Representation Learning for Completion and Prediction over Relational Temporal Data

Ali Ziat, Gabriella Contardo, Nicolas Baskiotis, Ludovic Denoyer*

Université Paris VI, LIP6

April 30, 2015

Abstract

We address the problem of learning over multiple inter-dependent temporal sequences where dependencies are modeled by a graph. We propose a model that is able to simultaneously fill in missing values and predict future ones. This approach is based on representation learning techniques, where temporal data are represented in a latent vector space so as to capture the dynamicity of the process and also the relations between the different sources. Information completion (missing values) and prediction are then performed on this latent representation. In particular, the model allows us to perform both tasks using a unique formalism, whereas most often they are addressed separately using different methods. Moreover, the models allows us to deal with heterogeneous information (labels and real values) at the same time. The model has been tested for a concrete application: car-traffic forecasting where each time series characterizes a particular road and where the graph structure corresponds to the road map of the city. We compare our method with different baselines for both completion and prediction on two large datasets and show the ability of our technique to jointly solve these problems. Note that the model is general and can be used as well in many different application fields.

1 Introduction

Temporal data correspond to a wide variety of phenomena from stock market to internet traffic forecasting. Different kind of temporal data can be produced: monovariate and multivariate time series where the produced values are real values, but also sequences of labels, events, ... The recent emergence of sensors everywhere - e.g mobile phones which typically produce temporal sequences of complex data (GPS, events, ...) - is an example that illustrates the need of new machine learning models for temporal data processing. Indeed, the produced information has particular characteristics that can't be handled by classical sequential and temporal models: they contain multiple missing values, they are heterogeneous, i.e. one source of information can produce different types of information, and one has to consider simultaneously multiple sources that can be somehow related, by spatial proximity for example.

Traffic forecasting over roads networks is a good illustration of that phenomenon: while traffic was usually monitored by using fixed sensors, one sensor for each road, the emergence of mobile sensors (i.e. GPS) in cars produces more complicated series: sensors are moving, measuring different roads at different time-steps. During a time period, some roads are monitored while others are not, resulting in series with many missing values - see Figure 1 and Figure 2.

On one side, several models have been proposed for multivariate time-series or sequences prediction. The most popular are probably neural networks [1] - an overview of related methods is given in Section 5. However, these models have not been conceived for dealing with missing data. On the other side, some models have been also proposed with the goal of automatically completing missing information (data imputation) [2], like matrix factorization techniques which have been used recently in the context of traffic forecasting [3] [4]. But data imputation and data prediction are usually seen as two different problems. Moreover, acquired sequences are usually heterogeneous, a sensor being able to produce different types of information, and also inter-dependent, two time series can be related to each other. There exist no model able to deal with all these characteristics and tasks conjointly.

We propose a novel method that aims at integrating all the aspects of complex temporal data in one single model.

*firstname.lastname@lip6.fr

The proposed approach is based on representation learning techniques aiming at projecting the observations in a continuous latent space, each sequence being modeled at each time-step by a point in this space. It has many advantages w.r.t existing techniques: (i) it is able to simultaneously learn how to fill missing values and to predict the future of the observed temporal data, avoiding to use two different models, (ii) it naturally allows one to deal with information sources that are organized among a graph structure (iii) it is also able to deal with heterogeneous information when sequences are composed of different types of information. Moreover, the model is based on continuous optimization schemes, allowing a fast optimization over large scale datasets.

The contributions of the paper are:

1. We propose a representation learning model for relational temporal data.
2. We show how this model can solve two different key problems conjointly: prediction and missing values completion.
3. We propose an extension of this model where each sequences is composed of different types of information (real values and labels in our case).
4. We make a large experiment set in the context of traffic prediction and compare our approach to baseline models in both prediction and completion.

This paper is organized as follow: Section 2 gives an overview of the context of the work and the definition of the two tasks. The section 3 describes the representation-learning models and gives details about the learning and inference algorithms. Section 4 defines the experimental protocol and presents the experimental results. Section 5 introduces a possible extension of our model. At last Section 6 describes the related work and Section 7 concludes our article and provides interesting perspectives.

2 Context

2.1 Notations and Tasks

Let us consider a set of n temporal sequences $\mathbf{x_1}, ..\mathbf{x_n}$ such that $x_i^{(t)} \in \mathcal{X}$ is the value of the i-th sequence[1] at time t defined by $\mathbf{x_i} = (x_i^{(1)}, .., x_i^{(T)})$. In the case where $\mathcal{X}$ is $\mathbb{R}^m$, the context corresponds to multiple multivariate time series, but our approach can also deal with sequences of labels for example. The sequences contain missing values so we also define a mask $m_i^{(t)}$ such that $m_i^{(t)} = 1$ if value $x_i^{(t)}$ is observed - and thus available for training the system - and $m_i^{(t)} = 0$ if $x_i^{(t)}$ is missing - and thus has to be predicted by the model. In addition, we consider that there exists a set of relations between the sequences which correspond to an external information, like spatial proximity for example when $\mathcal{X}$ is discrete. The sequences are thus organized in a graph $\mathcal{G} = \{e_{i,j}\}$ such that $e_{i,j} = 1$ means that $\mathbf{x_i}$ and $\mathbf{x_j}$ are related, and $e_{i,j} = 0$ elsewhere. For sake of simplicity, we consider that the graph structure connecting the sequences is static and does not change during time.

The two tasks that we want to (conjointly) solve - see Figure 3- are the following: (i) The problem of prediction consists in predicting *what happens next* given the observed temporal data. (ii) Data completion consists in missing values inference in the sequences based on the observed values. Let us denote $y_i^{(t)}$ the value of sequence i at time t predicted by the learned model, the two proposed tasks can be defined as follows:

Prediction aims at predicting the values at time $T + t'$ where T is the time of the observed sequences used for learning the model and t' the horizon. The quality of the prediction can be measured as the average loss between predicted values and groundthruth:

$$\mathcal{P}^{pred}(t') = \frac{1}{n}\sum_{i=1}^{n} \Delta(y_i^{(T+t')}, x_i^{(T+t')})$$

where Δ is a prediction error measure defined such that: $\Delta : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^+$. (the mean square error for example).

Completion aims at filling missing values (where $m_i^{(t)} = 0$) by predicted ones in the observed sequences. It can be measured by

$$\mathcal{P}^{comp} = \frac{1}{n.T}\sum_{t=1}^{t=T}\sum_{i=1}^{n}(1 - m_i^{(t)})\Delta(y_i^{(t)}, x_i^{(t)})$$

Note that, in the experimental part, these measures will be evaluated on a set of testing values in order to compare the quality of different approaches - see Section 4.

2.2 Main Idea

The goal of our model is to take into account the different types of information available in the dataset which are: (i) the observed values, (ii) the relations between the information sources and (iii) the dynamicity of the system. We

[1] We first present an homogeneous version of the model: the value of each sequence at each timestep is a single value in $\mathcal{X}$. We introduce an heterogeneous variant in Section 5 where $x_i^{(t)}$ is composed of different types of informations.

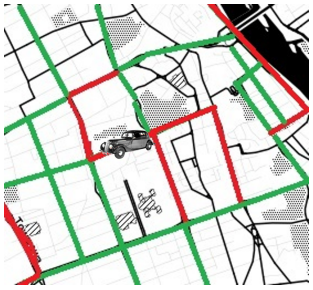

Figure 1: A part of a road network where roads are highlighted in red when traffic information is available during a time-step and in green otherwise. Sensors are moving and at each time-step these roads may change, which may results in highly sparse data.

T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
?	21	17	?	25	28	?	34	30	24
3	5	12	11	23	23	27	19	11	14
21	?	24	35	?	44	?	?	38	45
22	26	28	27	25	25	?	29	18	22
35	31	36	?	51	44	60	57	57	59
24	15	?	14	11	?	12	8	6	?
8	5	3	10	10	5	?	13	14	17

Figure 2: A part of the network: each row correspond to a road and each column is a time-step. Values represent the average speed on a given road during a time-step. Missing data are on red squares and correspond to roads crossed by no sensors (i.e. GPS).

propose to capture this information in a large dimensional latent space $\mathcal{Z}$ in which each observation will correspond to a particular point (or representation) at each time step, denoted $z_i^{(t)} \in \mathcal{Z}$. Facing n information sources during a duration of T, the model will thus learn $T \times n$ data points in $\mathcal{Z}$, each information source $\mathbf{x_i}$ being described by a learned trajectory $(z_i^{(1)}, ..., z_i^{(T)})$ in the latent space. This kind of approach has already been proposed for capturing complex information like in [5] for information diffusion, [6] for relational data or even [7] for Markov decision processes, but never explored for relational temporal data.

In order to capture all the observed information, the way the latent representations will be built will correspond to the following constraints:

No Information Loss: First, each representation of each series at each time-step $z_i^{(t)}$ must allow one to predict the value $x_i^{(t)}$ of the series. This property corresponds to the fact, that, from the learned representations, one will be able to build observations, and thus to fill missing values or to predict.

Dynamicity: Second, we want to capture the dynamicity of source $\mathbf{x_i}$. This particularly will be used when predicting the future behavior of each series. This will be done by enriching each latent point $z_i^{(t)}$ with a dynamical information.

Correlation between sources: At last, in order to integrate the relational information between sequences, we will consider that two related information sources tend to behave similarly, and thus the trajectory (in the latent space) of two connected sequences must be close.

Following these constraints, we describe how these representations can be learned from observations in the next section. Note, that, in comparison to classical inductive approaches (like neural networks for example) where high-level features are computed from the observations, our method is a transductive model where representations $z_i^{(t)}$ are parameters of the model from which observations can be build. The proposed model thus acts more as a generative model than as a discriminant one which is meaningful since it aims at being able to predict (i.e. generate) missing values and also to simulate the future of the sequences.

3 Representation-baSed temporal relational model

3.1 Loss-based approach

We now present how the constraints presented above can be integrated in a single model. The *RepresentAtIoN-baSed TempOral Relational Model* (RAINSTORM) is a loss-based model which is described through a continuous derivable loss function that will be optimized using classical optimization techniques. The approach proposed is close to the ones of the deep learning community but the proposed model is different from classical deep neural networks techniques since an explicit representation $z_i^{(t)}$ is learned for each time-step and each source as done in [7]. The main interest of this approach is to be able to deal easily with missing values while classical NN-based techniques are less suitable for this case.

Let us define $\mathcal{L}(\theta, \gamma, \mathbf{z})$ the loss function to minimize where $\mathbf{z}$ is the set of all the vectors $z_i^{(t)}$ for $i \in [1..n]$ and $t \in [1..T]$, T being the size of the observed time windows

i.e. the history of the time series. We define $\mathcal{L}$ as:

$$\mathcal{L}(\theta, \gamma, \mathbf{z}) = \frac{1}{O} \sum_{i=1}^{n} \sum_{t=1}^{T} m_i^{(t)} \Delta(f_\theta(z_i^{(t)}), x_i^{(t)}) \text{ (term 1)} + \lambda_{dyn} \sum_{i=1}^{n} \sum_{t=1}^{T-1} ||z_i^{(t+1)} - h_\gamma(z_i^{(t)})||^2 \text{ (term 2)} + \lambda_{struct} \sum_{i,j \in [1..N]^2} \sum_{t=1}^{T} e_{i,j} ||z_i^{(t)} - z_j^{(t)}||^2 \text{ (term 3)} \quad (1)$$

where O is the number of observed values i.e. values such that $m_i^{(t)} = 1$.

This loss function contains three terms, each one associated with one of the constraints that have been presented previously:

- Term 1 aims at simultaneously learn $\mathbf{z}$ and a function f_θ - called **decoding function** - such that, from $z_i^{(t)}$, f_θ can be used to predict the value $x_i^{(t)}$. The function $f_\theta(z_i^{(t)})$ is defined as $f_\theta : \mathbb{R}^N \to \mathcal{X}$. Δ is used to measure the error between predicting $f_\theta(z_i^{(t)})$ instead of $x_i^{(t)}$, $m_i^{(t)}$ playing the role of a mask restricting to compute this function only on the observed values, ignoring the missing values that are unknown and cannot be used in training.

- Term 2 aims at finding values $z_i^{(.)}$ and a dynamic model h_γ such that, when applied to $z_i^{(t)}$, h_γ allows us to predict the representation of the next state of time series i i.e. $z_i^{(t+1)}$. h_γ is the **dynamic function** which models the dynamicity of each series directly in the latent space: $h_\gamma : \mathbb{R}^N \to \mathbb{R}^N$. The parameters γ will be learned to minimize the mean square error between the prediction $h_\gamma(z_i^{(t)})$ and $z_i^{(t+1)}$ ensuring that h_γ is a good dynamic model of the series in $\mathbb{R}^N$

- At last, term 3 corresponds to a **structural regularity** over the graph structure that encourages the model to learn closer representations for time series that are related. This will force the model to learn representations that reflect the structure of the considered graph.

λ_{dyn} and λ_{struct} are manually defined coefficients that weight the importance of the different elements in the loss function. For f_θ and h_γ, different architectures can be chosen (they must be continuous and derivable) and we propose distinct architectures in the experimental section. These meta-parameters are chosen using classical validation techniques.

3.2 Learning Problem

The learning problem aims at minimizing the loss function $\mathcal{L}(\theta, \gamma, \mathbf{z})$ simultaneously on θ, γ and $\mathbf{z}$. By restricting the f_θ and h_γ to be continuous derivable functions, we can use gradient-descent based optimization approaches. The loss function is clearly not convex but such techniques will allow us to efficiently find a local minimum to this problem. The optimization of complex non-convex losses recently allowed to obtain very interesting results in many different applications (i.e. deep learning techniques) and we consider that since such a model is able to handle complex constraints, even stuck in a local minimum, it can produce nice predictions - see Section 4.

Different gradient-based approaches can be used: batch minimization, alternated gradient-decent, ... We propose to use a classical stochastic gradient method described in Algorithm 1. The algorithm takes as input the observed sequences, the mask over the sequences, randomly initialized decoding and dynamic functions, and a representation for each sequence for every time step (line 2-6). Then, iteratively, it modifies the parameters of the decoding function (line 12) and the corresponding representations (line 11) toward the direction of their gradient. This corresponds to the term 1 in equation (1). Parameters of the dynamic function and the concerned representations are modified the same way (line 15-17). The model also updates its parameters corresponding to the structural regularity (line 19-20). The algorithm produces as output both a set of latent points $\mathbf{z}$ but also the decoding and dynamic functions f_θ and h_γ. All this information will be then used for prediction and completion as explained in Section 3.3. The expression power of the model is mainly controlled by the size of the latent space N which will be chosen using classical validation techniques.

3.3 Inference

Now, we consider that, given a set of observed values, $\mathbf{z}$, f_θ and h_γ have been learned. Let us explain how this model can be concretely use.

3.3.1 Completion of missing values:

For all missing values $x_i^{(t)}$ such that $m_i^{(t)} = 0$, the proposed learning algorithm has learned a z-value $z_i^{(t)}$ in the latent space. This learned value has been mainly 'chosen' based on term 2 and 3 of the loss function (Equation 1) while the decoding term has not been tuned on $z_i^{(t)}$ since $x_i^{(t)}$ is unknown. In order to predict this missing value, our approach simply computes the value $f_\theta(z_i^{(t)})$ which produces a plausible output value. Note that, when choosing a linear decoding function, the predicted value is $\theta^T z_i^{(t)}$ which

Algorithm 1 Stochastic Gradient Descent Algorithm

input:
  $\forall i, t, x_i^{(t)} \in \mathcal{X}$
  $\forall i, t, m_i^{(t)}$
  $\epsilon \leftarrow$ learning rate
procedure Learning($\theta, \gamma, \mathbf{z}$)
  $\forall i, t, z_i^{(t)} \leftarrow rand, \gamma \leftarrow rand, \theta \leftarrow rand$
for number of iteration **do**
  Sample i, t, j
  %Gradient descent for term (1)
  if $m_i^{(t)} == 1$ **then**
    $z_i^{(t)} \leftarrow z_i^{(t)} + \epsilon \nabla_{z_i^{(t)}} \Delta(f_\theta(z_i^{(t)}), x_i^{(t)})$
    $\theta \leftarrow \theta + \epsilon \nabla_\theta \Delta(f_\theta(z_i^{(t)}), x_i^{(t)})$
  end if
  end
  %Gradient descent for term (2)
  $z_i^{(t)} \leftarrow z_i^{(t)} + \epsilon \lambda_{dyn} \nabla_{z_i^{(t)}} \Delta(h_\gamma(z_i^{(t)}), z_i^{(t+1)})$
  $z_i^{(t+1)} \leftarrow z_i^{(t+1)} + \epsilon \lambda_{dyn} \nabla_{z_i^{(t+1)}} \Delta(h_\gamma(z_i^{(t)}), z_i^{(t+1)})$
  $\gamma \leftarrow \gamma + \epsilon \lambda_{dyn} \nabla_\gamma \Delta(h_\gamma(z_i^{(t)}), z_i^{(t+1)})$
  %Gradient descent for term (3)
  $z_i^{(t)} \leftarrow z_i^{(t)} + \epsilon \lambda_{struct} \nabla_{z_i^{(t)}} ||z_i^{(t)} - z_j^{(t)}||^2 e_{i,j}$
  $z_j^{(t)} \leftarrow z_j^{(t)} + \epsilon \lambda_{struct} \nabla_{z_j^{(t)}} ||z_i^{(t)} - z_j^{(t)}||^2 e_{i,j}$
  end
end for
end procedure

is very close to the classical completion value computed through matrix factorization techniques, but θ and $\mathbf{z}$ have been learned based on multiple information (in particular the relational information into the graph).

3.3.2 Predicting the future:

Now, we explain how the model can be used to predict future values $x_i^{(t)}$. For all $t > T$, the model does not compute z-values and these $z_i^{(t)}$ are unknown. But our model learns a dynamic function h_γ which goal is to allow the prediction of $z_i^{(t+1)}$ given $z_i^{(t)}$. So, for any i, $z_i^{(T+1)}$ can be computed by $h_\gamma(z_i^{(T)})$, $z_i^{(T+2)}$ can be computed by $h_\gamma(h_\gamma(z_i^{(T)}))$ and so on. h_γ acts as a simulator of the sequences in the latent space. The future value $x_i^{(T+s)}$ can thus be predicted by simply computing $f_\theta(h_\gamma(h_\gamma(h_\gamma(.....h_\gamma(z_i^{(T)})..))))$ where h_γ is applied s times, the obtained vector being then transformed to prediction by using f_θ.

4 Traffic Forecasting and Experiments

4.1 Traffic Forecasting

We consider a road network as a graph where each node corresponds to a road, and edges are connections between roads: two roads are connected if they belong to the same crossroads. We consider that, at each time-step, some sensors return measures over a subset of roads (i.e. the roads such that $m_i^{(t)} = 1$). This measure is typically the average speed of cars on the road, or a measure of the volume of cars. Note that the subset of roads from which we obtain measures at time t can be different of the one at time $t + 1$ since the sensors are floating sensors moving in the city. The goal of traffic forecasting is first to fill missing values i.e. values on roads that have not been directly measured at time t and also to predict what will happen on the different roads for the next time-steps. In traffic forecasting, the time-step typically corresponds to some minutes (e.g. 15 minutes) and the prediction has to be made for the next hours.

4.2 Datasets

Experiences have been made on two datasets. Collected data consist of GPS trajectories of a large amount of vehicles in the cities of Beijing and Warsaw which are converted to relational time series through the following preprocessing steps: (i) trajectories are map-matched (in a similar way to what is done in [8]). (ii) Based on this matching, the speed and volume of cars are computed for each road at each timestep. (iii) The graph between series is built by connecting roads that share a same crossroads. Statistical details concerning the two datasets are given in Table 4. **Beijing dataset:** the dataset is provided by [9] and consists on trajectories of about 10500 taxis during a week, for a total of 17 millions of points. We keep traffic-volume and aggregate it on 15min windows. **Warsaw dataset:** The data were presented in 2010 in the context of the ICDM data mining challenge [10]. The contest's subject was traffic prediction and data were generated by a simulator in the city of Warsaw (Poland). 500 simulations of 10 hours each were provided with a very high sampling rate (one point every 10s), for a total of around 130 million of points. We aggregate data on $\sim$20 000 roads on a 10 minutes window and we compute average speed velocity for each road during this time-step.

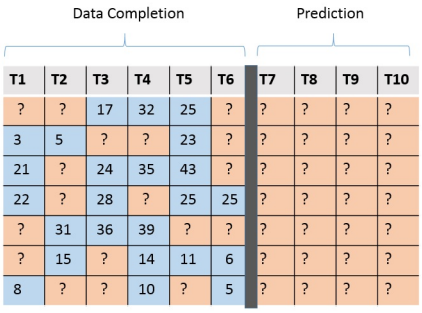

Figure 3: The tasks we want to achieve consist in data completion from T1 to T6 and prediction from T7 to T10. Blue squares are values such that $m_i^{(t)} = 1$

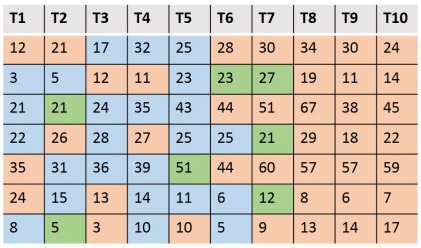

Figure 4: Boxes with a blue background are training data, those with a green background are used as validation and the red ones correspond to the test set.

Table 1: Statistics on the two Datasets: Volume is derivated of the presence of a vehicle on a road on given times-step. Speed information is only present for the Warsaw dataset. The sparsness is the percentage of missing data i.e. the percentage of roads at certain timesteps crossed by no cars.

Datasets	Beijing	Warsaw
Volume	Yes	Yes
Speed	No	Yes
Nb of roads/sequences	24 000	20 000
Size of time-step t	15 min	10 min
Nb of time-steps	672	100
Sparsness	69%	81%

4.3 Learning and Testing Protocol

In order to evaluate our method, we consider a set of training values and a set of testing values. Testing values are of two types: part of the testing values are sampled uniformly in the set of observed values for $t \in [1..T]$, T being the size of the observed data. These testing values will be used for evaluating the quality of the completion model.All values for $t > T$ are considered as testing values and will be used for evaluating our model in prediction. Moreover, a sub-part of the testing values will be used as validation data for tuning the hyperparameters and also the best architectures for f_θ and h_γ. The train/validation/test protocol is illustrated in Figure 4. Note that, from each dataset, we have extracted multiple problems of size T by using sliding windows over the collected data. The window size is 96 time-steps for the Beijing dataset and 40 time-steps for Warsaw.

4.4 Models

We propose to compare the RAINSTORM approach to the following baseline models, some baselines being used for data completion, and some others for prediction.

4.4.1 Completion

- **RecentObs**: This heuristic fills a missing value in a sequence by the most recent observation present in the sequence such as $x_i^{(t)} \leftarrow x_i^{(t-1)}$ if $m_i^{t-1} = 1$, or $x_i^{(t)} \leftarrow x_i^{(t-2)}$ if $m_i^{t-1} = 0$ and so on.
- **MF**: This correspond to the classical matrix factorization framework, described for instance for the task of traffic forecasting in [11].
- **MF-with geographic context**: This method is the one named TSE (traffic speed estimation) in [11]. It consists on minimizing a reconstruction cost on a traffic matrix for which external information such as geographic position in a city is incorporated.

4.4.2 Prediction

- **Last-Point**: This naive baseline assumes that the traffic at $t + 1$ will be the same that traffic at t such as $x_i^{(t)} \leftarrow x_i^{(t-1)}$.
- **NeuralNetwork**: This is the classical baseline method used in traffic forecasting based on a neural network architecture, described for instance in [12].
- **SAE**: This is the method described in [13] ; it consists on a deep architecture of stacked auto-encoders trained on the traffic history.

We also compare RAINSTORM with a model based on a heuristic able to perform both completion and prediction that we call **RoadMean** and can be described as follow: this model predicts and fills missing value with the mean of observed values on the sequence.
The RAINSTORM model has been configured as follows: (i) f_θ is a linear function (ii) three different architectures for h_γ have been tested:

- **RAINSTORM-lin** uses a linear h_γ function
- **RAINSTORM-trans** uses a h_γ such that $h_\gamma(z) = z + \gamma$. It involves less parameters that the previous model.
- **RAINSTORM-mlp** uses a one hidden layer neural network for h_γ with 200 hidden neurons and a hyperbolic tangent as activation function. It allows to model more complex behaviors.

4.5 Experiments and Results

4.5.1 Data Completion

For the first set of experiments, we focus on filling missing values in the two datasets. We have used 50% of the observations for training, 40% for testing and 10% for validation. Each performance has been computed by averaging the results obtained on 20 different runs. In that case, the Δ function is the classical RMSE. Table 3 illustrates the results obtained by the baselines and by our models, considering different sizes N of the latent space. First, one can see that the three RAINSTORM models outperforms the baselines for almost all the tested dimensions N. Particularly, on the Beijing dataset, the RAINSTORM_mlp model with $N = 50$ obtains a RMSE of 2.97 while the best baseline (SAE) only obtains 3.24. This corresponds to a $\sim 10\%$ improvement of the performance. This is due to the ability of the model to both capture the dynamics of the sequences, but also to benefit from the relational information.

4.5.2 Prediction

For the second set of experiments, we focus on the prediction problem. Here, the values at time $t > T$ are removed from the training set; 20% of the removed values are used as validation set and the 80% remaining are the test set. Table 2 shows the performance of the different models for the prediction task using a RMSE evaluation at $t = T + 1$ on the volume or average speed of the cars on each road. It also shows that, used as a prediction model, RAINSTORM obtains higher results than baseline techniques for the two cities. When considering long-term prediction quality (Figure 5), our model is still able to well predict from approximately $T + 1$ to $T + 7$ but fails in predicting long term values since the h_γ function is not a perfect model of the dynamicity.

4.5.3 Making Prediction and Completion simultaneously

In the third set of experiments, we evaluate our model for both completion and prediction. It means that the training set is built by removing observations both for $t \leq T$, and also all the values at time $t > T$ that will be used for evaluating the prediction quality. We compare the performance in completion and prediction to baselines model. Note that, in the pure prediction task, the baselines are learned without considering missing values in the training set for $t \leq T$ since these models are not able to deal with missing values. Table 4 presents the obtained performance. It shows that, if the prediction quality of our model has decreased with comparison to results obtained on the pure prediction task (Table 2), both the performance in completion and prediction remains higher than the performance of baseline models. The prediction decrease is due to the fact that now, our prediction model is trained with missing values for $t \leq T$.

5 Heterogeneous information

Now, we consider a novel experimental problem where, at each timestep, the observation $x_i^{(t)}$ is composed of both a label denoted $x_{i,lab}^{(t)}$ and a real value $x_{i,real}^{(t)}$. Note that we present the heterogeneous model in this case, but more complex data can be handled and the model is not limited to dealing with just one real value and one label. $x_{i,real}^{(t)}$ corresponds to the value used previously, and $x_{i,lab}^{(t)}$ is a label (in a set of three possible labels) that can be *high congestion*,*medium congestion* and *low congestion*. We extend our model by considering two different decoding functions $f_{\theta,real}$ which aims at predicting $x_{i,real}^{(t)}$ and $f_{\theta,lab}$ which predict one label between the three possible ones. These two decoding functions are naturally integrated in the loss of our model by using the following loss:

$$
\begin{aligned}
&\mathcal{L}(\theta, \gamma, \mathbf{z}) = \\
&\frac{1}{O} \sum_{i=1}^{n} \sum_{t=1}^{T} [m_{i,real}^{(t)} (f_{\theta,real}(z_i^{(t)}) - x_{i,real}^{(t)})^2 \\
&+ m_{i,lab}^{(t)} \Delta_{hinge}(f_{\theta,lab}(z_i^{(t)}), x_{i,lab}^{(t)})] \\
&+ \lambda_{dyn} \sum_{i=1}^{n} \sum_{t=1}^{T-1} ||z_i^{(t+1)} - h_\gamma(z_i^{(t)})||^2 \\
&+ \lambda_{struct} \sum_{i,j \in [1..N]^2} \sum_{t=1}^{T} e_{i,j} ||z_i^{(t)} - z_j^{(t)}||^2
\end{aligned}
\tag{2}
$$

Table 2: Prediction at $T+1$, comparison between described baselines models and the RAINSTORM model for different size of latent space N with a root mean square error (RMSE)

N	Model/Dataset	Beijing	Warsaw	
		Volume	Volume	Speed
	RoadMean	5.51	5.09	11.02
	Last-Point	5.28	4.73	12.01
	NeuralNetwork	4.77	4.27	8.05
	SAE	4.75	4.27	7.85
RAINSTORM				
5	Linear	5.07	4.28	7.48
	Translation	4.89	4.32	7.72
	MLP	4.82	4.28	7.74
10	Linear	4.91	4.25	7.40
	Translation	4.85	4.29	8.00
	MLP	4.78	**4.20**	7.21
20	Linear	4.72	4.24	7.33
	Translation	4.67	4.22	7.54
	MLP	**4.54**	4.21	**7.19**
50	Linear	4.73	4.27	7.22
	Translation	4.68	4.39	7.45
	MLP	4.66	**4.20**	7.60

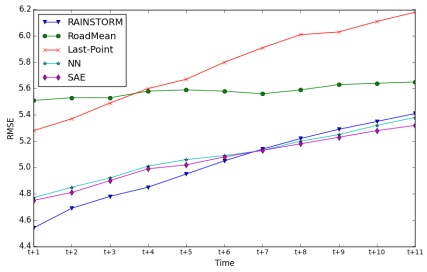

Figure 5: Evolution of RMSE when predicting future time-steps, from $T+1$ to $T+11$ on the Beijing dataset.

Note that, in our experiment, the "real" values and the "lab" values are not forced to be missing for the same cells and thus need to define two different masks $m^{(t)}_{i,real}$ and $m^{(t)}_{i,lab}$. Δ_{hinge} is a classification hinge-loss.

5.1 Experimental results

Table 5 shows the results of our approach when considering only the 'real' values, only the 'lab' values, and when considering the two types of values simultaneously. One can see that, when considering the two types of information simultaneously, our model is able to obtain better performance both in term of RMSE for the 'real' values, and in term of accuracy for the labels, showing that RAINSTORM can also integrate heterogeneous information and can benefit from it.

Table 3: Completion for 50% missing data, comparison between described baselines models and the RAINSTORM model for different sizes N of the latent space with a root mean square error (RMSE)

N	Model/Dataset	Beijing	Warsaw	
		Volume	Volume	Speed
	RoadMean	5.55	5.00	11.10
	RecentObs	5.31	5.11	11.38
	MF	3.58	3.16	6.80
	MF-Geo	3.24	2.99	6.49
RAINSTORM				
5	Linear	3.37	3.13	6.41
	Translation	3.19	3.12	6.32
	MLP	2.99	3.12	6.49
10	Linear	3.01	3.09	6.40
	Translation	3.05	3.17	6.51
	MLP	3.03	3.00	6.24
20	Linear	3.52	2.97	6.47
	Translation	3.21	**2.92**	6.45
	MLP	3.22	2.94	**6.23**
50	Linear	3.53	2.99	6.32
	Translation	3.08	2.95	6.35
	MLP	**2.97**	2.93	6.70

Table 4: Completion and Prediction preofrmance with different levels of missing training values on the Beijing dataset for RoadMean (RM) model and RAINSTORM-mlp (RS) (with a latent space of size 20)

	Completion		Prediction	
Percentage of missing values	RM	RS	RM	RS
5%	4.88	**2.89**	5.51	**4.63**
10%	4.89	**2.89**	5.55	**4.67**
20%	5.04	**2.91**	5.55	**4.79**
30%	5.11	**2.96**	5.64	**4.91**
50%	5.55	**3.22**	5.65	**5.06**
75%	5.90	**3.78**	5.81	**5.26**

Table 5: completion for 50% missing data with the heterogeneous model on the Beijing dataset. Results are given for RAINSTORM-mlp: we consider three setups with real values only, the labels only and an experiment with both types of values.

	$\mathbf{x}$ real only	$\mathbf{x}$ label only	both $\mathbf{x}$
	Accuracy	Rmse	Accuracy/Rmse
N=5	0.71	2.99	0.74/3.11
N=10	0.77	3.03	0.77/3.15
N=20	0.79	3.22	0.84/3.24
N=50	0.76	2.97	0.82/3.05
N=60	0.78	2.99	**0.81/2.96**
N=80	0.77	3.05	0.82/2.98

6 Related Work

The representation learning (and deep learning) is a very active field where different recent works target some of the aspects studied here. For example, the problem of learning representation over sequential data has been used with different approaches like Recurrent Neural Networks [14]. The main difference w.r.t our model is that the RNN-based methods are inductive (the representation is induced from observations) while RAINSTORM is a transductive model (the observations are induced from learned representations) which makes it more suitable for the completion problem: missing values are built from the representations. Some other methods concerning the problem of dealing with relational data has been also proposed with a close technique [6]. In the literature, relational data are usually handled by using semi-supervised models [16] or for graph, transductive classification methods [17].Our model is also related to multi-view deep neural networks like [18] . Here also, our transductive approach is more suitable for data where some of the views (the different types of observations) are missing. As far as we know, there is no existing model in this community that mix relational information, temporal information and heterogeneity. The problem of prediction and completion of time series has been the focus on many different approaches in the signal processing community. Different models that aims at building representations of series at each time step have been proposed like slow features analysis [19]. We do not detail this literature here since our model is different: it has been developed for dealing with more complex data - no only time series - like related heterogeneous temporal sequences composed of real values, labels, etc...

At last, the traffic prediction problem is an old topic. In particular, since the first research work [20], the problem has been studied from the "univariate series" point of view and different techniques have been evaluated like ARIMA (Autoregressive Integrated Moving Average) which have been applied soon after in the context of multivariate time-series [21]. For a large overview on this techniques in the field of traffic forecasting, you can refer to [23]. Machine learning approaches have been studied (SVR, neural networks) the same way (in univariate and multivariate temporal series) [24]. In practice, neural networks are predominant and are at the center of a large number of publications which propose different architectures [25] [12]. Neural Networks are also the baseline competitor which is often used [26]. More recently, deep-learning architecture have been also successfully used in this field [13]. Some models are dedicated to data completion [27]. Several approaches have been proposed to handle this problem: among them, the family of methods based on matrix factorization has shown its effectiveness on several occasions recently like in [28]. These methods were derived in the particular context of traffic forecasting in order to incorporate them external information such as geographic proximity like in [11].

7 Conclusion and Perspectives

We have presented a new way to learn over incomplete multiple sources of temporal relational data sources. The RAINSTORM approach is based on representation learning techniques and aims at integrating in a latent space the observed information, the dynamicity of the sequences of data, and their relations. Moreover, a simple modification of the model allows one to deal with heterogeneous sources. In comparison to baselines models that have been developed for prediction only or completion only, our approach shows interesting performance and is able to simultaneously complete missing values and predict the future evolution of the data. This model opens many perspectives: the first one is to integrate in the model a long-term memory while our approach (like classical neural networks) is limited to short time prediction. Indeed, we think that extending the model to very long temporal sequences is a key issue. Another perspective would be to conceive a model able to deal with temporal sources that produce information at different rates: for example, in the traffic forecasting problem, we want to propose a model that integrates both temporal information extracted from sensors, but also temporal information extracted from Web sources (like Twitter) which is produced at a lower rate.

References

[1] K. Chakraborty, K. Mehrotra, C. K. Mohan, and S. Ranka, "Forecasting the behavior of multivariate time series using

neural networks," *Neural networks*, vol. 5, no. 6, 1992.

[2] J. Honaker and G. King, "What to do about missing values in time-series cross-section data," *American Journal of Political Science*, vol. 54, no. 2, pp. 561–581, 2010.

[3] J. Shang, Y. Zheng, W. Tong, E. Chang, and Y. Yu, "Inferring gas consumption and pollution emission of vehicles throughout a city," in *KDD 2014*, ACM, 2014.

[4] M. T. Asif, N. Mitrovic, L. Garg, J. Dauwels, and P. Jaillet, "Low-dimensional models for missing data imputation in road networks," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, IEEE, 2013.

[5] S. Bourigault, C. Lagnier, S. Lamprier, L. Denoyer, and P. Gallinari, "Learning social network embeddings for predicting information diffusion," in *Proceedings of the 7th ACM international conference on Web search and data mining*, ACM, 2014.

[6] Y. Jacob, L. Denoyer, and P. Gallinari, "Classification and annotation in social corpora using multiple relations," in *Proceedings of the 20th ACM international conference on Information and knowledge management*, pp. 1215–1220, ACM, 2011.

[7] G. Contardo, L. Denoyer, T. Artieres, and P. Gallinari, "Learning states representations in pomdp," *arXiv preprint arXiv:1312.6042*, 2013.

[8] G. Jagadeesh, T. Srikanthan, and X. Zhang, "A map matching method for gps based real-time vehicle location," *Journal of Navigation*, vol. 57, no. 03, pp. 429–440, 2004.

[9] Y. Zheng, "T-drive trajectory data sample," August 2011.

[10] ICDM, "I.e.e.e icdm contest: Tomtom traffic prediction for intelligent gps navigation,"

[11] J. Shang, Y. Zheng, W. Tong, E. Chang, and Y. Yu, "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*, 2014.

[12] M. S. Dougherty and M. R. Cobbett, "Short-term inter-urban traffic forecasts using neural networks," *International journal of forecasting*, vol. 13, no. 1, pp. 21–31, 1997.

[13] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach,"

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, ACM, 2006.

[15] J. Solomon, R. Rustamov, L. Guibas, and A. Butscher, "Wasserstein propagation for semi-supervised learning," in *Proceedings of The 31st International Conference on Machine Learning*, pp. 306–314, 2014.

[16] D. Zhou, J. Huang, and B. Schölkopf, "Learning from labeled and unlabeled data on a directed graph," in *Proceedings of the 22nd international conference on Machine learning*, pp. 1036–1043, ACM, 2005.

[17] Y. Kang and S. Choi, "Restricted deep belief networks for multi-view learning," in *Machine Learning and Knowledge Discovery in Databases*, Springer, 2011.

[18] L. Wiskott and T. J. Sejnowski, "Slow feature analysis: Unsupervised learning of invariances," *Neural computation*, vol. 14, no. 4, pp. 715–770, 2002.

[19] M. S. Ahmed and A. R. Cook, *Analysis of freeway traffic time-series data by using Box-Jenkins techniques*. No. 722, 1979.

[20] A. Stathopoulos and M. G. Karlaftis, "A multivariate state space approach for urban traffic flow modeling and prediction," *Transportation Research Part C: Emerging Technologies*, vol. 11, no. 2, pp. 121–135, 2003.

[21] Y. Kamarianakis and P. Prastacos, "Forecasting traffic flow conditions in an urban network: comparison of multivariate and univariate approaches," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 1857, no. 1, 2003.

[22] E. I. Vlahogianni, J. C. Golias, and M. G. Karlaftis, "Short-term traffic forecasting: Overview of objectives and methods," *Transport reviews*, vol. 24, no. 5, pp. 533–557, 2004.

[23] Q. Li, "Short-time traffic flow volume prediction based on support vector machine with time-dependent structure," in *Instrumentation and Measurement Technology Conference, 2009. I2MTC'09. IEEE*, pp. 1730–1733, IEEE, 2009.

[24] C. De Fabritiis, R. Ragona, and G. Valenti, "Traffic estimation and prediction based on real time floating car data," in *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*, pp. 197–203, IEEE, 2008.

[25] M. Lippi, M. Bertini, and P. Frasconi, "Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 14, no. 2, pp. 871–882, 2013.

[26] G. E. Batista and M. C. Monard, "An analysis of four missing data treatment methods for supervised learning," *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 519–533, 2003.

[27] J. M. Hernandez-lobato, N. Houlsby, and Z. Ghahramani, "Probabilistic matrix factorization with non-random missing data," in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pp. 1512–1520, 2014.