Traffic Flow Forecasting on Data-Scarce Environments Using ARIMA and LSTM Networks

Bruno Fernandes[1(✉)], Fábio Silva[1,2], Hector Alaiz-Moretón[3], Paulo Novais[1], Cesar Analide[1], and José Neves[1]

[1] Department of Informatics, ALGORITMI Centre, University of Minho, Braga, Portugal
`bruno.fmf.8@gmail.com`, `{fabiosilva,pjon,analide,jneves}@di.uminho.pt`
[2] CIICESI, ESTG, Polytechnic Institute of Porto, Felgueiras, Portugal
[3] Department of Electrical and Systems Engineering, Universidad de León, Escuela de Ingenierías, León, Spain
`hector.moreton@unileon.es`

Abstract. Traffic flow forecasting has been in the mind of researchers for the last decades, remaining a challenge mainly due to its stochastic nonlinear nature. In fact, producing accurate traffic flow predictions would be extremely useful not only for drivers but also for those more vulnerable in the road, such as pedestrians or cyclists. With a citizen-first approach in mind, forecasting models can be used to help advise citizens based on the perception of outdoor risks, dangerous behaviors and time delays, among others. Hence, this work develops and evaluates the accuracy of different ARIMA and LSTM based-models for traffic flow forecasting on data-scarce and non-data-scarce environments. The obtained results show the great potential of LSTM networks while, in contrast, expose the poor performance of ARIMA models on large datasets. Nonetheless, both were able to identify trends and the cyclic nature of traffic.

Keywords: Traffic flow forecasting · Data-scarce environments · Long Short-Term Memory · AutoRegressive Integrated Moving Average · Road safety

1 Introduction

In our days, road safety has become a major concern of our society. This fact is easily explained with the substantial number of deaths happening on the road every day. Even though many stakeholders are focusing on vehicles and roads, one should not dismiss those more vulnerable at the road, known as Vulnerable Road Users (VRUs). Such road users are typically defined as pedestrians or cyclists, with their vulnerability arising from the lack of external protection, their age or disabilities, among others. Indeed, an approach that aims to increase VRUs' safety focuses on modelling traffic prediction. Works that already produce such outcomes are typically focused on vehicles and drivers. However, such information could be extremely useful for VRUs.

A simple example is the one of a cyclist who pretends to know which hour will have less traffic so that he could go cycling. As in all problems that involve forecast, one is required to have a set of data to work with. However, there may be situations where such data is less substantial. That should not be a motive to dismiss the problem.

Within the Machine Learning field, many different models and algorithms can be used to predict future points. In particular, for time series analysis there are two that stand out: AutoRegressive Integrated Moving Average (ARIMA) and Long Short-Term Memory (LSTM). In this work we develop models based on the referred algorithms, and evaluate and validate its accuracy and performance for traffic prediction in data-scarce and non-data-scarce environments. Such information is then shared with VRUs who can opt to avoid certain roads or certain hours to travel or do sports. Hence, with respect to this work the following research questions were elicited:

- RQ1. Which algorithm, ARIMA or LSTM, has better accuracy on data-scarce environments and which has better accuracy on non-data-scarce ones?
- RQ2. Do these algorithms behave better (in terms of accuracy and performance) on non-data-scarce environments when compared to data-scarce ones?

The remaining of this paper is structured as follows, viz. The next section reviews previous studies on traffic forecasting. The third section introduces the materials and methods, focusing on describing the dataset and data preparation, the assessed metrics and the developed ARIMA and LSTM models. The fourth section focuses on the performed experiments while section five gathers, analyses and discusses the obtained results. Finally, conclusions are gathered and future work is outlined.

2 Literature Review

This work focuses on the problem of traffic modelling on data-scarce and non-data-scarce environments using two distinct models. On the one hand we have ARIMA, developed over the Auto Regressive Moving Average (ARMA) model [1]. ARIMA has been implemented on many domains such as temperature and pollution prediction [2], and short-term traffic flow prediction [3], among others. On the other hand, LSTM consists of a special case of Recurrent Neural Networks (RNN), with recent years showing a significant increase in the use of LSTM in domains such as speech recognition [4], traffic flow prediction [5] and many others.

Researchers have been devoting many efforts to the use of ARIMA, LSTM or other models or algorithms for traffic flow prediction. However, for our knowledge, there is yet to be studied the accuracy and error rates on data-scarce environments against non-data-scarce ones. Nonetheless, an interesting work that studies the use of these models for traffic flow prediction is the one performed by Fu et al. [5], where these authors used LSTM to predict short-term traffic flow, showing that LSTM had a slightly better performance than ARIMA. In fact, short-term traffic forecast focuses in the near future, ranging from some minutes to some dozens of minutes. In terms of data, the authors used PeMS dataset, which has over 15 000 sensors deployed state-wide in California. Another study focused on short-term traffic flow is the one performed by Zhao et al. [6] where these authors propose a model applied to a dataset with 25.11 million records.

To evaluate the performance of the model they used Mean absolute error (MAE), mean square error (MSE) and mean relative error (MRE) as criteria. For their model and datasets, LSTM proved to behave better than ARIMA, specially for long forecast windows. On the other hand, Ma et al. [7] proposed a LSTM model to capture non-linear traffic dynamic in an effective manner. Their dataset consisted of 42387 records. The developed models were used to predict speed in the next 2 min based on speed and volume in the previous period on the same day. Interestingly, LSTM outperformed both RNN and Support Vector Machine models by, at least, 28%. Moreover, considering a multivariate model, LSTM also showed to be superior. The multivariate model also showed to be better than the univariate one. Another approach to the problem being addressed by this work involves the use of ensemble models based on ARIMA for traffic estimation on urban areas [8].

3 Methods and Materials

The next lines describe the materials and methods used in this work with the objective of creating time series forecasting models for traffic in smart cities, enabling the deployment of geographical models to advise VRUs in aspects related to traffic flow.

3.1 Dataset

The dataset was created from scratch and contains real world data. A software was developed to collect data from a set of public APIs. In particular, TOMTOM Traffic Flow API was the one used to create the traffic dataset, which has, as features, the city name; the functional road class describing the road type; the current average speed at the selected point; the free flow speed expected under ideal conditions; current travel time, in seconds; the travel time in seconds which would be expected under ideal free flow conditions; the confidence on the values; the ratio between live and the historical data; the coordinates; and a timestamp. The software went live on 24 July 2018 and has been collecting data uninterruptedly. It works by making an API call every twenty minutes using an HTTP Get request. It parses the received JSON object and saves the records on the database. The software was made so that any other road of any city or country can be easily added to the fetch list. For this work, the dataset included data until 11 October 2018, which consists of 80 days of data. The software is modular, configurable and is able to build more datasets specifically focused on pollution data, the weather and crowd sensing data. An initial approach was to consider a multi-variate model comprising both traffic and the weather. However, since during the 80 days that comprise the traffic dataset the weather was constant in terms of temperature, humidity and precipitation, the final decision was to develop a univariate model.

3.2 Data Preparation and Preprocessing

To answer our research questions, two distinct datasets were built. The first one, Dataset A, was used to evaluate the models on data-scarce environments and consisted of a total of 80 records without missing values. Each record corresponds to an entire day (grouped by day) and has, as main features the city name; the timestamp; the confidence average; the speed difference average, for each day, between the current speed at the selected points and the expected free flow speed; the expected free flow speed; the travel time difference average, for each day, between the current travel time in seconds and the expected travel time under free flow conditions; and the expected free flow travel time. On the other hand, the second dataset, Dataset B, was used to evaluate the models on non-data-scarce environments. The features of this dataset are the same as the ones describe in the previous lines. The difference was on the number of records. While the first dataset contained a record for each day, this second dataset consisted of a total of 1894 records. The same 80 days were used in both datasets, but while the first is grouped by day, the latter is grouped by hour. No missing values were present, but some hours are missing due to situations where the API limits were reached or the API was unavailable.

The criterion for dataset normalization was based on the following equation:

$$\frac{X_i - \min(X)}{\max(X) - \min(X)} \tag{1}$$

The MaxMinScaler data input pre-processing has been implemented with *sklearn. preprocessing.MinMaxScaler* [9]. In the case of ARIMA, data was transformed into a set of differences to make data stationary as requested by the model.

3.3 Assessed Metrics

Aiming to get the best possible combination of parameters for the developed models, we opted to optimize the MSE. The goal was to find the best combination of the parameter grid which would minimize MSE and give the best model. Root Mean Squared Error (RMSE) and MAE were also used.

3.4 Arima

ARIMA is a forecasting algorithm originally developed by Box and Jenkins [10]. It belongs to a class of univariate autoregressive algorithms used in forecasting applications based on timeseries data. The Arima models are generally defined by three parameters (q, d, p), where *q* is the order of the autoregressive components; *d* is the number of differencing operators; and *p* is the highest order of the moving average term. The parameters control the complexity of the model and, consequently, the Auto Regression, Integration and Moving Average components of the algorithm [11], i.e.:

$$\check{y}_t = \Phi_1 \Upsilon_{t-1} + \Phi_2 \Upsilon_{t-2} \ldots + \Phi_p \Upsilon_{t-p} + a_t - \theta_1 a_{t-1} \ldots - \theta_p a_{t-q}. \tag{2}$$

Where:

- y denotes a general time series;
- $\check{y}_t$ is the forecast of the time series y for time t;

- $\Upsilon_{t-1}\ldots\Upsilon_{t-p}$ are the previous p values of the time series y (and form the autoregression terms);
- $\Phi_1\ldots\Phi_p$ are coefficients to be determined by fitting the model;
- $a_t\ldots a_{t-q}$ is a zero mean white noise process (and forms the moving average terms);
- $\theta_t\ldots\theta_{t-q}$ are coefficients to be determined by fitting the model.

ARIMA works best when data has stable past correlations with few outliers, where its performance is usually better. On the other hand, the search for optimal parameters is not straightforward and requires analysis and experimentation. Nevertheless, ARIMA models should be kept simple and have no more than the value 3 for Autoregression or Moving Average terms.

3.5 LSTM

LSTM networks are a specific kind of convolutional networks. They are based on the idea that they can connect previous information to the present task, enabling memory capacities. Therefore, LSTM has been designed for long-term dependency problems as they are able to remember information for periods of time. LSTM core is based on contextual state cells that work as long-term or short-term memory cells. Thus, the output of a LSTM is handled by the state of these cells. This feature becomes interesting when it is necessary to develop a forecast model that is time dependent. An LSTM layer consists of a set of recurrently connected blocks, known as memory blocks, memory cells or just cells that are the computational units of the LSTM network [12]. These cells are composed by weights and gates. Each memory block contains one or more recurrently connected memory cells and three multiplicative units: input, output and forget gates. The gates allow the net to interact with the cells. The forget gate and the input gate update the internal state, while the output gate is the final limiter of the cells' output. These gates and the consistent dataflow called CEC, or Constant Error Carrousel, keep each cell stable. For the implementation of this network, we used the Keras framework for python with Tensor flow backend, using GPU processing and an improved performance version that uses CuDNN (CudnnLSTM).

4 Experiments

This section describes the set of experiments that have been implemented in this work, as well as the grid-search associated with each model. In fact, each model and their respective parameter grid were applied to the datasets explained before, i.e.:

- Dataset A: 80 records, grouped by day, describing 80 days;
- Dataset B: 1894 records, grouped by hour, describing the same 80 days:
  - Sub-dataset B0: all the 1894 records;
  - Sub-dataset B1: 1694 records as result of 1894-200;
  - Sub-dataset B2: 1294 records as result of 1894-600.

To obtain the best possible combination of parameters for ARIMA and LSTM, a nested cross-validation approach was implemented. This technique is based in the creation of several subsets. In this case, three was the number of created subsets for Dataset B: the first uses all data available (sub-dataset B0), the second had 200 records removed (sub-dataset B1) and the last had 600 records removed (sub-dataset B2). This strategy was applied to verify the model's forecast, to test how the models handle different amounts of data and if the produced errors are stable. Due to the small size of Dataset A, this sub-division was not implemented there. Dataset A and all sub-datasets B were then split into two slices. For Dataset A, 71 cases were used for training while 8 were used for calculating error metrics. These were the possible numbers due to the few records available in Dataset A. On the other hand, for sub-datasets B, the fist split retains, for training purposes, 75% of data. The second split, which contains the remaining data, is used to test the model and calculate the final error values.

4.1 Arima Experiments

A pre-condition of the ARIMA model is that data must be stationary. So, in order to successfully run the models, the dataset required preparation and its mean and variance needed to be stationary. That was accomplished by turning the original dataset into a dataset of differences values where an input is the difference between two sequential inputs. After successfully preparing the data, the problem is then proposed to different ARIMA models based on different (p, d, q) parameters in order to find the best performing models. This required repeated experiments with the sample dataset in the same conditions to directly compare algorithms errors under different conditions. The experiments were carried out for values in the range [0–10] for all parameters for the autoregressive component and in the range [0–2] for the remaining parameters. The decision to increase the order of the Auto Regression component was made because it yields better results each time, however the time required to run the model was noticeably larger. Safeguards were also implemented since some configurations of the ARIMA model do not converge, resulting in algorithm exceptions.

4.2 LSTM Experiments

Due to the random initialization of LSTM weights, five repetitions of each combination of parameters were performed, taking the mean of RMSE to verify the model's quality. A method to define a seed for eliminating the random feature of LSTM was not chosen mainly due to the small size of the datasets, specifically, Dataset A. The grid of parameters chosen for implementing LSTM in Dataset A is as follows:

- Batch size: for Dataset A = [2, 4, 10, 20]; for sub-datasets B = [12, 24, 48];
- Window size: for Dataset A = [3, 7, 10]; for sub-datasets B = [8, 24, 48].

The batch size parameter defines the number of cases that were included in each epoch. The windows size defines the number of past cases used for learning, considering that a big window may produce worst results before the presence of cyclic behaviours in times series. The remaining parameters are described in the following code:

```
model = Sequential()
model.add(CuDNNLSTM(128, inputshape=(window_size,
n_features=1]),return_sequences=True))
model.add(Dropout(0.2))
model.add(CuDNNLSTM (64, return_sequences=False))
model.add(Dropout(0.2))
model.add(Dense(32,activation='relu',  kernel_initializer
='uniform'))
model.add(Dropout(0.2))
model.add(Dense(1))
model.compile(loss='mse', optimizer='RMSprop')
```

CuDNNLSTM refers to LSTM layers oriented for execution in a GPU-CUNDD environment, while the ‘RMSprop’ parameter defines the optimizer. This restricts the oscillations in the vertical direction, thus improving the learning rate.

5 Results and Discussion

To validate the results, a set of 4 independent experiments were performed. The experiments made forecasts based on the average speed noticed on roads of a city. For Dataset A, the data-scarce one, the goal was to predict the last 7 days without prior knowledge. Results show that both models performed similarly, with the best ARIMA model being slightly worse than LTSM. However, training time for an ARIMA of order 10 for Auto Regression is more time intensive for each forecast. Table 1 shows the obtained results while Fig. 1 gives a graphic view of predictions.

Table 1. MAE, RMSE and MSE of ARIMA and LSTM models as well as the used parameters.

Dataset	Model	MAE	RMSE	MSE	Parameters
Dataset A	ARIMA	0.66160	0.91824	0.84316	(10, 1, 1)
Dataset A	LSTM	0.70257	0.90454	0.82301	Window size = 7 Batch size = 20
Sub-dataset B0	ARIMA	1.84567	2.79692	7.82278	(3, 1, 1)
Sub-dataset B0	LSTM	1.40967	2.16929	4.70613	Window size = 24 Batch size = 50
Sub-dataset B1	ARIMA	1.85845	2.65893	7.06989	(7, 0, 1)
Sub-dataset B1	LSTM	1.47858	2.30358	5.30649	Window size = 24 Batch size = 50
Sub-dataset B2	ARIMA	1.85311	2.68335	7.20038	(3, 0, 1)
Sub-dataset B2	LSTM	1.40726	1.91171	3.65474	Window size = 48 Batch size = 30

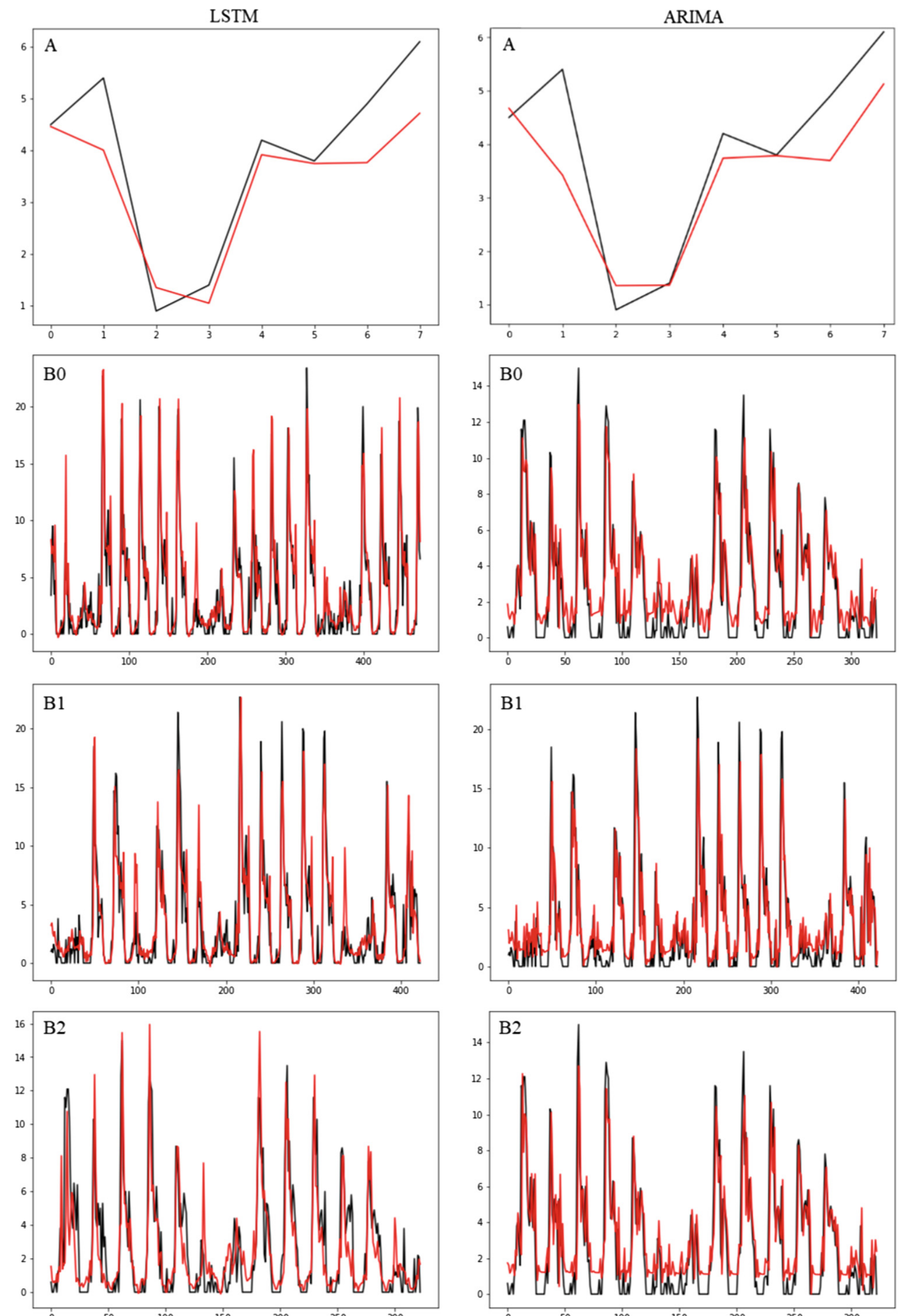

Fig. 1. Comparison of real values (black) vs predicted ones (red) for LSTM (left) and ARIMA (right), and for dataset A and sub-datasets B0, B1 and B2. X axis for time and Y axis for the speed difference average values.

Over the next set of experiments, the dataset was changed to an hourly dataset (Dataset B). Predictions are made using real data and produce one prevision at a time. The real past value is added to the dataset and compared to predicted values to assess MAE, RMSE and MSE. Regarding the obtained results for sub-dataset B0, LTSM showed to have better performance. The ARIMA model suffered from memory overflows, weight calculations and lack of convergence errors, and, so, the best computable model has a lower order of magnitude for the Auto Regressive parameter. Regarding sub-dataset B1, it has fewer instances but LTSM continues to show superiority over ARIMA, which still struggles with memory intensive operations. In the last sub-dataset, B2, even fewer cases were used. ARIMA was still not able to outperform LTSM. This provides evidence that ARIMA models may be better suited when data is scarce as in the case with dataset A. In addition, it becomes evident that the performance of LTSM is more constant over time, with consistently better results. In fact, the larger the dataset the worse ARIMA performs. In both Table 1 and Fig. 1, it is evident that LTSM yields better results than ARIMA, which presents similar performance on sub-datasets B0, B1 and B2. In fact, both Arima and LTSM have a constant error rate throughout the experiments, independently of the size of sub-dataset B.

In terms of computational time, ARIMA models are more time consuming when the (p, d, q) order increases to the point where it is unfeasible to train new models. LTSM optimized networks take significantly less time to train and once trained can yield constant predictions based on the data provided in contrast to ARIMA models, which need to be retrained.

6 Conclusions and Future Work

The use of forecasting techniques applied to specific roads and intersections has interesting applications in the context of smart cites, apart from the usual use case of traffic management. The approach presented in this paper compares two different algorithms used for mean velocity forecasting in real world roads. The results provide evidence that both algorithms can produce meaningful outputs and correct forecasting's using real world data both in data-scarce and non-data-scarce environments. This translates to accurately describing the velocity at which vehicles usually make their movement. To a smart city citizen such information is helpful to choose roads to perform outdoor activities or choose pedestrian routes. It was found that LTSM has better performance than ARIMA, however both identified trends and the cyclic nature of traffic. In the case of ARIMA, it is evident that the time needed to run the model is significantly lower in data-scarce environments when compared to non-data-scarce ones. LSTM has the ability to handle large sets of data and after the training period it runs very fast. Hence, as direct answers to the elicited research questions, we can say that LSTM has better accuracy on data-scarce and non-data-scarce environments. In terms of computational performance, LSTM proved to be better in both datasets, with ARIMA showing a poor performance on non-data-scarce environments.

In the context of traffic forecasting on smart cities, more focus will be given to the use of LTSM models or ensemble algorithms based on LTSM due to the continuous nature of data. Other ensemble methods using multi-variate estimations are also being considered. We will also focus on creating large geographical maps supported by LTSM models to predict traffic flow and velocities. The obtained results are promising and, so, next iterations will include an extension to massive online forecasting systems able to handle the complexity and volume of smart cities data streams.

Acknowledgments. This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT – *Fundação para a Ciência e Tecnologia* within the Project Scope: UID/CEC/00319/2013, being partially supported by a Portuguese doctoral grant, SFRH/BD/130125/2017, issued by FCT in Portugal.

References

1. Cortez, P., Rocha, M., Neves, J.: Evolving time series forecasting ARMA models. J. Heuristics **10**(4), 415–429 (2004)
2. Babu, C., Reddy, B.: Predictive data mining on Average Global Temperature using variants of ARIMA models. In: IEEE International Conference on Advances in Engineering, Science And Management (ICAESM 2012), pp. 256–260 (2012)
3. Li, K., Zhai, C., Xu, J.: Short-term traffic flow prediction using a methodology based on ARIMA and RBF-ANN. In: 2017 Chinese Automation Congress (CAC), pp. 2804–2807 (2017)
4. Graves, A., Mohamed, A., Hinton, G.: Speech recognition with deep recurrent neural networks. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649 (2013)
5. Fu, R., Zhang, Z., Li., L.: Using LSTM and GRU neural network methods for traffic flow prediction. In: 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC), pp. 324–328 (2016)
6. Zhao, Z., Chen, W., Wu, X., Chen, P., Liu, J.: LSTM network: a deep learning approach for short-term traffic forecast. IET Intel. Transp. Syst. **11**(2), 68–75 (2017)
7. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transp. Res. Part C: Emerg. Technol. **54**, 187–197 (2015)
8. Guan, W., Hua, X.: A combination forecasting model of urban ring road traffic flow. In: Proceedings of the IEEE Intelligent Transportation Systems Conference, pp. 671–676 (2006)
9. Scikit-learn: sklearn.preprocessing.MinMaxScaler (2018). http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. Accessed 10 Oct 2018
10. Box, G., Jenkins, G.: Time Series Analysis: Forecasting and Control (1976)
11. Van Der Voort, M., Dougherty, M., Watson, S.: Combining Kohonen maps with arima time series models to forecast traffic flow. Transp. Res. Part C: Emerg. Technol. **4**(5), 307–318 (1996)
12. Graves, A., Schmidhuber, J.: Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Netw. Off. J. Int. Neural Netw. Soc. **18**(5–6), 602–610 (2005)