Deep Learning to the Test: an Application to Traffic Data Streams

Metodologie di apprendimento approfondito applicate a dati di traffico stradale

Nina Deliu and Pierpaolo Brutti

Sommario Deep learning is a broad class of machine learning techniques based on learning data representation through multiple levels of abstraction. It has been successfully applied in several areas of research, but very few literature addressed the problem of traffic flow forecasting. Thus, driven by the belief that deep learning algorithms may capture the sharp traffic data non-linearities, we aimed to develop a deep architecture, namely a feed-forward neural network, and evaluate its performances in predicting short-term traffic streams. We illustrate our methodology, consisting in a predictors selection step and a subsequent training step, using traffic speed data of the Grande Raccordo Anulare road of Rome for the month of June 2016.

Sommario *Gli algoritmi di apprendimento approfondito costituiscono una vasta classe di algoritmi machine learning basati sulla rappresentazione dei dati tramite molteplici livelli di astrazione. Essi sono stati applicati con successo in diverse aree di ricerca, tuttavia solo una piccola parte della letteratura deep learning si è interessata al problema della previsione di dati di traffico. Considerando la capacità di questi algoritmi di catturare non linearità presenti all'interno dei dati, ci siamo proposti di sviluppare un'architettura deep learning per prevedere il traffico a breve termine. Illustriamo la nostra metodologia, consistente in una fase di selezione dei predittori e una di apprendimento della rete, considerando una dataset di dati di velocità di veicoli sulla rete autostradale Grande Raccordo Anulare di Roma.*

Nina Deliu
Dipartimento di Scienze Statistiche, Sapienza Università di Roma, Piazzale A. Moro, 5 00185, Roma, Italy e-mail: nina.deliu@uniroma1.it

Pierpaolo Brutti
Dipartimento di Scienze Statistiche, Sapienza Università di Roma, Piazzale A. Moro, 5 00185, Roma, Italy e-mail: pierpaolo.brutti@uniroma1.it

1 Introduction

Deep learning is a rather wide class of machine learning techniques based on learning data representation through multiple levels of abstraction. These methods have dramatically improved the state of the art in computer vision, natural language processing, speech recognition, object detection and many other domains such as drug discovery and genomics. However, according to the complexity of data, the prediction problem may be more or less difficult to perform. For instance, when aiming to model a time series, one should take into account the sequence dependence among the input values and handle the underlying temporal dynamics. While some areas relying on time series are well exploited and widely modelled with deep neural networks (e.g., speech recognition or financial applications [1]), some others are still poorly addressed, especially in the Italian research literature. One of these, is represented by road traffic modeling, which is a challenging and increasingly relevant field of research, as real-time forecasting may give travelers the ability to choose better routes and authorities the ability to manage the transportation system. Driven by the belief that deep learning algorithms might be able to capture nonlinear spatio-temporal effects in recurrent and non-recurrent traffic patterns, we aimed to develop a deep architecture and to asses if it may outperform standard modelling approaches in predicting short-term traffic streams.

2 Methods and Data

Deep learning algorithms, also known as deep artificial neural networks (ANNs) are statistical models directly inspired by biology, more precisely by the neural network of the human brain [2]. An ANN is organized as a graph, whose nodes or units are structured in multiple layers and interconnected by links to propagate activation from the initial to final units. Each link has a weight that determines the relative strength and sign of the connection, and each unit applies an activation function to the weighted sum of all incoming activations. The quintessential and most common deep learning architecture is the feed-forward neural network (FFNN), and it can be viewed as a directed acyclic graph [3].

The model: feed-forward neural network

Feed-forward neural networks are represented as a composition of many different non-linear functions, with the characteristic of flowing information hierarchically from the initial inputs X to all the neurons of the network through intermediate computations.

Formally, a deep feed-forward architecture can be described as follows. Let's suppose our network is composed by L hidden layers indexed as $l = 1,\dots,L$. Let $Z^{(l)}$ denote the vector of inputs into a generic layer $l+1$, $Z^{(0)}$ the vector of original inputs X and $Z^{(L+1)}$ the output vector of predictions $\hat{Y}(X)$. Using a matrix-vectorial notation, the model is given by

$$\begin{aligned}
X &= Z^{(0)} \\
Z^{(1)} &= f^{(1)}\left(W^{(1)}Z^{(0)} + b^{(1)}\right) \\
\dots \\
Z^{(L)} &= f^{(L)}\left(W^{(L)}Z^{(L-1)} + b^{(L)}\right) \\
\hat{Y}(X) &= Z^{(L+1)} = f^{(L+1)}\left(W^{(L+1)}Z^{(L)} + b^{(L+1)}\right).
\end{aligned}$$

where $f^{(1)}, \dots, f^{(L+1)}$ are the non-linear activation functions, $W^{(1)}, \dots, W^{(L+1)}$ the weight matrices and $b^{(1)}, \dots, b^{(L+1)}$ the bias or activation levels.
In summary, a feed-forward algorithm applies $L+1$ non-linear transformations to the input data X. Each transformation takes as input the output of the previous transformation, and gives a new abstract representation of the data.

Dataset

This study is based on a dataset of vehicle traffic streams obtained from QMap S.r.l. company, specialized in Intelligent Transport System and Info-mobility. All data were acquired by Global Positioning System (GPS) probes, which typically contain information on latitude, longitude, altitude, heading, speed and, most importantly, precision. This technology offers a low capital cost, a low installation cost, and a low data collection cost combined with a high location accuracy.
More precisely, data analyzed in this study consist in vehicle traffic speeds concerning the Grande Raccordo Anulare (GRA) road, that is a toll-free, ring-shaped orbital motorway, with 68.2 kilometres of circumference that encircles Rome. It represents one of the highest traffic volume Italian freeways, exceeding 160,000 vehicles per day (58 millions per year) [4]. In our dataset, the GRA road is partitioned in overall 279 locations (500 meters segments). For each location, it was recorded the respective traffic speed for the whole month of June 2016. With a length of 30 days and with observations available every 3-minutes interval, for this month were collected overall 14,400 observations for each GRA location.

3 Deep Learning for Traffic forecasting

Traffic speed data are represented as an ordered sequence of elements indexed by a regular (e.g., every day or ever year) time interval t and by a spatial index s. The goal in vehicle traffic forecasting is to develop a predictor of the quantity of interest. Suppose that for a certain location s we have observations up to time t, which we denote with $x_s^t := (x_{1,s}, \dots, x_{t,s})$, and we want to predict the traffic speed for that location at a future time $t+t'$, i.e., $X_{t+t',s}$. Considering S the overall number of locations, the input matrix X^t of all the available observations is given by

$$X^t = \begin{bmatrix} x_{1,1} & \dots & x_{t-k,s} & \dots & x_{1,S} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{t,1} & \dots & x_{t,s} & \dots & x_{t,S} \end{bmatrix}$$

The time point $t = 1$ is referred to the starting point of data collection or the first time for which we have information concerning the variable. However, it may be often reasonable to select an appropriate sub-interval time period and use a subset of observations for prediction, for example taking as initial time point $t-k$ and use $(x_{t-k,s}, \dots, x_{t,s})$ to predict $X_{t+t',s}$. In this case, our deep learning predictor of $X_{t+t',s}$ is given by

$$\begin{aligned} X^t_{t+t',s} &= \hat{Y}(x^t_s) = \hat{Y}(x_{t-k,s}, \dots, x_{t,s}) = \\ &= f^{(L+1)} \circ \dots \circ f^{(1)} \left(W^{(1)} x^t_s + b^{(1)} \right) = \\ &= f^{(L+1)} \circ \dots \circ f^{(1)} \left(Z^{(1),t}_s \right), \end{aligned}$$

where $f^{(1)}, \dots, f^{(L+1)}$ and L represent the hyperparameters of the deep artificial neural network.

Our prior assumption is that to predict traffic at a given location we might need to use only recent measurements. In addition, is computationally prohibitive and somehow unreasonable to use data from every single road segment to forecast the speed of that location. The problem of selecting an optimal initial time point $t-k$ and the relevant locations, thus an optimal input matrix, is equivalent to perform a predictor selection task to find the spatio-temporal relations in the data. A predictor selection problem requires an algorithm to find a sparse model [5].

Predictor Selection

Our predictors selection procedure involves two widely used models in the time series framework, with the first one being an Autoregressive Integrated Moving Average (ARIMA) model [6, 7]. By taking into account the temporal dependencies in the data, it allows us to estimate the optimal number of temporal lags to be used. The estimated number of lags k is then used to identify the spatial relation as well, by developing a sparse Vector Autoregressive Model (sVAR). We thus consider the problem of finding a sparse matrix A in the following model:

$$X^t_{t+t'} = Ax^t + \varepsilon_t, \;\; \varepsilon_t \sim N(0, V),$$

where A is a matrix of size $S \times Sk$, with S the number of locations and k the number of previous measurements or lags.

Our predictors selected as the result of finding the sparse linear model are then used as input values at the first layer to build the deep learning model.

The training process

The training problem of our supervised deep learning model consists in finding the set of learning parameters $(\hat{W}, \hat{b})$ that optimally predicts the outcome of interest. To do this, we use a training set $D = \{Y_i, X_i\}_{i=1}^N$ of input-output pairs to train the model by solving an optimization problem as the following

$$\arg\ \min_{W,b} \frac{1}{N} \sum_{i=1}^{N} ||Y_i - \hat{Y}^{W,b}(X_i)||_2^2. \tag{1}$$

When we have a non-linear model, the most loss functions cannot be optimized in a closed form [8] and are required iterative numerical optimization procedures, such as gradient descent [9]. Gradient descent algorithm tells us how to learn the the parameters we're interested in, but it supposes the ability to compute the gradient of the loss function, i.e., all the partial derivatives. A good algorithm employed to this purpose is Back-propagation [10]. It demonstrated to behave really better and faster than other earlier methods, becoming the workhorse of learning in neural networks. The feed-forward deep neural network was implemented with the `h2o` package in `R`. It was evaluated on both in-sample and out-of-sample metrics (i.e., mean squared error and R^2) based on a random split of 70%, 10% and 20% for the training, validation and test set, respectively.

4 Results and Conclusions

Before testing our deep learning model, we performed a Grid Search to find the optimal network (number of layers L, number of units N_l for each layer l, the activation functions f, and the regularization lasso λ_1 and ridge λ_2 penalty parameters). Based on $N = 10^4$ Monte Carlo samples, we obtained the following neural network

$$\begin{aligned} f &= \texttt{ReLU} \\ (\lambda_1, \lambda_2) &= (1\text{e}-5, 0) \\ \texttt{structure} &= (100, 50, 100, 50). \end{aligned}$$

In Table 1 we show the training and test error of our deep learning model compared with a general linear model (GLM). Both models are based on the same input sets, in a first step we consider the previous 10 time measurements and in a second step we only take into account the optimal spatio-temporal structure estimated with the ARIMA and sVAR models.

We noticed a non uniform model performance throughout the day, based on recurrent and non recurrent patterns of the proposed time series. However, compared to general linear models, the deep learning models showed much better results in terms of both in sample and out of sample outcomes, particularly with the optimal number of lags and the selected two adjacent locations.

Further significant improvements are likely to be obtained with other classes of deep learning, e.g., recurrent neural networks [11] or long short term memory networks [12], and future analyses should be carried out to evaluate their performances.

Tabella 1 In Sample and Out of Sample metrics according to a general linear model (GLM) and a deep learning model (DLM). 10 and 2 indicates 10 and 2 previous time measurements. All the values are based on a Monte Carlo sample.

	MSE		R^2	
Model	In Sample	Out of Sample	In Sample	Out of Sample
GLM10	36.48	37.95	0.89	0.89
DLM10	28.36	36.53	0.92	0.89
GLM2	37.47	37.57	0.89	0.89
DLM2	31.54	33.96	0.91	0.90

Riferimenti bibliografici

1. J. B. Heaton, N. G. Polson, and J. H. Witte, “Deep learning for finance: deep portfolios,” *Applied Stochastic Models in Business and Industry*, 2016. asmb.2209.
2. W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.
3. Y. Lecun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444, 5 2015.
4. ANAS, “ANAS, Grande Raccordo Anulare di Roma,” 2017. [Online; accessed 28-nov-2017].
5. N. Polson and V. Sokolov, “Deep Learning Predictors for Traffic Flows,” *ArXiv e-prints*, Apr. 2016.
6. G. J. G.E.P. Box, *Time Series Analysis, Forecasting and Control.* San Francisco, CA: Holden-Day, revised edition ed., 1970.
7. G. P. Zhang, “Time series forecasting using a hybrid arima and neural network model.,” *Neurocomputing*, vol. 50, pp. 159–175, 2003.
8. I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning.” Book in preparation for MIT Press, 2016.
9. L. A. Cauchy, “Methode generale pour la resolution des systemes d’equations simultanees,” *Compte Rendu a l’Academie des Sciences*, vol. 25, pp. 536–538,, 1847.
10. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” in *Neurocomputing: Foundations of Research* (J. A. Anderson and E. Rosenfeld, eds.), pp. 696–699, Cambridge, MA, USA: MIT Press, 1988.
11. A. Graves and J. Schmidhuber, “Offline handwriting recognition with multidimensional recurrent neural networks,” in *Advances in Neural Information Processing Systems 21* (D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, eds.), pp. 545–552, Curran Associates, Inc., 2009.
12. F. Gers, “Long short-term memory in recurrent neural networks,” 2001.