Computer Vision and Image Understanding 210 (2021) 103245

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

AC-VRNN: Attentive Conditional-VRNN for multi-future trajectory prediction

Alessia Bertugli [a,*], Simone Calderara [b], Pasquale Coscia [c], Lamberto Ballan [c], Rita Cucchiara [b]

[a] Dipartimento di Ingegneria e Scienza dell'Informazione, Universitá degli Studi di Trento, Italy
[b] Dipartimento di Ingegneria "Enzo Ferrari", Universitá degli Studi di Modena e Reggio Emilia, Italy
[c] Dipartimento di Matematica "Tullio Levi-Civita", Universitá degli Studi di Padova, Italy

ARTICLE INFO

Communicated by Nikos Paragios

Keywords:
Trajectory forecasting
Multi-future prediction
Time series
Variational recurrent neural networks
Graph attention networks

ABSTRACT

Anticipating human motion in crowded scenarios is essential for developing intelligent transportation systems, social-aware robots and advanced video surveillance applications. A key component of this task is represented by the inherently multi-modal nature of human paths which makes socially acceptable multiple futures when human interactions are involved. To this end, we propose a generative architecture for multi-future trajectory predictions based on Conditional Variational Recurrent Neural Networks (C-VRNNs). Conditioning mainly relies on prior belief maps, representing most likely moving directions and forcing the model to consider past observed dynamics in generating future positions. Human interactions are modelled with a graph-based attention mechanism enabling an online attentive hidden state refinement of the recurrent estimation. To corroborate our model, we perform extensive experiments on publicly-available datasets (e.g., ETH/UCY, Stanford Drone Dataset, STATS SportVU NBA, Intersection Drone Dataset and TrajNet++) and demonstrate its effectiveness in crowded scenes compared to several state-of-the-art methods.

1. Introduction

Trajectory forecasting has recently experienced exponential growth in several research areas such as video surveillance, sports analytics, self-driving cars and physical systems (Rudenko et al., 2020). Its main applications include pedestrians dynamics prediction (Helbing and Molnar, 1995; Alahi et al., 2016; Xue et al., 2018; Gupta et al., 2018; Yingfan et al., 2019; Zhang et al., 2019), vehicles behaviour analysis (Jiachen et al., 2019; Ma et al., 2018; Charlie et al., 2019; Lee et al., 2017) as well as intent estimation of people and cars on roads to avoid possible crashes. In sports analytics (Felsen et al., 2018; Yeh et al., 2019; Zhan et al., 2019; Sun et al., 2019; Chieh-Yu et al., 2018; Hsin-Ying et al., 2019), being able to predict players trajectories can improve the action interpretation of each player while in physical systems it can be fundamental to predict particles dynamics in complex domains (Kipf et al., 2018; Alet et al., 2019; Webb et al., 2019).

In this paper, we focus on predicting human dynamics in crowded contexts (e.g., shop entrances, university campuses and intersections) where people and autonomous vehicles mainly manifest their complex and multi-modal nature. Typically, two different strategies are employed to model human interactions: *pooling-based* and *graph-based* methods. Pooling-based methods (Alahi et al., 2016; Xue et al., 2018; Gupta et al., 2018; Javad et al., 2019; Liang et al., 2019; Lisotto et al., 2019) employ sequence-to-sequence models to extract features and generate subsequent time steps, interspersed with pooling layers to model interactions between neighbours. By contrast, graph-based methods (Yingfan et al., 2019; Zhang et al., 2019; Ma et al., 2018; Yeh et al., 2019; Sun et al., 2019; Vemula et al., 2018; Liang et al., 2020) apply graph neural networks to model interactions. Although these approaches have proven to be effective, some problems are still open, such as efficiently exploiting context cues and appropriately capturing human interactions in critical situations. Another relevant aspect to consider in trajectory prediction is represented by scene constraints like walls and other obstacles which strongly influence human motion. A common approach to overcome this issue is to introduce visual elements into the network such as images or semantic segmentation (Liang et al., 2019; Kosaraju et al., 2019; Sadeghian et al., 2018b) yet this implies the availability of video streams both at train and test time. To this end, we propose a novel method for multi-future trajectory forecasting that works in a completely generative setting, enabling the prediction of multiple possible futures. During online inference, we integrate human interactions at time step level, allowing other agents to affect the whole trajectory generation process. As a consequence, online interactions computation improves the predicted trajectories as the number of time steps increases limiting the error growth. To take into account past human motion, local belief maps steer future positions towards more frequent crossed areas when human interactions are limited or absent.

* Corresponding author.
E-mail address: alessia.bertugli@unitn.it (A. Bertugli).

https://doi.org/10.1016/j.cviu.2021.103245
Received 24 July 2020; Received in revised form 24 June 2021; Accepted 30 June 2021
Available online 7 July 2021

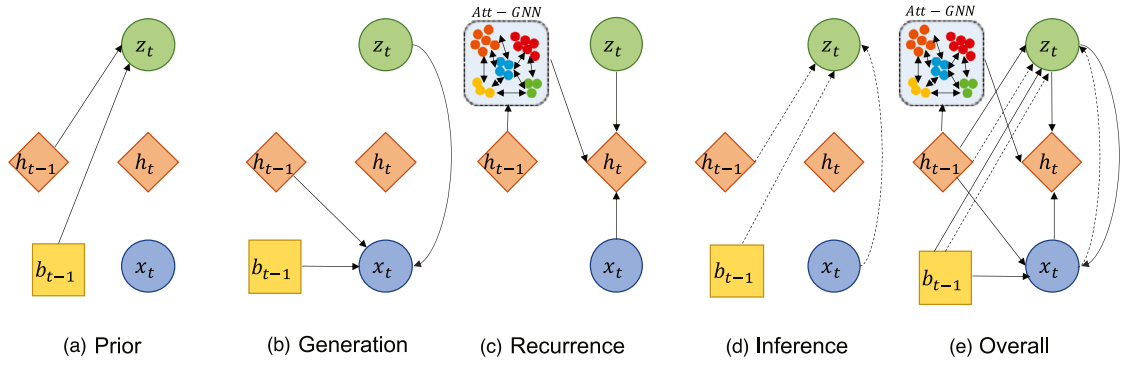

Fig. 1. Illustration of each phase of our AC-VRNN architecture for a time step t. A recurrent variational autoencoder is conditioned on prior belief maps b_{t-1}. The hidden state of the RNN h_{t-1} is refined with an attentive module obtaining h'_t, that replaces h_t in the next step of recurrence. At inference time, it generates future displacements using the prior network on $\mathbf{h}_t$ and makes an online computation of the adjacency matrix which defines connections between pairs of nodes.

Technically, our model is a Conditional-VRNN, conditioned by prior belief maps on pedestrians frequent paths, that predicts future positions one time step at a time, by relying on recurrent network hidden states refined with an attention-based mechanism.

The main contributions of this paper are two-fold:

(i) We propose a novel method to integrate human interactions into the model in an online fashion, relying on a hidden state refinement process with a graph attentive mechanism. We employ a similarity-based adjacency matrix to take into account pedestrians' neighbourhoods.

(ii) We introduce local belief maps to encourage the model to follow a prior transition distribution whenever the prediction is uncertain and to discourage unnatural behaviour such as crossing obstacles, avoiding employing additional visual inputs. In this way, future positions may take advantage of prior knowledge while being predicted. Such behaviour is imposed during training by a Kullback–Leibler (KL) divergence loss between ground-truths and samples contributing to the model performance refinement.

We demonstrate that our model achieves state-of-the-art performance on several standard benchmarks using different evaluation protocols. We also outperform our competitors on the challenging Stanford Drone Dataset (SDD) and the recent Intersection Drone Dataset (InD) showing the robustness of our architecture to more complex urban contexts. Furthermore, we test our model on human dynamics collected from basketball players to analyse its ability to capture complex interactions in confined areas. Finally, our architecture positions among the best models on the TrajNet++ benchmark.

2. Related work

Traditionally, trajectory prediction has been approached with rule-based and social force models (Helbing and Molnar, 1995; Mehran et al., 2009; Zanlungo et al., 2011) that have been proven to be effective in simple contexts, but fail to generalize to complex domains. In recent years, generative models (Gupta et al., 2018; Zhan et al., 2019; Sun et al., 2019; Kosaraju et al., 2019; Ivanovic and Pavone, 2019) have been focusing on the multi-modal nature of this task since multiple human paths could be regarded as socially acceptable despite being different from ground-truth annotations. In the following, we group related work into position-based models, which uses only spatial information, and graph-based models, which rely on connected structures.

Position-based models. Social-LSTM (Alahi et al., 2016) models individual trajectory as a long short-term memory (LSTM) encoder–decoder and considers interactions using a social pooling mechanism. Social GAN (Gupta et al., 2018) uses a pooling mechanism in combination with a generative model to predict socially acceptable trajectories. SoPhie (Sadeghian et al., 2018b) consists of a Generative Adversarial Network (GAN), which leverages the contribution of a social attention module and a physical attention module. SS-LSTM (Xue et al., 2018) uses different inputs to also take into account the influence of the environment and maps of the neighbourhood to narrow the field of mutual influences.

Graph-based models. Graph Neural Networks (GNNs) have been used to model interactions between different trajectories. Graph Variational RNNs (Yeh et al., 2019) model multi-agent trajectory data mainly focusing on multi-player sports games. Each agent is represented by a VRNN where prior, encoder and decoder are modelled as message passing GNNs, allowing the agents to weakly share information through nodes. Graph-structured VRNN network (Sun et al., 2019), based on relation networks, infers the current state and forecasts future states of basket and football players trajectories. SR-LSTM (Zhang et al., 2019) uses a state refinement module through a motion gate and pedestrian-wise attention. Social-BiGAT (Kosaraju et al., 2019) presents a graph-based generative adversarial network based on GAT (Veličković et al., 2018) that learns reliable future representations that encode the social interactions between humans in the scene and contextual images to incorporate scene information. ST-GAT (Yingfan et al., 2019) proposes a model based on two levels of LSTMs to incorporate interactions through a hidden state refinement, which uses a GAT in the encoding part, while the decoder generates future positions.

Compared to approaches based on Variational Autoencoders (VAEs) (Yeh et al., 2019; Sun et al., 2019; Kipf et al., 2018), our method does not model the prediction as a graph yet uses an attentive module to refine the hidden state of a recurrent network. Doing so, information about other agents influences the prediction. Unlike sports games, where all players share the same goal, in urban scenarios neighbourhood information is crucial since future paths may depend on mutual distances among people. Our model resembles SR-LSTM (Zhang et al., 2019) and STGAT (Yingfan et al., 2019) combining LSTMs and GNNs. Nevertheless, SR-LSTM (Zhang et al., 2019) exploits cell states of LSTMs limiting the observation horizon. STGAT (Yingfan et al., 2019) uses GAT (Veličković et al., 2018) as hidden state refinement, but it employes a sequence-to-sequence model without an online refinement. Both methods do not take into account contextual information or collective behaviours (e.g., belief maps) in order to avoid uncommon paths.

3. AC-VRNN model

Pedestrian dynamics are primarily affected by the neighbourhood space in urban areas. To avoid obstacles or other people, pedestrians continuously steer their motion gaining also the advantage of prior knowledge acquired in similar contexts. To this end, our model relies on past motions of monitored scenes as well as structured interactions in a generative setting.

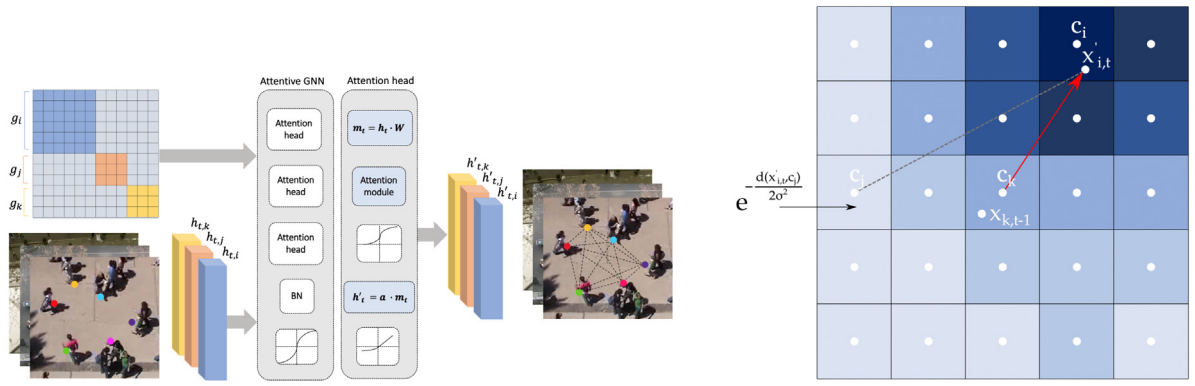

Fig. 2. Scheme of the proposed attentive hidden state refinement process. (a) The adjacency matrix is an irregular block matrix where each block size is defined by the number of pedestrians in the current scene. (b) Belief map during training for one sample using heat similarity-based strategy. The map is centred at $t-1$ to display the sampled displacements distribution at t.

Problem formulation. Given a pedestrian at time step t, his/her current position is represented by 2-D coordinates. Our model analyses T_{obs} time steps to predict motion dynamics during the next T_{pred} time steps. Similarly to Gupta et al. (2018), our model uses displacements with respect to the previous points. More specifically, given a sequence of displacements $(x_0, .., x_{T_{pred}})$, we observe a part of the sequence $(x_0, \dots, x_{T_{obs}})$ and predict the subsequent one $(x_{T_{obs}+1}, \dots, x_{T_{pred}})$.

Our Attentive Conditional Variational Recurrent Neural Network (AC-VRNN) is composed of three building blocks: (i) a VRNN to generate a sequences of displacements in a multi-modal way; (ii) a hidden state refinement based on an attentive mechanism to model the interactions within the neighbourhood, performed at a time step level during training and inference phases; (iii) a belief map to encourage the model to follow prior belief maps when it is uncertain, avoiding predicting samples that may fall within never crossed areas. A complete illustration of all phases of AC-VRNN is shown in Fig. 1.

Predictive VRNN. VRNNs (Chung et al., 2015) explicitly model dependencies between latent random variables $\mathbf{z}_t$ across subsequent time steps. They contain a Variational Autoencoder (VAE) (Kingma and Welling, 2014) at each time step conditioned on the hidden state variable $\mathbf{h}_{t-1}$ of an RNN to take into account temporal structures of sequential data. At each time step, prior, encoder and decoder output multivariate normal distributions, with three functions (f_{pri}, f_{enc} and f_{dec}) modelling their means and variances. Since the true posterior is intractable, it is approximated by a neural network q_ϕ, which also depends on the hidden state $\mathbf{h}_{t-1}$ under recurrency equations as follows:

$$p_\theta\left(\mathbf{z}_t|\mathbf{x}_{<t}, \mathbf{z}_{<t}\right) = \mathcal{N}\left(\mathbf{z}_t|\boldsymbol{\mu}_{\text{pri},t}, \left(\boldsymbol{\sigma}_{\text{pri},t}\right)^2\right), \quad \text{(prior)} \quad (1)$$

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

$$p_\theta\left(\mathbf{x}_t|\mathbf{x}_{<t}, \mathbf{z}_{\leq t}\right) = \mathcal{N}\left(\mathbf{x}_t|\boldsymbol{\mu}_{\text{dec},t}, \left(\boldsymbol{\sigma}_{\text{dec},t}\right)^2\right), \quad \text{(generation)} \quad (3)$$

$$\mathbf{h}_t = f_{\text{rnn}}\left(\mathbf{x}_t, \mathbf{z}_t, \mathbf{h}_{t-1}\right). \quad \text{(recurrence)} \quad (4)$$

These functions can be deep neural networks with learnable parameters θ and ϕ that output $\left(\boldsymbol{\mu}_{\text{pri},t}, \boldsymbol{\sigma}_{\text{pri},t}\right)$, $\left(\boldsymbol{\mu}_{\text{enc},t}, \boldsymbol{\sigma}_{\text{enc},t}\right)$ and $\left(\boldsymbol{\mu}_{\text{dec},t}, \boldsymbol{\sigma}_{\text{dec},t}\right)$, respectively. The generative and inference processes are jointly optimized by maximizing the following variational lower bound (ELBO) with respect to their parameters[1]:

$$ELBO = \mathbb{E}_{q_{\phi,t}(\mathbf{z}_t)}\left[\sum_{t=1}^{T}\left(-\text{KL}\left(q_{\phi,t}(\mathbf{z}_t) \parallel p_{\theta,t}(\mathbf{z}_t)\right) + \log p_{\theta,t}(\mathbf{x}_t)\right)\right]. \quad (5)$$

VRNNs are typically employed to generate sequences from scratch, in a fully generative setting. However, our task is to imitate training data rather than generate completely new data at evaluation time. In a predictive setting, the predicted positions must rely on the observed ones; without any information coming from the past, future positions would only be random. Using a fully generative setting, the model would not have any chance to exploit previous observations. For this reason, we have modified the inference protocol to generate sequences using the hidden state of the last observed time step. VRNN learns at each time step to generate the current displacement, given the input and the RNN's hidden state. At inference time, the model only uses the last hidden state from the observed sequence, then generates the subsequent time step. For the above reasons, AC-VRNN is a generative model used in a predictive setting: it generates one displacement at a time and becomes easy to embed human interactions at time step level.

Attentive Hidden State Refinement. Pedestrians dynamics are mainly influenced by surrounding agents. Our model handles human interactions using an attentive hidden state refinement of our recurrent network through a graph neural network, as shown in Fig. 2(a). Our hidden state refinement resembles the idea proposed by Veličković et al. (2018) which adopts an attention mechanism to learn relative weights between two connected nodes, through specific transformations called graph attentional layers. At time step t, our refinement strategy considers a set of hidden state nodes $\{\mathbf{h}_t^1, \dots, \mathbf{h}_t^N\}$, where each $\mathbf{h}_t^i \in \mathbb{R}^F$ represents the hidden state of the ith agent in the scene. The attention layer produces a new set of node features $\{\hat{\mathbf{h}}_t^1, \dots, \hat{\mathbf{h}}_t^N\}$, $\hat{\mathbf{h}}_t^i \in \mathbb{R}^{F'}$ as its output. The transformation is parametrized by a weight matrix $\mathbf{W} \in \mathbb{R}^{F' \times F}$ (shared between graph nodes) and a weight vector $\mathbf{a} \in \mathbb{R}^{2F'}$. Self-attention coefficients $\alpha_{i,j}$ between the nodes $\mathbf{h}_t^i$ and $\mathbf{h}_t^j$ are computed as follows:

$$\alpha_{i,j} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{h}_t^i \parallel \mathbf{W}\mathbf{h}_t^j\right]\right)\right)}{\sum_{k\in\mathcal{N}_i}\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T\left[\mathbf{W}\mathbf{h}_t^i \parallel \mathbf{W}\mathbf{h}_t^k\right]\right)\right)}, \quad (6)$$

where $\parallel$ represents the concatenation operator. The normalized attention coefficients are used to compute a linear combination of the features which represents the final output feature for every node, followed by a ELU non-linearity (Clevert et al., 2016) acting on the neighbourhood $\mathcal{N}_i$ of the ith node:

$$\hat{\mathbf{h}}_t^i = \text{ELU}\left(\sum_{j\in\mathcal{N}_i}\alpha_{i,j}\mathbf{W}\mathbf{h}_t^j\right). \quad (7)$$

The neighbourhood $\mathcal{N}_i$ defines the set of nodes with positive adjacency with respect to the ith agent. The adjacency matrix follows a similarity-based principle, and it is computed, inspired by proxemics interaction theory (Rios-Martinez et al., 2015), considering the heat kernel of the distance $d(i,j)$ between each pedestrian, $\exp\left(-\frac{d(i,j)}{2\sigma^2}\right)$, where σ is a smoothing hyperparameter. During training, our VRNN takes as input

[1] In order to keep the notation light we omit the conditioning variables.

Algorithm 1: Belief Maps Generation Algorithm.

```
function BELIEF_MAPS_GENERATION(trajectories)
    N, M, δ_x, δ_y ← get_grid_coarse(trajectories)
    x_min, y_min, x_max, y_max ← get_min_max(trajectories)
    global_grid ← make_global_grid(x_min, y_min, N, M, δ_x, δ_y)
    for all bin ∈ global_grid do
        maps ← [0, .., 0]
        for all trajectory ∈ trajectories do
            neighbour_centres ← get_neighbour_centres(bin, δ_x, δ_y)
            for all index, coord ∈ trajectory do
                if coord_x ∈ [bin_x, bin_x + δ_x] and coord_y ∈ [bin_y, bin_y + δ_y] then
                    next_coord ← trajectory[index + 1]
                    map ← similarity_matrix(next_coord, neighbour_centres, map)
                end if
            end for
        end for
        map ← normalize(map)
        maps ← insert(map)
    end for
    return maps
end function

function GET_GRID_COARSE(trajectories)
    μ_x, μ_y ← mean_displacements(trajectories)
    σ_x, σ_y ← standard_deviation_displacements(trajectories)
    x_min, y_min, x_max, y_max ← get_min_max(trajectories)
    N ← (x_max − x_min) / ((μ_x + σ_x)/2) ; M ← (y_max − y_min) / ((μ_y + σ_y)/2)
    δ_x ← (x_max − x_min)/N ; δ_y ← (y_max − y_min)/M
    return N, M, δ_x, δ_y
end function

function SIMILARITY_MATRIX(next_coord, neighbour_centres, map)
    for all index, centre ∈ neighbour_centres do
        map[index] ← accumulate(e^{−sqrt((next_coord_x − centre_x)^2 + (next_coord_y − centre_y)^2)})
    end for
    return map
end function
```

a set of sequences for a time step t. Then, it samples the next position $\mathbf{x}_t^i$ for each pedestrian i. Finally, the graph attention mechanism acts on the hidden state $\mathbf{h}_t^i$ (provided by Eq. (4)) to compute the corresponding interaction-refined state $\hat{\mathbf{h}}_t^i$. The refined hidden state $\hat{\mathbf{h}}_t^i$ is concatenated to the original one and a final linear projection is applied as follows:

$$\mathbf{h}_t'^i = \text{Linear}\left(\mathbf{h}_t^i \parallel \hat{\mathbf{h}}_t^i\right). \tag{8}$$

At the next time step, our VRNN uses the refined hidden state $\mathbf{h}_t'^i$ which carries information about interactions of previous time steps.

Conditional-VRNN on Belief Maps. Since AC-VRNN is a stochastic model, it could potentially exhibit high predictive variance hence generating predictions far from expected ones. To balance the bias/variance trade-off of the predictor, we introduce belief maps on displacements. Belief maps collect data about crossed areas at training time; therefore, they contain information about the collective behaviour of monitored agents. Conditioning the prediction to such maps, may lead the model to follow past behaviours and, at the same time, discourage it to predict displacements far from past crossed areas, avoiding the generation of non-realistic paths.

Belief maps are computed dividing the coordinate space for each scene into a $N \times M$ grid. The boundaries of this grid are defined by minimum and maximum coordinates along x and y directions. Both past and future information of training trajectories are considered. These values could also be obtained manually defining the allowed area for predicting new coordinates. The values of N and M define the grid coarse and are computed considering the average displacement μ and its standard deviation σ as follows:

$$N = \left\lfloor \frac{(x_{max} - x_{min})}{\frac{\mu+\sigma}{2}} \right\rfloor, \quad M = \left\lfloor \frac{(y_{max} - y_{min})}{\frac{\mu+\sigma}{2}} \right\rfloor. \tag{9}$$

For each grid location (bin), a $L \times L$ neighbourhood is then considered (with $L = 5$). For each (x, y) location, we get the corresponding $L \times L$ neighbourhood and compute heat kernels between the next location and the neighbourhood bins centres[2]. This procedure is repeated for all the trajectories and bins values are accumulated by summation. Each belief map $\mathbf{b}$, i.e. a $L \times L$ sub-grid indexed by the (x, y) location in the scene, is subsequently normalized in order to transform the cumulative grid into a probability distribution. Unlike the recurrent process within the VRNN, the creation of belief maps is a Markov process, as their generation only depends on single-step transitions. Details in Algorithm 1 each step for generating our belief maps.

Conditional-VRNN. We exploit the belief maps to encourage the model to follow the average behaviour shown by previously observed agents. In our work, we use a recurrent version of CVAE (Kingma et al., 2014), conditioning VRNN on belief maps. At each time step, *prior*, *encoder* and *decoder* networks take the belief map at $t-1$ as input, conditioning the resultant Gaussian distribution. We embed belief maps with a linear projection before feeding them into the VRNN blocks:

$$\boldsymbol{\mu}_{\text{pri},t}, \boldsymbol{\sigma}_{\text{pri},t} = f_{\text{pri}}\left(\mathbf{h}_{t-1}, \mathbf{b}_{t-1}; \theta\right) \tag{10}$$

$$\boldsymbol{\mu}_{\text{enc},t}, \boldsymbol{\sigma}_{\text{enc},t} = f_{\text{enc}}\left(\mathbf{x}_t, \mathbf{h}_{t-1}, \mathbf{b}_{t-1}; \boldsymbol{\phi}\right) \tag{11}$$

$$\boldsymbol{\mu}_{\text{dec},t}, \boldsymbol{\sigma}_{\text{dec},t} = f_{\text{dec}}\left(\mathbf{z}_t, \mathbf{h}_{t-1}, \mathbf{b}_{t-1}; \theta\right) \tag{12}$$

In addition to conditioning the model on belief maps, a further loss term is inserted, in order to optimize the affinity between ground-truth maps and those generated by the model. By sampling multiple displacements from the model, we obtain the sampled candidate belief map $\mathbf{b}'_{t-1}$, which identifies a probability distribution over local bin transitions. For each sampled displacement and subsequent location, we firstly index the corresponding grid bin, then the heat kernel value between the sampled next location and the $L \times L$ neighbourhood bin centres is used to fill the grid (see Fig. 2(b)). To build the non-ground truth belief maps, we only use the information about the position at x_{t-1}, and then draw N samples from our model. The aforementioned procedure allows the model to unroll the sub-grids, obtaining for every location a discrete probability density of possible transitions. Thus, it is possible to compare generated belief maps $\mathbf{b}'_{t-1}$ and ground-truth ones $\mathbf{b}_{t-1}$ by means of the KL divergence, exploiting the histogram loss term proposed by Ustinova and Lempitsky (2016). We add this contribution to the ELBO loss in Eq. (5) encouraging the model to be compliant to the collective behaviour of all agents. Such a divergence measure is multiplied by a constant k for loss balancing to ensure that its weight is comparable to the other loss components:

$$\mathcal{L} = \mathbb{E}_{q_{\phi,t}(\mathbf{z}_t)}\Big[\sum_{t=1}^{T}\Big(-\text{KL}\left(q_{\phi,t}(\mathbf{z}_t) \parallel p_{\theta,t}(\mathbf{z}_t)\right) + \log p_{\theta,t}(\mathbf{x}_t) + k\,\text{KL}(\mathbf{b}_{t-1} \parallel \mathbf{b}'_{t-1})\Big)\Big]. \tag{13}$$

4. Experiments

4.1. Datasets

We present experiments on different datasets to prove the robustness of our model on various scenarios and protocols. More specifically, we define multiple experiments on ETH (Pellegrini et al., 2009), UCY((Lerner et al., 2007)), Stanford Drone Dataset (Robicquet et al., 2016), STATS SportVU NBA,[3] Intersection Drone Dataset (inD) (Bock et al., 2020), and TrajNet++ (Kothari et al., 2021).

ETH-UCY. ETH (Pellegrini et al., 2009) consists of two scenes, *Eth* and *Hotel*, while UCY (Lerner et al., 2007) consists of three scenes, *Zara1*, *Zara2* and *Univ*. The benchmark contains different types of interactions among pedestrians and fixed obstacles such as buildings or parked cars.

[2] 5×5 belief maps along with the proposed global grid's partition guarantee that future displacements fall into the corresponding belief maps.

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

Table 1
Quantitative results of considered methods for ETH and UCY datasets. We report Average Displacement Error (ADE) and Final Displacement Error for unimodal methods and TopK and TopK FDE (with $K = 20$) for multi-modal ones. The results were obtained for $t_{obs} = 8$ and $t_{pred} = 12$ (in metres). The first block of experiments regards the using of data employed by S-GAN and STGAT models; the second one uses the SR-LSTM version of data while the last experiments are trained with the S-Ways protocol. On average, our model outperforms several methods showing a slightly worse FDE error when the S-Ways protocol is employed. No belief maps appear necessary for SR-LSTM data version.

	Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
(A)	S-LSTM (Alahi et al., 2016)	1.09/2.35	0.79/1.76	0.67/1.40	0.47/1.00	0.56/1.17	0.72/1.54
	S-GAN-P (Gupta et al., 2018)	0.87/1.62	0.67/1.37	0.76/1.52	0.35/0.68	0.42/0.84	0.61/1.21
	S-GAN (Gupta et al., 2018)	0.81/1.52	0.72/1.61	0.60/1.26	0.34/0.69	0.42/0.84	0.58/1.18
	Trajectron (Ivanovic and Pavone, 2019)	**0.59**/1.14	0.35/0.66	0.54/1.13	0.43/0.83	0.43/0.85	0.56/1.14
	SoPhie (Sadeghian et al., 2018b)	0.70/1.43	0.76/1.67	0.54/1.24	**0.30**/0.63	0.38/0.78	0.54/1.15
	Social-BiGAT (Kosaraju et al., 2019)	0.69/1.29	0.49/1.01	0.55/1.32	**0.30**/**0.62**	0.36/0.75	0.48/1.00
	Next (Liang et al., 2019)	0.73/1.65	**0.30**/0.59	0.60/1.27	0.38/0.81	0.31/0.68	0.46/1.00
	STGAT (Yingfan et al., 2019)	0.78/1.60	**0.30**/**0.54**	**0.51**/**1.08**	0.33/0.72	0.29/0.63	0.44/0.91
	A-VRNN (Ours)	0.73/1.45	0.34/0.65	0.53/1.14	0.33/0.69	**0.26**/**0.54**	0.44/0.89
	AC-VRNN (Ours)	0.61/**1.09**	**0.30**/0.55	0.58/1.22	0.34/0.68	0.28/0.59	**0.42**/**0.83**
(B)	SR-LSTM (Zhang et al., 2019)	0.63/1.25	**0.37**/**0.74**	**0.51**/**1.10**	0.41/0.90	0.32/0.70	0.45/0.94
	A-VRNN (Ours)	**0.60**/**1.18**	**0.37**/**0.74**	0.55/1.20	**0.39**/**0.83**	**0.27**/**0.59**	**0.44**/**0.91**
(C)	S-Ways (Javad et al., 2019)	**0.39**/**0.64**	0.39/0.66	**0.55**/**1.31**	0.44/**0.64**	0.51/0.92	0.46/**0.83**
	A-VRNN (Ours)	0.60/1.24	0.22/0.45	0.61/1.34	0.46/1.06	**0.30**/**0.67**	**0.44**/0.95
	AC-VRNN (Ours)	0.55/1.06	**0.18**/**0.26**	0.76/1.59	**0.37**/0.72	0.33/0.70	**0.44**/0.87

Table 2
Results for $t_{obs} = 8$ and $t_{pred} = 12$ on Stanford Drone Dataset (in metres). AC-VRNN significantly reduces TopK ADE and TopK FDE error metrics. Average NLL is the best one among all approaches while collision errors are below 1% for all methods.

Method	SDD				
	TopK ADE (↓)	TopK FDE (↓)	Avg NLL (↑)	Col-I (↓)	Col-II (↓)
S-GAN-P (Gupta et al., 2018)	0.65	1.26	−3.79	0.00	0.33
STGAT (Yingfan et al., 2019)	0.57	1.09	−2.70	0.00	0.40
DAG-Net (Monti et al., 2020)	0.54	1.07	−2.54	0.49	0.25
A-VRNN (Ours)	0.55	0.98	−1.11	**0.00**	**0.11**
AC-VRNN (Ours)	**0.51**	**0.90**	**−0.18**	0.16	0.22

Table 3
Results for $t_{obs} = 10$ and $t_{pred} = 40$ in feet on STATS SportVU NBA dataset.

Team	Method	STATS SportVU NBA				
		TopK ADE (↓)	TopK FDE (↓)	Avg NLL (↑)	Col-I (↓)	Col-II (↓)
ATK	STGAT (Yingfan et al., 2019)	9.94	15.80	−8.65	0.21	0.32
	Weak-Supervision (Zhan et al., 2019)	9.47	16.98	**−6.29**	0.57	0.20
	A-VRNN (Ours)	**9.32**	**14.91**	−7.60	**0.09**	**0.18**
DEF	STGAT (Yingfan et al., 2019)	7.26	11.28	−7.88	0.20	**0.27**
	Weak-Supervision (Zhan et al., 2019)	7.05	10.56	**−5.69**	0.70	0.57
	A-VRNN (Ours)	**7.01**	**10.16**	−6.70	**0.13**	0.43

Table 4
Results for $t_{obs} = 8$ and $t_{pred} = 12$ in metres on inD dataset.

Method	**inD**				
	TopK ADE (↓)	TopK FDE (↓)	Avg NLL (↑)	Col-I (↓)	Col-II (↓)
S-GAN (Gupta et al., 2018)	0.48	0.99	−1.84	**0.51**	0.55
STGAT (Yingfan et al., 2019)	0.48	1.00	−1.55	0.60	0.58
A-VRNN (Ours)	0.45	0.97	−1.69	0.61	**0.52**
AC-VRNN (Ours)	**0.42**	**0.80**	**−0.29**	0.78	0.61

Stanford Drone Dataset (SDD) (Robicquet et al., 2016). SDD is a large scale dataset, containing urban scenes of a university campus, streets and intersections, shot by a drone. More specifically, it is composed of 31 videos of 8 different scenarios. This dataset provides more complex scenes compared to the previous ones, involving various types of human interactions. We use the version proposed by TrajNet benchmark (Sadeghian et al., 2018a; Becker et al., 2018) which contains only pedestrian annotations. We split the training set into three sets for the learning process selecting 70% of data as training, 10% as validation and the remaining part as testing.

STATS SportVU NBA.[3] It consists of tracked trajectories of 10 basketball players (5 attackers, 5 defenders) during the 2016 NBA season monitoring 1600 matches. Each trajectory contains 50 time steps sampled at 5 Hz with x, y, and z coordinates expressed in feet. 40 time steps are used as observations and 10 time steps for predictions. All trajectories are normalized and shifted to obtain zero-centred sequences to the middle of court.

Intersection Drone Dataset (inD) (Bock et al., 2020). It captures four different German intersections from a bird's-eye-view perspective and contains more than 11000 trajectories of various road users (e.g., pedestrians, cars, cyclists) saved in 33 recordings. Data is collected at 25 Hz using a drone.

TrajNet++ (Kothari et al., 2021). It is a large scale interaction-centric trajectory prediction benchmark composed of a real-world dataset and a synthetic dataset. The real-world dataset contains selected trajectories of different datasets (ETH (Pellegrini et al., 2009), UCY (Lerner et al., 2007), WildTrack (Chavdarova et al., 2018), L-CAS (Yan et al., 2017) and CFF (Alahi et al., 2014)). This benchmarks defines a *primary* pedestrian per scene and his/her categorization into four different types: static, linear, interacting and non-interacting.

4.2. Metrics

TopK Average Displacement Error (TopK ADE): Average Euclidean distance over all estimated points and ground-truth positions of a trajectory as proposed in Pellegrini et al. (2009):

$$ADE = \sum_{i=1}^{\mathcal{P}} \sum_{t=T_{obs}+1}^{T_{pred}} \frac{\sqrt{(\hat{x}_t^i - x_t^i)^2 + (\hat{y}_t^i - y_t^i)^2}}{T_{pred} \cdot \mathcal{P}}; \quad (14)$$

TopK Final Displacement Error (TopK FDE): Average Euclidean distance between predicted and ground-truth final destinations:

$$FDE = \sum_{j=1}^{\mathcal{P}} \frac{\sqrt{(\hat{x}_{T_{pred}}^j - x_{T_{pred}}^j)^2 + (\hat{y}_{T_{pred}}^j - y_{T_{pred}}^j)^2}}{\mathcal{P}}. \quad (15)$$

$\mathcal{P}$ represents the number of pedestrians and T_{pred} is the predicted time horizon. The above metrics are evaluated using the top-k (or best-of-N) i.e., we sample N trajectories and consider the ADE and FDE of the lowest-error trajectory.

Average Log-Likelihood (Avg NLL): Average Log-Likelihood of ground truth trajectories over the predicted time horizon considering a distribution fitted with N output predictions. We compute this metric as in Kothari et al. (2021).

TopK Collisions: Similarly to Kothari et al. (2021), we consider two types of collisions, Col-I and Col-II, measuring the collisions of a pedestrian w.r.t his/her neighbours considering a fixed neighbourhood. Col-I (or prediction collision) uses the neighbours' predicted trajectories to check a collision, while Col-II relies on their ground-truth annotations. Nevertheless, since we use these metrics in a multi-modal context, we consider predictions with the lowest ADE (TopK ADE) for both primary and neighbours pedestrians. We report the percentage of collisions averaged over all test scenes.

4.3. Quantitative results

ETH-UCY. We evaluate our model using different versions of ETH-UCY datasets since multiple data and protocols are available for these scenes. Quantitative results are reported in Table 1. We indicate with AC-VRNN our full model including the hidden state refinement process and belief maps and with A-VRNN our model without belief maps. Firstly, we consider a leave-one-out training protocol (A) as in S-GAN (Gupta et al., 2018). Our model outperforms all baselines on *Eth* (FDE) and *Zara2* (TopK ADE and TopK FDE with $K = 20$) scenes and exhibits the best values on average metrics. AC-VRNN significantly outperforms A-VRNN suggesting the beneficial effect of belief maps conditioning. For the remaining scenes, slightly worse performance of AC-VRNN could be ascribed to the leave-one-out protocol since training belief maps may not entirely comply with test scenes increasing uncertainty for future predictions. SR-LSTM (Zhang et al., 2019) defines different *Eth* annotations considering 6 frames at $0.4s$ instead of 10 frames due to a frame rate issue of original annotations, affecting each cross-validation fold (B). In this case, our model outperforms SR-LSTM baseline or achieve comparable results on all scenes for both metrics. Finally, S-Ways (Javad et al., 2019) does not use a leave-one-out protocol. Each dataset is split into 5 subsets, using 4 subsets for training and the remaining ones for testing purpose (C). We achieve better performance on TopK ADE and slightly worse performance on TopK FDE. Without the leave-one-out protocol, AC-VRNN significantly outperforms A-VRNN on FDE suggesting the beneficial effect of belief maps conditioning.

Stanford Drone Dataset. To consider more complex urban scenarios, we test our model also on Stanford Drone Dataset. We compare our results with S-GAN-P (Gupta et al., 2018) and STGAT (Yingfan et al., 2019). As shown in Table 2, AC-VRNN outperforms A-VRNN version and both selected baselines. With more complex trajectories and scene topologies, our attentive module is able to better capture interactions among pedestrians and belief maps help to avoid incorrect behaviours following the prior distribution of displacements in the monitored scene.

Table 5
Results for $t_{obs} = 9$ and $t_{pred} = 12$ in metres on TrajNet++. For unimodal methods ADE and FDE metrics are reported while for multimodal ones we reported the TopK ADE and TopK FDE metrics with $K = 3$.

Method	TajNet++	
	ADE/TopK ADE (↓)	FDE/TopK FDE (↓)
S-LSTM (Alahi et al., 2016)	0.55	1.18
S-ATT (Vemula et al., 2018)	0.56	1.22
S-GAN (Gupta et al., 2018)	**0.51**	**1.09**
D-LSTM (Kothari et al., 2021)	0.57	1.23
AC-VRNN (Ours)	0.57	1.17

STATS SportVU NBA. Additionally, we test our model using basketball players trajectories whose dynamics are clearly different from ones exhibited by pedestrians in urban scenes. As reported in Table 3, our A-VRNN reduces TopK ADE and TopK FDE metrics on both offencive and defensive players trajectories compared to STGAT (Yingfan et al., 2019) and Weak-Supervision (Zhan et al., 2019). Avg NLL are similar for all methods, whereas collision errors given by A-VRNN are mainly smaller than the errors generated by competitive approaches. In this case, belief maps cannot properly steer future positions since basketball courts do not have obstacles and never-crossed areas. Moreover, basketball players do not typically follow a collective behaviour.

Intersection Drone Dataset. On InD dataset, we adopt the same evaluation protocol used for Stanford Drone Dataset considering, for each scene, 70% of data as training, 10% as validation while the remaining part for testing. We retain only pedestrians' trajectories and downsample each scene to obtain 20 time steps in 8 s. In Table 4 we compare our model to S-GAN (Gupta et al., 2018) and STGAT (Yingfan et al., 2019). AC-VRNN overcomes all the competitive methods on TopK ADE and TopK FDE and Avg NLL. S-GAN gives a smaller Col-I error with respect to AC-VRNN and S-GAN, while A-VRNN shows a smaller Col-II error.

TrajNet++. Finally, we test our model on TrajNet++ (Kothari et al., 2021) real-world dataset. The results are reported in Table 5 where ADE and FDE metrics are used for unimodal methods and TopK ADE and TopK FDE (with $K = 3$) metrics for multimodal ones. We find that our model reaches competitive performance with respect to other approaches, especially for TopK FDE. Our results are obtained by submitting the results to the evaluation server averaging the results on different types of scene considering only the real dataset. We compare AC-VRNN against published competitive approaches as competing with a lead-board that is updated every day is out of the scope of this quantitative analysis. Other methods results are reported from (Kothari et al., 2021; Liu et al., 2020). Since the Avg NLL for competitive methods is missing, we do not report this metric in Table 5. However, our method attains an Avg NLL of −8.33.

4.4. Ablation experiments

We also present an ablation study to show the contribution of different components of our model on the prediction task. In the following, we detail each component and report quantitative results in Tables 6 and 7.

Vanilla Variational Recurrent Network. We investigate the ability of Vanilla VRNNs to predict accurate trajectories on ETH, UCY and SDD datasets. This model does not consider any human interactions or prior scene knowledge. ETH scenes appear mainly affected by the lack of additional information while UCY scenes attain comparable results to our AC-VRNN model, especially for TopK ADE metric. Such a result highlights the importance of trajectory forecasting task to go beyond a time-series problem and the need of including contextual information about the scene, such as human interactions or experience gained in similar contexts.

Fig. 3. Illustration of predicted trajectories using AC-VRNN, baselines and competitive methods on *Eth* (left) and *Zara1* (middle) scenes of ETH and UCY datasets and *gates_0* and *deathCircle_1* of SDD (right).

Table 6
Ablation experiments showing TopK ADE and TopK FDE for $t_{obs} = 8$ and $t_{pred} = 12$ in metres on ETH, UCY and SDD datasets. AVG column reports average results for ETH and UCY datasets.

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Vanilla VRNN	0.79/1.61	0.46/0.94	0.55/1.20	0.34/0.75	0.26/0.58	0.48/1.02
GCN-VRNN	0.81/1.58	0.41/0.85	0.59/1.31	0.38/0.84	0.41/0.96	0.52/1.11
AC-VRNN w/o KLD	0.73/1.41	0.52/1.07	0.64/1.36	0.43/0.89	0.39/0.83	0.54/1.11
All-1 ADJ Matrix	0.77/1.52	0.37/0.73	0.55/1.19	0.34/0.75	0.26/0.58	0.46/0.95
kNN ADJ Matrix	0.76/1.54	0.47/0.99	0.57/1.26	0.42/0.95	0.26/0.58	0.50/1.01
A-VRNN (Ours)	0.73/1.45	0.34/0.65	**0.53/1.14**	**0.33**/0.69	**0.26/0.54**	0.44/0.89
AC-VRNN (Ours)	**0.61/1.09**	**0.30/0.55**	0.58/1.22	0.34/**0.68**	0.28/0.59	**0.42/0.83**

Table 7
Ablation experiments showing TopK ADE and TopK FDE for $t_{obs} = 8$ and $t_{pred} = 12$ for SDD dataset.

Method	SDD	
	TopK ADE (↓)	TopK FDE (↓)
Vanilla VRNN	0.56	1.15
GCN-VRNN	0.53	1.05
AC-VRNN w/o KLD	0.60	1.11
All-1 ADJ Matrix	0.57	1.11
kNN ADJ Matrix	0.73	1.43
A-VRNN	0.56	1.14
AC-VRNN ($L = 3$)	0.67	1.31
AC-VRNN ($L = 5$)	**0.51**	**0.92**
AC-VRNN ($L = 7$)	0.68	1.33

Table 8
Main hyperparameters used to train both AC-VRNN and A-VRNN models on tested datasets.

Hyperparameter	ETH/UCY	SDD	STATS SportVU NBA	TrajNet++	**inD**
Optimizer	Adam	Adam	Adam	SGD	Adam
Learning rate	10^{-3}	10^{-3}	10^{-3}	3×10^{-4}	$10^{-4}/5 \times 10^{-4}$
Batch size	16	16	32	8	16
Latent space size	16	16	32	16	16
Warm-up epochs	50	50	–	3	50

Hidden State Refinement with Graph Convolutional Neural Network. This experiment models interactions with a hidden state refinement based on a standard Graph Convolutional Networks (GCN) . The model has worse performance compared to AC-VRNN and Vanilla VRNN models on ETH and UCY datasets while obtains comparable results to AC-VRNN on SDD dataset. The experiments suggest that, for complex contexts, attention mechanisms are able to capture more useful information in order to model interactions among pedestrians compared to simple scenarios where interactions may be reduced.

AC-VRNN without KLD Loss on Belief Maps. To demonstrate the importance of KL-divergence loss on belief maps, we train our model without this term yet still conditioning the model on them. We obtain the worst results on all datasets proving that the network is not able to integrate belief maps information conditioning only VAE components. KL-divergence allows the network to generate displacement distributions similar to the ground-truth ones and to follow prior knowledge about local behaviours.

Adjacency Matrix. We also evaluate our model using different kinds of adjacency matrices to corroborate the use of the similarity one. We consider an *all-1* adjacency matrix where edges are equally weighted and all pedestrians in the scene are connected. This model attains good performance but slightly worse than the ones obtained with a similarity matrix on both ETH/UCY and SDD, proving that assuming the same importance for all involved agents negatively affects the results. k-NN matrix only considers nearby pedestrians. The neighbourhood is computed by sorting mutual distances between each pedestrian, retaining only the first k nearest neighbours (with $k = 3$), defined as a set S_i. Each element is set to 1 if $a_{i,j} \in S_i$, to 0 otherwise. k-NN matrix obtains quite the worst results on ETH and UCY datasets and performs poorly on SDD dataset. This experiment demonstrates that a

Table 9
Detailed description of each module of our AC-VRNN architecture.

Module	Architecture
Features extraction (trajectory)	Linear (2, 64) → LeakyReLU → Linear(64, 64) → LeakyReLU
Features extraction (belief map)	Linear (64, 64) → LeakyReLU
Prior	Linear(128, 64) → LeakyReLU
Mean	Linear(64, 16)
Log-variance	Linear(61, 16)
Encoder	Linear(192, 64) → LeakyReLU → Linear(64, 64) → LeakyReLU
Mean	Linear(64, 16)
Log-variance	Linear(64, 16)
Latent space	Linear(16, 64) → LeakyReLU
Decoder	Linear(192, 64) → LeakyReLU → Linear(64, 64) → LeakyReLU
Mean	Linear(64, 16) → HardTanH(−10, 10)
Log-variance	Linear(64, 16)
Recurrence	GRU(128, 64, 1)
Graph	GraphAttentionLayer(64, 64, hidden_units = 8, heads = 4, $\alpha = 0.2$) → BatchNorm1D → TanH

small neighbourhood is not able to capture interactions in large scenes where pedestrians show mutual influences also at long distances.

Belief Maps Dimension. Since belief maps define the probability that a pedestrian in a cell will move towards another one, it is important to consider a proper cell dimension. If we consider a fine-grained grid ($L = 3$), we could discard information about pedestrians whose displacement is greater than the defined one. Likewise, if we consider a course-grained grid ($L = 9$), outermost cells may not be properly filled. To select the best value of the parameter L, we test our model using different cell dimensions and found that $L = 5$ is the best choice for our datasets.

Hidden State Initialization. The hidden state initialization has a strong impact on the RNN training process. We experiment with three different initialization approaches:

- *Zero initialization*: a simple zero-tensor initialization.
- *Learned initialization*: a linear layer is trained to learn an optimal initialization.
- *Absolute coordinate initialization*: the tensor is initialized with the first absolute coordinates to provide spatial information to the learning process that is based on displacements generation.

We experimentally notice that the *absolute coordinate initialization* has a significant impact on the recurrent process leading to a performance improvement on ETH/UCY dataset and on SDD, while on STATS SportVU NBA InD and TrajNet++ the *zero initialization* is preferable.

Block Irregular Adjacency Matrix. AC-VRNN is based on a single Variational Recurrent Neural Network with shared parameters. To jointly compute a unique adjacency matrix for each time step, we build a block matrix where each block contains the matrix corresponding to a single scene, randomly chosen from the training dataset. Blocks can have different dimensions since a variable number of agents may be present in the scene.

4.5. Qualitative results

Fig. 3 presents some qualitative experiments, comparing our model with baselines and competitive methods. On *Eth*, GCN-VRNN, based on a Graph Convolutional Neural Network, generates trajectories that significantly drift from the ground-truth ones. On *Zara1*, all considered models are able to follow correct paths, but AC-VRNN appears more able to predict complex trajectory such as the entrance into a building, following the collective agents' behaviour. For SDD, we randomly select two scenes and show our model samples against competitive methods. All methods predict plausible paths, but AC-VRNN generates more realistic trajectories in some cases, following the sidewalk rather than crossing the road diagonally.

Long-term predictions. Since AC-VRNN is a completely generative model, it is possible to generate an unlimited number of future positions as well as creating trajectories without any observations. This could be especially useful for applications that require sampling a large number of trajectories to simulate realistic motion dynamics as required by synthetic scenarios mimicking real-life situations. Obviously, as the number of time steps increases, the predicted paths tend to drift from realistic ones, but our model qualitatively predicts plausible trajectories even after several time steps. To this end, we show in Fig. 4 some qualitative experiments considering up to 200 time steps.

Multimodal predictions. Fig. 5 depicts other qualitative examples generated by AC-VRNN model showing multiple paths to demonstrate the ability of our model to predict multi-modal trajectories. Finally, Fig. 6 shows probability distributions of future paths. When interactions among pedestrians is limited or absent, our model correctly predicts continuous linear paths. By contrast, the increasing number of human interactions leads the predictions to simulate complex patterns.

4.6. Implementation details

We train our model for 500 epochs on ETH-UCY and SDD, for 300 epochs on STATS SportVU NBA, for 300 epochs on inD and for 25 epochs on TrajNet++. Except for ETH/UCY, we re-train all competitive methods for the same number of epochs and report the best results after performing a hyperparameter search retaining the best model on the validation set. For ETH/UCY we report results from the original paper except for STGAT (Yingfan et al., 2019) that has been re-trained with the best hyperparameters proposed by the authors. We use gradient clipping set to 10. For SGD optimizer we use a momentum of 0.9. The RNN is a GRU with 1 layer and hidden size equals to 64. The attentive GNN has a hidden size of 8 with 4 attention heads. Each belief map during training is generated by sampling 100 displacements. In Eq. (13), k is set to 100 for all the datasets. Other hyperparameters that vary according to the dataset are reported in Table 8. In Table 9 an overall description of AC-VRNN architecture is reported.[4]

Warm-up on VRNN KL-Divergence. VRNN is trained with the ELBO loss that is composed of two terms: Negative Log-Likelihood and KL-Divergence. To correctly balance these two terms, we use a warm-up method that increases the weight in the range $[0, 1]$ of the KL-Divergence up to N epochs. After this learning period, we fix the KL weight to 1. This technique favours the reconstruction error during the early epochs in order to firstly teach the network to generate correct samples and then to approach both *encoder*'s and *prior*'s means and log-variances.

[4] For a more detailed explanation see Veličković et al. (2018) and https://github.com/Diego999/pyGAT.

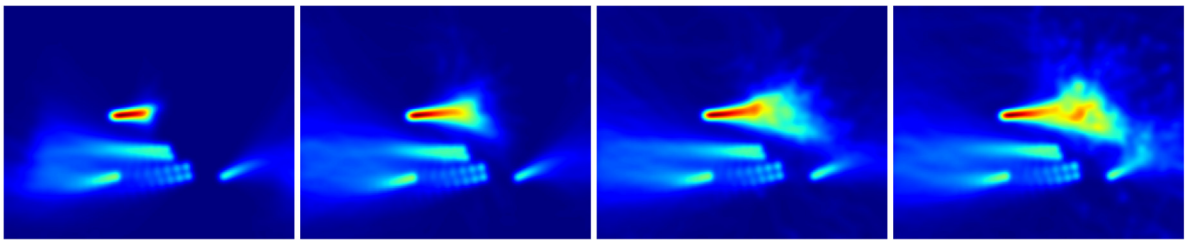

Fig. 4. Heatmaps of the predictions probability distribution for long-term predictions. The predictions are made for $t_{obs} = 8$ and $t_{pred} = 20, 60, 120$ and 200, respectively (from left to right). We select *Zara1* scene and observe that the trajectories are coherent with the scene topology.

Fig. 5. Multiple predictions of AC-VRNN trajectories to highlight the multi-modality nature of our model on ETH and UCY datasets.

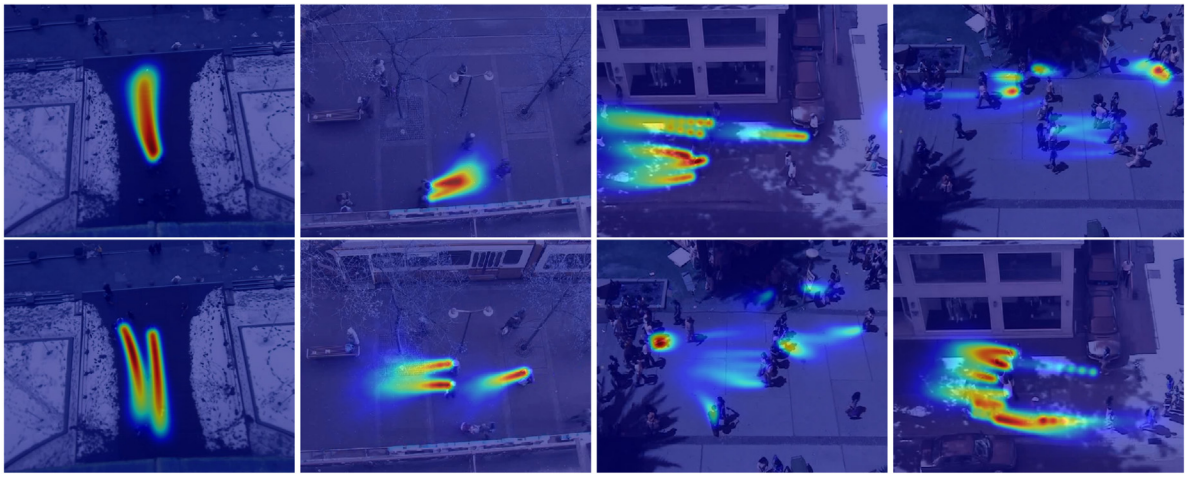

Fig. 6. Heatmaps representing probability distributions generated by our model for ETH and UCY datasets.

5. Conclusion

In this paper, we proposed a novel architecture for multi-future trajectory forecasting. Our framework uses VRNNs in a predictive setting. An attentive module includes interactions through a hidden state refinement process based on a graph neural network in an online fashion at a time step level. Finally, local belief maps encourage the model to follow a future displacement probability grid when the model is not confident about its prediction. We refer to our model as AC-VRNN and test it on several trajectory prediction datasets collected in different urban scenarios achieving the best performance compared to state-of-the-art methods. Our future work will be towards a detailed analysis of long-term predictions in order to deal with more complex and uncertain scenarios. Furthermore, an interesting aspect would be to include into the model additional scene context (e.g., depth data or WiFi/BLT signals) in order to design a multi-modal architecture to gain the advantage of multiple modalities.

CRediT authorship contribution statement

Alessia Bertugli: Conceptualization, Methodology, Software, Data curation, Visualization, Writing - original draft, Writing - review & editing, Formal analysis. **Simone Calderara:** Conceptualization, Methodology, Writing - review & editing, Supervision, Funding acquisition. **Pasquale Coscia:** Validation, Investigation, Data curation, Supervision,

Visualization, Writing - original draft, Writing - review & editing. **Lamberto Ballan:** Writing - review & editing, Supervision, Project administration, Funding acquisition, Resources. **Rita Cucchiara:** Supervision, Funding acquisition, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

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

References

Alahi, A., Goel, K., Ramanathan, V., Robicquet, A., Fei-Fei, L., Savarese, S., 2016. Social LSTM: Human trajectory prediction in crowded spaces. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Alahi, A., Ramanathan, V., Fei-Fei, L., 2014. Socially-aware large-scale crowd forecasting. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Alet, F., Weng, E., Lozano-Pérez, T., Kaelbling, L.P., 2019. Neural relational inference with fast modular meta-learning. In: Neural Information Processing Systems.

Becker, S., Hug, R., Hübner, W., Arens, M., 2018. RED: A simple but effective baseline predictor for the trajnet benchmark. In: European Conference on Computer Vision Workshops.

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., Eckstein, L., 2020. The inD dataset: A drone dataset of naturalistic road user trajectories at german intersections. In: IEEE Intelligent Vehicles Symposium.

Charlie, Y., Tang, Ruslan, Salakhutdinov, 2019. Multiple futures prediction. In: Neural Information Processing Systems.

Chavdarova, T., Baqué, P., Bouquet, S., Maksai, A., Jose, C., Bagautdinov, T., Lettry, L., Fua, P., Van Gool, L., Fleuret, F., 2018. WILDTRACK: A multi-camera HD dataset for dense unscripted pedestrian detection. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Chieh-Yu, C., Wenze, L., Hsin-Ying, H., Wen-Hao, Z., Yu-Shuen, W., Jung-Hong, C., 2018. Generating defensive plays in basketball games. In: ACM International Conference on Multimedia.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A., Bengio, Y., 2015. A recurrent latent variable model for sequential data. In: Neural Information Processing Systems.

Clevert, D.-A., Unterthiner, T., Hochreiter, S., 2016. Fast and accurate deep network learning by exponential linear units (ELUs). In: International Conference on Learning Representations.

Felsen, P., Lucey, P., Ganguly, S., 2018. Where will they go? predicting fine-grained adversarial multi-agent motion using conditional variational autoencoders. In: European Conference on Computer Vision.

Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A., 2018. Social GAN: Socially acceptable trajectories with generative adversarial networks. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Helbing, D., Molnar, P., 1995. Social force model for pedestrian dynamics. Phys. Rev. E 51 (5), 4282.

Hsin-Ying, H., Chieh-Yu, C., Yu-Shuen, W., Chuang, J.-H., 2019. BasketballGAN: Generating basketball play simulation through sketching. In: ACM International Conference on Multimedia.

Ivanovic, B., Pavone, M., 2019. The trajectron: probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In: IEEE/CVF International Conference on Computer Vision.

Javad, A., Jean-Bernard, H., Julien, P., 2019. Social ways: Learning multi-modal distributions of pedestrian trajectories with GANs. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition Workshops.

Jiachen, L., Hengbo, M., Masayoshi, T., 2019. Conditional generative neural system for probabilistic trajectory prediction. In: IEEE/RSJ International Conference on Intelligent Robots and Systems.

Kingma, D.P., Rezende, D.J., Mohamed, S., Welling, M., 2014. Semi-supervised learning with deep generative models. In: Neural Information Processing Systems.

Kingma, D.P., Welling, M., 2014. Auto-encoding variational Bayes. In: International Conference on Learning Representations.

Kipf, T., Fetaya, E., Wang, K.-C., Welling, M., Zemel, R., 2018. Neural relational inference for interacting systems. In: International Conference on Machine Learning.

Kosaraju, V., Sadeghian, A., Martín-Martín, R., Reid, I.D., Rezatofighi, S.H., Savarese, S., 2019. Social-bigat: Multimodal trajectory forecasting using bicycle-GAN and graph attention networks. In: Neural Information Processing Systems.

Kothari, P., Kreiss, S., Alahi, A., 2021. Human trajectory forecasting in crowds: A deep learning perspective. IEEE Trans. Intell. Transp. Syst. 1–15.

Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H.S., Chandraker, M.K., 2017. DESIRE: Distant future prediction in dynamic scenes with interacting agents. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Lerner, A., Chrysanthou, Y., Lischinski, D., 2007. Crowds by example. Comput. Graph. Forum 26 (3), 1186–1194.

Liang, J., Jiang, L., Murphy, K., Yu, T., Hauptmann, A., 2020. The garden of forking paths: Towards multi-future trajectory prediction. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Liang, J., Jiang, L., Niebles, J.C., Hauptmann, A.G., Fei-Fei, L., 2019. Peeking into the future: predicting future person activities and locations in videos. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Lisotto, M., Coscia, P., Ballan, L., 2019. Social and scene-aware trajectory prediction in crowded spaces. In: IEEE/CVF International Conference on Computer Vision Workshops.

Liu, Y., Yan, Q., Alahi, A., 2020. Social NCE: Contrastive learning of socially-aware motion representations. ArXiv abs/2012.11717.

Ma, Y., Zhu, X., Zhang, S., Yang, R., Wang, W., Manocha, D., 2018. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In: AAAI Conference on Artificial Intelligence.

Mehran, R., Oyama, A., Shah, M., 2009. Abnormal crowd behavior detection using social force model. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition Workshops.

Monti, A., Bertugli, A., Calderara, S., Cucchiara, R., 2020. DAG-Net: Double attentive graph neural network for trajectory forecasting. In: IAPR International Conference on Pattern Recognition.

Pellegrini, S., Ess, A., Schindler, K., van Gool, L., 2009. You'll never walk alone: Modeling social behavior for multi-target tracking. In: IEEE/CVF International Conference on Computer Vision.

Rios-Martinez, J., Spalanzani, A., Laugier, C., 2015. From proxemics theory to socially-aware navigation: A survey. Int. J. Soc. Robot. 7 (2), 137–153.

Robicquet, A., Sadeghian, A., Alahi, A., Savarese, S., 2016. Learning social etiquette: Human trajectory understanding in crowded scenes. In: European Conference on Computer Vision.

Rudenko, A., Palmieri, L., Herman, M., Kitani, K.M., Gavrila, D.M., Arras, K.O., 2020. Human motion trajectory prediction: A survey. Int. J. Robot. Res. 39 (8), 895–935.

Sadeghian, A., Kosaraju, V., Gupta, A., Savarese, S., Alahi, A., 2018a. Trajnet: Towards a benchmark for human trajectory prediction. ArXiv.

Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Savarese, S., 2018b. SoPhie: An attentive GAN for predicting paths compliant to social and physical constraints. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Sun, C., Karlsson, P., Wu, J., Tenenbaum, J.B., Murphy, K., 2019. Stochastic prediction of multi-agent interactions from partial observations. In: International Conference on Learning Representations.

Ustinova, E., Lempitsky, V., 2016. Learning deep embeddings with histogram loss. In: Neural Information Processing Systems.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks. In: International Conference on Learning Representations.

Vemula, A., Muelling, K., Oh, J., 2018. Social attention: modeling attention in human crowds. In: IEEE International Conference on Robotics and Automation.

Webb, E., Day, B., Andres-Terre, H., Lió, P., 2019. Factorised neural relational inference for multi-interaction systems. In: International Conference on Machine Learning.

Xue, H., Huynh, D.Q., Reynolds, M., 2018. SS-LSTM: A hierarchical LSTM model for pedestrian trajectory prediction. In: IEEE Winter Conference on Applications of Computer Vision.

Yan, Z., Duckett, T., Bellotto, N., 2017. Online learning for human classification in 3D lidar-based tracking. In: IEEE/RSJ International Conference on Intelligent Robots and Systems.

Yeh, R.A., Schwing, A.G., Huang, J., Murphy, K., 2019. Diverse generation for multi-agent sports games. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.

Yingfan, H., Huikun, B., Zhaoxin, L., Tianlu, M., Zhaoqi, W., 2019. STGAT: Modeling spatial-temporal interactions for human trajectory prediction. In: IEEE/CVF International Conference on Computer Vision.

Zanlungo, F., Ikeda, T., Kanda, T., 2011. Social force model with explicit collision prediction. EPL (Europhys. Lett.) 93 (6), 68005.

Zhan, E., Zheng, S., Yue, Y., Sha, L., Lucey, P., 2019. Generating multi-agent trajectories using programmatic weak supervision. In: International Conference on Learning Representations.

Zhang, P., Ouyang, W., Zhang, P., Xue, J., Zheng, N., 2019. SR-LSTM: State refinement for LSTM towards pedestrian trajectory prediction. In: IEEE/CVF International Conference on Computer Vision and Pattern Recognition.