Graph Conditional Variational Models: Too Complex for Multiagent Trajectories?

Yannick Rudolph
Leuphana University of Lüneburg, Germany
SAP SE, Machine Learning R&D, Berlin, Germany
yannick.rudolph@stud.leuphana.de
yannick.rudolph@sap.com

Ulf Brefeld
Leuphana University of Lüneburg, Germany
brefeld@leuphana.de

Uwe Dick
Leuphana University of Lüneburg, Germany
uwe.dick@leuphana.de

Abstract

Recent advances in modeling multiagent trajectories combine graph architectures such as graph neural networks (GNNs) with conditional variational models (CVMs) such as variational RNNs (VRNNs). Originally, CVMs have been proposed to facilitate learning with multi-modal and structured data and thus seem to perfectly match the requirements of multi-modal multiagent trajectories with their structured output spaces. Empirical results of VRNNs on trajectory data support this assumption. In this paper, we revisit experiments and proposed architectures with additional rigour, ablation runs and baselines. In contrast to common belief, we show that prior results with CVMs on trajectory data might be misleading. Given a neural network with a graph architecture and/or structured output function, variational autoencoding does not seem to contribute statistically significantly to empirical performance. Instead, we show that well-known emission functions do contribute, while coming with less complexity, engineering and computation time.

1 Introduction

Modeling the whereabouts of agents over space and time is an emerging research topic and important in many areas including the prediction of pedestrian movements (e.g. Alahi et al., 2016; Gupta et al., 2018), sports analytics (e.g. Le et al., 2017) and autonomous driving (e.g. Casas et al., 2020).

Trajectory data over multiple agents is usually highly multi-modal and comes with a structured output space: Firstly, assuming stochastic actions of each agent, there are multiple modes with respect to the density of future trajectories. Intuitively, one can think of different futures being all likely given a current setting. Secondly, the output space is naturally structured, since multiple trajectories have to be estimated per time step, while individual trajectories have to account for spatial dimensionality.

Graph architectures such as graph neural networks (GNNs, Scarselli et al., 2008), or graph networks in general (cf. Battaglia et al., 2018), are providing convincing architectural benefits for learning tasks on multiagent trajectories which come as unordered sets of individual trajectories. This is of special importance in learning scenarios where the number of agents are not fixed and/or the order of the agents is not predetermined, such as predicting the movements of pedestrians or players in team sports (e.g. Alahi et al., 2016; Gupta et al., 2018; Le et al., 2017; Felsen et al., 2018).

Conditional variational models (CVMs) such as conditional variational autoencoders (CVAEs, Sohn et al., 2015) and variational recurrent neural networks (VRNNs, Chung et al., 2015) have both

1st I Can't Believe It's Not Better Workshop (ICBINB@NeurIPS 2020), Vancouver, Canada.

been proposed to learn representations that help with structured prediction. Being an extension to variational autoencoders (VAEs, Kingma and Welling, 2014; Rezende et al., 2014), both architectures involve autoencoding an output variable, while conditioning on some input variable. In the case of CVAEs, the concept of the input variable is generic. In the case of VRNNs, a specific decomposition of sequential data and assumed latent variables result in time-step-wise conditioning with respect to the past. The inherent stochasticity of the models is suggested to help with "modeling variability" (Sohn et al., 2015) and making "diverse predictions" (Chung et al., 2015).

Recent research towards modeling multiagent trajectories proposes neural network architectures that combine components operating on graph structured data with conditional variational models. In this paper, we mainly focus on two examples: Ivanovic and Pavone (2019), who leverage spatiotemporal graphs with dynamic edges for neighboring agents with per agent CVAE and recurrent decoder for pedestrian prediction, and Yeh et al. (2019), who model interaction in team sports and propose to extend the VRNN model with GNNs. Both papers report empirical results which support the assumption that the employed CVMs are important for the performance of the architectures. By contrast, in this paper we provide evidence that a statistically significant contribution of the conditional variational model components to the empirical results for either architecture cannot be established.

Our approach is motivated by the observation that the structure of the output space of an individual trajectory in two dimensions is not overly complex. Revising an experiment on modeling handwriting data conducted in Chung et al. (2015), we show that mixture density networks (MDNs, Bishop, 1994), with Gaussian mixture models (GMMs) as output distributions, are indeed sufficient too capture the multi-modality and structure of two-dimensional trajectories. Lifting these results to interactions in multiagent trajectories with graph networks shows that the output space decomposes due to permutation equivariance and thus preserves the low-dimensional structure locally. We provide experimental evidence, that variational autoencoding does not result in higher likelihoods with respect to modeling multiagent trajectories when added to neural networks of certain structure.

We thus argue, that conditional variational autoencoding (as considered in this paper) is not the right choice for modeling multiagent trajectories. By contrast, structured emission functions such as MDNs, perform on par or even better and seem to render variational autoencoding unnecessary. Additionally, MDNs come with less complexity, engineering and compute.

Specifically, our contributions are: (i) We revise an experiment from Chung et al. (2015) (Section 5.2) and thus show, that MDN emission functions render the VRNN architecture unnecessary for modeling single two-dimensional trajectories. (ii) We provide a new ablation study with respect to the trajectron model (Ivanovic and Pavone, 2019) (Section 5.3) and new baseline experiments with graph VRNNs (Yeh et al., 2019) (Section 5.4). We thus also show, that MDN emission functions are more significant than variational autoencoding for modeling multiagent trajectories with graph architectures.

With this paper we hope to correct expectations towards CVMs for modeling trajectories to an appropriate level and mark the proper inclusion of CVM components for such problems as open research. Our results are intended to encourage practitioners (interested in fast returns) to focus on well understood and possibly simpler results, before incorporating components with increasing complexity, when modeling multiagent trajectories.

2 Related work

Mixture density networks (MDNs) have been introduced in Bishop (1994) and are by now well established. This holds especially for Gaussian component densities and thus GMM output distributions (cf. Bishop, 2006, for a textbook introduction). Combining RNNs with GMM emission for spatiotemporal tasks specifically, has been popularized by Graves (2013), who employed this architecture to model handwriting data from the IAM-OnDB dataset (cf. Marti and Bunke, 2002). The IAM-OnDB dataset was subsequently used in Chung et al. (2015) and is thus also used in our experiment in Section 5.2. Ha and Eck (2018) model structurally similar data (i.e. vector images) with an (unconditional) VAE paired with a mixture density RNN decoder and GMM emission.

Regarding multiagent trajectories, especially prior work on modeling team sports usually involves heuristics to account for player ordering (Le et al., 2017; Zhan et al., 2019). While there are arguments supporting this approach (Hobbs et al., 2019), the use of graph networks (Battaglia et al., 2018) or related attention mechanisms (Xu et al., 2015; Bahdanau et al., 2015) to model agent interaction is widespread. Notable early work includes Sukhbaatar et al. (2016) in general and Hoshen (2017) as well as Kipf et al. (2018) for multiagent trajectories in sports. Variational autoencoding approaches towards sports trajectories include Felsen et al. (2018) and Zhan et al. (2019). We consider Sanchez-Gonzalez et al. (2018) one of the earliest works on explict graph RNNs (GRNNs).

With respect to Ivanovic and Pavone (2019), follow-up work (Salzmann et al., 2020) uses the same overall design, but proposes to improve the latent variable component via the InfoVAE objective (Zhao et al., 2019). Preliminary ablation studies regarding the combination of CVAE and GMM emission in the architecture have been conducted in Schmerling et al. (2018) and Ivanovic et al. (2018). While they suggest the CVAE to provide a small benefit, the significance of the results is not clear. Casas et al. (2020) is another recent example for the combination of CVAEs and graph networks. Different from the architectures discussed in this paper however, a non-recurrent and non-probabilistic decoder is employed.

3 Conditional variational models

3.1 Conditional variational autoencoders

Variational autoencoders (VAEs, Kingma and Welling, 2014; Rezende et al., 2014) assume a generative model $p(\mathbf{x}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z})$, where a latent variable $\mathbf{z}$ is introduced to better model a data point $\mathbf{x}$. Specifically, VAEs come with a stochastic optimization method for a lower bound (the *evidence lower bound*, or ELBO) of the marginal log-likelihood $\log p(\mathbf{x})$. Distributions p_θ and q_ϕ involved in the ELBO formulation are modeled with neural networks. Parameters θ and ϕ can be learned via stochastic gradient ascent. In this framework, q_ϕ is considered a variational approximation to the true but unknown posterior of the latent variable.

Transferring variational autoencoding to conditional density estimation of some target variable $\mathbf{y}$, Sohn et al. (2015) proposes to optimize the ELBO of $\log p(\mathbf{y}|\mathbf{x})$ instead:

$$\log p_\theta(\mathbf{y}|\mathbf{x}) \geq \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{y})} \left[\log p_\theta(\mathbf{y}|\mathbf{x}, \mathbf{z})\right] - D_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})||p_\theta(\mathbf{z}|\mathbf{x})).$$

Here, $D_{\mathrm{KL}}(q||p)$ denotes the Kullback-Leibler (KL) divergence of two distributions q and p. When the prior $p_\theta(\mathbf{z}|\mathbf{x})$ and the variational posterior $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{y})$ are modeled as Gaussians, the KL divergence can be calculated in closed form. Optimization of the expected log-likelihood of the target variable $\mathbf{y}$ with respect to the variational parameters ϕ is done via reparameterization (cf. Mohamed et al., 2019). Latent variable instances $\mathbf{z}$ are thus reparameterized as $\mathbf{z} = g(\mathbf{x}, \mathbf{y}, \boldsymbol{\epsilon})$, with g being a deterministic function with respect to $\mathbf{x}$ and $\mathbf{y}$, while $\boldsymbol{\epsilon}$ is an auxiliary noise variable, usually taken to be componentwise standard normal, i.e. $\boldsymbol{\epsilon}_i \sim \mathcal{N}(0, 1)$.

Besides CVAEs, Sohn et al. (2015) proposes Gaussian stochastic neural networks (GSNNs) as a simpler alternative, where sampling stays consistent between learning and test time: GSNNs are obtained from CVAEs by setting the variational posterior equal to the prior, thereby reducing the model to straightforward density estimation with (learnable) noise injected during training.

As such, GSNNs make it natural to discuss two properties of CVAEs from an architectural perspective: Firstly, the target variable $\mathbf{y}$ is fed into lower parts of the model. Secondly, learnable noise is injected, enabling the network to bridge the gap between the approximate posterior (having access to the target) and the prior. We conjecture, that autoencoding the target variable via CVMs is not necessary for some tasks, on which the noise nevertheless helps with convergence. In a first hypothesis, which we will address in the experiments in Section 5, we state this even stronger:

Hypothesis 1: *Conditional autoencoding* ***does not*** *contribute towards density estimation for low-dimensional trajectories, at least if we account for its regularization properties via noise injection.*

3.2 Variational recurrent neural networks

The variational RNN (Chung et al., 2015) is a conditional variational model for sequential data, which assumes a specific decomposition of a latent variable $\mathbf{z}$ and a data sequence $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$ of length T. This decomposition leads to the following (empirical) lower bound on a sequence:

$$\log p_\theta(\mathbf{x}_{\leq T}) \geq \sum_{t=1}^{T} \left[\log p_\theta(\mathbf{x}_t|\mathbf{x}_{<t}, \mathbf{z}_{\leq t}) - D_{\mathrm{KL}}(q_\phi(\mathbf{z}_t|\mathbf{x}_{\leq t}, \mathbf{z}_{<x})||p_\theta(\mathbf{z}_t|\mathbf{x}_{<t}, \mathbf{z}_{<t}))\right],$$

where $\mathbf{z}_{\leq t} \sim q_\phi(\mathbf{z}_{\leq t}|\mathbf{x}_{\leq t})$ is sampled via ancestral sampling. Modeling the joined distribution $p_\theta(\mathbf{x}_{<t}, \mathbf{z}_{<t})$ with a recurrent neural network f, such that hidden representations are $h_t = f(\mathbf{x}_t, \mathbf{z}_t, h_{t-1})$, makes the ELBO eligible to optimization even for sequences with variable length. From an architectural standpoint, the VRNN features time-step-wise CVAEs, with $\mathbf{x}_t$ as output, and conditioning with respect to the past (of trajectory data as well as of latent variables).

Inter alia, Chung et al. (2015) experimented on density estimation for two-dimensional handwriting. Notably, the paper compared RNN and VRNN models with single Gaussian and GMM emission functions. Their results suggest that the VRNN-GMM model is best for the task. However, the experiment featured one run with train and validation splits only. In Section 5.2, we revisit this experiment with additional rigour, questioning the assumption that variational autoencoding and structured emission functions are indeed orthogonal. Specifically, we test our own conjecture, which we phrase as a second hypothesis:

Hypothesis 2: *Given an MDN with GMM emission function, conditional variational models* ***are not*** *significant components anymore regarding the task of modeling* single *two-dimensional trajectories.*

4 Modeling multiagent trajectories with graph architectures

4.1 Permutation equivariance via graph networks

Graph architectures such as graph networks (Battaglia et al., 2018) allow neural networks to process a set of inputs and aligned outputs independently of their permutation. Also, the input set does not have to be of fixed size. Specifically, graph networks assume a graph over the data, representing inputs, outputs and hidden states as nodes or agents and (directed) edges or relations.

Disregarding global attributes, each graph network layer consists of update and aggregation functions with respect to node and edge representations. Edge representations are updated with respect to their affiliated nodes. Node attributes are updated via aggregation functions which operate on incoming set of edges. When the aggregation functions are permutation invariant with respect to their input, the resulting node representations are permutation equivariant. Elementwise operations, such as sum or mean, are permutation invariant. Considering graph networks operating on a set of K partial trajectories $\mathbf{x}^{1:K}_{<t} = \{\mathbf{x}^1_{<t}, \mathbf{x}^2_{<t}, \dots, \mathbf{x}^K_{<t}\}$, e.g. to learn parameter estimates for densities of individual trajectories, thus leads to a third hypothesis, which we test experimentally in Section 5.4:

Hypothesis 3: *Results with respect to conditional variational models for density estimation over* single *trajectories* ***do*** *translate to* multiagent *trajectories, if computations are permutation equivariant.*

4.2 Graph conditional variational models

Specifically, we are looking at two recently proposed graph architectures for trajectory modeling, which feature either a CVAE with RNN decoder or VRNN components respectively:

Ivanovic and Pavone (2019) propose the trajectron, an architecture for modeling trajectories of pedestrians, given their past. Inputs to the model are position, velocity and acceleration. Agents are considered typed nodes, while edges are created heuristically when agents are close in space. Influences from these dynamic edges are modeled with separate RNNs in the spirit of structural-RNNs (Jain et al., 2016). Node and edge types are accounted for by parameter sharing over involved functions. Output of the model is a density estimation over velocities, which equal the differences in consecutive positions: $\Delta\mathbf{x}_t = \mathbf{x}_t - \mathbf{x}_{t-1}$. Results can readily be used to calculate a density over future positions via integration. While the full trajectron architecture features several more components, notable architectural choices (with respect to our research question) are the use of (structural-)RNN encoders for past trajectories, trajectory-wise CVAEs with discrete latent variables, a bidirectional RNN encoder for future trajectories and an RNN decoder with GMM emission functions. We refer to the original paper for more details.

Yeh et al. (2019) propose the graph VRNN (GVRNN) to model interactions in multiagent trajectories. Within the VRNN framework, this architecture models the parameterization of Gaussian prior and variational posterior, as well as the parameterization of a Gaussian decoder with GNNs which operate over individual trajectories. Featuring recurrence over agents, i.e. $h^k_t = f(\mathbf{x}^k_t, \mathbf{z}^k_t, h^k_{t-1})$, the model thus assumes a factorization of the latent variables over all (K) agents. With $\boldsymbol{\mu}$ denoting a mean vector, $\boldsymbol{\sigma}$ denoting a vector of standard deviations relating to a diagonal covariance matrix and $[\cdot, \cdot]$

indicating concatenation, the architecture amounts to the following variational autoencoder structure (abstracting from implementation details):

$$\begin{aligned}
\left[\boldsymbol{\mu}^{1:K}_{\mathrm{pri},t}, \boldsymbol{\sigma}^{1:K}_{\mathrm{pri},t}\right] &= \underset{\mathrm{pri}}{\mathrm{GNN}}\left(h^{1:K}_{t-1}\right), \\
\left[\boldsymbol{\mu}^{1:K}_{\mathrm{enc},t}, \boldsymbol{\sigma}^{1:K}_{\mathrm{enc},t}\right] &= \underset{\mathrm{enc}}{\mathrm{GNN}}\left(\left[\mathbf{x}^{1:K}_{t}, h^{1:K}_{t-1}\right]\right), \\
\left[\boldsymbol{\mu}^{1:K}_{\mathrm{dec},t}, \boldsymbol{\sigma}^{1:K}_{\mathrm{dec},t}\right] &= \underset{\mathrm{dec}}{\mathrm{GNN}}\left(\left[\mathbf{z}^{1:K}_{t}, h^{1:K}_{t-1}\right]\right).
\end{aligned}$$

The input to the model is positional data, while output densities are predicted over velocities also. Skip connections are employed and node and edge types are encoded via embeddings, which are passed to the GNNs. Notably, GMM emission functions were not experimented with.

Results on density estimation as well as metrics regarding prediction performance reported in the two papers, suggest that it is beneficial to combine graph architectures with CVMs. We argue however, that the significance of these results was not established. Thus, we conduct a further ablation study with respect to the trajectron in Section 5.3 and run experiments regarding GVRNN architectures with and without GMM emission functions on synthetic data in Section 5.4. We thereby gather evidence regarding CVMs in specific graph architectures, aiming at a fourth (and last) hypothesis:

Hypothesis 4: *Graph architectures modeling multiagent trajectories with CVM components* ***do not*** *generally result in better log-likelihoods than alike models without the CVM component.*

5 Experiments

5.1 About evaluation

We evaluate the modeling capacity of trajectory models via log-likelihood estimation on unseen test data $\mathcal{D}_{\mathrm{test}}$. For models with Gaussian or GMM emission not involving stochastic auxiliary variables, this is done analytically by calculating $\mathbb{E}_{\mathcal{D}_{\mathrm{test}}}\left[\sum_{t=t_0}^{T} \log p(\mathbf{x}_t|\mathbf{x}_{<t})\right]$. In our experiments on synthetic data in Section 5.4 we set $t_0 = 2$, while following the evaluation scheme of the reference experiments in Section 5.2 and 5.3. For variational models, we report the expectation of the sum over ELBOs per time step instead. For the experiment on modeling handwriting, we further report an importance sampled log-likelihood estimate, as has been done in the original experiment (and as originally proposed in Rezende et al., 2014). For the models with prior noise, i.e. GSNNs, we simply approximate the log-likelihood via heuristically injecting mean noise (alike the usual procedure for dropout, cf. Srivastava et al., 2014).

5.2 IAM-OnDB: modeling single trajectories with a VRNN model

In this section, we revise the experiment with respect to modeling handwriting from the IAM-OnDB dataset (cf. Marti and Bunke, 2002), as conducted by Chung et al. (2015). While the data comes with an additional binary indicator relating to pen lifting (which is modeled alongside), we consider the IAM-OnDB dataset an example of two-dimensional single trajectory data.

As Chung et al. (2015) used the experimental setting of Graves (2013), the data originally was split into training and validation sets only. Further, the experiment only consisted of one data split. This introduced the possibility of overfitting on the validation set, especially as early stopping was employed. For our reimplementation of the experiment, we considered 5 data splits, always training on $3/5$ of the data, validating on $1/5$ and testing the models with the best validation results on the last $1/5$ of the data. We split the trajectories over writer IDs, as was done in the data splits used by Chung et al. (2015).

We conducted our experiment faithfully following the model description in the paper, as well as according to code provided by the authors.[1] Specifically, we applied differencing to the time series and conducted normalization with respect to the training statistics. An LSTM cell (Hochreiter and Schmidhuber, 1997) with *tanh* activation functions is used in the RNN. For the variational methods, no further regularization is conducted with respect to the KL divergence term. Feature extractors (one

[1] https://github.com/jych/nips2015_vrnn

layer neural networks) for inputs $\mathbf{x}$ and latent variables $\mathbf{z}$ are learned, as they have been considered crucial by Chung et al. (2015).

Our reimplementation was done in PyTorch (Paszke et al., 2019). Different from Chung et al. (2015), we did not choose to model Gaussians within the emission function with full covariance matrices. Recent work suggests this to be hindering optimization (cf. Rybkin et al., 2020). We used diagonal Gaussians instead for both, Gaussian as well as GMM emission. Further, we bounded the variances from below (e.g. as in Goyal et al., 2017). This allowed us to forgo further weight normalization or gradient clipping as well as to use a higher learning rate (0.001, using Adam, Kingma and Ba, 2014). We further used a larger batch size (128 instead of 64), to make use of modern hardware. Next to the models from the original experiment, we included two GSNN variants (one with Gaussian and one with GMM emission), as well as a VRNN-I model with GMM emission. The VRNN-I is a model, in which the VRNN prior is input-*independent*. It has been proposed by Chung et al. (2015), to test whether conditional priors are necessary.

Running our models on the *original* data splits, showed comparable results to the original experiment, although our implementation results in slightly better log-likelihoods overall. However, we did not observe the reported performance gap between the VRNN and RNN models with GMM emission functions. Test results on *our* data splits can be seen in Figure 1. The results do confirm these preliminary findings. While a paired t-test shows that differences between VRNN-Gauss and RNN-GMM are highly significant (p-value < 0.0001) a paired t-test regarding the RNN-GMM and VRNN-GMM log-likelihoods, indicates that there is no significant difference between models. At the same time, mean KL divergences of the VRNN models with GMM emissions are considerably smaller than the KL divergences of the VRNN models with Gaussian emission (see Table 1). Note, that the effect of importance sampling for the variational GMM models is thus negligible and can only be seen in the figure at higher zoom factors. The performance of GSNNs on this task is comparable to that of RNNs. Noise alone does not seem to increase log-likelihoods even for Gaussian emission.

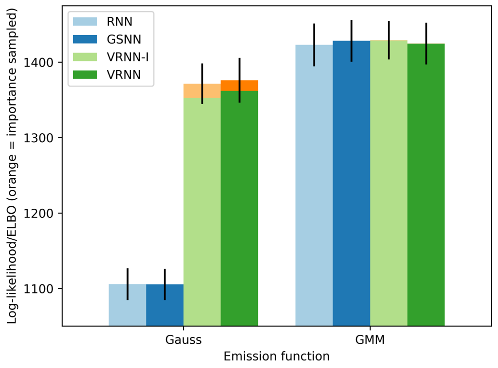


Figure 1: Log-likelihood, respectively ELBO, of compared models on the IAM-OnDB handwriting dataset (mean results over 5 test splits with standard errors; orange = importance sampling)

Table 1: Mean KL divergences of VRNN models over the IAM-OnDB test splits

Model class	Emission function	$D_{\mathrm{KL}}(q\|\|p)$
VRNN-I	Gauss	51.73
VRNN	Gauss	67.44
VRNN-I	GMM	0.32
VRNN	GMM	0.08

5.3 The trajectron: an ablation study with respect to its CVAE component

In this section, we report an ablation study on the full trajectron model as proposed in Ivanovic and Pavone (2019). The paper experiments with respect to pedestrian trajectory datasets ETH (Pellegrini et al., 2009) and UCY (Lerner et al., 2007), which have been preprocessed into coordinates and split into train, validation and test sets by Gupta et al. (2018). In total, there are splits for 5 different scenes.

The code of the model, as well as the training and evaluation procedures including all hyperparameters, has been made available by the authors.[2] This made it possibly to readily conduct an ablation study with respect to the density modeling contribution of the CVAE component in the full model.

Specifically, we disabled the CVAE component of the trajectron to obtain two model variants: For the first model variant, we interchanged the CVAE with a GSNN component. For the second model variant, we removed the latent variable component altogether. We did not conduct any additional hyperparameter tuning, but rather trained our models with the parameters chosen for the trajectron.

Table 2: Log-likelihood and runtime evaluation over all 5 pedestrian datasets (SE = standard error)

	Validation		Testing		Runtime (%)	
Latent variable component	Mean	SE	Mean	SE	Mean	SE
CVAE	38.53	±0.91	29.34	±7.26	100.00	0.00
GSNN	39.06	±0.72	31.51	±4.89	77.68	±1.98
None	38.39	±1.00	31.34	±5.34	45.13	±5.33

Test and validation log-likelihood results of the fully trained models are reported in Table 2. Here, all log-likelihood expectations are exact, due to integration over the low-dimensional categorical latent variables. We further report the runtime of our model variants compared to the full trajectron (in percent). A paired t-test shows no significant difference in the model variants with respect to their mean log-likelihoods over all five datasets. However, the runtimes of the simpler models are significantly lower when compared to the trajectron. Notably, removing the CVAE altogether reduces the mean runtime by over 50% without any further optimization.

5.4 GVRNNs: modeling synthetic multiagent trajectories

In this section, we report an experiment regarding the modeling capacity of GVRNNs (Yeh et al., 2019) with respect to small scale synthetic multiagent trajectories. Specifically, we compare the density estimation capabilities of simpler GRNN models (cf. Sanchez-Gonzalez et al., 2018) to GVRNNs, again including a version in which we interchanged the CVM component with a GSNN. Other than Yeh et al. (2019), we experiment with both, Gaussian as well as GMM emission functions.

We sampled the dataset from trajectories of interacting reinforcement learning (RL) agents, trained within a confined two-dimensional space. Specifically, we chose a continuous hunter–prey scenario (referred to as *simple tag*) from the *multi-agent particle environment* (Mordatch and Abbeel, 2018). Within the scenario, we constrained the entities to two hunters and one prey agent only. We deem this one of the simplest settings for the agents to show coordinated behaviour. All agents were trained with the MADDPG (Lowe et al., 2017) RL algorithm. To obtain more sensible trajectories, we introduced reward shaping via the L_2 distance between agents and added a further penalty to soft-force the agents within a square area. Even though the MADDPG algorithm is in principle deterministic, data generation was conducted with training noise. The dataset thus contains stochasticity. Training, validation and test episodes were sampled with 25 time steps each. We constructed 5 data splits, such that we trained models on 1000 episodes, validated on 2000 episodes and tested on 20000 episodes each. See Figure 2 for an example of trajectories and interaction.

Like Yeh et al. (2019), we used positional data as input and modeled the density of next step velocities. We further used feature extractors on inputs and latent variables as in Chung et al. (2015). For graph networks, we choose to use interaction networks (Battaglia et al., 2016), with learned embeddings for node and edge types. We however used elementwise differences between nodes for the edge updates, inspired by the importance of the relative positions of agents. This resulted in better performance during preliminary experiments. We further use a GRU cell (Cho et al., 2014) for recurrence.

The overall architectures of our models were designed such that all models are comparable in parameter number to the trajectory models employed by Chung et al. (2015). That is, models have around 8 million parameters, with variational models granted a bit more each. For optimization we resorted to Adam and introduced gradient norm clipping (Pascanu et al., 2013). A small learning rate decay (gamma=0.999) was added for stability and convergence, after an initial parameter sweep. We

[2]https://github.com/StanfordASL/Trajectron

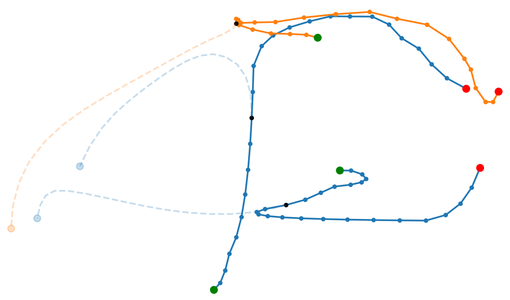
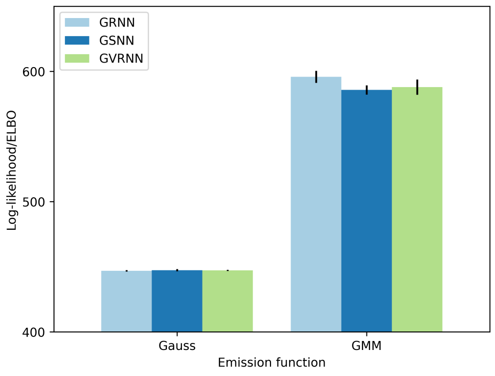


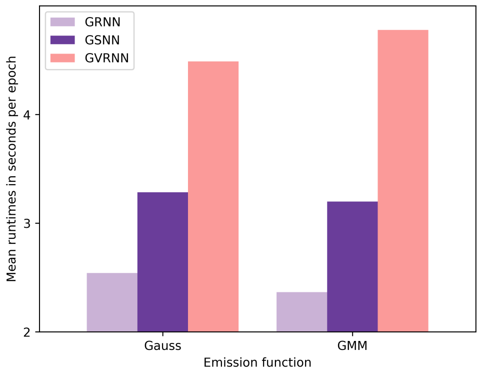
Figure 2: Example episode with 25 time steps from the synthetic data trajectories. Dark lines denote ground truth, while light dotted lines denote roll-outs with mean predictions from a fully trained GRNN-GMM model. Prey (orange) and hunters (blue) start and end their trajectories at the green and red dots respectively. Roll-outs start after 10 observations (indicated by black dots). The figure is supposed to provide an intuition about agent behaviour, stochasticity in the data, as well as predictions over multi-modal futures.

further conducted a grid search over initial learning rates and weight decay parameters on the first of our data splits, resulting in higher weight decay parameters for the models with Gaussian emission.

Log-likelihood results on test splits are shown in Figure 3a. Again, all models with GMM emission resulted in much better density estimates than their Gaussian counterparts. Indeed, only the differences with respect to emission functions are significant according to paired t-tests. Even though a KL scaling parameter was annealed (cf. Sønderby et al., 2016) from 0 to 1 over the first 200 epochs for the GVRNN models, the final KL divergence terms remained very small compared to the log-likelihood: smaller than 10^{-5} and 10^{-7} respectively for Gaussian and GMM emission. This led us to forgo importance sampling for these results as increases in the bound are deemed negligible.



(a) Log-likelihoods, respectively ELBO for GVRNN (mean results over 5 test splits with standard errors)



(b) Runtimes per epochs in seconds (note that Gaussian models need less epochs to reach their results)

Figure 3: Mean log-likelihood or ELBO as well as runtimes evaluated for synthetic trajectories

Figure 3b shows mean runtimes per epoch for the different models. As in the trajectron ablation study in Section 5.3, the models without conditional variational components are considerably faster.

GSNN models do not seem to provide any benefit on the synthetic trajectories. However, different from the reimplementation of the IAM-OnDB experiment in Section 5.2, we do not observe a performance gain with CVMs for Gaussian emission on this task either. The density estimation capabilities of our models seem to only depend on the choice of emission function.

6 Discussion

In this section, we summarize our findings and relate the empirical results from Section 5 to the four hypotheses established in Sections 3 and 4. Connections to further related literature are pointed out.

Regarding Hypothesis 1, which states that conditional autoencoding does not contribute towards density estimation for low-dimensional trajectories, we make the following observations: Statistical evidence gathered with respect to the contribution of the VRNN components for models with Gaussian emission on *single* trajectories allows us to reject the hypothesis in its broad scope. With regards to this, our results support the findings in Chung et al. (2015). Nevertheless, models without a conditional prior might be considered instead of the full VRNN model, as the performance of the VRNN-I model suggests. Noise regularization however, does not seem to be crucial for trajectory data, as GSNN models perform on par with deterministic models over all considered experiments.

With regards to Hypothesis 2, the IAM-OnDB experiment provides further evidence which implies that MDN emission dominates the contribution of conditional variational models for *single* trajectories: Modeling the emission via GMMs seems to render the contribution of CVMs for single trajectory modeling statistically insignificant. As such, this leads us to reject Hypothesis 1 only in the case of a weak decoder with single Gaussian emission. These findings are in line with literature on VAEs having problems with autoregressive or/and powerful decoders (cf. Chen et al., 2017; Zhao et al., 2019). See Bowman et al. (2015) and Fraccaro et al. (2016) for early results on sequential data.

Evidence with respect to Hypothesis 3 is mixed: The above results on *single* trajectories seem to translate only in part to *multiagent* trajectories. On the one hand, we do not observe any significant contribution of CVMs in case of our experiments with GVRNNs with Gaussian emission. With respect to that, the hypothesis seems false. However, as this partial result is only supported on one dataset, it might not be conclusive. Regarding Gaussian emission, Yeh et al. (2019) reports better log-likelihood results on one of two datasets for comparable variational models (i.e. typed GVRNNs modeling interactions). On the other hand, the significant contribution of GMM emission does translate to graph architectures in our experiments. For model performance, this is the important part.

Lastly, both, the experiments with trajectron variants as well as the experiment with GVRNNs on synthetic data do not allow us to reject Hypothesis 4. It thus stays likely, that graph architectures modeling multiagent trajectories with CVM components do not generally result in better log-likelihoods than alike models without a CVM component. Given the empirical evidence, we conjecture this to be especially true in the case of architectures which make use of MDN emission functions.

Broader Impact

This paper revises possibly misleading results in prior work with respect to conditional variational models such as CVAEs and VRNNs used for (multiagent) trajectory modeling. Both CVAEs and VRNNs are highly influential models (measured in citation), while multiagent trajectory data receives a great deal of attention in current research. At the same time, proposed models are becoming increasingly complex, which renders attribution of model performance to individual components more difficult.

This paper corrects expectations towards CVMs for trajectory modeling, identifies the proper inclusion of CVM components for trajectory modeling as open research question and encourages practitioners to rather run models without variational components for the time being. We thus believe that, both, researchers and practitioners may benefit from our contribution.

Acknowledgments and Disclosure of Funding

We would like to thank Boris Ivanovic for his helpful insights and comments with respect to our ablation study on the trajectron model. Further, we would like to thank Johannes Höhne for being essential in driving forward the research collaboration between SAP SE and the Leuphana University of Lüneburg, of which this paper is one result. In addition, Yannick Rudolph would like to thank Johannes Höhne for his fundamental supervision at SAP SE, as well as Marius Lehne and Christian Reisswig at SAP SE for their help and support while writing this paper.

References

Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human Trajectory Prediction in Crowded Spaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–971, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate. In *International Conference on Learning Representations*, 2015.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction Networks for Learning about Objects, Relations and Physics. In *Advances in Neural Information Processing Systems*, pages 4502–4510, 2016.

Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational Inductive Biases, Deep Learning, and Graph Networks. *arXiv preprint arXiv:1806.01261*, 2018.

Christopher M. Bishop. Mixture Density Networks. 1994.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, New York, 2006.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Sergio Casas, Cole Gulino, Simon Suo, Katie Luo, Renjie Liao, and Raquel Urtasun. Implicit Latent Variable Model for Scene-Consistent Motion Forecasting. In *European Conference on Computer Vision*, 2020.

Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational Lossy Autoencoder. In *International Conference on Learning Representations*, 2017.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv preprint arXiv:1406.1078*, 2014.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A Recurrent Latent Variable Model for Sequential Data. In *Advances in Neural Information Processing Systems*, pages 2980–2988, 2015.

Panna Felsen, Patrick Lucey, and Sujoy Ganguly. Where Will They Go? Predicting Fine-Grained Adversarial Multi-Agent Motion using Conditional Variational Autoencoders. In *European Conference on Computer Vision*, pages 732–747, 2018.

Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential Neural Models with Stochastic Layers. In *Advances in Neural Information Processing Systems*, pages 2199–2207, 2016.

Anirudh Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. Z-Forcing: Training Stochastic Recurrent Networks. In *Advances in Neural Information Processing Systems*, pages 6713–6723, 2017.

Alex Graves. Generating Sequences With Recurrent Neural Networks. *arXiv preprint arXiv:1308.0850*, 2013.

Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially Acceptable Trajectories With Generative Adversarial Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264, 2018.

David Ha and Douglas Eck. A Neural Representation of Sketch Drawings. In *International Conference on Learning Representations*, 2018.

Jennifer Hobbs, Matthew Holbrook, Nathan Frank, Long Sha, and Patrick Lucey. Improved Structural Discovery and Representation Learning of Multi-Agent Data. *arXiv preprint arXiv:1912.13107*, 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

Yedid Hoshen. Vain: Attentional Multi-Agent Predictive Modeling. In *Advances in Neural Information Processing Systems*, pages 2701–2711, 2017.

Boris Ivanovic and Marco Pavone. The Trajectron: Probabilistic Multi-Agent Trajectory Modeling With Dynamic Spatiotemporal Graphs. In *International Conference on Computer Vision*, pages 2375–2384, 2019.

Boris Ivanovic, Edward Schmerling, Karen Leung, and Marco Pavone. Generative Modeling of Multimodal Multi-human Behavior. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-RNN: Deep Learning on Spatio-Temporal Graphs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5308–5317, 2016.

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*, 2014.

Thomas Kipf, Ethan Fetaya, Kuan-Chieh Wang, Max Welling, and Richard Zemel. Neural Relational Inference for Interacting Systems. In *International Conference on Machine Learning*, pages 2693–2702, 2018.

Hoang M. Le, Yisong Yue, and Peter Carr. Coordinated Multi-Agent Imitation Learning. In *International Conference on Machine Learning*, pages 1995–2003, 2017.

Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by Example. *Computer Graphics forum*, 3 (26), 2007.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems*, pages 6379–6390, 2017.

U.-V. Marti and Horst Bunke. The IAM-database: an English sentence database for offline handwriting recognition. *International Journal on Document Analysis and Recognition*, 5(1):39–46, 2002.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte Carlo Gradient Estimation in Machine Learning. *arXiv preprint arXiv:1906.10652*, 2019.

Igor Mordatch and Pieter Abbeel. Emergence of Grounded Compositional Language in Multi-Agent Populations. In *AAAI Conference on Artificial Intelligence*, 2018.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the Difficulty of Training Recurrent Neural Networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019.

Stefano Pellegrini, Andreas Ess, K. Schindler, and Luc van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *IEEE International Conference on Computer Vision*, 2009.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *International Conference on Machine Learning*, pages 1278–1286, 2014.

Oleh Rybkin, Kostas Daniilidis, and Sergey Levine. Simple and Effective VAE Training with Calibrated Decoders. *arXiv preprint arXiv:2006.13202*, 2020.

Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control. *arXiv preprint arXiv:2001.03093*, 2020.

Alvaro Sanchez-Gonzalez, Nicolas Heess, Jost Tobias Springenberg, Josh Merel, Martin Riedmiller, Raia Hadsell, and Peter Battaglia. Graph Networks as Learnable Physics Engines for Inference and Control. In *International Conference on Machine Learning*, pages 4470–4479, 2018.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

Edward Schmerling, Karen Leung, Wolf Vollprecht, and Marco Pavone. Multimodal Probabilistic Model-based Planning for Human-robot Interaction. In *IEEE International Conference on Robotics and Automation*, 2018.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning Structured Output Representation using Deep Conditional Generative Models. In *Advances in Neural Information Processing Systems*, pages 3483–3491, 2015.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. Ladder Variational Autoencoders. In *Advances in Neural Information Processing Systems*, pages 3738–3746, 2016.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1): 1929–1958, 2014.

Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus. Learning Multiagent Communication with Backpropagation. In *Advances in Neural Information Processing Systems*, pages 2244–2252, 2016.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.

Raymond A. Yeh, Alexander G. Schwing, Jonathan Huang, and Kevin Murphy. Diverse Generation for Multi-Agent Sports Games. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

Eric Zhan, Stephan Zheng, Yisong Yue, Long Sha, and Patrick Lucey. Generating Multi-Agent Trajectories using Programmatic Weak Supervision. In *International Conference on Learning Representations*, 2019.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Balancing Learning and Inference in Variational Autoencoders. In *AAAI Conference on Artificial Intelligence*, volume 33, pages 5885–5892, 2019.