DAG-Net: Double Attentive Graph Neural Network for Trajectory Forecasting

Alessio Monti*, Alessia Bertugli*, Simone Calderara and Rita Cucchiara
AImageLab, University of Modena and Reggio Emilia, Modena, Italy
Email: {alessio.monti, alessia.bertugli, simone.calderara, rita.cucchiara}@unimore.it

***Abstract*—Understanding human motion behaviour is a critical task for several possible applications like self-driving cars or social robots, and in general for all those settings where an autonomous agent has to navigate inside a human-centric environment. This is non-trivial because human motion is inherently multi-modal: given a history of human motion paths, there are many plausible ways by which people could move in the future. Additionally, people activities are often driven by goals, e.g. reaching particular locations or interacting with the environment. We address the aforementioned aspects by proposing a new recurrent generative model that considers both single agents' future goals and interactions between different agents. The model exploits a double attention-based graph neural network to collect information about the mutual influences among different agents and to integrate it with data about agents' possible future objectives. Our proposal is general enough to be applied to different scenarios: the model achieves state-of-the-art results in both urban environments and also in sports applications.**

I. Introduction

Trajectory prediction has become an essential component for several applications: self-driving cars and social robots may leverage useful insights about human motion to preventively forecast pedestrian actions and avoid collisions [1], [2], [3], [4], while surveillance systems can benefit from knowing how crowds will move to better monitor huddled environments [5], [6]. There is also a high interest outside the smart cities context, like for example in team sports, where such predictions could give important insights for tactical analysis. However, designing a model to help to predict agents' trajectories is as desirable as difficult: a series of demanding challenges has to be faced.

First of all, the task results particularly tough because human motion is inherently *multi-modal*: when moving, people may follow several plausible trajectories, as there is a rich distribution of potential human behaviours. This means that, given a particular set of past observations, there is no unique correct future since several behaviours could be equally appropriate [7], [8]. In an urban environment, a pedestrian could choose to reach his/her destination following several plausible paths: cut straight to the objective crossing the road, or maybe take a longer walk following physical clues like sidewalks and pedestrian crossings. Similarly, in a basketball match, when an attacking player is running towards an opponent, several modes of behaviour develop: the player may choose to avoid the defender by passing the ball to a teammate, or perhaps achieve the same result by dribbling the adversary directly. Furthermore, each of these possible modes may exhibit a high variance in return: agents can modify step after step some of their features, like speed.

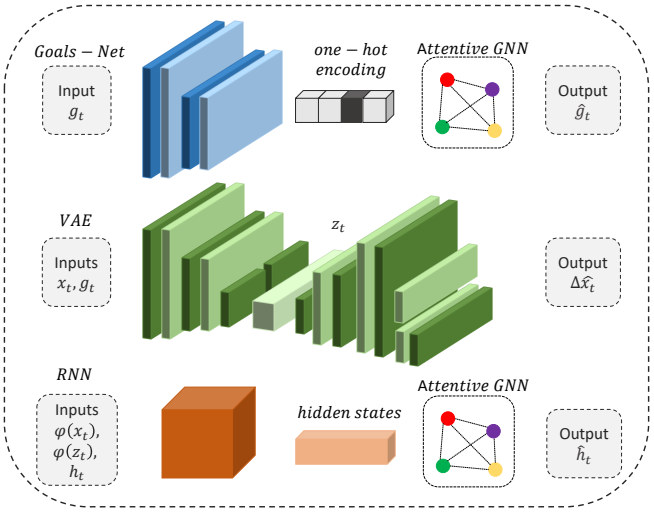


Fig. 1. Scheme of DAG-Net architecture. It is composed of a Goal-Net that learns to predict agents' future goals; a VAE to generate displacements at every time-step; a RNN to consider the temporal nature of the sequence.

Another challenge, especially when agents move in crowded scenarios, is represented by *social interactions*. Interactions heavily impact on future trajectories [7], [9]: since people plan their paths reading each other future possible behaviours, each person's motion is influenced by the subjects around them. Interactions do not have to be necessarily as explicit as walking in groups or talking to each other: a moving pedestrian can interact with other people just by implicitly considering the positions of the surrounding agents to avoid future collisions. In team sports, interactions take on an even more important role: an attacker could put in place a certain set of movements just because the rest of his team has a specific disposition, as well as the whole defending team could in turn react and put in place a predefined tactic only because some opponents are arranged in a certain way. The varied nature of interactions and their heavy impact on agents' behaviour leads to develop sophisticated methods to accurately interpret and integrate such information into the prediction method.

Even the *future knowledge* about interacting agents positions can be a relevant feature that affects the development of each

*Equal contribution.

path. Taking into account this aspect during the prediction can improve the accuracy of the model.

To address these challenges we propose DAG-Net, a double attentive graph neural network for trajectory forecasting. The network backbone is a recurrent version of the Variational Autoencoder (VAE) [10]: time-step after time-step, the autoencoder is used to generate the future position in terms of the displacement from the current location. The modules of our recurrent autoencoder are conditioned on subjects' objectives so that the model can accordingly produce likely future positions. The backbone is integrated with a double Graph Neural Network (GNN)-based mechanism: the first GNN defines the future objectives of each agent in a structured way, distilling each goal with proximity knowledge; the second GNN models agents' interactions, filtering the hidden states of the recurrent network through neighbourhood information. Both the GNNs use a self-attention mechanism to assign different weights to each edge of the graph. The entire model is depicted in Fig. 1.

II. Related Works

Literature extensively demonstrated that trajectory prediction cannot leave aside modelling the interactions between different agents. Early works in trajectory prediction took advantage of hand-crafted features and energy potential parameters [11], [12], [13], [14]. In particular, Helbing and Molnar [11] modelled an interaction between two agents as a social force: the idea is to use attractive forces to guide agents toward their destination while employing repulsive forces to encourage collision avoidance. Other noteworthy examples are the Discrete Choice framework by Antonini et al. [12], Continuum Dynamics by Treuille et al. [13], and Gaussian Processes by Wang et al. [14]. Although these methods exhibit robust performance, they share a common weakness: the drawback is represented by the very same hand-crafted features, which fail to generalise properly and struggle in complex scenarios, limiting the results in terms of prediction accuracy. More modern approaches [9], [15], [16] rely on recurrent neural networks (RNNs) or one of their more efficient variants (LSTMs or GRUs) to learn these features directly from data: such architectures are specifically designed to exploit the temporal dependencies that characterise time-series; hence they are particularly suited for predicting trajectories, where every position is strictly correlated with the previous. To model all the agents inside a particular scene, the naïve use of such models foresees the employing of a single RNN with shared parameters. However, without further solutions, such a network would predict single agents trajectories independently. To share information across different subjects, several mechanisms have been proposed.

In this sense, one of the most important contributions is Social Pooling by Alahi et al. [9]: the mechanism merges agents' recurrent hidden states inside a *social tensor*. To build the social tensor the model employs a *grid-based pooling*: given an agent i and its neighbourhood $\mathcal{N}_i$, all the hidden states of agent i's neighbours are pooled together inside the social tensor, which is then fed to the recurrent cell as one of the inputs. The main limitation of this approach is the neighbourhood itself: such a local solution fails to capture the global context, as it does not allow the model to consider the interactions between all the possible agents inside the scene in a computationally efficient manner.

Social GAN by Gupta et al. [7] introduces, inside the Generator, a new Pooling Module that instead combines information coming from all the possible agents present in the scene. For every agent i, all the hidden states $\mathbf{h}_t^j$ of the other agents are processed by an MLP and max-pooled together element-wise into the tensor P_i.

Social Ways by Amirian et al. [17] takes up and customise this solution: instead of a simple max-pooling, the influence of the other agents on the generic agent i is evaluated by applying an attention weighting procedure. The model builds a so-called *interaction feature vector*: the i^{th} vector is created by combining new social features that come from predefined geometric properties. Given two agents i and j, the vector is built by stacking information like the Euclidean distance between the agents or the bearing angle from agent j to agent i. A simple scalar product between the i^{th} interaction vector and the hidden $\mathbf{h}^j$ brings to the attention coefficient a^{ij}.

Another relevant work is STGAT by Huang et al. [18]. Instead of employing custom pooling modules, the authors exploited the recent progress in Graph Neural Networks (GNNs). To share information across different pedestrians, STGAT treats each agent as a node of a graph. In this manner, the model can employ a GNN [19] as its sharing mechanism, allowing to aggregate information from neighbours by performing self-attention [20] on graph nodes.

A different approach comes instead from Zhan et al. [21]: rather than sharing the hidden states, the authors tried to induce coordination by working on agents' future intentions. The method is described as *weakly supervised*, since it requires a preliminary extraction from ground-truth trajectories of some low-dimensional features (called *macro-intents*) that will hopefully provide a tractable way to capture the coordination between agents. By conditioning the generation of the future intent on the hidden state of a recurrent cell that watches the whole set of agents, the produced macro-intents will somehow be also influenced by what other agents are willing to do. This allows the model to keep in consideration the coordination between the different subjects.

DAG-Net is inspired by [21] since it employs a similar idea of intents. However, our objectives are generated and used in a different way. We treat goals as structured components by exploiting them as graph nodes and we accordingly condition the generation process on the resulting interrelated future objectives. Furthermore, we use an attentive module to associate importance weights to different interactive nodes.

III. Method

A. Problem definition

The position of a generic agent i at time t is represented by $\mathbf{x}_t^i = (x^i, y^i)_t$, where $(x^i, y^i)_t$ are the coordinates that localise

the agent in the scene at the given time-step; agent i's trajectory can be therefore defined as a series $X_i = \{\mathbf{x}_1^i, ..., \mathbf{x}_T^i\}$ of consecutive positions. Every trajectory X_i is split into past and future: given a certain number T_{obs} of past positions, the goal is to predict in most accurate way the next T_{pred} future positions. All the coordinates are taken with respect to a real-world reference system, thus are expressed in meters or feet and do not refer to single pixels.

Before feeding the data to the model, every trajectory is transformed into a series of relative positions: every absolute position $\mathbf{x}_t^i$ is transformed in a couple of displacements $(\Delta x_t^i, \Delta y_t^i)$ that express the movement along the two axes with respect to the previous absolute position $\mathbf{x}_{t-1}^i$. Coming back to the original coordinates is always possible: given the initial absolute position $\mathbf{x}_1^i$, the other absolute coordinates can be obtained with a cumulative sum along the remaining time-steps. To simplify the notation, we use $\mathbf{x}_t$ instead of Δx_t to denote a displacement at time-step t.

B. Recurrent VAE

The network backbone is realised with a recurrent version of the Variational Autoencoder [22]. For non-sequential data, Variational Autoencoders have already been shown to be effective in recovering and modelling complex multi-modal distributions over the data space: for this purpose, a VAE introduces a set of latent random variables $\mathbf{z}$ within the latent space $\mathcal{Z}$, specifically designed to capture the characterising variations in the observed input variables $\mathbf{x}$.

In order to model and generate sequences that can be both highly variable and highly structured (e.g. trajectories), Chung et al. [23] extended this approach by proposing a recurrent version of the VAE, namely the Variational Recurrent Neural Network (VRNN). The network retains the necessary flexibility to model highly non-linear dynamics, but also explicitly models the dependencies between latent variables across subsequent time-steps. The VRNN can be described as a combination of the VAE and the RNN architectures: more specifically, the VRNN contains a variational autoencoder for every time-step of the input sequence, whose prior distribution over $\mathbf{z}$ is conditioned on the hidden state $\mathbf{h}_t$ of a common RNN. The combination with a recurrent cell helps the variational autoencoder architecture to keep into consideration the temporal structure of the input data. Generally, the model is employed in a completely generative setting: after training, the model is used to generate brand new sequences that however resemble the original dataset examples. Since we want to accurately reproduce agents' future path, we instead employ the model in a predictive setting: after a burn-in period of T_{obs} observation time-steps, the model is used to generate T_{pred} new future positions that have to be as close as possible to the ground-truth positions we want to forecast.

The model can be decomposed in four sub-components: the encoder, the decoder, the prior, and the RNN, that are implemented like neural networks as φ^{enc}, φ^{dec}, φ^{prior} and φ^{rnn}.

The first stage is represented by the encoder: the encoder receives raw data (in our case, single $\mathbf{x}_t$ displacements at a given time-step), embeds them in a fixed-length feature vector, incorporates the vector with the last hidden state $\mathbf{h}_{t-1}$ of the recurrent cell, and obtains the representation of their combination in the latent space $\mathcal{Z}$. Thus, the approximate posterior will not only be a function of $\mathbf{x}_t$ as in the VAE but also a function of $\mathbf{h}_{t-1}$:

$$\boldsymbol{\mu}_{\mathbf{z},t}, \boldsymbol{\sigma}_{\mathbf{z},t} = \varphi^{enc}\left(\varphi^{\mathbf{x}}(\mathbf{x}_t), \mathbf{h}_{t-1}\right), \tag{1}$$

$$q_\phi\left(\mathbf{z}_t | \mathbf{x}_{\leq t}, \mathbf{z}_{<t}\right) = \mathcal{N}\left(\mathbf{z}_t | \boldsymbol{\mu}_{\mathbf{z},t}, \left(\boldsymbol{\sigma}_{\mathbf{z},t}\right)^2\right), \tag{2}$$

where $\varphi^{\mathbf{x}}$ is a neural network that extracts the features from the input and ϕ are the approximation function parameters.

The decoder network takes the latent variable $\mathbf{z}_t$, embeds it in a fixed-size vector, incorporates the embedding with the last hidden state $\mathbf{h}_{t-1}$ of the recurrent cell, and provides a reconstruction $\hat{\mathbf{x}}_t$ of $\mathbf{x}_t$:

$$\boldsymbol{\mu}_{\hat{\mathbf{x}},t} \boldsymbol{\sigma}_{\hat{\mathbf{x}},t} = \varphi^{dec}\left(\varphi^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1}\right), \tag{3}$$

$$p_\theta\left(\mathbf{x}_t | \mathbf{x}_{<t}, \mathbf{z}_{\leq t}\right) = \mathcal{N}\left(\mathbf{x}_t | \boldsymbol{\mu}_{\hat{\mathbf{x}},t}, \left(\boldsymbol{\sigma}_{\hat{\mathbf{x}},t}\right)^2\right), \tag{4}$$

where $\varphi^{\mathbf{z}}$ is a feature extracting neural network and θ are the approximation function parameters. In the end, the objective of the decoder is to output a sample $\hat{\mathbf{x}}_t$ that resembles as much as possible the original input $\mathbf{x}_t$.

The prior network is instead able to reach the latent space $\mathcal{Z}$ starting only from the last hidden state $\mathbf{h}_{t-1}$ of the recurrent cell. It is computed as follow:

$$\boldsymbol{\mu}_{\mathbf{0},t}, \boldsymbol{\sigma}_{\mathbf{0},t} = \varphi^{prior}\left(\mathbf{h}_{t-1}\right), \tag{5}$$

$$p_\theta\left(\mathbf{z}_t | \mathbf{x}_{<t}, \mathbf{z}_{<t}\right) = \mathcal{N}\left(\mathbf{z}_t | \boldsymbol{\mu}_{\mathbf{0},t}, \left(\boldsymbol{\sigma}_{\mathbf{0},t}\right)^2\right). \tag{6}$$

The prior is essential to *generate* new data: to take a step forward and not just reconstruct the input (e.g. when we want to predict the next future displacement), the encoding network is detached. Here the prior network ensures that we are still able to reach the latent space $\mathcal{Z}$, even without the encoder: furthermore, since the produced latent variables resemble the ones returned by the encoder, passing them to the decoder allows to obtain likely (yet new) examples, as if they were actual inputs coming from the dataset.

Finally, the RNN updates its hidden state by taking into account both the input $\mathbf{x}_t$ and the latent variable $\mathbf{z}_t$: this encourages the explicit modelling of the temporal dependencies across subsequent time-steps.

$$\mathbf{h}_t = \varphi^{rnn}\left(\mathbf{x}_t, \mathbf{z}_t, \mathbf{h}_{t-1}\right) \tag{7}$$

The entire model is trained by maximising the sequential *evidence lower-bound* (ELBO):

$$\mathbb{E}_{q_\phi(\mathbf{z}_{\leq T} | \mathbf{x}_{\leq T})}\left[\sum^{T} \log p_\theta(\mathbf{x}_t \mid \mathbf{z}_{\leq t}, \mathbf{x}_{<t}) - D_{KL}\left(q_\phi(\mathbf{z}_t \mid \mathbf{x}_{\leq t}, \mathbf{z}_{<t}) \,||\, p_\theta(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t})\right)\right]. \tag{8}$$

The loss can be interpreted as the variational autoencoder ELBO summed over each time-step t of the input sequence.

C. Conditioning VAE to agents' goals

Inspired by [24], [25], [21], [26], [27], [28], we provide an additional input to our backbone in order to condition the displacements generation process to agents' future objectives. We choose to describe agents' future goals in terms of spatial information (Fig. 2a). To make our model as invariant as possible with respect to the different characteristics of the environment, we divided the top-down view of the scene in a grid of macro-areas: each cell can potentially represent the future objective of a single agent. Agent i's goal at time t, $\mathbf{g_t^i}$, is then represented by a one-hot encoding of the grid, where the cell in which the agent will land in the future is filled with a 1.

To obtain ground-truth objectives, a sliding window approach has been used: a window of size w slides through the original absolute trajectory and captures a goal every w time-steps. This information is used to condition the *prior*, the *encoder* and the *decoder* networks, as shown in Eq. (9), (10) and (11) where we drop the superscripts to refer to the behaviour of a general agent.

$$\boldsymbol{\mu}_{\mathbf{0},t}, \boldsymbol{\sigma}_{\mathbf{0},t} = \varphi^{prior}\left(\mathbf{h}_{t-1}, \mathbf{g}_t\right), \tag{9}$$

$$\boldsymbol{\mu}_{\mathbf{z},t}, \boldsymbol{\sigma}_{\mathbf{z},t} = \varphi^{enc}\left(\varphi^{\mathbf{x}}(\mathbf{x}_t), \mathbf{h}_{t-1}, \mathbf{g}_t\right), \tag{10}$$

$$\boldsymbol{\mu}_{\hat{\mathbf{x}},t}\boldsymbol{\sigma}_{\hat{\mathbf{x}},t} = \varphi^{dec}\left(\varphi^{\mathbf{z}}(\mathbf{z}_t), \mathbf{h}_{t-1}, \mathbf{g}_t\right). \tag{11}$$

To produce likely goals during the inference phase, we employ a further network. This network is again conditioned on the hidden state $\mathbf{h}_{t-1}$ of the recurrent cell, and takes as additional inputs the last predicted objective for the agent and the concatenation $\mathbf{d}_{t-1}$ of the absolute positions of the other agents in the scene (i.e. their disposition).

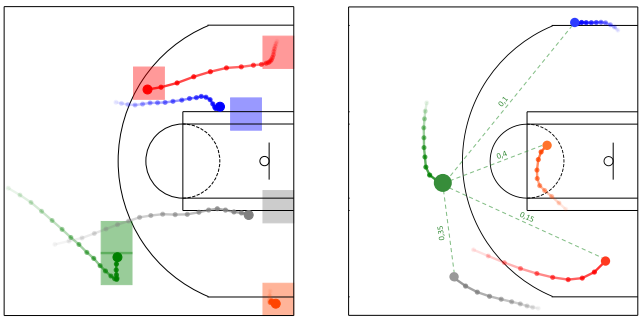
$$\mathbf{g}'_t = \varphi^{goal}(\mathbf{g}'_{t-1}, \mathbf{d}_{t-1}, \mathbf{h}_{t-1}) \tag{12}$$

$$\mathbb{E}_{q_\phi(\mathbf{z}_{\leq T}|\mathbf{x}_{\leq T})}\Bigg[\sum_{t=1}^{T}\sum_{k=1}^{K}\log p_\theta(\mathbf{x}_t \mid \mathbf{z}_{\leq t}, \mathbf{x}_{<t}) - \mathbf{g}_t^k log(\mathbf{g}_t'^k)$$
$$-D_{KL}\left(q_\phi(\mathbf{z}_t \mid \mathbf{x}_{\leq t}, \mathbf{z}_{<t}) \,||\, p_\theta(\mathbf{z}_t \mid \mathbf{x}_{<t}, \mathbf{z}_{<t})\right)\Bigg] \tag{13}$$

The additional loss term is computed as a Cross-Entropy between the ground-truth goal $\mathbf{g}_t$ and the predicted one $\mathbf{g'}_t$, where K is total the number of cells inside their one-hot encoding.

D. Double Attentive Graph Neural Networks

DAG-Net leverages two graph attentive networks to model two different kinds of interactions: the interactions between agents and the relationships between future goals. In structured motion environments, where agents' behaviours are moved not only by single intentions but also by social rules and/or common goals, it is important to condition the prediction to both mutual interactions and neighbours objectives. DAG-Net jointly employs past data and future intentions to improve forecasting in such contexts.



(a) (b)

Fig. 2. In (a) we can observe how goals deeply influence past and future trajectories, guiding agents to specific portions of the court. In (b) we can observe the similarities between the green player and his teammates: these values will directly influence the recombination of both goals and hidden states at the green node.

1) Goals relationships: As seen in Eq. (9), (10) and (11), agents' future objectives are used to condition the backbone: nevertheless, without further solutions, a single predicted goal $\mathbf{g}'_t$ focuses only on the corresponding agent objective. To effectively capture the coordination between the different subjects in the scene, DAG-Net shares goals information among agents relying on group interactions. To model the structure of future interactions, an attentive GNN [19] is employed. At every time-step, the network takes as an input node the one-hot encoding of each agent's predicted goal $\mathbf{g}'_t$, and produces a new distilled goal $\tilde{\mathbf{g}}_t$ built on proximity notions. After the concatenation of the distilled goal with the original one, the final refined goal is obtained through a linear projection:

$$\hat{\mathbf{g}}_t = W\left(\mathbf{g}'_t \parallel \tilde{\mathbf{g}}_t\right) \tag{14}$$

where the parameter matrix $W \in \mathcal{R}^{dxd}$ (with d the number of goals grid cells) is learnt in an end-to-end fashion during training. The new produced goal $\hat{\mathbf{g}}_t$ will then take the place of $\mathbf{g}'_t$ inside the ELBO loss presented in Eq. (13).

2) Agents' interactions: To model the interactions between the different agents in the scene, our model uses again a graph-based approach. Each agent is connected to the others as a node of a graph where edges weights are defined by a self-attention mechanism. A distance-based adjacency matrix is used in addition to the attentive module to consider proximity information (Fig. 2b). At every time-step, the hidden state $\mathbf{h}_t$ of each agent is fed to the GNN: the network outputs a new distilled hidden state $\tilde{\mathbf{h}}_t$ that takes into account the history of the neighbours. After the concatenation of the distilled hidden state with the original one, a linear layer is used to achieve the final output:

$$\hat{\mathbf{h}}_t = H\left(\mathbf{h}_t \parallel \tilde{\mathbf{h}}_t\right) \tag{15}$$

where the parameter matrix H is learnt during the training phase. The new refined hidden state $\hat{\mathbf{h}}_t$ will then be used in the next time-step as the current hidden state of the agent.

E. Training protocol

During training, we let the network see the entire $T = T_{obs} + T_{pred}$ time-steps from ground-truth sequences. The solution gives the model the opportunity to collect important features also from the latest time-steps of the sequence: this is useful in urban contexts and results particularly effective in sports, where we have long trajectories that usually start as linear but seldom continue in the same way, often bending and turning back upon themselves.

During validation and testing, we instead divide the trajectories into an *observation* and a *prediction* split: specifically, the network burns in for T_{obs} time-steps observing the first portion of the ground-truth trajectory, then it's let predicting the remaining T_{pred} time-steps.

IV. Experiments

A. Implementation Details

The VRNN recurrent cell is a GRU with 1 recurrent layer and a hidden state dimension of 64; the dimension of the latent variable is set to 32. For each graph, we then employ two attentive GNN layers: the first layer reduces the input to lower-dimensional hidden space, the second layer returns instead to the original input space. Each GNN layer uses 4 attention heads. The entire model has been optimised with Adam optimiser. To cope with the differences between urban and sport settings, we employ different sets of hyper-parameters.

For the urban setting, we use a learning rate of 10^{-4} and a batch size of 16; the Cross-Entropy contribution is weighted with a factor of 10^{-2}. The hidden state dimension between the two graph layers is set to 4. The model has been trained for 500 epochs.

For the sports setting, we use a learning rate of 10^{-3} and a batch-size of 64; the Cross-Entropy contribution is weighted with a factor of 10^{-2}. The hidden state dimension between the two graph layers is set to 8. The model has been trained for 300 epochs.

B. Metrics

Similar to prior works in literature, the model is evaluated with respect to two error metrics on prediction results, *Average Displacement Error* (ADE) and *Final Displacement Error* (FDE).

The Average Displacement Error represents the average Euclidean distance, over the entire predicted sequence, between the ground-truth positions and the predicted ones:

$$ADE = \frac{\sum_{i \in \mathcal{P}} \sum_{t=0}^{T_{pred}} \sqrt{((\hat{x}_t^i, \hat{y}_t^i) - (x_t^i, y_t^i))^2}}{\mid \mathcal{P} \mid \cdot T_{pred}} \tag{16}$$

where $\mathcal{P}$ is the set of pedestrians considered, $\mid \mathcal{P} \mid$ is its cardinality, $(\hat{x}_t^i, \hat{y}_t^i)$ are the predicted absolute coordinates at time t, and (x_t^i, y_t^i) are the ground-truth absolute coordinates at time t.

The Final Displacement Error follows the same logic but focuses only on the last time-step, evaluating the Euclidean distance between the final ground-truth position and the predicted one.

$$FDE = \frac{\sum_{i \in \mathcal{P}} \sqrt{((\hat{x}_{T_{pred}}^i, \hat{y}_{T_{pred}}^i) - (x_{T_{pred}}^i, y_{T_{pred}}^i))^2}}{\mid \mathcal{P} \mid} \tag{17}$$

C. Datasets

1) Stanford Drone Dataset: The Stanford Drone Dataset [29] is composed of a series of top-down videos recorded by a hovering drone in 8 different college campus scenes. This large scale dataset collects complex and crowded scenarios with various types of interacting targets: apart from classic pedestrians, we can also find bikes, skateboarders, cars, buses, and other vehicles, therefore the navigation inside such environments results particularly tough. We use the TrajNet benchmark version of the dataset [30]: trajectories are composed of a series of consecutive positions expressed as (x, y) world coordinates and recorded at 2.5FPS. Due to the lack of annotation in the test set, we split the training set into three sub-sets for the training, test and validation phases.

2) STATS SportVU NBA Dataset: The dataset comes from the player tracking data provided by STATS SportVU [31]. The dataset contains tracking positions from the 2016 NBA regular season on a span of over 1200 different games: the data are recorded with a series of cameras that surround the court and give back a bird-eye view of players' positions. The games are split into offensive plays, hence every sequence starts when the ball crosses the middle of the court. A play ends when one the following conditions is met: a shot is made (missed or scored), the ball exceeds the court bounds, the ball is intercepted by the defending team, or the shot clock runs out. Each play composes of 50 time-steps sampled at 5Hz, where each time-step contains the positions (expressed as (x, y, z) world coordinates) for all the 10 players on the court (5 attackers, 5 defenders) plus the ball. All the data have been subsequently normalised and shifted to have zero-centred sequences to the middle of court and plays that always develop towards the right basket.

D. Quantitative Results

For the basketball setting, we evaluated separately offence and defence: since agents are placed in an explicit competitive setting, their nature is intrinsically different, both from the goals and the trajectories points of view. The attackers call the shots trying to score, while the defenders usually react to their moves: training the network simultaneously on both teams would distract the final results.

The values reported in Table I support our choice, with defence trajectories being clearly easier than attacking ones. All the metrics are expressed in feet and refer to a prediction horizon of 40 time-steps, with 10 initial steps of observation.

TABLE I
BASKETBALL SPORTVU RESULTS

Team	Model	ADE	FDE
ATK	STGAT [18]	9.94	15.80
	Social-Ways [17]	9.91	15.19
	Weak-Supervision [21]	9.47	16.98
	DAG-Net (Our)	**8.98**	**14.08**
DEF	STGAT [18]	7.26	11.28
	Social-Ways [17]	7.31	10.21
	Weak-Supervision [21]	7.05	10.56
	DAG-Net (Our)	**6.87**	**9.76**

TABLE II
LONG-TERM EVALUATIONS

Model	Team	20-10 Split ADE	20-20 Split ADE	20-30 Split ADE
C-VAE [27]	ATK	3.95	5.80	7.08
DAG-Net (Our)	ATK	**2.09**	**4.58**	**6.66**
C-VAE [27]	DEF	3.01	4.10	**4.98**
DAG-Net (Our)	DEF	**2.05**	**4.07**	5.01

TABLE III
STANFORD DRONE DATASET RESULTS

Model	ADE	FDE
STGAT [18]	0.58	1.11
Social-Ways [17]	0.62	1.16
DAG-Net (Our)	**0.53**	**1.04**

The model achieves impressive results compared to state-of-the-art methods. STGAT shows quite strong performances, being able to weight all the possible contributions from the agents in the scene, but lacks additional information about what could happen in the long-term. With its attention-based pooling, Social-Ways returns very similar results: the combination of its geometric social features gives useful clues on such complicated trajectories and interactions as the basketball ones. Even though its different approach, Weak Supervision gain promising results too: the exploitation of future intentions allow agents to plan correctly their future path. However, by jointly considering agents' interactions and future goals, our model is more able to capture the nature of real paths and to reach smaller errors with respect to all these competitive methods.

To evaluate whether our model could show appreciable performance on different prediction horizons, we produced some *long-term evaluations*: since basketball trajectories offered a high number of time-steps with which we could produce various splits, we focused again on sports. For producing such evaluations, we concentrated on different observation-prediction sequences: given 10 time-steps of observation, we evaluated all the methods on increasing prediction splits, from 10 time-steps to 40 time-steps, with steps of 10. As Fig. 3 shows, our method globally outperforms the competitors in

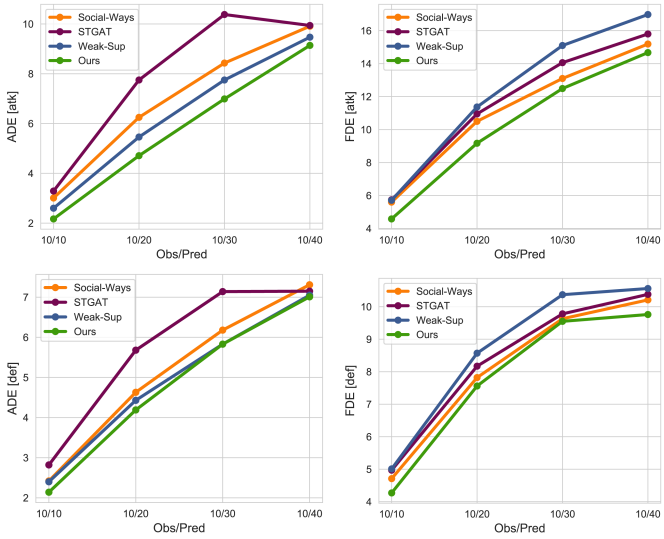


Fig. 3. Long-term evaluations: the method is evaluated both in ADE and FDE for increasing prediction lengths, from 10 to 40 time-steps. Attack on the top, defence on the bottom. All the metrics are in feet.

all the different evaluations and in both the metrics. As for the numbers in Table I, the difference is more pronounced for the attack than for the defence.

We have also run some long-term evaluations considering a longer observation period (Table II), mainly to observe how the prediction accuracy changes when the model is allowed to adjust to a greater initial period: we let the model burn-in for 20 initial time-steps and then predict the remaining ones, again with increasing steps of 10. In this setting we are able to compare our model to a further autoencoder architecture, by Felsen et al. [27], that briefly employs a C-VAE [25] conditioned on players' role. Our model, even without additional information about players' identities, shows better metrics in terms of the average distance from ground-truth positions.

In urban settings such as the ones in SDD (Table III), the aforementioned distinction between different categories of agents is no longer necessary, because all the pedestrians share the same nature and actively cooperate to not interfere with each other. The trajectories are split into segments of 8s: we observe 3.2s of history and predict over a 4.8s future horizon. Operating at 0.4s per time-step results in 8 time-steps of observation and a future prediction span of 12 time-steps; all the metrics are in meters. We can not report results of Weak-Supervision for SDD: since the model adopts a separate VRNN for each agent, its architecture is not suitable for less constrained scenes that exhibit a variable number of agents. For this reason, Weak-Supervision could not be tested outside the basket environment.

The results are in line with the ones presented for basketball. The attentive sharing of agents' goals and the graph distillation step for the hidden states outperforms the competitors in both ADE and FDE: individual goals force agents to pass

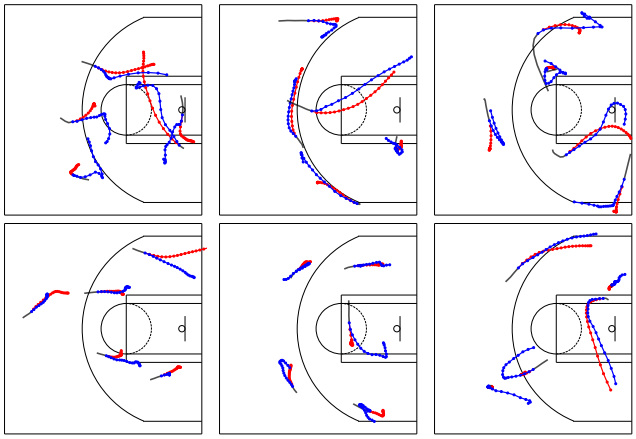


Fig. 4. Basketball roll-outs. After an initial observation stage (black), model predictions (red) are evaluated against the ground-truth (blue), The top roll-outs refer to three different attack plays, while the bottom one represent three different defensive actions.

Fig. 5. Qualitative samples that compare DAG-Net and state-of-the-art methods on Stanford Drone Dataset.

through specific future areas coherent to their past trajectory, while the two attentive mechanisms help them to keep in consideration both other agents' immediate will and their long-term objectives.

E. Qualitative Results

In competitive settings such as sports, the opposing teams are intrinsically different. The attacking team drives the game trying to score, while the defenders often limit to counter react to its moves. These behaviours deeply affect the resulting trajectories, as can be clearly seen in the roll-outs presented in Fig. 4. Attackers trajectories tend to be particularly varied and intricate, often bending and intersecting; on the contrary, defenders tend to move linearly and occasionally deflect to follow an opponent or to close a gap. Despite some sudden changes of direction in the real trajectories, especially for the attackers, our model is able to correctly predict the overall future movement of the players and rather trace the ground-truth. Because of the complexity of such trajectories, the predictions do not always precisely resemble the expected output: nevertheless, even when the predictions fail to follow the real ground-truth trajectories, the model still predicts a likely behaviour coherent with the play development, proving its strength in capturing the multi-modal nature of players' movements.

TABLE IV
STATS SPORTVU

Team	Model	Agents' interact.	Future object.	ADE	FDE
ATK	Vanilla VRNN [23]	✗	✗	9.41	15.56
	A-VRNN	✓	✗	9.48	15.52
	DAG-Net (Our)	✓	✓	**8.98**	**14.08**
DEF	Vanilla VRNN [23]	✗	✗	7.16	10.50
	A-VRNN	✓	✗	7.05	10.34
	DAG-Net (Our)	✓	✓	**6.87**	**9.76**

TABLE V
STANFORD DRONE DATASET

Model	Agents' interact.	Future object.	ADE	FDE
Vanilla VRNN [23]	✗	✗	0.58	1.17
A-VRNN	✓	✗	0.56	1.14
DAG-Net (Our)	✓	✓	**0.53**	**1.04**

On the other hand, urban trajectories are more straightforward, because pedestrian obviously tend to move linearly, doing only some occasional deviations to avoid collisions or to turn. Nevertheless, the adoption of agents' goals still gives the model the possibility to produce more likely trajectories. Since agents are constrained to pass through specific portions of the scene coherent with their motion behaviour, predictions can closely resemble real future movements: in both the plot reported in Fig. 5, DAG-Net is able to keep closer to the ground-truth, while both the competitors tend to predict more linear trajectories and consequently deviate from the expected output. For the very same reasons, final predictions can also be more precise: having important insights about the regions the agent will occupy in the future can help the model to appropriately predict the overall portion of the scene where the agent will land at the end of his trajectory. DAG-Net predicted final location resembles the agent's real destination, while both the competitors fail to approximately forecast such information.

V. ABLATION EXPERIMENTS

In this section we present ablation experiments to show the improvements introduced by each component of our model (Table IV and Table V). Results present two baselines: the Vanilla VRNN and the Attentive-VRNN (A-VRNN), i.e. a version of our network that presents only the attentive graph for the hidden states refinement. DAG-Net outperforms both baselines on STATS SportVU and SDD. The Vanilla VRNN experiments show that using a stand-alone network without considering interactions between agents does not allow the model to capture the nature of real paths. For this reason, A-VRNN achieves better performance than Vanilla VRNN;

still, this version is not able to capture future structured dependencies between agents. The results obtained with DAG-Net highlight the importance of inserting future information into the prediction and combining humans objectives in a structured way.

VI. Conclusions

We propose a novel architecture called DAG-Net, a double graph-based network that deals with both past interactions and future goals through attentive mechanisms. By facing trajectory prediction as a structured problem, our model overcomes state-of-the-art performances on both STATS SportVU NBA Dataset and Stanford Drone Dataset, proving its strength on team sports and urban contexts. It shows impressive results also on long-term predictions. Our future work will focus on the application of our model to more general settings, involving time-series forecasting such as finance and health care.

Acknowledgements

Funded by the PRIN PREVUE - PRediction of activities and Events by Vision in an Urban Environment" project (CUP E94I19000650001), PRIN National Research Program, MIUR.

References

[1] W.-C. Ma, D.-A. Huang, N. Lee, and K. M. Kitani, "Forecasting interactive dynamics of pedestrians with fictitious play," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 774–782.

[2] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 126–12 134.

[3] A. Bera, S. Kim, T. Randhavane, S. Pratapa, and D. Manocha, "Glmp-realtime pedestrian path prediction using global and local movement patterns," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 5528–5535.

[4] J. Mainprice, R. Hayne, and D. Berenson, "Goal set inverse optimal control and iterative replanning for predicting human reaching motions in shared workspaces," *IEEE Transactions on Robotics*, vol. 32, no. 4, pp. 897–908, 2016.

[5] S. Oh, A. Hoogs, A. Perera, N. Cuntoor, C.-C. Chen, J. T. Lee, S. Mukherjee, J. Aggarwal, H. Lee, L. Davis *et al.*, "A large-scale benchmark dataset for event recognition in surveillance video," in *CVPR 2011*. IEEE, 2011, pp. 3153–3160.

[6] R. Mehran, A. Oyama, and M. Shah, "Abnormal crowd behavior detection using social force model," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2009, pp. 935–942.

[7] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[8] V. Kosaraju, A. A. Sadeghian, R. Martín-Martín, I. D. Reid, S. H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[9] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 961–971.

[10] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013.

[11] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[12] G. Antonini, M. Bierlaire, and M. Weber, "Discrete Choice Models for Pedestrian Walking Behavior," *Transportation Research Part B: Methodological*, vol. 40, pp. 667–687, 09 2006.

[13] A. Treuille, S. Cooper, and Z. Popovic, "Continuum crowds," *ACM Trans. Graph.*, vol. 25, pp. 1160–1168, 07 2006.

[14] J. Wang, A. Hertzmann, and D. J. Fleet, "Gaussian Process Dynamical Models," in *Advances in Neural Information Processing Systems 18*, Y. Weiss, B. Schölkopf, and J. C. Platt, Eds. MIT Press, 2006, pp. 1441–1448.

[15] T. Fernando, S. Denman, S. Sridharan, and C. Fookes, "Soft+ hardwired attention: An lstm framework for human trajectory prediction and abnormal event detection," *Neural networks*, vol. 108, pp. 466–478, 2018.

[16] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. K. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2165–2174, 2017.

[17] A. Javad, H. Jean-Bernard, and P. Julien, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2019, pp. 0–0.

[18] H. Yingfan, B. Huikun, L. Zhaoxin, M. Tianlu, and W. Zhaoqi, "STGAT: Modeling Spatial-Temporal Interactions for Human Trajectory Prediction," in *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[19] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph Attention Networks," *International Conference on Learning Representations (ICLR)*, 2018.

[20] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 5998–6008.

[21] E. Zhan, S. Zheng, Y. Yue, L. Sha, and P. Lucey, "Generating multi-agent trajectories using programmatic weak supervision," in *International Conference on Learning Representations (ICLR)*, 2019.

[22] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic Backpropagation and Approximate Inference in Deep Generative Models," in *Proceedings of the 31st International Conference on Machine Learning, Cycle 2*, ser. JMLR Proceedings, vol. 32. JMLR.org, 2014, pp. 1278–1286.

[23] J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio, "A Recurrent Latent Variable Model for Sequential Data," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 2980–2988.

[24] D. P. Kingma, D. J. Rezende, S. Mohamed, and M. Welling, "Semi-supervised learning with deep generative models," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'14. Cambridge, MA, USA: MIT Press, 2014, pp. 3581–3589.

[25] K. Sohn, X. Yan, and H. Lee, "Learning structured output representation using deep conditional generative models," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, ser. NIPS'15. Cambridge, MA, USA: MIT Press, 2015, pp. 3483–3491.

[26] A. Bhattacharyya, M. Hanselmann, M. Fritz, B. Schiele, and C.-N. Straehle, "Conditional flow variational autoencoders for structured sequence prediction," *arXiv*, vol. abs/1908.09008, 2019.

[27] P. Felsen, P. Lucey, and S. Ganguly, "Where will they go? predicting fine-grained adversarial multi-agent motion using conditional variational autoencoders," in *ECCV*, 2018.

[28] L. Jiachen, M. Hengbo, and T. Masayoshi, "Conditional generative neural system for probabilistic trajectory prediction," in *in 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019.

[29] A. Robicquet, A. Alahi, A. Sadeghian, B. Anenberg, J. Doherty, E. Wu, and S. Savarese, "Forecasting Social Navigation in Crowded Complex Scenes," *CoRR*, vol. abs/1601.00998, 2016.

[30] A. Sadeghian, V. Kosaraju, A. Gupta, S. Savarese, and A. Alahi, "Trajnet: Towards a benchmark for human trajectory prediction," *arXiv preprint*, 2018.

[31] "SportVU - STATS Perform," https://www.statsperform.com/team-performance/basketball/optical-tracking/.