^t = \phi_1(x_j^t - x_i^t; W_r), \]  
\[ e_{ji}^t = RNN(e_{ji}^{t-1}, r_{ji}^t; W_{RNN}), \]  
\[ p_i^t = \sum_j a_{ji}^t \cdot e_{ji}^t, \]  

where \( \phi_1 \) is an MLP and the embedding weights \( W_r, W_{RNN} \) are learned. The weights \( a_{ji}^t \) are derived using an attention mechanism. We denote this architecture by [Att-LSTM].

Similarly, Ivanovic et al. [55] defines an LSTM-based ‘edge encoder’ connecting the primary pedestrian to the rest of the pedestrians in the scene. At each time-step, the states of the neighbouring pedestrian are summed and passed as input to the ‘edge encoder’ to handle variable number of pedestrians. In other words, the authors utilize an LSTM to provide a representation of the aggregated vector.

\[ e_i^t = [s^{t_i}; \sum_{j \in N(i)} s^{t_j}], \]  
\[ p_i^t = LSTM([p_i^{t-1}; e_i^t; W_{EE}]), \]  

where \( W_{EE} \) denote the LSTM weights and \( s_i^t \) signifies the state of pedestrian \( i \) at time-step \( t \). We denote this architecture by [SumPool-LSTM].

We argue that encoding the aggregated vector using LSTMs offers the advantage of modelling higher-order interactions in the temporal domain. In other words, the interaction module learns how the interaction representations evolve over time. We now propose our non-grid based interaction module called DirectConcat combining the strengths of LSTM-based interaction modelling and aggregation through concatenation. Mathematically, our proposed module has the following recurrence:

\[ r_{ji}^t = \phi_1([x_j^t - x_i^t, v_j^t - v_i^t]; W_r), \]  
\[ e_i^t = Concat(r_{1i}^t, r_{2i}^t, \ldots, r_{ni}^t), \]  
\[ h_i^t = LSTM(h_i^{t-1}, e_i^t; W_{EE}), \]  
\[ p_i^t = \phi_2(h_i^t; W_p), \]  

where \( \phi_1, \phi_2 \) are MLP and the weights \( W_r, W_p \) and \( W_{EE} \) are learned. In our design, we augment the relative velocity of the neighbours with their relative positions. We will demonstrate in the experimental section that this step greatly boosts the performance metrics. We denote this architecture by [Concat-LSTM]. The different LSTM-encoding based interaction modules are illustrated in Fig 5.

### 3.3 Forecasting Model

We now describe the rest of the components of the forecasting model. To claim that a particular design of the interaction module is superior, it is essential to keep the rest of the forecasting model components constant. Only then we can be sure that it was the interaction module design that boosted performance, and not one of the extra added components. We choose the time-sequence
encoder to be an LSTM due to its capability to handle varying input length and capture long-term dependencies. Moreover, most works have LSTMs as their base motion-encoding architecture.

The rest of the architecture we describe now is identical for all the methods described in the previous section. The state of person $i$ at time-step $t$, $s_t^i$, is embedded using a single layer MLP to get the state embedding $e_t^i$. We represent each person’s state using his/her velocity, as switching the input representation from absolute coordinates to velocities increases the generalization power of sequence encoder. We obtain the interaction vector $p_t^i$ of person $i$ from the interaction encoder. We concatenate the interaction vector with the velocity embedding and provide the resultant vector as input to the sequence-encoding module. Mathematically, we obtain the following recurrence:

$$e_t^i = \phi(v_t^i; W_{emb}),$$

$$h_t^i = \text{LSTM}(h_{t-1}^i, [e_t^i; p_t^i]; W_{encoder}),$$

where $\phi$ is the embedding function, $W_{emb}, W_{encoder}$ are the weights to be learned. The weights are shared between all persons in the scene.

The hidden-state of the LSTM at time-step $t$ of pedestrian $i$ is then used to predict the distribution of the velocity at time-step $t+1$. Similar to Graves [78], we output a bivariate Gaussian distribution parametrized by the mean $\mu_t^{i+1} = (\mu_x, \mu_y)_t^{i+1}$, standard deviation $\sigma_t^{i+1} = (\sigma_x, \sigma_y)_t^{i+1}$ and correlation coefficient $\rho_t^{i+1}$:

$$[\mu_t^i, \sigma_t^i, \rho_t^i] = \phi_{dec}(h_{t-1}^i, W_{dec}),$$

where $\phi_{dec}$ is modelled using an MLP and $W_{dec}$ is learned.

**Training:** All the parameters of the forecasting model are learned by minimizing the negative log-likelihood (NLL) loss:

$$L(w) = - \sum_{t=T_{obs}+1}^{T_{pred}} \log(P(v_t^i|\mu_t^i, \sigma_t^i, \rho_t^i)).$$

Contrary to the general practice of training the model by minimizing the NLL loss for all the trajectories in the training dataset, we minimize the loss for only the primary pedestrian (defined
in the next section) in each scene of the training dataset. We will demonstrate how this training procedure helps the model better capture social interactions in the experimental section.

**Testing:** During test time, till time-step $T_{\text{obs}}$, we provide the ground truth position of all the pedestrians as input to the forecasting model. From time $T_{\text{obs}+1}$ to $T_{\text{pred}}$, we use the predicted position (derived from the predicted velocity) of each pedestrian as input to the forecasting model and predict the future trajectories of all the pedestrians.

4 TrajNet++: A Trajectory Forecasting Benchmark

In this section, we present TrajNet++, our interaction-centric human trajectory forecasting benchmark. To demonstrate the efficacy of a trajectory forecasting model, the standard practice is to evaluate these models against baselines on a standard benchmark. However, current methods have been evaluated on different subsets of available data without proper sampling of scenes in which social interactions occur. In other words, a data-driven method cannot learn to model agent-agent interactions if the benchmark comprises primarily of scenes where the agents are static or move linearly. Therefore, our benchmark comprises largely of scenes where social interactions occur. To this extent, we propose the following trajectory categorization hierarchy.

![Figure 6: Our proposed hierarchy for Trajectory Categorization. Using our defined trajectory categorization, we construct the TrajNet++ benchmark by sampling trajectories corresponding largely to ‘Type III: Interacting’ category.](image)

4.1 Trajectory Categorization

We provide a detailed trajectory categorization (Fig 6). This detailed categorization helps us not only to better sample trajectories for TrajNet++ dataset but also glean insights into the model performance in diverse scenarios, i.e., to verify whether the model captures all the different kinds of interactions.
To aid our categorization, we introduce the notion of a primary pedestrian as a reference pedestrian with respect to which we categorize scenes. Each scene has a primary pedestrian whose motion we want to forecast. We refer to the other pedestrians in the scene as neighbouring pedestrians.

We explain in detail our proposed hierarchy for trajectory categorization (Fig 6). We also provide example scenarios for the same in Fig 7:

- **Static (Type I):** If the euclidean displacement of the primary pedestrian in the scene is less than a specific threshold.

- **Linear (Type II):** If the trajectory of the primary pedestrian can be correctly forecasted with the help of an Extended Kalman Filter (EKF). A trajectory is said to be correctly forecasted by EKF if the FDE between the ground truth trajectory and forecasted trajectory is less than a specific threshold.

The rest of the scenes are classified as ‘Non-Linear’. We further divide non-linear scenes into Interacting (Type III) and Non-Interacting (Type IV).

- **Interacting (Type III):** These correspond to scenes where the primary trajectory undergoes social interactions. For a detailed categorization coherent with commonly observed social interactions, we divide interacting trajectories into the following sub-categories (shown in Fig 8).
  - **Leader Follower [LF] (Type IIIa):** Leader follower phenomenon refers to the tendency to follow pedestrians going in relatively the same direction. The follower tends to regulate his/her speed and direction according to the leader. If the primary pedestrian is a follower, we categorize the scene as Leader Follower.
  - **Collision Avoidance [CA] (Type IIIb):** Collision avoidance phenomenon refers to the tendency to avoid pedestrians coming from the opposite direction. We categorize the scene as Collision avoidance if the primary pedestrian to be involved in collision avoidance.
  - **Group (Type IIIc):** The primary pedestrian is said to be a part of a group if he/she maintains a close and roughly constant distance with at least one neighbour on his/her side during prediction.
  - **Other Interactions [Others] (Type IIId):** Trajectories where the primary pedestrian undergoes social interactions other than LF, CA and Group. We define social interaction as follows: We look at an angular region in front of the primary pedestrian. If any neighbouring pedestrian is present in the defined region at any time-instant during prediction, the scene is classified as having the presence of social interactions.

- **Non-Interacting (Type IV):** If a trajectory of the primary pedestrian is non-linear and undergoes no social interactions during prediction.

Using our defined trajectory categorization, we construct the TrajNet++ benchmark by sampling trajectories corresponding mainly to the Type III category. Moreover, having many Type I scenes in a dataset can hamper the training of the model and result in misleading evaluation. Therefore, we remove such samples in the construction of our benchmark. A few examples of our categorization in the real world are displayed in Fig 9. In addition to comprising well-sampled trajectories, TrajNet++ provides an extensive evaluation system to understand model performance better.

### 4.2 Evaluation Metrics

**Unimodal Evaluation:** Unimodal evaluation refers to the evaluation of models that propose a single future mode for a given past observation. The most commonly used metrics of human trajectory
forecasting in the unimodal setting are Average Displacement Error (ADE) and Final Displacement Error (FDE) defined as follows:

1. **Average Displacement Error (ADE)**: Average $L_2$ distance between ground truth and model prediction overall predicted time steps.

2. **Final Displacement Error (FDE)**: The distance between the predicted final destination and the ground truth final destination at the end of the prediction period $T_{pred}$.

These metrics essentially define different distance measures between the forecasted trajectory and the ground truth trajectory. With respect to our task, one of the most important aspects of human behavior in crowded spaces is collision avoidance. To ensure that models forecast feasible collision-free trajectories, we propose two new collision-based metrics in our framework (see Fig 10):

3. **Collision I - Prediction collision (Col-I)**: This metric calculates the percentage of collision between the primary pedestrian and the neighbors in the forecasted future scene. This metric indicates whether the predicted model trajectories collide, i.e., whether the model learns the notion of collision avoidance.

4. **Collision II - Groundtruth collision (Col-II)**: This metric calculates the percentage of collision between the primary pedestrian’s prediction and the neighbors in the groundtruth future scene.

We want to stress further the importance of the collision metrics in the unimodal setup. As mentioned earlier, human motion is multimodal. A model may forecast a physically-feasible future, which is different from the actual ground truth. Such a physically-feasible prediction can result in
a large ADE/FDE, which can be misleading. Our Col-I metric can help overcome this limitation of ADE/FDE metrics and provides a solution to measure ’physical feasibility’ of a prediction (aversion to a collision in this case). Col-II metric indicates whether the model understood the intention of the neighbours and predicted the desired trajectory mode indicated by fewer collisions with neighbours in ground truth. We believe our proposed collision metrics are an important step towards capturing the understanding of the model of human social etiquette in crowds.

**Multimodal Evaluation:** For models performing multimodal forecasting, *i.e.*, outputting a future trajectory distribution, we provide the following metrics to measure their performance:

5. **Top-k ADE:** Given $k$ output predictions for an observed scene, this metric calculate the ADE of the prediction closest to the groundtruth trajectory, similar in spirit to Variety Loss [52].

6. **Top-k FDE:** Given $k$ output predictions for an observed scene, this metric calculate the FDE of the prediction closest to the groundtruth trajectory, similar in spirit to Variety Loss [52].

For the Top-k metrics, we propose $k$ be small (3 as opposed to 20) as a model outputting
uniformly-spaced predictions, irrespective of the input observation, can result in a much lower Top-20 ADE/FDE.

7. **Average NLL**: This metric was proposed by Boris et. al. [55]. At each prediction step, the authors utilize a Kernel Density Estimate (KDE) [79]. From these estimates, the log-likelihood of ground truth trajectory is computed at each time step and is subsequently averaged over the prediction horizon. This metric provides a good indication of the probability of the ground truth trajectory in the model prediction distribution.

### 4.3 Datasets

We now describe the datasets used in the TrajNet++ benchmark. Since the focus of this work is to tackle agent-agent interactions in crowded settings, we explicitly select datasets where scene constraints do not play a significant role in determining the future trajectory. For each real world dataset, we utilize only the information regarding the pedestrian locations from the respective annotations files, i.e., spatial coordinates of each pedestrian at each time frame. Furthermore, we provide no information regarding the destination of each pedestrian or structure of the scene. Our goal is to forecast only the 2D spatial coordinates for each pedestrian.

#### 4.3.1 TrajNet++ Real Datasets

- **ETH**: ETH dataset provides for two locations: Univ and Hotel, where pedestrian trajectories are observed. This dataset contains a total of approximately 750 pedestrians exhibiting complex interactions (Pellegrini et. al. [25]). The dataset is one of the widely used benchmarks for pedestrian trajectory forecasting. It captures diverse real-world social interactions like leader follower, collision avoidance, and group forming and dispersing.

- **UCY**: UCY dataset consists of three scenes: Zara01, Zara02 and Uni, with a total of approximately 780 pedestrians (Lerner et. al. [2]). This dataset, in addition to the ETH dataset, is widely used as benchmarks for pedestrian trajectory forecasting, offering a wide range of non-linear trajectories arising out of social interactions.

- **WildTrack**: This is a recently proposed benchmark [80] for pedestrian detection and tracking captured in front of ETH Zurich. Since the dataset comprises of diverse crowd interactions in the wild, we utilize it for our task of trajectory forecasting.
- **L-CAS:** This is a recently proposed benchmark for pedestrian trajectory forecasting (Sun et. al. [72]). The dataset, comprising over 900 pedestrian tracks, comprises diverse social interactions that are captured within indoor environments. Some of the challenges scenarios in this dataset include people pushing trolleys and running children.

- **CFF:** This is a large-scale dataset of 42 million trajectories extracted from real-world train stations [36]. It is one of the biggest datasets that capture agent-agent interactions in crowded settings during peak travel times. Due to the high density of people, we observe higher instances of social interactions like leader-follower in this dataset.

### 4.3.2 TrajNet++ Synthetic Dataset

Interaction-centric synthetic datasets can provide the necessary controlled environment to compare the performances of different model components. We provide synthetic data in TrajNet++ to evaluate the performance of a model under controlled interaction scenarios.

**Simulator Selection:** It is a necessary condition that the interactions in the synthetic dataset are similar to those in the real world. Empirically, we find that in comparison to Social Force [13], ORCA [29] provides a better similarity to real world human motion with respect to collision avoidance. We choose ORCA parameters, which demonstrate a reaction distance and reaction curvature similar to real data during collision avoidance (Fig 11).

**Dataset Generation:** Given the ORCA parameters, we generated the synthetic dataset using the following procedure: \( n \) pedestrians were initialized at random on a circle of radius \( r \) keeping a certain minimum distance \( d_{\text{min}} \) between their initial positions. The goal of each pedestrian was defined to be the point diametrically opposite to the initial position on the circle. For the TrajNet++ synthetic dataset: We ran different simulations with \( n \) chosen randomly from the range \([4, 7)\) on a circle of radius \( r = 10 \) meters and \( d_{\text{min}} = 2 \) meters.

Given the generated trajectories, we selected only those scenes which belonged to the Type III: ‘Interacting’ category. The ORCA simulator demonstrates sensitive dependence on initial conditions. This can be attributed to the fact that all the agents are expected to collide near the same point (at the origin), so slight perturbations can greatly affect the future trajectory of all agents. Sensitivity to initial conditions, also known as the Butterfly Effect, is a well-studied phenomenon of Chaos theory [81]. To identify such sensitive initial conditions, the practice which is often followed is to perturb the initial conditions with arbitrary small noise and observe the effect. Along similar lines, we propose an additional step to filter out such ‘sensitive’ scenes: in each scene, we perturb all trajectories at the point of observation with a small uniform noise \( \text{noise} \in U[−\text{noise}\_\text{thresh}, \text{noise}\_\text{thresh}] \), and forecast the future trajectories using ORCA. We perform this procedure \( k \) times. If any of the \( k \) ORCA predictions have a significant ADE compared to the ground truth, we filter out such scenes. Fig 12 visualizes the sample outputs of our filtering process (with \( \text{noise}\_\text{thresh} = 0.01, k = 20, n = 5 \)). We passed the selected scenes through a final additional filter that identifies sharp unrealistic turns in trajectories. Fig 13 illustrates a few sample scenes in our TrajNet++ synthetic dataset.
5 Experiments

In this section, we perform extensive experimentation on both TrajNet++ synthetic and real-world datasets to understand the efficacy of interaction module designs for human trajectory forecasting. Moreover, we demonstrate how our proposed metrics help to provide a complete picture of model performance. Our proposed simple yet powerful method outperforms competitive baselines on both real-world and synthetic datasets in terms of forecasting physically-acceptable trajectories.

5.1 Implementation Details

The velocity of each pedestrian is embedded into a 64-dimensional vector. The dimension of the interaction vector is fixed to 256. The dimension of the goal direction vector is fixed to 64. For grid-based interaction encoding, we construct a grid of size $16 \times 16$ with a resolution of 0.6 meters. The dimension of the hidden state of both the encoder LSTM and decoder LSTM is 128. As mentioned earlier, each pedestrian has his/her own encoder and decoder. The batch size is fixed to 8. We train using ADAM optimizer [82] with a learning rate of 1e-3. We perform interaction encoding at every time-step.

Our proposed trajectory categorization allows one to train the model focusing on the non-linear interacting trajectories. Since each scene is categorized with respect to the primary pedestrian, during training, the loss is calculated only with respect to the prediction of the primary trajectory.
<table>
<thead>
<tr>
<th>Name</th>
<th>Total</th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>LF</th>
<th>CA</th>
<th>Grp</th>
<th>Oth</th>
<th>IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>54513</td>
<td>0</td>
<td>0</td>
<td>54513</td>
<td>495</td>
<td>7183</td>
<td>0</td>
<td>46853</td>
<td>0</td>
</tr>
<tr>
<td>BIWI Hotel.</td>
<td>238</td>
<td>22</td>
<td>91</td>
<td>109</td>
<td>24</td>
<td>29</td>
<td>41</td>
<td>39</td>
<td>16</td>
</tr>
<tr>
<td>Zara01.</td>
<td>1017</td>
<td>4</td>
<td>184</td>
<td>542</td>
<td>109</td>
<td>160</td>
<td>231</td>
<td>134</td>
<td>287</td>
</tr>
<tr>
<td>Zara03.</td>
<td>960</td>
<td>16</td>
<td>152</td>
<td>634</td>
<td>108</td>
<td>222</td>
<td>232</td>
<td>200</td>
<td>158</td>
</tr>
<tr>
<td>Stud01.</td>
<td>5719</td>
<td>94</td>
<td>605</td>
<td>4772</td>
<td>712</td>
<td>2030</td>
<td>1862</td>
<td>1364</td>
<td>248</td>
</tr>
<tr>
<td>Stud03.</td>
<td>4302</td>
<td>46</td>
<td>350</td>
<td>3598</td>
<td>537</td>
<td>1508</td>
<td>1469</td>
<td>1118</td>
<td>308</td>
</tr>
<tr>
<td>WildTr.</td>
<td>1098</td>
<td>115</td>
<td>43</td>
<td>668</td>
<td>43</td>
<td>75</td>
<td>145</td>
<td>422</td>
<td>272</td>
</tr>
<tr>
<td>L-CAS</td>
<td>874</td>
<td>180</td>
<td>87</td>
<td>310</td>
<td>10</td>
<td>85</td>
<td>12</td>
<td>210</td>
<td>297</td>
</tr>
<tr>
<td>CFF06</td>
<td>20972</td>
<td>22</td>
<td>4267</td>
<td>15384</td>
<td>5194</td>
<td>8239</td>
<td>267</td>
<td>4841</td>
<td>1299</td>
</tr>
<tr>
<td>CFF07</td>
<td>21145</td>
<td>16</td>
<td>4251</td>
<td>15635</td>
<td>5352</td>
<td>8361</td>
<td>252</td>
<td>4991</td>
<td>1243</td>
</tr>
<tr>
<td>CFF08</td>
<td>19840</td>
<td>13</td>
<td>3950</td>
<td>14619</td>
<td>4805</td>
<td>7521</td>
<td>216</td>
<td>4881</td>
<td>1258</td>
</tr>
<tr>
<td>CFF09</td>
<td>10548</td>
<td>10</td>
<td>2579</td>
<td>6717</td>
<td>1733</td>
<td>3010</td>
<td>203</td>
<td>2742</td>
<td>1242</td>
</tr>
<tr>
<td>CFF12</td>
<td>20962</td>
<td>11</td>
<td>4242</td>
<td>15445</td>
<td>5309</td>
<td>8294</td>
<td>268</td>
<td>4990</td>
<td>1264</td>
</tr>
<tr>
<td>CFF13</td>
<td>19792</td>
<td>17</td>
<td>3746</td>
<td>14679</td>
<td>4768</td>
<td>7519</td>
<td>263</td>
<td>4898</td>
<td>1350</td>
</tr>
<tr>
<td>CFF14</td>
<td>20509</td>
<td>12</td>
<td>4041</td>
<td>15135</td>
<td>5099</td>
<td>7893</td>
<td>274</td>
<td>4927</td>
<td>1321</td>
</tr>
<tr>
<td>CFF15</td>
<td>19866</td>
<td>15</td>
<td>3824</td>
<td>14741</td>
<td>4815</td>
<td>7563</td>
<td>277</td>
<td>4984</td>
<td>1286</td>
</tr>
<tr>
<td>CFF16</td>
<td>10044</td>
<td>5</td>
<td>2523</td>
<td>6258</td>
<td>1626</td>
<td>2689</td>
<td>249</td>
<td>2566</td>
<td>1258</td>
</tr>
<tr>
<td>CFF17</td>
<td>9250</td>
<td>8</td>
<td>2458</td>
<td>5694</td>
<td>1508</td>
<td>2694</td>
<td>198</td>
<td>2227</td>
<td>1090</td>
</tr>
<tr>
<td>CFF18</td>
<td>19437</td>
<td>13</td>
<td>4067</td>
<td>14042</td>
<td>4744</td>
<td>7211</td>
<td>248</td>
<td>4669</td>
<td>1315</td>
</tr>
<tr>
<td>Total</td>
<td>250k</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1: TrajNet++: Statistics of the Training Split.

<table>
<thead>
<tr>
<th>Name</th>
<th>N</th>
<th>I</th>
<th>II</th>
<th>III</th>
<th>LF</th>
<th>CA</th>
<th>Grp</th>
<th>Oth</th>
<th>IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>3842</td>
<td>0</td>
<td>0</td>
<td>3842</td>
<td>73</td>
<td>632</td>
<td>0</td>
<td>3142</td>
<td>0</td>
</tr>
<tr>
<td>BIWI ETH.</td>
<td>1139</td>
<td>11</td>
<td>227</td>
<td>640</td>
<td>189</td>
<td>153</td>
<td>172</td>
<td>244</td>
<td>261</td>
</tr>
<tr>
<td>UNI.</td>
<td>244</td>
<td>0</td>
<td>50</td>
<td>100</td>
<td>7</td>
<td>11</td>
<td>38</td>
<td>49</td>
<td>94</td>
</tr>
<tr>
<td>Zara02.</td>
<td>1881</td>
<td>80</td>
<td>496</td>
<td>998</td>
<td>192</td>
<td>355</td>
<td>452</td>
<td>270</td>
<td>307</td>
</tr>
<tr>
<td>Total</td>
<td>7106</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: TrajNet++: Statistics of the Testing Split.
Figure 12: Illustration of our filtering procedure to generate Trajnet++ Synthetic dataset. Given a
ground-truth scene (in black) generated by ORCA, we perturb the positions of agents and forecast
the future with ORCA, iteratively, to obtain a distribution (in blue). This procedure helps us identify
the sensitive scenes and consequently remove them.

Figure 13: Illustration of our synthetically generated samples using the calibrated ORCA parameters.

5.2 Interaction Models

We consider the following grid based baselines:

- Vanilla [Van]: No interaction encoding takes place.
- Occupancy Pooling [O-Grid] [16, 45]: Grid based interaction model where each cell indicates
  the presence of the neighbour
- Social Pooling [S-Grid] [16, 45, 47, 51]: Grid based interaction model where each cell comprises
  of the hidden-state of the neighbour
- Directional Pooling [D-Grid] [Ours]: Grid based interaction model where each cell comprises
  of the relative velocity of the neighbour.

The grid is then passed through a two layer MLP to get the interaction vector. We experimented
with various feedforward architectures and a two layer MLP performs the best for encoding the grid.

We consider the following are the non-grid based baselines:

- Positional Concatenation MLP [O-Concat-MLP]: The relative position of each neighbour
  is embedded and concatenated with each other. The resulting vector is then passed through
  an MLP to get the interaction vector. We consider the top-$n$ neighbours based on euclidean
distance ($n$ being a hyperparameter).
• Directional Concatenation MLP [D-Concat-MLP]: The relative position and relative velocity of each neighbour are embedded and concatenated with each other. The resulting vector is then passed through an MLP to get the interaction vector.

• Directional MaxPool MLP [D-MaxPool-MLP]: The relative position and relative velocity of each neighbour are embedded and max-pooled. The resulting vector is then passed through an MLP to get the interaction vector.

• Directional Attention MLP [D-Attn-MLP]: The relative position and relative velocity of each neighbour are embedded and passed through a self-attention block. The resulting vector is then passed through an MLP to get the interaction vector.

• Hidden State MaxPool MLP [H-MaxPool-MLP] [52]: The relative position and hidden-state of each neighbour are embedded and max-pooled. The resulting vector is then passed through an MLP to get the interaction vector.

• Social Attention MLP [H-Att-MLP] (similar to [61, 58, 57, 53, 62]): The relative position, relative velocity and hidden-state of each neighbour are embedded. The embeddings are passed through a self-attention block [68]. The resulting vector is then passed through an MLP to get the interaction vector.

• Sum Pool LSTM [D-SumPool-LSTM] [55]: The absolute position and velocity of each neighbour is summed and concatenated to that of primary pedestrian. The resulting vector is then passed through an LSTM to get the interaction vector.

• Social Attention LSTM [O-Att-LSTM] [77]: The relative position of each neighbour are encoded through LSTMs. The hidden-states of LSTMs are passed through a self-attention block to get the interaction vector.

• Directional Concatenation LSTM [D-Concat-LSTM] (Ours): The relative position and relative velocity of each neighbour are embedded and concatenated. The resulting vector is then passed through an LSTM to get the interaction vector.

Please note that for an objective comparison between interaction modules, we fix the base sequence encoder architecture to be an LSTM. Data augmentation is another technique that can help increase accuracy, which can get wrongly attributed to the interaction encoder. We use rotation augmentation as the data augmentation technique to regularize all the models.

5.3 Synthetic Experiments

Synthetic datasets are the ideal testbeds to validate model performances in noise-free controlled scenarios explicitly. The synthetic dataset is generated using the procedure described in Section 4.3.2. For the synthetic dataset, since ORCA has access to the goals to the pedestrian, we embed the goal-direction and concatenate it to the velocity embedding (see Eq 23). We utilize synthetic datasets to validate the efficacy of various interaction modules in a controlled setup.

Unimodal Evaluation: Table 3 quantifies the performance of the different designs of interaction modules published in the literature on TrajNet++ synthetic dataset. Among the grid-based models, our proposed D-Grid outperforms O-Grid, especially in terms of Col-I, i.e., D-Grid learns better to avoid collisions. It is interesting to note that even though the motion encoder (LSTM) has the potential to infer the relative velocity of neighbours over time, there is significant difference in performance when we provide relative velocity of the neighbours as input to the pooling grid.
### Table 3: Baseline models compared according to their interaction encoder designs when forecasting 12 future time-steps, given the previous 9 time-steps on TrajNet++ synthetic dataset. The interaction model design is categorized with respect to neighbour information aggregation (Merge) strategy and type of the encoder architecture (Enc.). Errors reported are ADE / FDE in meters, Col-I / Col-II in % as defined in Section 4.2.

<table>
<thead>
<tr>
<th>Model (Acronym)</th>
<th>Merge</th>
<th>Enc.</th>
<th>ADE</th>
<th>FDE</th>
<th>Col-I</th>
<th>Col-II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vanilla</td>
<td></td>
<td></td>
<td>0.32</td>
<td>0.62</td>
<td>19.2</td>
<td>7.1</td>
</tr>
<tr>
<td>O-LSTM [16] (O-Grid)</td>
<td>Grid</td>
<td>MLP</td>
<td>0.27</td>
<td>0.53</td>
<td>10.1</td>
<td>5.9</td>
</tr>
<tr>
<td>S-LSTM [16] (S-Grid)</td>
<td>Grid</td>
<td>MLP</td>
<td>0.24</td>
<td>0.50</td>
<td>2.0</td>
<td>4.4</td>
</tr>
<tr>
<td>D-LSTM (Ours) (D-Grid)</td>
<td>Grid</td>
<td>MLP</td>
<td>0.25</td>
<td>0.50</td>
<td>2.4</td>
<td>4.8</td>
</tr>
<tr>
<td>S-GAN [52] (H-MaxPool-MLP)</td>
<td>MaxPool</td>
<td>MLP</td>
<td>0.27</td>
<td>0.52</td>
<td>6.8</td>
<td>5.2</td>
</tr>
<tr>
<td>S-BiGAT [61] (H-Att-MLP)</td>
<td>Attention</td>
<td>MLP</td>
<td>0.25</td>
<td>0.50</td>
<td>2.5</td>
<td>5.8</td>
</tr>
<tr>
<td>DirectConcat-MLP (Ours) (D-Concat-MLP)</td>
<td>Concat</td>
<td>MLP</td>
<td>0.25</td>
<td>0.50</td>
<td>1.3</td>
<td>5.6</td>
</tr>
<tr>
<td>Trajectron [55] (D-SumPool-LSTM)</td>
<td>SumPool</td>
<td>LSTM</td>
<td>0.29</td>
<td>0.57</td>
<td>14.0</td>
<td>6.5</td>
</tr>
<tr>
<td>Social Attention [77] (O-Att-LSTM)</td>
<td>Attention</td>
<td>LSTM</td>
<td>0.24</td>
<td>0.48</td>
<td>1.0</td>
<td>5.1</td>
</tr>
<tr>
<td>DirectConcat (Ours) (D-Concat-LSTM)</td>
<td>Concat</td>
<td>LSTM</td>
<td>0.24</td>
<td>0.48</td>
<td>0.7</td>
<td>5.2</td>
</tr>
</tbody>
</table>

Further, **D-Grid** performs at par with **S-Grid**, along with being computationally less expensive, thereby rendering it more suitable for real-world deployment tasks.

Among the non-grid based models, we focus on the information aggregation strategies for MLP-based encoders. It is evident that our baseline **D-Concat-MLP** of concatenating the neighbourhood information performs at par, if not better than weighting-based and max-pooling-based alternatives. This performance can be attributed to the fact that the interaction vector obtained using **D-Concat-MLP** preserves the identity of the surrounding neighbours.

Among the non-grid LSTM-based designs, the drop in performance of **D-SumPool-LSTM** module [55] can be attributed to (1) sum pooling which loses the individual identity of the neighbours and (2) encoding of absolute neighbour coordinates instead of relative coordinates: relational coordinates of agents to the target agent are easier to train than exact coordinates of agents. On the other hand, the relatively higher Col-I metric for **O-Att-LSTM** [77] can be attributed to its design as that it does not account for the relative velocity between agents. Finally, we notice that encoding the interaction information using LSTM, **D-Concat-LSTM**, improves performance over its MLP-based counterpart **D-Concat-MLP**. MLP encoders, due to their non-recurrent nature, have no information regarding the representation at the previous step. We argue that LSTM can capture the evolution of interaction and therefore provide a better neighbourhood representation as the scene evolves. Moreover, having a separate LSTM for encoding interactions can reduce the load on the sequence-encoding LSTM that monitors past motion as well.
5.4 Real World Experiments

We now discuss the performances of forecasting models on TrajNet++ real world data. With the help of our defined trajectory categorization, we construct the TrajNet++ real-world benchmark by sampling trajectories corresponding mainly to ‘Interacting’ category. Moreover, many real-world trajectories are static (people in groups standing and talking to each other). Having many static scenes in a dataset can provide misleading results during evaluation. Therefore, we remove such samples from our benchmark. Having gained insights on the performance of different modules on controlled synthetic data, we explore the question, ‘Do these findings generalize to the real world datasets comprising much more diverse interactions?’

Additional Baselines: We compare the data-driven baselines with the classical trajectory forecasting models, namely, Extended Kalman Filter (EKF), Social Force [13], and ORCA [29]. Both Social Force and ORCA models forecast the future trajectory based on the assumption that each pedestrian has an intended direction of motion and a preferred velocity, as a result of his/her intended goal. However, estimating this goal from the observed trajectory required us to know the person’s intention, which we cannot access. Since we focus on short-term human trajectory forecasting, we interpolate the observed trajectory to identify the virtual goals for each agent.

Unimodal Evaluation: Table 4 provides an extensive evaluation of existing baselines on the Type III ‘Interacting’ trajectories of the TrajNet++ real dataset. The first part of the table compares the classical methods. The high error of EKF can be attributed to the fact that the filter does not model social interactions. The classical methods of Social Force and ORCA are calibrated to fit the TrajNet++ training data by minimizing ADE/FDE metrics, along with the constraint that collisions should be avoided.

The second part of Table 4 compares the performance of the various NN-based interaction encoder designs. The interaction-based NN models outperform the handcrafted models in terms of the distance-based metrics. Our collision metrics help to differentiate the NN-based model performance in terms of the physical acceptability of predictions. In contrast to synthetic experiments, our proposed D-Grid performs superior to S-Grid in terms of avoiding collisions in the real world. Furthermore, our proposed baseline D-Concat-LSTM, built from simple principles, performs at par, if not better, than the existing non-grid counterparts.

Multimodal Evaluation: TrajNet++ synthetic dataset was generated using ORCA, which simulates the scene following a deterministic unimodal policy. However, human motion in real world...
<table>
<thead>
<tr>
<th>Model (Acronym)</th>
<th>Merge</th>
<th>Enc.</th>
<th>ADE</th>
<th>FDE</th>
<th>Col-I</th>
<th>Col-II</th>
<th>NLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hand-crafted methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kalman Filter</td>
<td>–</td>
<td>–</td>
<td>0.87</td>
<td>1.69</td>
<td>16.20</td>
<td>22.1</td>
<td>–</td>
</tr>
<tr>
<td>Social Force</td>
<td>–</td>
<td>–</td>
<td>0.89</td>
<td>1.53</td>
<td>0.0</td>
<td>13.1</td>
<td>–</td>
</tr>
<tr>
<td>ORCA</td>
<td>–</td>
<td>–</td>
<td><strong>0.68</strong></td>
<td>1.40</td>
<td>0.0</td>
<td>15.0</td>
<td>–</td>
</tr>
<tr>
<td>Top submitted methods*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AMENet [83]</td>
<td>–</td>
<td>–</td>
<td>0.62</td>
<td>1.30</td>
<td>14.1</td>
<td>16.90</td>
<td>–</td>
</tr>
<tr>
<td>AIN [84]</td>
<td>–</td>
<td>–</td>
<td>0.62</td>
<td>1.24</td>
<td>10.7</td>
<td>17.10</td>
<td>–</td>
</tr>
<tr>
<td>Grid based methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vanilla</td>
<td>–</td>
<td>–</td>
<td>0.61</td>
<td>1.31</td>
<td>14.5</td>
<td>16.1</td>
<td>12.29</td>
</tr>
<tr>
<td>O-LSTM [16] (O-Grid)</td>
<td>Grid</td>
<td>MLP</td>
<td>0.56</td>
<td>1.21</td>
<td>11.3</td>
<td>15.6</td>
<td>11.43</td>
</tr>
<tr>
<td>S-LSTM [10] (S-Grid)</td>
<td>Grid</td>
<td>MLP</td>
<td><strong>0.55</strong></td>
<td><strong>1.19</strong></td>
<td>7.8</td>
<td>15.8</td>
<td>10.01</td>
</tr>
<tr>
<td>D-LSTM (Ours) (D-Grid)</td>
<td>Grid</td>
<td>MLP</td>
<td>0.57</td>
<td>1.25</td>
<td><strong>7.3</strong></td>
<td><strong>14.8</strong></td>
<td>11.22</td>
</tr>
<tr>
<td>Non-Grid based methods</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S-GAN [52] (H-MaxPool-MLP)</td>
<td>MaxPool</td>
<td>MLP</td>
<td>0.58</td>
<td>1.26</td>
<td>14.1</td>
<td>16.0</td>
<td>11.93</td>
</tr>
<tr>
<td>S-BigAT [61] (Att-MLP)</td>
<td>Attention</td>
<td>MLP</td>
<td>0.60</td>
<td>1.29</td>
<td>8.3</td>
<td>16.4</td>
<td><strong>9.22</strong></td>
</tr>
<tr>
<td>Trajectron [55] (SumPool-LSTM)</td>
<td>SumPool</td>
<td>LSTM</td>
<td>0.58</td>
<td>1.25</td>
<td>15.6</td>
<td>16.4</td>
<td>12.70</td>
</tr>
<tr>
<td>Social Attention [77] (Att-LSTM)</td>
<td>Attention</td>
<td>LSTM</td>
<td><strong>0.55</strong></td>
<td><strong>1.19</strong></td>
<td>9.8</td>
<td>16.1</td>
<td>10.65</td>
</tr>
<tr>
<td>DirectConcat (Ours) (D-Concat-LSTM)</td>
<td>Concat</td>
<td>LSTM</td>
<td>0.57</td>
<td>1.24</td>
<td><strong>7.4</strong></td>
<td><strong>16.0</strong></td>
<td>9.78</td>
</tr>
</tbody>
</table>

Table 4: Baseline models compared according to their interaction encoder designs (see acronyms) when forecasting 12 future time-steps, given the previous 9 time-steps on TrajNet++ real world dataset. The model design is categorized with respect to the neighbour information mixing (Merge) strategy and the type of encoder architecture (Enc.). Errors reported are ADE / FDE in meters, Col I / Col II in %, NLL in units as defined in Sec 4.2.
is multimodal. Therefore, as described in Sec 4.2, our TrajNet++ framework provides multimodal evaluation metrics in addition to unimodal metrics. For a complete evaluation, we report the performance of various methods trained in multimodal settings using the variety loss defined in [52], on TrajNet++ real dataset. In Table 4, we report the NLL metric that provides an estimate of the probability of the ground truth trajectory in the model prediction distribution. Among the grid-based models, S-Grid performs the best while among the non-grid based models, Att-MLP performs superior. Exploring techniques to output accurate yet diverse multimodal distributions is an avenue for future research.

To summarize, despite claims in literature that specific interaction modules better model interactions, we observe that under identical conditions, all modules perform similar in terms of the distance-based ADE and FDE metrics. There certainly exists room for improvement, and we hope that our benchmark provides the necessary resources to advance the field of trajectory forecasting.

5.5 Ablation Studies

While benchmarking the interaction modules on both synthetic and real datasets, we empirically observed important design choices and training strategies. We highlight them in this subsection through a series of ablation studies. Moreover, we open-source our code for reproducibility. We hope that such practices will help to accelerate the development of interaction modules in future research.

1. **Col-I is an essential evaluation metric**: Table 3 and Table 4 emphasize the importance of our proposed Col-I metric, i.e., the percentage of collision of primary pedestrian with neighbors in the forecasted scene. This metric indicates the ability of the model to learn the social etiquette of collision avoidance. In safety-critical scenarios, it is more important for a model to prevent collisions in comparison to minimizing ADE/FDE. We hope that in future, researchers will incorporate this metric while reporting their model performances on trajectory forecasting datasets.

2. **Embedding relative velocity provides a boost**: Table 5 illustrates the improvement in performance on providing the relative velocity to the interaction modules. A key to the success of the interaction modules is to have informed input features. Having the relative velocity embedding significantly improves the performance of both grid and non-grid based models, especially in learning to avoid collisions. Empirically, one can argue that it is easier to provide the relative velocity to the interaction model compared to relying on the sequence encoder to infer the relative velocity through time.

3. **Concatenation of embeddings are simple yet powerful baselines**: Table 6 illustrates
Table 6: Concatenation of embeddings are simple yet powerful baselines for comparing different neighbour information aggregation strategies. For concatenation, we consider the top-4 neighbours based on euclidean distance.

<table>
<thead>
<tr>
<th>Model</th>
<th>Dataset</th>
<th>Rel. Pos.</th>
<th>Rel. Vel.</th>
<th>Merge</th>
<th>ADE</th>
<th>FDE</th>
<th>Col-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>D-MLP</td>
<td>Synthetic</td>
<td>✓</td>
<td>✓</td>
<td>max-pool</td>
<td>0.28</td>
<td>0.55</td>
<td>14.3</td>
</tr>
<tr>
<td>D-Attn</td>
<td>Synthetic</td>
<td>✓</td>
<td>✓</td>
<td>attn</td>
<td>0.27</td>
<td>0.52</td>
<td>8.1</td>
</tr>
<tr>
<td>D-NN</td>
<td>Synthetic</td>
<td>✓</td>
<td>✓</td>
<td>concat</td>
<td>0.25</td>
<td>0.50</td>
<td>1.3</td>
</tr>
<tr>
<td>D-MLP</td>
<td>Real</td>
<td>✓</td>
<td>✓</td>
<td>max-pool</td>
<td>0.59</td>
<td>1.24</td>
<td>13.6</td>
</tr>
<tr>
<td>D-Attn</td>
<td>Real</td>
<td>✓</td>
<td>✓</td>
<td>attn</td>
<td>0.56</td>
<td>1.23</td>
<td>7.6</td>
</tr>
<tr>
<td>D-NN</td>
<td>Real</td>
<td>✓</td>
<td>✓</td>
<td>concat</td>
<td>0.59</td>
<td>1.26</td>
<td>8.1</td>
</tr>
</tbody>
</table>

Table 7: Our proposed training objective that penalizes only the prediction of the primary pedestrian, instead of penalizing all the pedestrians in the scene, provides superior performance with respect to helping the model learn to avoid collisions.

<table>
<thead>
<tr>
<th>Strategy</th>
<th>Dataset</th>
<th>ADE</th>
<th>FDE</th>
<th>Col-I</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard Training [52, 77, 55]</td>
<td>Synthetic</td>
<td>0.25</td>
<td>0.48</td>
<td>11.9</td>
</tr>
<tr>
<td>Proposed Training</td>
<td>Synthetic</td>
<td>0.24</td>
<td>0.48</td>
<td>2.5</td>
</tr>
<tr>
<td>Standard Training [52, 77, 55]</td>
<td>Real</td>
<td>0.59</td>
<td>1.27</td>
<td>14.8</td>
</tr>
<tr>
<td>Proposed Training</td>
<td>Real</td>
<td>0.56</td>
<td>1.21</td>
<td>11.3</td>
</tr>
</tbody>
</table>

4. A Different Training Objective: We employ a different training objective in comparison to the standard practice to train the forecasting model. As mentioned earlier, we penalize only the primary pedestrian during training. Moreover, during training, we provide the ground truth of neighbouring trajectories during the prediction period and forecast only the primary trajectory. This training scheme is different from the common practice where the neighbouring trajectories are also predicted during training [52, 77, 55]. Table 7 illustrates the effectiveness of our training objective in helping the model to learn collision avoidance better. During test time, we do not provide the ground truth neighbour trajectories.

6 Conclusions

In this work, we tackled the challenge of modelling social interactions between pedestrians in crowds. While modelling social interactions is a central issue in human trajectory forecasting, the literature lacks a definitive comparison between the many existing interaction models on identical grounds. We presented an in-depth analysis of the design of interaction modules proposed in the literature and developed a simple yet powerful method DirectConcat, which serves two advantages: (1) it retains
the uniqueness of neighbouring pedestrians, and (2) the recurrent modelling of interactions helps to better model interactions.

A significant yet missing component in this field is an objective and informative evaluation of these interaction-based methods. To solve this issue, we propose \textit{TrajNet++}: (1) \textit{TrajNet++} is interaction-centric as it largely comprises scenes where interactions take place thanks to our defined trajectory categorization, both in the real world and synthetic settings, (2) \textit{TrajNet++} provides an extensive evaluation system that includes novel collision-based metrics that can help measure the \textit{physical feasibility} of model predictions. The superior quality of \textit{TrajNet++} is highlighted by the improved performance of interaction-based models on real world datasets on all metrics (4 of the top 5 methods on \textit{TrajNet} \cite{85}, an earlier benchmark, do not model social interactions). Further, we demonstrated how our collision-based metrics provide a more concrete picture regarding the model performance.

DirectConcat, our method built from simple principles, outperforms competitive baselines on \textit{TrajNet++} synthetic dataset by benchmarking against several popular interaction module designs in the field. On the real dataset, there is no clear winner amongst all the designs, when compared on equal grounds. There is room for improvement, and we hope that our benchmark facilitates researchers to objectively and easily compare their methods against existing works so that the quality of trajectory forecasting models can keep increasing, allowing us to tackle more challenging scenarios.
References


