Spatio-Temporal Neural Networks for Space-Time Series Forecasting and Relations Discovery

Ali Ziat, Edouard Delasalles, Ludovic Denoyer, Patrick Gallinari

Spatio-Temporal Neural Networks for Space-Time Series Forecasting and Relations Discovery

Ali Ziat, Edouard Delasalles, Ludovic Denoyer, Patrick Gallinari

Spatio-Temporal Neural Networks for Space-Time Data Modeling and Relation Discovery

Edouard Delasalles[1], Ali Ziat[1,2], Ludovic Denoyer[1] and Patrick Gallinari[1]

[1] Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6, F-75005 Paris, France

[2]Vedecom Institute, Eco-Mobility Department, 77000, Versailles, France

firstname.name@lip6.fr

Abstract. We introduce a dynamical spatio-temporal model formalized as a recurrent neural network for modeling time series of spatial processes, i.e. series of observations sharing temporal and spatial dependencies. The model learns these dependencies through a structured latent dynamical component, while a decoder predicts the observations from the latent representations. We consider several variants of this model, corresponding to different prior hypothesis about the spatial relations between the series. The model is used for the tasks of forecasting and data imputation. It is evaluated and compared to state-of-the-art baselines, on a variety of forecasting and imputation problems representative of different application areas: epidemiology, geo-spatial statistics and car-traffic prediction. The experiments also show that this approach is able to learn relevant spatial relations without prior information.

Keywords: time-series, spatio-temporal, forecasting, data imputation, deep learning, neural networks

1. Introduction

Time series exhibiting spatial dependencies are present in many domains including ecology, meteorology, biology, medicine, economics, traffic, and vision. The observations can come from multiple sources e.g. GPS, satellite imagery, video cameras, etc. Several difficulties arise when modeling spatio-temporal data, among them: 1) their size: sensors can cover very large space and temporal lags; 2) the complexity of the underlying generation process, which might be highly non linear; and 3)

Received xxx
Revised xxx
Accepted xxx

the inherent uncertainty of the measurements: sensors are not perfect, and data points are frequently missing or noisy. Answering these challenges, i.e. reducing the spatial dimensionality, uncovering the underlying data generation process, and modeling data uncertainty naturally leads to consider latent dynamic models. This has been exploited both in statistics (Cressie & Wikle 2011) and in machine learning (ML) (Bahadori et al. 2014, Koppula & Saxena 2013).

Deep learning has also developed a whole range of dynamic latent models for capturing relevant information in sequences. Recurrent neural networks (RNN) and their many variants have been used in different contexts for sequence classification, sequence to sequence prediction, sequence generation and many other tasks (Bengio 2008, Chung et al. 2015*a*, Li et al. 2015). These models are able to capture meaningful features of the sequential data generation processes, but the spatial structure, essential in many applications, has been seldom considered in Deep Learning. Very recently, convolutional RNNs (SHI et al. 2015, Srivastava et al. 2015) and video pixel networks (Kalchbrenner et al. 2017) have been used to handle both spatiality and temporality, but they have mainly been designed for the restrictive case of video applications.

We introduce a general class of deep spatio-temporal models for time series of spatial processes. They allow us to explicitly model both spatial and temporal dependencies. In the paper we focus on two tasks: forecasting and imputation (i.e. inferring missing values). The model, denoted Spatio-Temporal Neural Network (STNN), is designed to capture the dynamics and correlations in multiple series at the spatial and temporal levels. This is a dynamical system model with two components: one for capturing the spatio-temporal dynamics of the process into latent states, and one for decoding these latent states into actual series observations. The model is tested and compared to state of the art alternatives, including recent RNN approaches, on several datasets for imputation and forecasting tasks. Tests were performed on time series coming from various domain: health, traffic, meteorology and oceanography. Besides a quantitative evaluation on forecasting and imputation tasks, the ability of the model to discover relevant spatial relations between series is also analyzed.

The paper is organized as follows: in section 2 we introduce the related work in machine learning and spatio-temporal statistics. The model is presented for the forecasting task in sections 3 and 4 with its different variants, and for the imputation task in section 5. The experiments are described in section 6 for forecasting 6.2, relations discovery 6.3 and imputation 6.4.

2. Related Work

The classical topic of time series modeling and forecasting has given rise to an extensive literature, both in statistics and machine learning. In statistics, classical linear models are based on auto-regressive and moving average components. Most assume linear and stationary time dependencies with a noise component (De Gooijer & Hyndman 2006). In machine learning, non linear extensions of these models based on neural networks were proposed as early as the nineties, opening the way to many other non linear models developed both in statistics and ML, like kernel methods (Muller et al. 1999) for instance.

For the imputation task, the canonical approaches in statistics and machine learning are based on the Expectation Maximization (EM) algorithm and on Matrix Factorization (MF) methods. Bańbura & Modugno (2014) proposed an

adaptation of the EM algorithm for time series with missing data, claiming good results for long consecutive missing values. Recently, several adaptations of MF have been proposed for data completion in time series (Shang et al. 2014, Shi et al. 2016, Song et al. 2012).

Dynamical state space models, such as recurrent neural networks, have been used for time series modeling in different contexts since the early nineties (Connor et al. 1994). Recently, these models have witnessed important successes for several sequence modeling problems, leading to breakthrough in domains like speech (Graves et al. 2013), language generation (Sutskever et al. 2011), translation (Cho et al. 2014) and many others. Most of these applications are formalized as classification problems either at the level of discrete sequence events (e.g. words for translation) or at the level of subsequences in continuous signal (e.g. for speech decoding). Forecasting of complex multivariate sequences has been recently considered for the task of next frame prediction for video data (Oord et al. 2016, Denton & Birodkar 2017). Here again, spatial dependency is not explicitly modeled even if it can be implicitly captured by the models. Explicit spatio-temporal modeling with deep learning approaches has only been very recently considered (de Bezenac et al. 2017). Compared to forecasting, imputation has seldom been addressed in the deep learning literature. Mirowski & LeCun (2009) proposed a dynamic factor graph model designed for multiple series modeling where the evaluation is mainly performed on imputation tasks. This model is close to ours: it is a generative model with a latent component that captures the temporal dynamics and a decoder for predicting the series. However, spatial dependencies are not considered, and the learning and inference algorithms are different. Che et al. (2016) proposed a modified GRU unit to take into account missing input values for health-care related tasks.

Recently, the development of non parametric generative models has become a very popular research direction in Deep Learning, leading to different families of innovative and promising models. For example, the Stochastic Gradient Variational Bayes algorithm (SGVB) (Kingma & Welling 2013) provides a framework for learning stochastic latent variables with deep neural networks, and has recently been used by some authors to model time series (Bayer & Osendorfer 2014, Chung et al. 2015*b*, Krishnan et al. 2015). In our context, which requires to model explicitly both spatial and temporal dependencies between multiple time series, variational inference as proposed by such models is still intractable, especially when the number of series grows, which is the case in our experiments.

Spatio-temporal statistics already have a long history (Cressie & Wikle 2011, Wikle & Hooten 2010). The traditional methods rely on a descriptive approach using the first and second-order moments of the process for modeling the spatio-temporal dependencies. More recently, dynamical state space models, where the current state is conditioned on the past have been explored (Wikle 2015). For these models, time and space can be either continuous or discrete. The usual way is to consider discrete time, leading to the modeling of time series of spatial processes. When space is continuous, models are generally expressed by linear integro-difference equations, which is out of the scope of our work. With discrete time and space, models come down to general vectorial autoregressive formulations. These models face a curse of dimensionality in the case of a large number of sources. Different strategies have been adopted to solve this problem, such as parameter reduction or latent space modeling. This leads to model families that are close to the ones used in machine learning for modeling dynamical phenomena while incorporating spatial components. An interesting feature of these approaches

is the incorporation of prior knowledge inspired from physical models of space-time processes. This consists in taking inspiration from prior background of physical phenomena, e.g. diffusion laws in physics, and using this knowledge as guidelines for designing dependencies in statistical models (de Bezenac et al. 2017). In climatology, models taking into account both temporal and geographical components have also been used such as Gaussian Markov Random Fields (Rue & Held 2005).

In the machine learning domain, spatio-temporal modeling has been seldom considered, even though some spatio-temporal models have been proposed (Ceci et al. 2017). Bahadori et al. (2014) introduce a tensor model for kriging and forecasting. Koppula & Saxena (2013) use conditional random fields for detecting activity in video, where time is discretized at the frame level and one of the tasks is the prediction of future activity. Brain Computer Interface (BCI) is another domain for spatio-temporal data analysis with some work focused on learning spatio-temporal filters (Dornhege et al. 2005, Ren & Wu 2014), but this is a very specific and different topic.

3. The STNN Model

3.1. Notations and Task

Let us consider a set of n temporal series, m is the dimensionality of each series and T their length [1]. $m = 1$ means that we consider n univariate series, while $m > 1$ correspond to n multivariate series each with m components. We will denote X the values of all the series between time 1 and time T. X is then a tensor in $\mathbb{R}^{T\times n\times m}$, such that $X_t^i \in \mathbb{R}^m$ is a m-dimensional vector containing values of series i at time t. X_t will denote the slice of X at time t, such that $X_t \in \mathbb{R}^{n\times m}$ denotes the values of all the series at time t.

We consider two tasks: forecasting and imputation. The model is first presented for forecasting in this section and in section 4. The imputation version of the model is introduced in section 5. For simplicity, we first present our forecasting model in a mono-relational setting. An extension to multi-relational series where different relations between series are observed is described in section 3.4. We consider that the spatial organization of the sources is captured through a matrix $W \in \mathbb{R}^{n\times n}$. Ideally, W would indicate the mutual influence between sources. In practice, it might be a proximity or similarity matrix between the sources: for geo-spatial problems, this might correspond to the inverse of a physical distance - e.g. geodesic - between sources. For other applications, this might be provided through local connections between sources using a graph structure (e.g. adjacency matrix for connected roads in a traffic prediction application or graph kernel on the web). In a first step, we make the hypothesis that W is provided as a prior on the spatial relations between the series. An extension where these relations are learned is presented in section 4.

We then consider in the remainder of this section the problem of spatial time series forecasting i.e predicting the future of the series, knowing their past. We want to learn a model $f : \mathbb{R}^{T\times n\times m} \times \mathbb{R}^{n\times n} \to \mathbb{R}^{\tau\times n\times m}$ able to predict the future at τ time-steps of the series based on X and on their spatial dependency.

[1] We assume that all the series have the same dimensionality and length. This is often the case for spatio-temporal problems otherwise this restriction can be easily removed.

3.2. Modeling Time Series with Continuous Latent Factors

Let us first introduce the model in the simpler case of multiple time series prediction, without considering spatial relations. The model has two components.

The first one captures the dynamic of the process and is expressed in a latent space. Let Z_t be the latent representation, or latent factors, of the series at time t. The dynamical component writes $Z_{t+1} = g(Z_t)$. The second component is a decoder which maps latent factors Z_t onto a prediction of the actual series values at t: $\tilde{X}_t = d(Z_t)$, $\tilde{X}_t$ being the prediction computed at time t. In this model, both the representations Z_t and the parameters of the dynamical and decoder components are learned. Note that this model is different from the classical RNN formulations (Hochreiter & Schmidhuber 1997, Cho et al. 2014). The state space component of a RNN with self loops on the hidden cells writes $Z_{t+1} = g(Z_t, X'_t)$, where X'_t is the ground truth X_t during training, and the predicted value $\tilde{X}_t$ during inference. In our approach, latent factors Z_t are learned during training and are not an explicit function of past inputs as in RNNs: the dynamics of the series are then captured entirely in the latent space.

This formal definition makes the model more flexible than RNNs since not only the dynamic transition function $g(.)$, but also the state representations Z_t are learned from data. A similar argument is developed in Mirowski & LeCun (2009). It is similar in spirit to Hidden Markov models or Kalman filters.

Learning problem Our objective is to learn the two mapping functions d and g together with the latent factors Z_t, directly from the observed series. We formalize this learning problem with a bi-objective loss function that captures the dynamics of the series in the latent space and the mapping from this latent space to the observations. Let $\mathcal{L}(g, d, Z)$ be this objective function:

$$\mathcal{L}(d, g, Z) = \frac{1}{T} \sum_t \Delta(d(Z_t), X_t) + \lambda \frac{1}{T} \sum_{t=1}^{T-1} ||Z_{t+1} - g(Z_t)||^2 \tag{1}$$

The first term of the right hand side of (1) measures the ability of the model to reconstruct the observed values X_t from the latent factor Z_t. It is based on loss function Δ which measures the discrepancy between predictions $d(Z_t)$ and ground truth X_t. The second term aims at capturing the dynamics of the series in the latent space. This term forces the system to learn latent factors Z_{t+1} that are as close as possible to $g(Z_t)$. Note that in the ideal case, the model converges to a solution where $Z_{t+1} = g(Z_t)$, which is the classical assumption made when using RNNs. The hyper-parameter λ is used here to balance this constraint and is fixed by cross-validation. The solution d^*, g^*, Z^* to this problem is computed by minimizing $\mathcal{L}(d, g, Z)$:

$$d^*, g^*, Z^* = \arg\min_{d,g,Z} \mathcal{L}(d, g, Z) \tag{2}$$

Learning algorithm In our setting, functions d and g, described in the next section, are differentiable parametric functions. Hence, the learning problem can

be solved end-to-end with Stochastic Gradient Descent (SGD) techniques[2] directly from (2). At each iteration, a pair (Z_t, Z_{t+1}) is sampled, and Z_t, Z_{t+1}, g and d are updated according to the gradient of (1). Training can also be performed via mini-batch, meaning that for each iteration several pairs (Z_t, Z_{t+1}) are sampled, instead of a single pair. This results in a high learning speed-up when using GPUs which are the classical configuration for running such methods.

Inference Once the model is learned, it can be used to predict future values of the series. The inference method is the following: the latent factors of any future state of the series is computed using the g function, and the corresponding observations is predicted by using d on these factors. Formally, let us denote $\tilde{Z}_\tau$ the predicted latent factors at time $T + \tau$. The forecasting process computes $\tilde{Z}_\tau$ by successively applying the g function τ times on the learned vector Z_T:

$$\tilde{Z}_\tau = g \circ g \circ ... \circ g(Z_T) = g^{(\tau)}(Z_T) \tag{3}$$

and then computes the predicted outputs : $\tilde{X}_\tau = d(\tilde{Z}_\tau)$

3.3. Modeling Spatio-Temporal Series

Let us now introduce a spatial component in the model. We consider that each series has its own latent representation at each time step. Z_t is thus a $n \times N$ matrix such that $Z_{t,i} \in \mathbb{R}^N$ is the latent factor of series i at time t, N being the dimension of the latent space. This is different from approaches like Mirowski & LeCun (2009) or RNNs for multiple series prediction, where Z_t is a single vector common to all the series. The decoding and dynamic functions d and g are respectively mapping $\mathbb{R}^{n \times N}$ to $\mathbb{R}^{n \times m}$ and $\mathbb{R}^{n \times N}$ to $\mathbb{R}^{n \times N}$.

The spatial information is integrated in the dynamic component of our model through a matrix $W \in \mathbb{R}_+^{n \times n}$ with n the number of sources. In a first step, we consider that W is provided as prior information on the series' mutual influences. In 4, we remove this restriction, and show how it is possible to learn the weights of the relations, and even the spatial relations themselves, directly from the observed data. The latent representation of any series at time $t+1$ depends on its own latent representation at time t (intra-dependency) and on the latent representations of the other series at t (inter-dependency). Intra-dependency will be captured through a linear mapping denoted $\Theta^{(0)} \in \mathbb{R}^{N \times N}$ and inter-dependency will be captured by averaging the latent vector representations of the neighboring series using matrix W, and computing a linear combination denoted $\Theta^{(1)} \in \mathbb{R}^{N \times N}$ of this average. Formally, the dynamic model $g(Z_t)$ is designed as follow:

$$Z_{t+1} = h(Z_t\Theta^{(0)} + WZ_t\Theta^{(1)}) \tag{4}$$

Here, h is a non-linear function. In the experiments we set $h = tanh$ but h could also be a more complex parametrized function like a multi-layer perceptron

[2] In the experiments, we used the Nesterov's Accelerated Gradient (NAG) method (Sutskever et al. 2013).

(MLPs) for example - see section 6. The resulting optimization problem over d, Z, $\Theta^{(0)}$ and $\Theta^{(1)}$ writes:

$$\begin{aligned} d^*, Z^*, \Theta^{(0)*}, \Theta^{(1)*} = \underset{d,Z,\Theta^{(0)},\Theta^{(1)}}{\arg\min} & \frac{1}{T}\sum_t \Delta(d(Z_t), X_t) \\ & + \lambda \frac{1}{T}\sum_{t=1}^{T-1} ||Z_{t+1} - h(Z_t\Theta^{(0)} + WZ_t\Theta^{(1)})||^2 \end{aligned} \tag{5}$$

3.4. Modeling different types of relations

The model in section 3.3 considers that all the spatial relations are of the same type (e.g. based on sources proximity). For many problems, we will have to consider different types of relations. For instance, when sensors correspond to physical locations and the target is some meteorological variable, the relative orientation or position of two sources may imply a different type of dependency between the sources. In the experimental section, we consider problems with relations based on the relative position of sources: $north, south, west, east, ...$ The multi-relational framework generalizes the previous formulation of the model, and allows us to incorporate more abstract relations, like different measures of proximity or similarity between sources. For instance, when sources are spatiality organized in a graph, it is possible to define graph kernels, each one of them modeling a specific similarity. The following multi-relational formulation is based on adjacency matrices, and can directly incorporate such graph kernels.

Each possible relation type is denoted r and is associated to a matrix $W^{(r)} \in \mathbb{R}_+^{n\times n}$. For now, and as before, we consider that the $W^{(r)}$ are provided as prior knowledge. Each type of relation r is associated to a transition matrix $\Theta^{(r)}$. This learned matrix captures the spatio-temporal relationship between the series for this particular type of relation. The model dynamics writes:

$$Z_{t+1} = h(Z_t\Theta^{(0)} + \sum_{r\in\mathcal{R}} W^{(r)}Z_t\Theta^{(r)}) \tag{6}$$

where $\mathcal{R}$ is the set of all possible types of relations. The learning problem is similar to equation (5) with the argument of h replaced by the expression in (6). The corresponding model is illustrated in figure 1. This dynamic model aggregates the latent representations of the series for each type of relation, and then applies $\Theta^{(r)}$ on this aggregate. Each $\Theta^{(r)}$ is able to capture the dynamics specific to relation (r).

4. Capturing spatio-temporal correlations

In the previous sections, we made the hypothesis that the spatial relational structure and the strength of influence between series were provided as prior to the model as priors through the $W^{(r)}$ matrices. We introduce below an extension of the model where weights on these relations are learned. This model is denoted STNN-R(efining). We further show that this model can be easily extended to learn both the relations and their weights directly from the data, without any

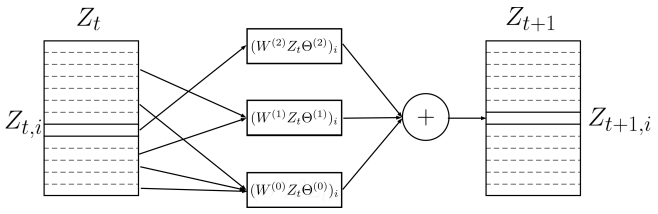


Fig. 1. Architecture of the STNN model as described in Section 3.4

.

prior on the existence and on the weights of the relations. This extension is denoted STNN-D(iscovering).

We will first introduce the STNN-R extension. Let $\Gamma^{(r)} \in \mathbb{R}^{n \times n}$ be a matrix of weights such that $\Gamma^{(r)}_{i,j}$ is the strength of the relation between series i and j in the relation r. Let us extend the formulation in Equation (6) as follows:

$$Z_{t+1} = h(Z_t\Theta^{(0)} + \sum_{r \in \mathcal{R}} (W^{(r)} \odot \Gamma^{(r)}) Z_t \Theta^{(r)}) \tag{7}$$

where $\Gamma^{(r)}$ is a matrix to be learned, $W^{(r)}$ is a prior i.e a set of observed relations, and $\odot$ is the element-wise multiplication between two matrices. The learning problem can be now be written as:

$$\begin{aligned} d^*, Z^*, \Theta^*, \Gamma^* = &\arg\min_{d,Z,\Gamma} \frac{1}{T} \sum_t \Delta(d(Z_t), X_t) + \gamma|\Gamma| \\ &+ \lambda \frac{1}{T} \sum_{t=1}^{T-1} ||Z_{t+1} - h(\sum_{r \in (R)} (W^{(r)} \odot \Gamma^{(r)}) Z_t \Theta^{(r)})||^2 \end{aligned} \tag{8}$$

where $|\Gamma^{(r)}|$ is a l_1 regularizing term that aims at sparsifying $\Gamma^{(r)}$. We thus add an hyper-parameter γ to tune this regularization factor.

If no prior is available, then simply removing the $W^{(r)}s$ from equation (7) leads to the following model:

$$Z_{t+1} = h(Z_t\Theta^{(0)} + \sum_{r \in \mathcal{R}} \Gamma^{(r)} Z_t \Theta^{(r)}) \tag{9}$$

where $\Gamma^{(r)}$ is no more constrained by the prior $W^{(r)}$ so that it will represent both the relational structure and the relation weights. Both models are learned with SGD, in the same way as described in 3.2. The only difference is that a gradient step on the $\Gamma^{(r)}s$ is added.

5. STNN for Data Imputation

We investigate here how the STNN model can be adapted for the *data imputation* problem. Data imputation, or missing values completion, is a classical problem in statistics and can come in different instances. We focus on the case where the information for some or all of the series is missing at different time steps, and

can be inferred from information available at other time steps and/or at different locations.

This setting covers several situations. Spatio-temporal data are often acquired by networks of physical sensors. These sensors are not always reliable. They can stop recording or transmitting data, or data can be too noisy to provide useful information. Observations can also be blurred or occulted by external factors. For instance, satellite imaging in the visible domain is sensible to clouds that occult parts of the earth surface. When needed, this information should be reconstructed using available data recorded at different time and locations, or data coming from other types of sensors. Another example that often occurs in traffic applications is when no signal is recorded at some places because of the absence of vehicles equipped with sensors at these places. This does not mean, of course, that traffic is absent. Hence, the values should be inferred from data available at other places.

For all these examples, having a reliable model for imputing missing data is essential. In this section, we propose an adaptation of the Spatio-Temporal Neural Network model in order to address this data imputation problem.

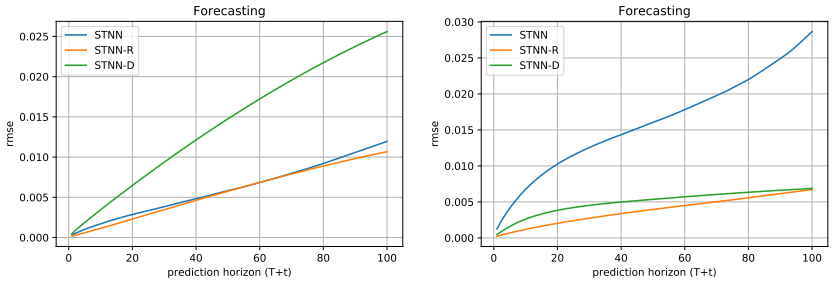
In the formulation of the data imputation task, in addition to the series values $X \in \mathbb{R}^{T \times n \times m}$ (with T the number of time-steps, n the number of series and m the dimensionality of the series), we also consider a missing data mask $M \in \{0,1\}^{T \times n}$. M_t^i is the binary mask on the series i at time-step t, and is equal to 1 when vector $X_t^i \in \mathbb{R}^m$ (observed values of series i at time-step t) is missing, and 0 if it is present. The goal is to minimize the prediction error of missing data points.

As apposed to the forecasting task, we suppose that observations from every time-step (past and future) are present, and missing data may appear at any time-step. If X_t^i is a missing value for series i at time t, the prediction $\hat{X}_t^i$ will be computed based on all the available $X_{t'}^j$ for $j \in \{1, n\}$ and $t' \in \{1, T\}$.

The objective function for imputation is the following:

$$\mathcal{L}(d, g, Z) = \frac{1}{\sum_t \sum_i (1 - M_t^i)} \sum_t \sum_i (1 - M_t^i) \Delta(d(Z_t^i), X_t^i) + \sum_{t=1}^{T-1} ||Z_{t+1} - g(Z_t)||^2 \tag{10}$$

In this expression, supervision comes from the available X_t^i, so that the Z_t^i value inferred for a missing X_t^i will depend on all available observations $X_t'^j$. The second term $\sum_{t=1}^{T-1} ||Z_{t+1} - g(Z_t)||^2$ acts as a regularizer for the Z_t^i value associated to a missing X_t^i. Intuitively, the Z_t^i value for a missing observation should be coherent with the neighboring latent states Z associated to observations X. For example if one suppose that X_t is missing while X_{t-1} and X_{t+1} are observed, the term Z_t will be directly constrained by the two loss terms $||g(Z_t) - Z_{t-1}||^2$ and $||Z_t - g(Z_{t-1})||$. Since our model learns one latent state for each series and each time-step, it will learn these latent states over missing values that could then be retrieved using the decoding function. Indeed, inference in this model is straightforward: once learned, a missing value of series i at time-step t can be computed as $X_t^i = d(Z_t^i)$.



(a) Heat diffusion experiment (b) Modulated heat diffusion experiment

Fig. 2. Forecasting performances (RMSE) for the synthetic heat diffusion experiments. Left: standard heat diffusion. Right: heat diffusion with modulated diffusion constant. Datasets are simulated for 200 timesteps. Models are learned on the first 100 timesteps, and forecast the next 100 timesteps.

6. Experiments

The following section contains experiments and results on different tasks and settings [3]. First, we present experiments on a synthetic dataset in order to demonstrate some properties of the STNN model and its variants. Then we present results on real world datasets for the forecasting task, followed by a qualitative analysis of the relation discovery capabilities of our model. Finally, we present results on the data imputation task.

6.1. Synthetic experiments on heat diffusion

We begin by evaluating and analyzing the STNN model and its variants on a heat diffusion simulation dataset. It is a simple problem whose characteristics, in particular the spatio-temporal dependencies, are perfectly known. Hence, its complexity can be controlled. We consider a 1-D segment where a heat source is applied on its center. The diffusion of the heat is then governed by the following differential equation:

$$\frac{\partial u}{\partial t} = a\left(\frac{\partial^2 u}{\partial x^2}\right)$$

where u is the heat value, a a diffusion constant, and x and t are respectively the space and time variables. This equation can be discretized in space and time by the explicit Euler method as follows:

$$u_{t+1}^i = u_t^i + a\Delta t\left(\frac{u_t^{i-1} - 2u_t^i + u_t^{i+1}}{\Delta x^2}\right) \quad \forall i \in 1, ..., n$$

We construct a dataset by simulating heat diffusion on a segment with $n = 41$ points through 200 time-steps. We use the first 100 time-steps for training and

[3] Code available at https://github.com/edouardelasalles/stnn

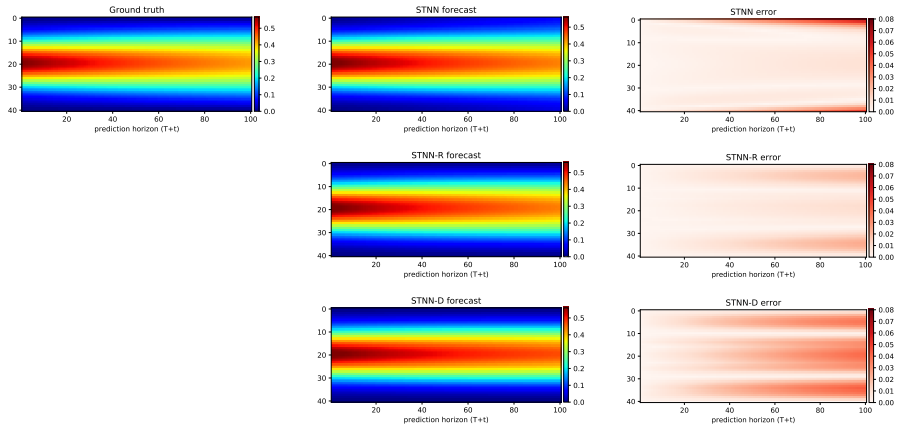


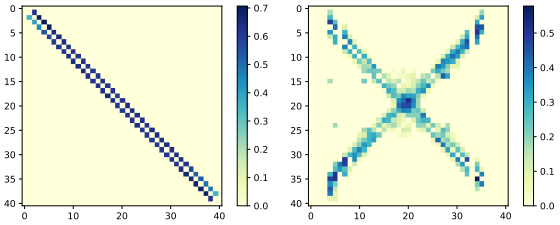
Fig. 3. One hundred time step forecasting of heat diffusion by our three different models.

the remaining 100 for the forecasting evaluation. The adjacency matrix W we use for STNN and STNN-R is the one connecting direct neighbors in the diffusion segment. Figure 2a show the RMSE scores for prediction at $t+1$ to $t+100$, and figure 3 shows the predicted values and the ground truth. As expected, the STNN-R model performs best. It has both a strong relational prior (i.e only adjacent points interact with each other) and enough flexibility to adjust the relation weights and well capture the spatio-temporal correlations. STNN-D has no spatial prior, and fails to learn the dynamics of the process.

On the top right image of Figure 3, we can see that the errors made by the STNN model are concentrated on the borders: the model over-estimates the heat diffusion at these points. Indeed, the two border points have only one neighbor each, whereas all other points have two. STNN-R, on the other hand, is able to adapt relation weights in order to cope with this effect. This is illustrated on the right image, second row of Figure 3 where the absolute error on the borders is clearly lower for STNN-D than for the other variants.

The relations weights learned by STNN-R (denoted by Γ on eq. (7)) are shown on Figure 4a. A pixel at position (i, j) on this image corresponds to the weight that STNN-R puts on series j at time t when updating the latent representation of series i at time $t+1$. High values mean stronger influence of series j in the update of series i. One can see that STNN-R learns asymmetrical and low value weights between points close to the borders (upper left and bottom right pixels on Figure 4a), allowing it to cope with the border effect described in the previous paragraph. We also show on Figure 4b the relation weights discovered by STNN-D. Similarly to STNN-R, it learns low relation weights between points near the edges. The horizontal and vertical axial symmetries in figure 4b reflect the symmetry of the data itself (figure 3 top left image).

To further explore the adaptivity of STNN, we make the diffusion process more complex. The diffusion constant a is replaced by an RBF kernel positioned on the center of the diffusion segment:



(a) STNN-R (b) STNN-D

Fig. 4. Relation weights learned by STNN-R (left) and STNN-D (right). The images represent adjacency matrix weights (denoted by Γ on eq. (7) and (9))

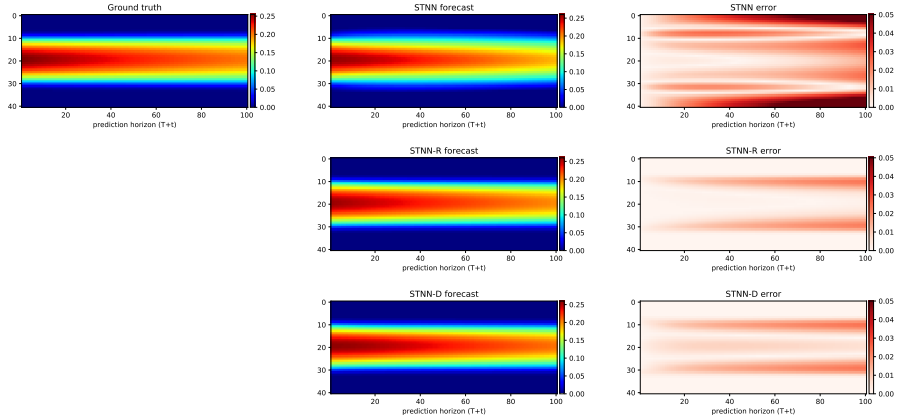


Fig. 5. One hundred time step forecasting of modulated heat diffusion by our three different models.

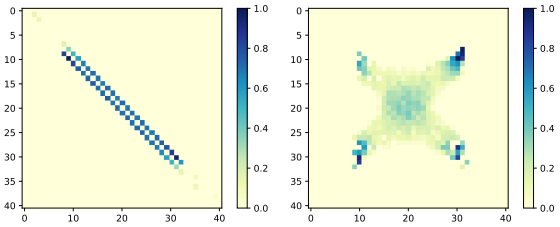
$$u_{t+1}^i = u_t^i + a^i \Delta t \left(\frac{u_t^{i-1} - 2u_t^i + u_t^{i+1}}{\Delta x^2} \right) \; s.t. \; a^i = a' K(|i - \frac{n-1}{2}|, \frac{n-1}{2})$$

where K is a RBF kernel. This modification results in heat propagating itself faster in the center, and slower as it reaches the borders.

Results are shown in figure 2b. In this setting, both STNN-R and STNN-D perform significantly better than STNN. While the latter learns a dynamics which is the same for all positions, both STNN-R and STNN-D learn position dependent spatial weights. Figure 5 shows the predicted values. It is easy to see that, once again, STNN over-estimates heat propagation speed. Figure 6 shows the learned relations: both STNN-R and STNN-D put very low values on relations between points at the extremities of the segment.

6.2. Spatio-Temporal Series Forecasting

For this first task, experiments are performed on a series of spatio-temporal forecasting problems representative of different domains. We consider predictions



(a) STNN-R (b) STNN-R

Fig. 6. Relation weights learned by STNN-R (left) and STNN-D (right) on modulated heat diffusion.

Table 1. Datasets statistics. n is the number of series, m is the dimension of each series, $time-step$ corresponds to the duration of one time-step and *#folds* corresponds to the number of temporal folds used for validation. For each fold, evaluation has been made on the next 5 values at $T+1, T+2, ..., T+5$. The relation columns specifies the number of different relation types used in the experiments i.e the number of $W^{(r)}$ matrices used in each dataset. 1 to 3 means that the best among 1 to 3 relations was selected using cross validation

Dataset	n	m	nb relations	time-step	total length	training length	#folds
Google Flu	29	1	1 to 3	weeks	$\approx$ 10 years	2 years	50
GHO (25 datasets)	91	1	1 to 3	years	45 years	35 years	5
Wind	500	2	1 to 3	hours	30 days	10 days	20
PST	2520	1	8	months	$\approx$ 33 years	10 years	15
Bejing	5000	1	1 to 3	15 min	1 week	2 days	20

within a +5 horizon i.e. given a training series of size T, the evaluation of the quality of the model will be made over $T+1$ to $T+5$ time steps. The different model hyper-parameters are selected using a time-series cross-validation procedure called rolling origin as in Ben Taieb & Hyndman (2014), Ganeshapillai et al. (2013). This protocol makes use of a sliding window of size T': on a series of length T, a window of size T' is shifted several times in order to create a set of train/test folds. The beginning of the T' window is used for training and the remaining for test. The value of T' is fixed so that it is large enough to capture the main dynamics of the different series. Each series was re-scaled between 0 and 1.

We performed experiments with the following models:
(i) **Mean**: a simple heuristic which predicts future values of a series with the mean of its observed past values computed on the T' training steps of each training fold.
(ii) **AR**: a classical univariate Auto-Regressive model. For each series and each variable of the series, the prediction is a linear function of R past lags of the variable, R being a hyper-parameter tuned on a validation set.
(iii) **VAR-MLP**: a vectorial auto-regressive model where the predicted values of the series at time $t+1$ depend on the past values of all the series for a lag of size R. The predictive model is a multi-layer perceptron with one hidden layer. Its performance were uniformly better than a linear VAR. Here again the hidden layer size and the lag R were set by validation
(iv) **RNN-tanh**: a vanilla recurrent neural network with one hidden layer of

recurrent units and tanh non-linearities. As for the **VAR-MLP**, one considers all the series simultaneously, i.e. at time t the RNN receives as input X_{t-1} the values of all the series at $t-1$ and predicts X_t. A RNN is a dynamical state-space model but its latent state Z_t explicitly depends through a functional dependency both on the preceding values of the series X_{t-1} and on the preceding state Z_{t-1}. Note that this model has the potential to capture the spatial dependencies since all the series are considered simultaneously, but does not model them explicitly.
(v) **RNN-GRU**: same as the **RNN-tanh**, but recurrent units is replaced with gated recurrent units (GRU) units, which are considered state of the art for many sequence prediction problems today [4]. We have experimented with several architectures, but using more than one layer of GRU units did not improve the performance, so we used 1 layer in all the experiments.
(vi) **Dynamic Factor Graph (DFG)**: the model proposed in Mirowski & LeCun (2009) is the closest to ours but uses a joint vectorial latent representation for all the series as in the RNNs, and does not explicitly model the spatial relations between series.
(vii) **STNN**: our model where g is the function described in equation (6), h is the *tanh* function, and d is a linear function. Note that other architectures for d and g have been tested (e.g. multi-layer perceptrons) without improving the quality of the prediction. The λ value has been set by cross validation.
(viii and ix) **STNN-R** and **STNN-D**: for the forecasting experiments the γ value of the L_1 penalty (see equation (8)) were set to 0 since higher value decreased the performance, a phenomena often observed in other models such as L_1-regularized SVMs. The influence of γ on the discovered spatial structure is further discussed and illustrated in figure 10.

We detail the set of hyper-parameters used in our grid-searches in section 6.2.2.

6.2.1. Datasets

The different forecasting problems and the corresponding datasets are described below. The dataset characteristics are provided in table 1.

- **Disease spread forecasting:** The **Google Flu** dataset contains for 29 countries, about ten years of weekly estimates of influenza activity computed by aggregating Google search queries (see `http://www.google.org/flutrends`). We extract binary relations between the countries, depending on whether or not they share a border, as a prior W.
- **Global Health Observatory (GHO):** This dataset made available by the Global Health Observatory, (`http://www.who.int/en/`) provides the number of deaths for several diseases. We picked 25 diseases corresponding to **25 different datasets**, each one composed of 91 time series corresponding to 91 countries (see table 1). Results are averages over all the datasets. As for Google Flu, we extract binary relations W based on borders between the countries.
- **Geo-Spatial datasets:** The goal is to predict the evolution of geophysical phenomena measured on the surface of the Earth.
  The **Wind** dataset (`www.ncdc.noaa.gov/`) consists of hourly summaries of meteorological data. We predict wind speed and orientation for approximately

[4] We also performed tests with LSTM and obtained similar results as with GRU.

Table 2. Average RMSE for the different datasets computed for T+1, T+2,...,T+5. Standard deviation was computed by re-training the models on different seeds.

Models	Google Flu	GHO[5]	Beijing	Speed	Direction	PST
MEAN	.175	.335	.201	0.191	0.225	.258
AR	.101±.004	.299 ± .008	.075 ± .003	.082 ± .005	0.098±.016	.15 ± .002
VAR-MLP	.095 ± .004	.291 ± .004	.07 ± .002	.071 ± .005	0.111±0.14	.132 ± .003
DFG	.095 ± .008	.288 ± .002	.068 ± .005	.07 ± .004	.092 ± .006	.99 ± .019
RNN-tanh	.082 ± .008	.287 ± .011	.075 ± .006	.064 ± .003	.09 ± .005	.141 ± .01
RNN-GRU	.074 ± .007	.268 ± .07	.074 ± .002	.059 ± .009	.083 ± .005	.104 ± .008
STNN	.066 ± .006	**.261** ± .009	.056 ± .003	**.047** ± .008	**.061** ± .008	.095 ± .008
STNN-R	**.061** ± .008	**.261** ± .01	**.055** ± .004	**.047** ± .008	**.061** ± .008	**.08** ± .014
STNN-D	.073 ± .007	.288 ± .09	.069 ± .01	.059 ± .008	.073 ± .008	.109 ± .015

Table 3. RMSE of STNN-R over the 25 datasets in GHO for T+1, T+2,...,T+5

Disease \ Model	AR	VAR-MLP	RNN-GRU	Mean	DFG	STNN-R
All causes	0.237	0.228	0.199	0.35	0.291	**0.197**
Tuberculosis	0.407	0.418	**0.37**	0.395	0.421	0.377
Congenital syphilis	0.432	0.443	0.417	0.459	0.422	**0.409**
Diphtheria	0.406	0.396	0.387	0.404	0.419	**0.385**
Malignant neoplasm of esophagus	0.355	**0.341**	**0.341**	0.363	0.372	0.345
Malignant neoplasm of stomach	0.44	0.434	0.431	0.455	0.452	**0.43**
	0.267	0.254	0.282	0.303	0.301	**0.253**
Malignant neoplasm of intestine	0.281	0.29	0.278	0.314	0.305	**0.275**
Malignant neoplasm of rectum	0.501	0.499	**0.481**	0.504	0.509	0.498
Malignant neoplasm of larynx	0.321	0.313	0.32	0.314	0.329	**0.310**
Malignant neoplasm of breast	0.375	0.375	0.382	0.394	0.38	**0.36**
Malignant neoplasm of prostate	0.111	0.113	**0.109**	0.184	0.138	**0.109**
Malignant neoplasm of skin	0.253	0.243	0.227	0.264	0.256	**0.221**
Malignant neoplasm of bones	0.103	0.099	0.097	0.204	0.173	**0.08**
Malignant neoplasm of all other and unspecified sites	0.145	0.157	**0.147**	0.164	0.169	0.156
Lymphosarcoma	0.15	0.132	0.13	0.231	0.135	**0.122**
Benign neoplasms	0.366	0.362	0.332	0.398	0.331	**0.331**
Avitaminonsis	0.492	0.474	0.449	0.571	0.58	**0.414**
Allergic disorders	0.208	0.217	0.221	0.342	0.24	**0.202**
Multiple sclerosis	0.061	0.057	0.061	0.242	0.152	**0.056**
Rheumatic fever	0.325	0.31	0.287	0.345	0.313	**0.256**
Diseases of arteries	0.302	0.301	0.269	0.345	0.328	**0.238**
Influenza	0.141	0.141	0.155	0.23	0.217	**0.125**
Pneumonia	0.119	0.128	**0.1**	0.187	0.187	**0.1**
Pleurisy	0.246	0.246	0.247	0.29	0.272	**0.245**
Gastro-enteritis	0.386	0.369	**0.291**	0.394	0.398	0.295
Disease of teeth	0.344	0.312	0.305	0.413	0.361	**0.302**

500 land stations on U.S. locations. In this dataset, the relations correspond to a thresholded spatial proximity between the series. Given a selected threshold value d, two sources are connected ($w_{i,j} = 1$) if their distance is below d and not connected ($w_{i,j} = 0$) otherwise.
The **Pacific Sea Temperature (PST)** dataset represents gridded (at a 2 by 2 degrees resolution, corresponding to 2520 spatial locations) monthly Sea Surface Temperature (SST) on the Pacific for 399 consecutive months from January 1970 through March 2003. The goal is to predict future temperatures at the different spatial locations. Data were obtained from the Climate Data Library at Columbia University (`http://iridl.ldeo.columbia.edu/`). Since the series are organized on a 2D grid, we extract 8 different relations : one for each cardinal direction (north, north-west, west, etc...). For instance, the relation *north*, is associated to a binary adjacency matrix $W^{(north)}$ such that $W^{(north)}_{i,j}$ is set to 1 if and only if source j is located 2 degree at the north of source i (the pixel just above on the satellite image).

– **Car Traffic Forecasting:** The goal is to predict car traffic on a network of streets or roads. We use the **Beijing dataset** provided in Yuan et al. (2011, 2010) which consists of GPS trajectories for ~ 10500 taxis during a week, for a total of 17 millions of points corresponding to road segments in Beijing. From this dataset, we extracted the traffic-volume aggregated on a 15 min window for 5,000 road segments. The objective is to predict the traffic at each segment. We connect two sources if they correspond to road segments with a shared crossroad.

For all the datasets but PST (i.e. Google Flu, GHO, Wind and Bejing), we defined the relational structure using a simple adjacency matrix W. Based on this matrix, we defined K different relations by introducing the powers of this matrix: $W^{(1)} = W$, $W^{(2)} = W \times W$, etc. In our setting K took values from 1 to 3 and the optimal value for each dataset has been selected during the validation process.

6.2.2. Hyper-parameters selection

In order to achieve the best possible results on our model, and also on our baselines, we grid-searched hyper-parameters on each models, for each datasets. Hence, the results presented in the rest of this section come from models optimized and fine-tuned independently across all datasets. Each hyper-parameter is selected by cross-validation on a large grid of hyper-parameters values. We detailed below that values tested for each hyper-parameters:

– **AR:**
  · Number of lags $R \in \{1, 2, 5, 10, 15, 25\}$

– **VAR-MLP:**
  · Number of lags $R \in \{1, 2, 5, 10, 15, 25\}$
  · Size of the hidden state $\in \{20, 50, 80, 150, 300, 500\}$

– **RNN-models:**
  · Size of the hidden state $= (20, 50, 80, 150, 300, 500)$

– **DFG:**

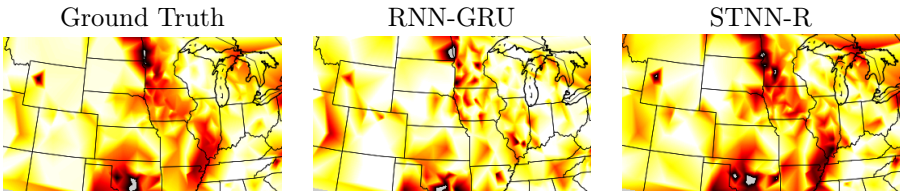


Fig. 7. Prediction of wind speed over around 500 stations on the US territory. prediction is shown at time-step $T+1$ for RNN-GRU (center) and STNN-R (right).

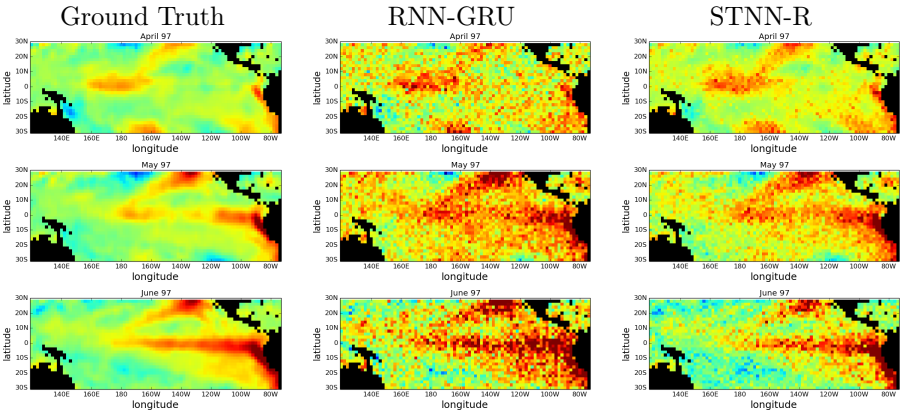


Fig. 8. Example of a 3 months prediction of Pacific temperature. Left column is the ground truth, central and right columns correspond respectively to RNN-GRU and STNN-R predictions at horizon $T+1$, $T+2$ and $T+3$ (top to bottom).

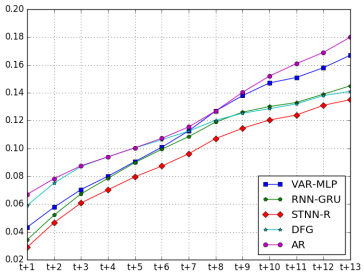
- Size of the hidden states $\in \{10, 20, 50, 100, 300, 500\}$

– **STNN:**

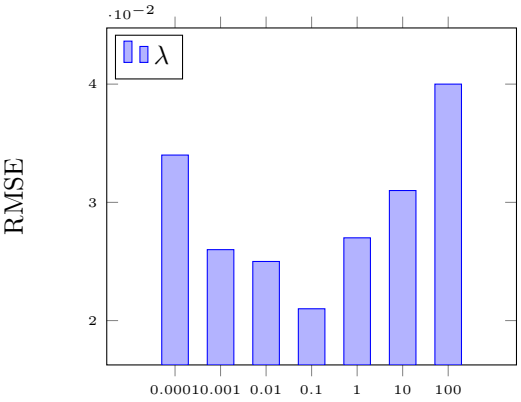
- Dimension of the latent space $N \in \{5, 10, 20, 50, 80, 10\}$
- Soft-constraint parameter $\lambda \in \{0.001, 0.01, 0.1, 1, 10\}$
- Sparsity regulation $\gamma \in \{0.001, 0.01, 0.1, 1\}$
- K value $\in \{1, 2, 3\}$ (for datasets composed of 1 unique relation)

6.2.3. Results

A quantitative evaluation of the different models and the baselines, on the different datasets is provided in table 2. All the results are average prediction error for $T+1$ to $T+5$ predictions. The score function used is the Root Mean Squared Error (RMSE). A first observation is that STNN and STNN-R models which make use of prior spatial information significantly outperform all the other models on all the datasets. For example, on the challenging PST dataset, our models increase by 23% the performance of the GRU-RNN baseline. The increase is more important when the number of series is high (geo-spatial and traffic datasets) than when it is small (disease datasets). In these experiments, STNN-D is on par with RNN-GRU. The two models do not use prior information on spatial



(a) RMSE on the Google Flu dataset at horizon $T+1$ to $T+13$



(b) RMSE on Google Flu w.r.t λ

Fig. 9. Quantitative study on the Google Flu dataset.

proximity. STNN makes use of a more compact formulation than RNN-GRU for expressing the series mutual dependency but the results are comparable. Vectorial AR logically improves on mono-variable AR (not shown here) and non linear MLP-VAR improves on linear VAR.

We also provide in table 3 the score for each of the 25 diseases of the GHO dataset. STNN-R obtained the best performance compared to STNN and STNN-D. It outperforms state-of-the-art methods in 20 out of 25 datasets, and is very close to the RNN-GRU model on the 5 remaining diseases where RNN-GRU performs best. It thus shows that our model is able to benefit from the neighbour information in the proximity graph.

Figures 7 and 8 illustrate respectively the prediction of *STNN-R* and *RNN-GRU* on the meteorology and on the oceanography datasets along with the ground truth. Clearly on these datasets, STNN qualitatively performs much better than RNNs by using explicit spatial information. STNN is able to predict fine details corresponding to local interactions when RNNs produce a much more noisy prediction. These illustrations are representative of the general behavior of the two models.

We also provide the performance of the models at different prediction horizons $T+1, T+2, ..., T+13$ on Figure 9a for the Google Flu dataset. Results show that STNN performs better than the other approaches for all the prediction horizons and is thus able to better capture longer-term dependencies.

Figure 9b illustrates the RMSE of the STNN-R model when predicting at $T+1$ on the Google Flu dataset for different values of λ. One can see that the best performance is obtained for an average value of λ: low values corresponding to weak temporal constraints do not allow the model to learn the dynamicity of the series while high values degrade the performance of STNN.

6.3. Discovering the Spatial Correlations

In this subsection, we illustrate the ability of STNN to discover relevant spatial correlations on different datasets. Figures 10 and 11 illustrate the values of Γ obtained by STNN-D where no structure (e.g. adjacency matrix W) is provided

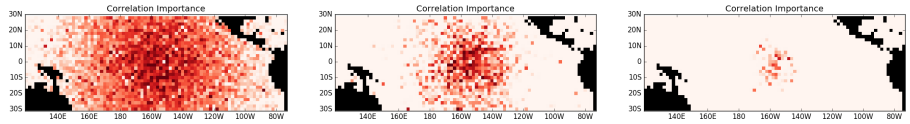


Fig. 10. Illustrations of correlations Γ discovered by the STNN-D model, with γ in $\{0.01, 0.1, 1\}$ (from top to bottom).

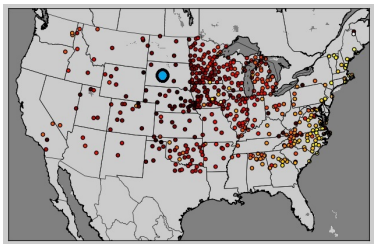


Fig. 11. Spatial correlation discovery with STNN-D on the Wind dataset

to the model on the PST and Wind dataset respectively. Each pixel corresponds to a particular time series and the figure shows the correlation $\Gamma_{i,j}$ discovered between each series j with a series i. The series i is roughly located at the center of the picture in Figure 10, and is represented by a blue circle in Figure 11. The darker a pixel is, the higher the absolute value of $\Gamma_{i,j}$ is (note that black pixels correspond to countries and not sea). Different levels of sparsity are illustrated from low (up) to high (down). Even if the model does not have any knowledge about the spatial organization of the series (no W matrix provided), it is able to re-discover this spatial organization by detecting strong correlations between close series, and low ones for distant series.

Figure 12 illustrates the correlations discovered on the PST dataset. We used as priors 8 types of relations corresponding to the 8 cardinal directions (South, South-West, etc...). In this case, STNN-R learns weights (i.e $\Gamma^{(r)}$) for each relation based on the prior structure. For each series, we plot the direction with the highest learned weight. The strongest direction for each series is illustrated by a specific color in the figure. For instance, a dark blue pixel indicates that the stronger spatial correlation learned for the corresponding series is the North-West direction. The model extracts automatically relations corresponding to temperature propagation directions in the pacific, providing relevant information about the spatio-temporal dynamics of the system.

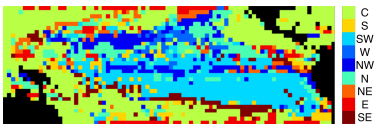


Fig. 12. Spatial correlations extracted by the STNN-R model on the PST dataset. The color of each pixel correspond to the principal relation extracted by the model.

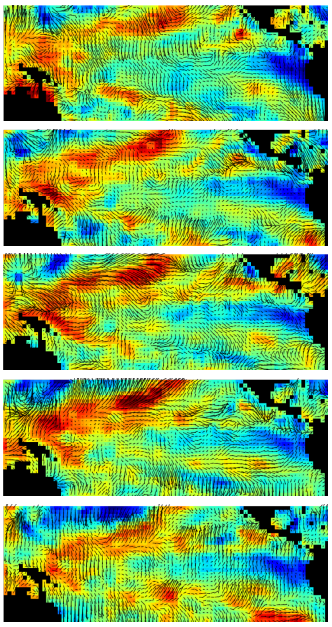


Fig. 13. Dynamic spatio-temporal relations extracted from the PST dataset on the training set on 3 consecutive time-steps. The color represents the actual sea surface temperature. The arrows represent the extracted spatial relations that evolve through time.

The model can be adapted to different situations. Figure 13 represents captured temporal evolution of the spatial relations on the PST dataset. For this experiment, we have slightly changed the STNN-R model by making the $\Gamma^{(r)}$ time dependent according to:

$$\Gamma^{(r)}_{t,j,i} = f_r(Z^i_t) \tag{11}$$

This means that with this modified model, the spatial relation weights depend on the current latent state of the corresponding series and may evolve with time. In the experiment, f_r is a logistic function. On figure 13, the different plots correspond to successive time steps. The color represent the actual sea surface temperatures, and the arrows represent the direction of the stronger relation weights $\Gamma^{(r)}_t$ among the eight possible directions (N, NE, etc). One can see that the model captures coherent dynamic spatial correlations such as global currents directions or rotating motions that gradually evolve with time.

6.4. Data Imputation

This section presents the experiments concerning data imputation. We first introduce the experimental protocol, we briefly describe the baselines and then detail and comment our quantitative and qualitative results.

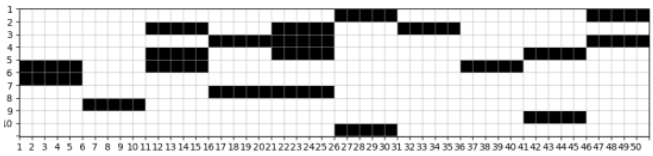


Fig. 14. The figure represents 10 time-series over 50 time-steps, white squares corresponding to observed values and black squares corresponding to missing ones. This missing values have been generated from a fully observed set of time series using a corruption schema where $p_m = 20$ and $l_m = 5$ (see Section 6.4.1).

6.4.1. Experimental protocol

As mentioned in section 5, we focus on the case where the information for some or all of the series is missing at different time steps. This setting is quite general and covers different situations. For instance, missing values may affect only some of the series at a given time step, or all the series may be affected at the same time steps. In addition, the number of time steps with missing values may be extremely different from a problem to the other. For evaluating the models, one needs a generic protocol. In order to provide a quantitative evaluation of the quality of our model, we defined a protocol common to all the datasets. For a given dataset, we remove a random subset of the data as detailed below and the resulting dataset with missing values will then be used for training and testing. Missing values are generated at random as follows.

We choose a missing percentage p_m - different values are used in the experiments - which indicates the proportion of the series values that is going to be considered as missing. For instance $p_m = 20$ means that 20% of the dataset values we be considered as missing. We also choose a missing value length l_m, which determines the size of the missing chunks. For instance, if $l_m = 5$, a missing data chunk is composed of 5 consecutive time-steps in a given series. Figure 14 shows a sample of a missing data mask M for 50 time-steps with 10 series, where $p_m = 20$ and $l_m = 5$.

The training set denotes all available observations (non missing values - aka white squares in Figure 14) while test set denotes the missing values (black squares in Figure 14). In order to select hyper-parameters, we held out a validation set from the training set. More specifically, during the validation phase, we take out a proportion p_m of the training set that we keep for evaluating hyper-parameters, and train on the remaining $(1 - p_m)\%$ of the training set. Once the hyper-parameters are selected, we train the model from scratch on the entire training set, and evaluate on the test set.

We evaluate our model on the following datasets GFlu, Wind, Beijing car traffic, and PST. For the Wind dataset, we jointly consider the speed characteristic and the direction characteristic in order to evaluate our model on a multi-variate setting.

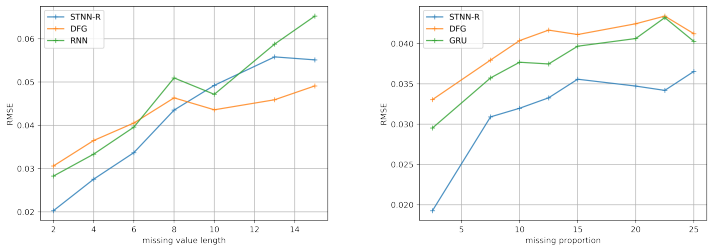
6.4.2. Baselines

We compared our model to the following baselines:

– **Mean:** missing data are imputed with corresponding series' average value.
– **Last:** missing data are imputed with the series' last observed value.

Models	Google Flu	Beijing	Wind	PST
MEAN	1.08e−1	8.37e−2	2.28e−1	6.14e−2
LAST	6.96e−2	6.94e−2	1.51e−1	9.86e−2
Amelia II	7.98e−2	6.99e−2	1.87e−1	X
GRU	3.77e−2	5.25e−2	1.32e−1	1.04e−2
DFG	4.04e−2	5.26e−2	1.37e−1	7.77e−3
STNN	3.65e−2	4.85e−2	1.17e−1	**2.59e−3**
STNN-R	**3.20e−2**	**4.52e−2**	1.21e−1	2.76e−3
STNN-D	3.31e−2	4.60e−2	**1.15e−1**	3.78e−3

Table 4. RMSE for the imputation task on the different datasets. These results where obtained for $p_m = 10$ and $l_m = 5$. (X means that the dataset is to large for the available implementation)



(a) Corruption length variation (b) Corruption proportion variation

Fig. 15. Evolution of our model and baselines score when the missing value proportion change. On figure 15a, the length of the occulted chunks varies, while the corruption proportion stays at 10%. On figure 15b, the missing proportion changes, while the corruption length stays at 5 times-steps

– **Amelia II (Honaker et al. 2011) :** a statistical model for missing data imputation based on a bootstrapped version of the EM algorithm. For our experiments, we sample m times for each missing value, given the observed variables, and take the mean of these samples.
– **GRU:** we used the "GRU-simple" baseline proposed by Che et al. (2016). This simple baseline for imputation using GRU works as follow: for a time series X, each missing value is replaced with the average value of the series, giving a new series $\tilde{X}$. At each time-step t, the GRU is fed with the concatenation of $\tilde{X}_t$ and M_t, the missing value mask at time step t. The loss is a standard MSE, where the gradient for missing values is not backpropagated.
– **DFG:** the DFG model (Mirowski & LeCun 2009) is another latent dynamical model, which has been developed for imputation.

Note that "Mean" and "Last" are frequently used heuristics for handling missing values, "GRU-simple" and "DFG" are state of the art latent dynamical models designed for imputation in time series or sequences.

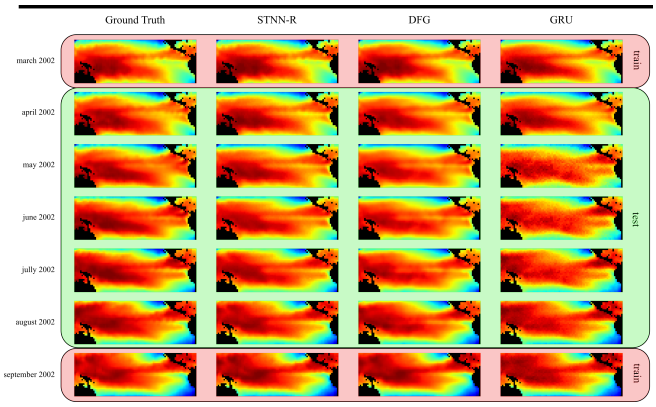


Fig. 16. Complete time-step imputation visualization. August and September 2002, shown with a pink border, are observed and used for training. April to august 2002 included, shown with a green border, are not observed during training and are used as test set for imputation.

6.4.3. Quantitative Results

Table 4 presents quantitative test results. The scores are the RMSE on the missing test values. These results were obtained with following parameters: missing value proportion rate $p_m = 10$ and missing value length of 5 time-steps, $l_m = 5$. As for the forecasting results, the STNN models and its variants perform consistently better than the baselines. For the Google Flu dataset, STNN-R is 18% better than the strongest baseline (GRU). On the Wind dataset, STNN-D achieves the best results, performing 15% better than DFG. It is on the PST dataset that we obtain the strongest results: STNN performs 3 times better than DFG. Amelia II baseline, fails to reach the performance of deep models (STNN, DFG, GRU) by a large margin. This is due to its over-simplistic normal prior and the lack of spatial prior. This illustrates the difficulty of the imputation task on large real world datasets.

We also performed a quantitative study of the model robustness for different levels of missing values by comparing "STNN-R", "DFG" and "GRU-simple". Results are shown on Figure 15a where l_m is varying with a fixed $p_m = 10$, and in Figure 15b where p_m is varying with a fixed $l_m = 5$. On Figure 15a one can see that STNN-R performs better than the two baselines for all missing value proportions. Concerning the missing values length, for lengths higher than 10 time-steps, DFG gets better results than STNN-R. Our model learns one explicit latent factor per time-series, and when too many consecutive values are missing for one time-series, the predicted latent factors tend to collapse. The DFG model only learns one factor common to all the series and is then more robust to this type of corruption: since in our setting, the missing values only consider a subset of the series, DFG benefits from the other observed values at a given time-step to better infer a correct latent factor.

Table 5. Test results (RMSE) on the PST dataset where all values from chunks of 5 consecutive time-steps are missing, with overall 10% values missing. Theses results comes from the same experiment that yields the images on Figure 16

	STNN-R	DFG	RNN
PST	**2.21**e−**2**	2.83e−2	3.25e−2

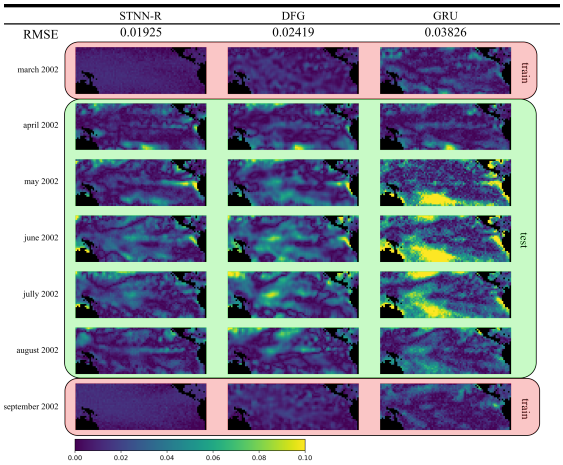


Fig. 17. Absolute error visualization for imputed data of figure 16 i.e absolute difference between reconstructed values and ground truth values. The RMSE line corresponds to the RMSE computed only on the 5 test time-steps (green background) indicated in the figure.

6.4.4. Complete time-step reconstruction

We also experiment a configuration where, at a given time-step, the values of all the series are missing simultaneously. This is an extreme scenario, but it can happen for instance on earth observation problems. For these experiments, we keep a missing value ratio at 10% and a missing sequence length of 5 time-steps, with all the values in any chunks of 5 time-steps completely occulted during training.

Figure 16 shows a sample of the data reconstructed by our models and baselines on the Pacific surface temperature dataset from March to September 2002. March and September 2002 where observed (pink border on the Figure) and used for training, while observations from April to August 2002 where occulted (green border on the Figure) and used in test as missing values. Figure 16 shows that the GRU baseline performs worse than both STNN-R and DFG: the predictions are not locally smooth. The predictions from STNN-R and DFG look very similar on this figure. In order to analyze better the differences, we also plot the absolute error performed by the three models on Figure 17. The larger error of GRU is again clearly visible on this Figure, and it also appears clearly that STNN-R imputations are of better quality than those of DFG which does not model explicitly the spatial dependencies.

The RMSE printed above each column on Figure 17 is the RMSE of the 5

reconstructed time-steps for each model. This confirms the visual results: STNN-R actually achieves a better performance on these 5 time-steps. We show on the table 5 the RMSE for all the dataset. We can see that STNN-R performs better on average for all the missing chunks.

6.5. Discussion

By using prior information on the existence of spatial dependencies, the model is able to learn both the strength of these relations and the temporal dynamics of the underlying process. The proposed model compares well with classical forecasting methods on a series of benchmarks. It then offers an interesting solution for both the forecasting and the imputation problems. The model still has some restrictions:

- It is designed for short prediction horizons only (5 time steps ahead) and is not adapted for longer term forecast.
- The model should be retrained when the testing conditions are different from the training ones. This is the case for climate applications for example, for data from different locations
- The learned dynamics is stationary in time, but we propose a method to learn a dynamic transition function at then end of Section 6.3.

7. Conclusion

We have proposed a new latent model for addressing multivariate spatio-temporal time series modeling problems with applications to forecasting and imputation. For this model the dynamics are captured in a latent space and the prediction makes use of a decoder mechanism. Extensive experiments on datasets representative of different domains show that this model is able to capture spatial and temporal dynamics, and performs better than state of the art competing models for both the forecasting and th imputation tasks. This model is amenable to different variants concerning the formulation of spatial and temporal dependencies between the sources.

Acknowledgements. Locust project ANR-15-CE23-0027-01, funded by Agence Nationale de la Recherche.

References

Bahadori, M. T., Yu, Q. R. & Liu, Y. (2014), Fast multivariate spatio-temporal analysis via low rank tensor learning, *in* 'Advances in Neural Information Processing Systems', pp. 3491–3499.

Bańbura, M. & Modugno, M. (2014), 'Maximum likelihood estimation of factor models on datasets with arbitrary pattern of missing data', *Journal of Applied Econometrics* **29**(1), 133–160.

Bayer, J. & Osendorfer, C. (2014), 'Learning stochastic recurrent networks', *arXiv preprint arXiv:1411.7610* .

Ben Taieb, S. & Hyndman, R. (2014), Boosting multi-step autoregressive forecasts, *in* 'Proceedings of The 31st International Conference on Machine Learning', pp. 109–117.

Bengio, Y. (2008), 'Neural net language models', *Scholarpedia* **3**(1), 3881.

Ceci, M., Corizzo, R., Fumarola, F., Malerba, D. & Rashkovska, A. (2017), 'Predictive modeling of pv energy production: How to set up the learning task for a better prediction?', *IEEE Transactions on Industrial Informatics* **13**(3), 956–966.

Che, Z., Purushotham, S., Cho, K., Sontag, D. & Liu, Y. (2016), 'Recurrent neural networks for multivariate time series with missing values', *arXiv preprint arXiv:1606.01865* .

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. (2014), 'Learning phrase representations using rnn encoder-decoder for statistical machine translation', *arXiv preprint arXiv:1406.1078* .

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C. & Bengio, Y. (2015*a*), A recurrent latent variable model for sequential data, *in* 'Advances in neural information processing systems', pp. 2962–2970.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C. & Bengio, Y. (2015*b*), A recurrent latent variable model for sequential data, *in* C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama & R. Garnett, eds, 'Advances in Neural Information Processing Systems 28', Curran Associates, Inc., pp. 2980–2988.

Connor, J. T., Martin, R. D. & Atlas, L. E. (1994), 'Recurrent neural networks and robust time series prediction', *Neural Networks, IEEE Transactions on* **5**(2), 240–254.

Cressie, N. A. C. & Wikle, C. K. (2011), *Statistics for spatio-temporal data*, Wiley series in probability and statistics, Hoboken, N.J. Wiley.

de Bezenac, E., Pajot, A. & Gallinari, P. (2017), Deep learning for physical processes: Incorporating prior scientific knowledge, *in* 'Proceedings of the International Conference on Learning Representations (ICLR)'.

De Gooijer, J. G. & Hyndman, R. J. (2006), '25 years of time series forecasting', *International journal of forecasting* **22**(3), 443–473.

Denton, E. L. & Birodkar, v. (2017), Unsupervised learning of disentangled representations from video, *in* I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan & R. Garnett, eds, 'Advances in Neural Information Processing Systems 30', Curran Associates, Inc., pp. 4417–4426.

Dornhege, G., Blankertz, B., Krauledat, M., Losch, F., Curio, G. & robert Muller, K. (2005), Optimizing spatio-temporal filters for improving brain-computer interfacing, *in* 'in NIPS'.

Ganeshapillai, G., Guttag, J. & Lo, A. (2013), Learning connections in financial time series, *in* 'Proceedings of the 30th International Conference on Machine Learning (ICML-13)', pp. 109–117.

Graves, A., Mohamed, A.-r. & Hinton, G. (2013), Speech recognition with deep recurrent neural networks, *in* 'IIIE ICASSP'.

Hochreiter, S. & Schmidhuber, J. (1997), 'Long short-term memory', *Neural computation* **9**(8), 1735–1780.

Honaker, J., King, G. & Blackwell, Matthew, e. a. (2011), 'Amelia ii: A program for missing data', *Journal of statistical software* **45**(7), 1–47.

Kalchbrenner, N., Oord, A. v. d., Simonyan, K., Danihelka, I., Vinyals, O., Graves, A. & Kavukcuoglu, K. (2017), Unsupervised learning of video representations using lstms, *in* 'Proceedings of the 34nd ICML-17'.

Kingma, D. P. & Welling, M. (2013), Auto-encoding variational bayes, *in* 'Proceedings of the 2nd International Conference on Learning Representations (ICLR)'.

Koppula, H. & Saxena, A. (2013), Learning spatio-temporal structure from rgb-d videos for human activity detection and anticipation, *in* 'Proceedings of ICML'.

Krishnan, R. G., Shalit, U. & Sontag, D. (2015), 'Deep kalman filters', *arXiv preprint arXiv:1511.05121* .

Li, Y., Tarlow, D., Brockschmidt, M. & Zemel, R. (2015), 'Gated graph sequence neural networks', *arXiv preprint arXiv:1511.05493* .

Mirowski, P. & LeCun, Y. (2009), Dynamic factor graphs for time series modeling, *in* 'Machine Learning and Knowledge Discovery in Databases', Springer, pp. 128–143.

Muller, K. R., Smola, A. J., Ratsch, G., Scholkopf, B., Kohlmorgen, J. & Vapnik, V. (1999), 'Using support vector machines for time series prediction', *Advances in kernel methods—support vector learning* .

Oord, A. V., Kalchbrenner, N. & Kavukcuoglu, K. (2016), Pixel recurrent neural networks, *in* M. F. Balcan & K. Q. Weinberger, eds, 'Proceedings of The 33rd International Conference on Machine Learning', Vol. 48 of *Proceedings of Machine Learning Research*, PMLR, pp. 1747–1756.

Ren, Y. & Wu, Y. (2014), Convolutional deep belief networks for feature extraction of eeg signal, *in* 'Neural Networks (IJCNN), 2014 International Joint Conference on', IEEE, pp. 2850–2853.

Rue, H. & Held, L. (2005), *Gaussian Markov random fields: theory and applications*, CRC Press.

Shang, J., Zheng, Y., Tong, W., Chang, E. & Yu, Y. (2014), Inferring gas consumption and pollution emission of vehicles throughout a city, *in* 'Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, pp. 1027–1036.

Shi, W., Zhu, Y., Philip, S. Y., Huang, T., Wang, C., Mao, Y. & Chen, Y. (2016), 'Temporal dynamic matrix factorization for missing data prediction in large scale coevolving time series', *IEEE Access* **4**, 6719–6732.

SHI, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-k. & WOO, W.-c. (2015), Convolutional lstm network: A machine learning approach for precipitation nowcasting, *in* C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama & R. Garnett, eds, 'Advances in Neural Information Processing Systems 28', Curran Associates, Inc., pp. 802–810.

Song, Y., Liu, M., Tang, S. & Mao, X. (2012), Time series matrix factorization prediction of internet traffic matrices, *in* 'Local Computer Networks (LCN), 2012 IEEE 37th Conference on', IEEE, pp. 284–287.

Srivastava, N., Mansimov, E. & Salakhudinov, R. (2015), Unsupervised learning of video representations using lstms, *in* D. Blei & F. Bach, eds, 'Proceedings of the 32nd ICML-15'.

Sutskever, I., Martens, J., Dahl, G. & Hinton, G. (2013), On the importance of initialization and momentum in deep learning, *in* 'Proceedings of the 30th ICML'.

Sutskever, I., Martens, J. & Hinton, G. E. (2011), Generating text with recurrent neural networks, *in* 'Proceedings of the 28th International Conference on Machine Learning, ICML 2011'.

Wikle, C. K. (2015), 'Modern perspectives on statistics for spatio-temporal data', *Wiley Interdisciplinary Reviews: Computational Statistics* **7**(1), 86–98.

Wikle, C. K. & Hooten, M. B. (2010), 'A general science-based framework for dynamical spatio-temporal models', *Test* **19**(3), 417–451.

Yuan, J., Zheng, Y., Xie, X. & Sun, G. (2011), Driving with knowledge from the physical world, *in* 'Proceedings of the 17th ACM SIGKDD', ACM, pp. 316–324.

Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G. & Huang, Y. (2010), T-drive: driving directions based on taxi trajectories, *in* 'Proceedings of the 18th SIGSPATIAL', ACM, pp. 99–108.

Author Biographies

Edouard Delasalles received his Master in computer science from Sorbonne Université, Paris, France in 2016. He is currently a Ph.D. student in the Machine Learning and Information Access (MLIA) team at Laboratoire d'Informatique de Paris 6 (LIP6), Sorbonne Université. His research interests include representations learning with time series, temporal generative models and deep dynamic models.

Ali Ziat received his Ph.D. in machine learning from Sorbonne Université in 2017, during which he studied representation learning for time-series classification and time-series-forecasting. After his Ph.D., he joined the Boston Consulting Group, a consulting firm, where he is currently a Senior Data-scientist helping different industries improving or developing their AI capabilities.

Ludovic Denoyer is a Full Professor in the Machine Learning team at University Pierre et Marie Curie (France). He obtained his Master of Artificial Intelligence in 2001 and his Ph.D. thesis in 2004 on the problem of classification with structured inputs. He then focused on two main research topics: 1) representation learning for structures, and particularly complex graphs (multi-relational, dynamic, heterogeneous) and social networks 2) budgeted reinforcement learning models with applications to text classification, image classification, features acquisition and structured output prediction. His research is now mainly centered on the desire to develop human-machine interaction systems.

Patrick Gallinari is professor at Sorbonne University (SU), Paris, France. He has been a pioneer in France in the field of Neural Networks and has been working on several machine learning frameworks including Deep Learning and on their application to different fields of engineering. He is co-author of around 200 publications on these topics. He was from 2005 to 2013 the director of the computer science laboratory (LIP6) at Sorbonne and he is head of the Machine Learning and Information Access (MLIA) team at LIP6.