POLITECNICO DI TORINO
Repository ISTITUZIONALE

# Short term urban traffic forecasting using deep learning

*Original*
Short term urban traffic forecasting using deep learning / Albertengo, G.; Hassan, W.. - ELETTRONICO. - IV-4/W7:(2018), pp. 3-10. (Intervento presentato al convegno 3rd International Conference on Smart Data and Smart Cities tenutosi a DELFT (NL) nel October 4–5, 2018) [10.5194/isprs-annals-IV-4-W7-3-2018].

*Availability:*
This version is available at: 11583/2714742 since: 2018-10-05T10:48:31Z

*Publisher:*
Copernicus

*Published*
DOI:10.5194/isprs-annals-IV-4-W7-3-2018

*Terms of use:*

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

*Publisher copyright*

(Article begins on next page)

12 October 2024

# SHORT TERM URBAN TRAFFIC FORECASTING USING DEEP LEARNING

G. Albertengo[1], W. Hassan[1,*]

[1] Dept. of Electronics and Telecommunications, Politecnico di Torino,
Corso Duca degli Abruzzi, 24, 10129 Turin, Italy - (guido.albertengo, waqar.hassan)@polito.it

**KEY WORDS:** Deep Learning, traffic flow prediction, urban traffic forecasting.

**ABSTRACT:**

In today's world, the number of vehicles is increasing rapidly in developing countries and China and remains stable in all other countries, while road infrastructure mostly remains unchanged, causing congestion problems in many cities. Urban Traffic Control systems can be helpful in counteracting congestion if they receive accurate information on traffic flow. So far, these data are collected by sensors on roads, such as Inductive Loops, which are rather expensive to install and maintain. A less expensive approach could be to use a limited number of sensors combined with Artificial Intelligence to forecast the intensity of traffic at any point in a city. In this paper, we propose a simple yet accurate short-term urban traffic forecasting solution applying supervised window-based regression analysis using Deep Learning algorithm. Experimental results show that is it possible to forecast the intensity of traffic with good accuracy just monitoring its intensity in the last few minutes. The most significant result, in our opinion, is that the machine can generate accurate predictions even with no knowledge of the current time, the day of the week or the type of the day (holiday, weekday, etc).

## 1. INTRODUCTION

Urban Traffic Control (UTC) system is an integral part of any smart city scenario. Efficient use of road resources and infrastructure became essential due to the ever-growing number of road vehicles. To achieve traffic flow control and coordination, UTC systems monitor, distribute and control the traffic flows (Wood and K, 1993). In summary, a UTC system consists of a model of the physical infrastructure, sensors, a single or multiple controllers and actuators. The most common type of road-traffic sensor is the Inductive Loop (IL) detector which is usually buried under the road surface and identifies the passage of a vehicle through the changes in its inductance. The actuator (such as a traffic light) controls the flow of traffic according to the instructions of the controller. The controller constantly monitors and forecasts the traffic status and optimises the control strategy according to flow efficiency and/or environmental criteria. Adaptive Traffic Control System (ATCS) is a traffic light control program which provides fully responsive traffic signal control based on real-time traffic conditions (Skehan, 1996).

Without traffic forecasting, UTC systems can only rely on the current situation of traffic, estimated by road sensors, which is not sufficient for planning and optimization (Williams and Hoel, 2003). The aim of traffic flow prediction is to estimate the number of vehicles per unit time at a given location point or road segment (Zhang et al., 2011). It allows the implementation of several ITS solutions, such as ATCS, traffic management systems, advanced public transportation systems and logistics management. Forecasting is achieved using multi-sourced historical and real-time data processed by multiple types of forecasting and prediction models (Lv et al., 2015).

Machine learning is seeing more advancements and application than ever. Ever since the revival of Deep Learning (Krizhevsky et al., 2017), it is finding more and more applications in self-driving cars, robotics, image/object recognition, voice recognition, healthcare, cancer diagnosis, earthquake prediction and weather forecasting. Deep Learning is a type of machine learning which uses multiple-layer architectures to extract inherent features and structure in data from lowest to highest level. According to the Gartner hype cycle for emerging technologies of 2017, machine learning is at its peak of inflated expectations and is expected to reach a plateau within 2 to 5 years (Walker, 2017).

*Corresponding author

### 1.1 Current proposals

Traffic prediction approaches have been extensively researched in literature and generally can be grouped into three basic categories, i.e. *parametric*, *non-parametric* and *simulation-based* models. Parametric models include time-series models (Ghosh et al., 2007), Kalman filtering models (Okutani and Stephanedes, 1984), etc. Non-parametric models include Support Vector Regression (SVR) methods (Castro-Neto et al., 2009), Artificial Neural Networks (ANNs) (Smith and Demetsky, 1994), etc. Simulation based approaches use traffic simulation tools to predict traffic flow (Qiao et al., 2001).

Reviews of short-term traffic forecasting indicate that models are becoming more and more data-driven rather than analytical, due to an increase of computational intelligence (Vlahogianni et al., 2014). Researchers have been proposing traffic flow prediction models as early as in the seventies. For example, the AutoRegressive Integrated Moving Average (ARIMA) model and its derivations have been used to predict short-term flows of traffic for over 3 decades (Box and Pierce, 1970). Although, now the focus has moved from classical statistical models to neural-network based models (Vlahogianni and Karlaftis, 2011). Non-parametric models have also been extensively researched due to the non-linear nature of traffic flow. Apart from three basic categories of prediction models, hybrid models, which combine one or more techniques have also been proposed (Hong et al., 2011). A Stacked AutoEncoder (SAE) based model was proposed for 15 to 60 min traffic flow prediction (Lv et al., 2015). This model generates predictions with reasonable accuracy but is computationally complex due to the need for higher number of hidden layers and hidden units in the learner.

In summary, many traffic prediction models have been proposed in the literature to provide real-time traffic flow information in ITS, UTC and smart cities. Extensive comparison and review studies suggest that there is no technique that clearly outperforms other methods in all situations (van Hinsbergen and Sanders, 2007, Vlahogianni et al., 2014, Bolshinsky and Freidman, 2012).

### 1.2 Our contribution

In this paper, we propose a short-term urban traffic forecasting solution applying supervised window-based regression analysis using Deep Learning. We experimented with multiple configurations and prediction schemes to optimise the learning and prediction process. The resulting model is much simpler than other Deep Learning based traffic models proposed in the literature, yet it still outperforms them.

The rest of this paper is organized as follows: Section 2 introduces and analyses the traffic flow dataset used for this study; Section 3 extensively presents the Deep Learning prediction work-flow, pre-processing steps and schemes and hyper-parameter optimization; Section 3.5 outlines the error measures used to evaluate the performance of models; Section 4 discusses the experimental results and compares them to other state-of-the-art solutions. Finally, concluding remarks are presented in Section 5.

## 2. THE DATASET

The dataset used to train the AI machine is publicly available at the Uniform Resource Locator (URL) http://opendata.5t.torino.it/get_fdt in eXtensible Markup Language (XML) format (5T srl, 2011). It provides information about the number and average speed of the vehicles travelling in the urban area of Turin, detected by the means of ILs or aerial sensors. The dataset contains a single table with the following columns:

- **start_time**: Timestamp for the start of the period to which the data are referred
- **end_time**: Timestamp for the end of the period to which the data are referred
- **period**: Aggregation period [min]
- **lat**: Latitude in the WGS84 system of the measuring station
- **lng**: Longitude in the WGS84 system of the measuring station
- **Road_name**: Name of the road where the measuring station is located
- **Road_LCD**: Traffic Message Channel (TMC) code of the road on which the measuring station is present
- **lcd1**: Code of the initial node or TMC code of the initial location of the source arc
- **direction**: TMC direction (positive or negative)
- **accuracy**: Accuracy of the measurement [percentage]
- **offset**: Distance from the start of the arc along the direction of travel [m]
- **flow**: Vehicular flow [vehicle/hour]
- **speed**: Average speed [km/h]

The table contains roughly 130 data sources that are updated every 5 minutes. In the analysis that follows, we will only consider the data sources that are inside the boundaries of Metropolitan City of Turin (some 126 observation points). This data is collected by Urban Traffic OPtimization by Integrated Automation (UTOPIA) project (Mauro and Taranto, 1990). UTOPIA is a hierarchical decentralized traffic light control system that has been implemented and tested in a large area within the city of Turin.

### 2.1 Dataset analysis

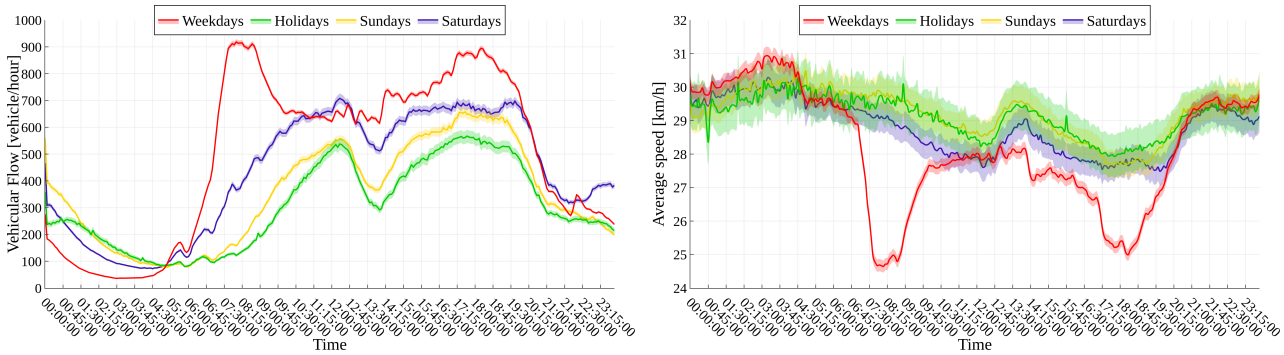
Figure 1. Geographical location of all data sources (red dots) in the urban area of Turin

Figure 1 shows the map of all data sources in the urban region of Turin. The data sources are well spread out over major city roads and intersections. For the sake of this analysis and tests, data was collected from 01-Oct-2017 till 28-Feb-2018. During this period there were 103 weekdays, 20 Saturdays, 21 Sundays and 7 holidays.

The analysis of the dataset shows that the data can be categorized into four types of days (Fox and Clark, 1998):

1. Weekdays (from Monday till Friday)
2. Saturdays
3. Sundays
4. Holidays (national and local holidays)

The average vehicular flow in the city changes according to the categories of days, as seen in the Figure 2(a). As expected, weekdays show the highest flow while holidays experience the lowest flow. During weekdays, the two highest peaks are from 07:40 until 8:50 and from 17:10 until 18:35. These peaks are due to commuters travelling between home and workplace and back. Saturdays and Sundays experience much less traffic than weekdays and the highest peaks are between 12:05 and 12:50. The average speed in the city is shown in Figure 2(b). Saturdays, Sundays and holidays follow a very similar trend and the average stays at some 29 km/h. On weekdays traffic slows down during peak hours (*i.e.* from 07:20 till 09:55 and from 16:55 till 19:25). This is due to the high vehicular flow during those periods, which generates congestion.




(a) Average vehicular flow

(b) Average vehicular speed

Figure 2. Vehicular flows in the city of Turin categorized by type of day (bands represent 95% confidence interval)

## 3. METHODOLOGY

To predict the intensity of traffic, we used supervised window-based regression analysis with Deep Learning. *Regression analysis* is a set of statistical processes for estimating the relationships among variables. *Supervised learning* is the machine learning task of inferring a function from labelled training data. *Deep Learning* is based on a multi-layer feed-forward artificial neural network that is trained with stochastic gradient descent using back-propagation. A *feed-forward neural network* is an artificial neural network wherein connections between the units do not form a cycle.

The process of machine learning usually involves gathering data, preparing data, selecting a model, training the model, evaluating its performance, tuning model parameters and finally generating predictions. Figure 3 shows prediction work-flow for forecasting this time series. Firstly, the labelled training dataset is read from the database. Then it passes through a few pre-processing steps depending on the type of test being performed. An explanation related to the different pre-processing steps can be found in Section 3.1. The labelled training dataset is passed to the Deep Learning machine which creates a model of the data. The unlabelled testing dataset is read and passed through the same pre-processing steps to evaluate the performance of the generated model. Once the testing dataset is labelled, it is evaluated using measures described in Section 3.5.

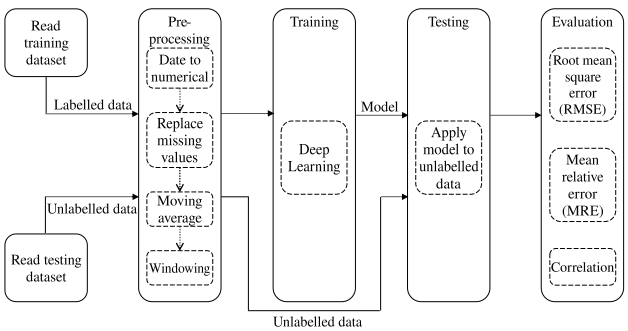



Figure 3. Prediction work-flow for supervised Deep Learning based regression for windowed time series forecasting

For the sake of conducting tests, we randomly selected multiple intersections across the city as data sources. We ran each test on all of them and results were very consistent across all intersections. In the following, as an example, the intersection between *Corso Giovanni Agnelli* (CGA) and *Via Filadelfia* (VF) is chosen as the data source. At this intersection, two data sources are available in the open data, specifically for northbound traffic (towards city centre) and southbound traffic (away from city centre). We selected the data source with northbound traffic for the discussion that follows.

### 3.1 Pre-processing

To further enhance and enrich the collected data, some pre-processing operations are performed. Their description is provided below.

**3.1.1 Date to numerical** converts a text-based timestamp into an integer number indicating the minutes from the origin of the dataset. For example, in case of weekdays and weeks dataset, 00:00 Monday is set to number 0 and 00:05 Monday to 5 and so on. For weekends dataset, 0 is set to Saturday midnight. This allows the machine to treat the string-based timestamp attribute as an index.

**3.1.2 Replace missing values** fills in gaps in a series due to missing data. A simple linear interpolation is applied to the data if one value is missing in the series. If more than one value is missing, the missing portion is replaced with the minimum value of the series.

**3.1.3 Moving average** is commonly used with time series data to smooth out short-term fluctuations and highlight long-term trends or cycles. In other words, moving average works like a low-pass filter. A triangular weighted window function is applied to the data with a window width of five ($5 \times 5\ mins = 25\ mins$) and the result of the weighted average is inserted at the end of the window.

**3.1.4 Time series windowing** is a very popular pre-processing step used in time series analysis and prediction. It takes any time series data and transforms it into a cross-sectional format. It essentially converts time values into cross-sectional attributes on which any predictive modelling algorithm can be used to predict future values. In other words, it transforms the given example set containing series data into a new example set containing single valued examples. For this purpose, windows with a specified *window size* (the width of the used window) and *step size* (the distance between the first values) are moved across the series and the attribute value lying *horizon* (the distance between the last window value and the value to predict) values after the window end is used as the label which should be predicted.

As an example, in case of the vehicular flow time series of one data source, a window size of six, a step size of one and a horizon of one would mean:

- The window considers the vehicular flow of last 30 mins;
- The window is moved 5 mins per step;
- The window is used to predict the vehicular flow of the next 5 mins.

Figure 4 shows a visual representation of the same example. The solid line boxes represent the input values while the dotted box represents the future value to be predicted.

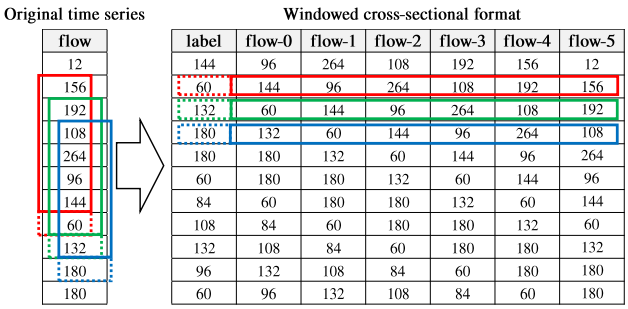



Figure 4. An example of original time series to windowed cross-sectional format conversion

**3.1.5 Shuffled order** pre-processing step shuffles the order or sequence of a windowed data table in a random fashion. The purpose of adding a shuffled order pre-processor is to further verify that the learner can correctly predict the traffic flow without timestamp even when the order of samples is shuffled (after windowing). Figure 5 shows a sample windowed data table randomly shuffled.

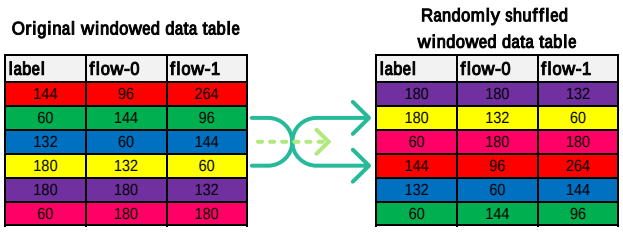



Figure 5. An example of random shuffling of windowed data table

### 3.2 Pre-processing schemes

To investigate the best approach for traffic prediction, we experimented with six different combinations of pre-processing schemes:

A. *Standard*: This scheme involves no special pre-processing other than converting the date to a number, replacing missing values and windowing.

B. *Moving average*: This scheme uses data passed through the date to a number, replacing missing values, moving average and windowing blocks.

C. *Moving average without timestamp input*: The dataset for this test does not contain any attribute related to time. The pre-processing steps include replacing missing values, moving average and windowing.

D. *Moving average without timestamp input and shuffled order*: The dataset for this test does not contain any attribute related to time. At the testing stage, the test dataset is shuffled to remove any chronology associated with it. The pre-processing steps include replacing missing values, moving average and windowing.

E. *No timestamp input*: The dataset for this test does not contain any attribute related to time. The pre-processing steps include replacing missing values and windowing.

F. *No timestamp input with shuffled order*: The dataset for this test does not contain any attribute related to time. At the testing stage, the test dataset is shuffled to remove any chronology associated with it. The pre-processing steps include replacing missing values and windowing.

Figure 6 shows a comparison between all pre-processing scheme pipelines. Note that the alphabets on the left of the figure correspond with items in the list.

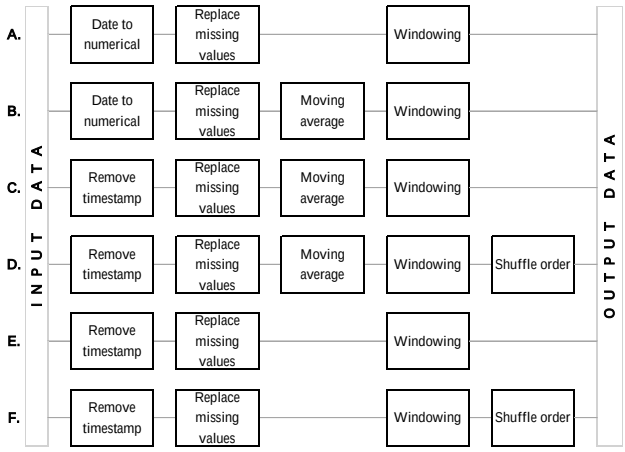



Figure 6. A comparison of all pre-processing scheme pipelines

### 3.3 Hyper-parameter optimization

A very important step involved in machine learning is tuning the model and the pre-processing parameters. *Hyper-parameter optimization* is the process of choosing a set of optimal hyper-parameters for a learning algorithm. A *hyper-parameter* is a parameter whose value is set before the learning process begins. By contrast, the values of other parameters are derived via training. We used an exhaustive grid search based hyper-parameter optimization to optimize the window size of windowing pre-processor and the number of epochs and learning rate of the Deep Learning machine. The loss function is based on Mean Relative Error (MRE). The grid search configuration for all hyper-parameters is shown in Table 1. This results in 726 total combinations of hyper-parameters per pre-processing scheme. The best combination of hyper-parameters is chosen based on the minimum value of MRE. Please note that the number of hidden layers and the number of neurons per hidden layer are fixed to two and 50, respectively.

### 3.4 Software configuration

For the experimentation, we used RapidMiner version 7.6 (RapidMiner, 2017). RapidMiner is a data science software platform that provides an integrated environment

| Hyper-parameter | Grid range | | | |
|---|---|---|---|---|
| | Min | Max | Steps | Scale |
| Window size | 2 | 12 | 10 | Linear |
| Epochs | 10 | 200 | 10 | Linear |
| Learning rate | 0.0 | 1.0 | 5 | Linear |

Table 1. Grid search based hyper-parameter optimization configuration

for data preparation, machine learning, Deep Learning, text mining, and predictive analytics. The specific implementation of Deep Learning algorithm used in these tests is included in $H_2O$ 3.8.2.6 (H2O.ai, 2016). *$H_2O$* is an open source in-memory platform for distributed and scalable machine learning.

### 3.5 Error measures

The error measures used to evaluate the accuracy of the predictions are listed below.

- *Root Mean Square Error (RMSE)* is the averaged root-mean-squared error such that:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\hat{f}_i - f_i)^2} \quad (1)$$

  where $\hat{f}$ is the predicted value, $f$ is the true value and $N$ is the total number of readings.

- *Mean Relative Error (MRE)* is the averaged absolute deviation of the prediction from the actual value divided by the actual value such that:

$$MRE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{\hat{f}_i - f_i}{f_i}\right| \quad (2)$$

  where $\hat{f}$ is the predicted value, $f$ is the true value and $N$ is the total number of readings.

- *Correlation* is the averaged correlation coefficient between the label and prediction attributes.

## 4. RESULTS

We present here the results for six different pre-processing schemes applied to three different sets of data. Table 3 shows the summary of all test results for all schemes. The results presented in the table are the best results after hyper-parameter optimization for each pre-processing scheme. In general, the machines can follow the general trend of vehicular flow and the results are quite promising.

The moving average pre-processor improves the accuracy of the predictions significantly. On average for all datasets, RMSE, MRE and Correlation for all schemes without moving average are 120.82, 27.69% and 93.51% respectively. The same values for schemes with moving average improve as much as 36.29, 8.15% and 99.38%. This constitutes up to 70% improvement in RMSE and MRE and 6% improvement in correlation due to moving average. Figure 7(a) and Figure 7(b) shows the comparison between predicted and true value of vehicular flow for the same dataset without and with moving average respectively.

The machine can generate an accurate model of the data with and without timestamp as an input. On average for all datasets, it shows a degradation of only 0.9% in MRE without timestamp as an input. This implies that the machine can model traffic without being aware of the time domain at all. It is possible to generate a very high accuracy prediction just by observing a few samples of vehicular flow in the past. Also, by removing time as an input attribute to the learner, simplifies the complexity of the learner, resulting in faster training and predictions.

Figure 9 shows the comparison between the predicted and true value of vehicular flow for the same dataset with moving average and without timestamp as an input. In this case, a single learner was trained to predict the entire week's vehicular flow. Please note that the x-axis of the figure was just added for illustration purposes since the actual dataset does not contain any time frame reference. The best overall results are achieved by pre-processing schemes B and C.

Our assumption is that the learner can correctly predict the flow of traffic without timestamp or index or any information about the order or sequence of a prediction. Results show that this assumption is true since on average for all datasets, with and without shuffled order schemes show negligible difference in error measures (4 RMSE, 0.4% MRE and 0.18% correlation). This proves that the learner predicts every prediction individually and does not rely on previous or future predictions.

As far as the window size is concerned, results show that it does not have a major impact on the performance of the learner. The best three combinations of hyper-parameters show negligible difference in error measures with a major variation in the value of window size. Overall, in our experience, a window size of 10 to 30 minutes is reasonable for generating high accuracy predictions.

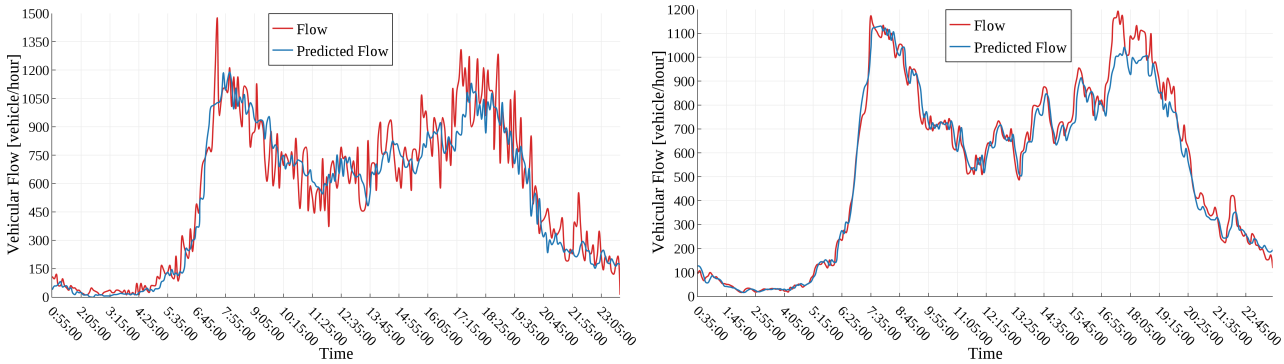
To further demonstrate the robustness of our approach, we performed a cross-examination among different datasets. We trained a machine using one type of dataset (such as two weekdays) and used this machine to predict the traffic flow of a different dataset (such as a weekend). Table 2 shows a summary of best results achieved after this cross-examination. Please note that all tests are performed using pre-processing scheme C (moving average without timestamp input). Results suggest that a machine trained with an entire week of traffic flow is the most suited to predict traffic flow on any day of the week. This hypothesis is confirmed by the machine trained using a set of data covering two full weeks instead of two weekdays or two weekends shows an improvement of on average 12.88 in RMSE, 1.6% in MRE and 0.23% in correlation.

| Training data | Testing data | RMSE | MRE | Correlation |
|---|---|---|---|---|
| 2 weekdays | 1 weekend | 36.41 | 8.49% | 98.90% |
| | 1 week | 43.30 | 9.03% | 99.20% |
| 2 weekends | 1 weekday | 61.97 | 10.57% | 98.90% |
| | 1 week | 64.61 | 10.65% | 98.70% |
| 2 weeks | 1 weekday | 41.78 | 7.99% | 99.30% |
| | 1 weekend | 35.60 | 8.19% | 99.00% |

Table 2. Summary of vehicular flow prediction errors with cross-examination among different datasets using Deep Learning

### 4.1 Comparison with state of the art

In comparison, the best results achieved by Deep Learning Architecture (DLA) based forecasting method by (Huang et al., 2014)




(a) using pre-processing scheme A (No special pre-processing)

(b) using pre-processing scheme B (Moving average)

Figure 7. Predicted traffic flow vs actual traffic flow for CGA intersection VF northbound dataset consisting of a weekday

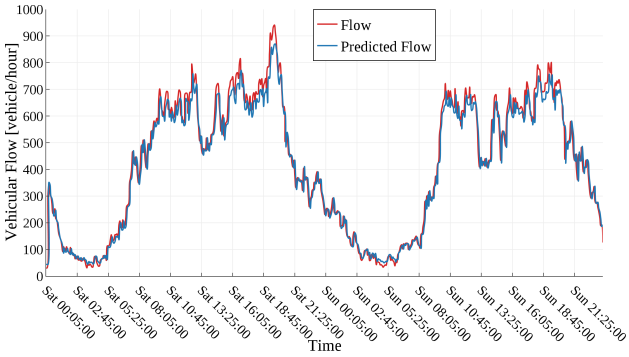



Figure 8. Predicted traffic flow vs actual traffic flow for CGA intersection VF northbound dataset consisting of weekends (Saturday and Sunday) using pre-processing scheme B (Moving average)

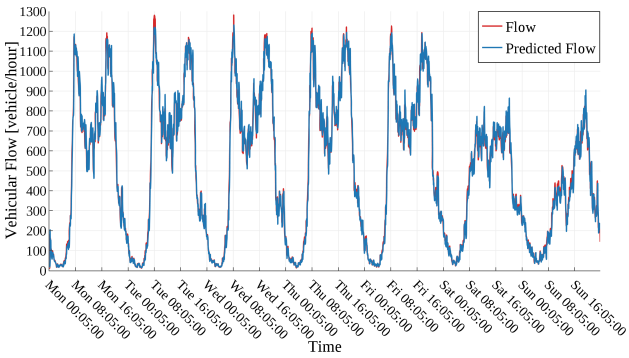



Figure 9. Predicted traffic flow vs actual traffic flow for CGA intersection VF northbound dataset consisting of an entire week using pre-processing scheme C (Moving average without timestamp input)

are around 90% Mean Accuracy ($MA = 1 - MRE$) using three hidden layers with 128 hidden units each and 40 epochs with data aggregated using a window size of 60 mins. Authors claim that these results are better than those of the ARIMA model (Box and Pierce, 1970), the Bayesian model (Sun et al., 2006), the SVR model (Castro-Neto et al., 2009), the LWL model (Shuai et al., 2008), the multivariate non-parametric regression model (Clark, 2003), the NN model (Smith and Demetsky, 1994), and the NN-S model (Chan et al., 2012). Our solution outperforms these results with MA of 94.2% while being much less complex (2 hidden layers with 50 hidden units each with data aggregated using a window size of 25). The best results achieved by SAE based Deep Learning forecasting method by (Lv et al., 2015) are around 6.48% MRE using 3 hidden layers with 400 hidden units each with data aggregated using a window size of 15 mins. According to authors, these results are better than those achieved by the BackPropagation Neural Network (BPNN), the Random Walk (RW), the Support Vector Machine (SVM) and the Radial Basis Function Neural Network (RBFNN). Again, our solution outperforms these results in terms of higher accuracy and lower complexity (MRE of 5.8%; two hidden layers with 50 hidden units each).

## 5. CONCLUSION

In this paper, we proposed a system for forecasting urban traffic over a short time period using Deep Learning. The prediction is accurate while being simple in terms of complexity. We also proposed and analysed the effects of multiple pre-processing schemes to improve the accuracy of forecasting. From experiments on a real traffic flow dataset from the City of Turin, we showed that our Deep Learning machine performs better than state of the art DLAs. Results show that our solution outperforms other DLAs with nearly 4% accuracy improvements while being much simpler in terms of complexity (hidden layers and hidden neurons). Furthermore, we presented that the most effective way to pre-process data is to use a simple moving average without timestamp as an input. This means that the machine can generate a prediction with only the traffic flow of past few minutes without any knowledge of the time. Moreover, to establish the robustness of our approach, we cross-examined our architecture with different datasets. Results show that a single Deep Learning machine with only two hidden layers of 50 units each trained with traffic flow data of a week can predict the flow on any day of the week as well as holidays with remarkable accuracy.

The results are not only accurate, but they have significant applications. By accurately determining the intensity of traffic in the near future, ATCS can optimize the traffic light control program to minimize congestion. Moreover, when applied at multiple intersections, a forecasted traffic congestion can be distributed over multiple intersections to lower the overall impact. Furthermore, when using a multi-plan based traffic light control (*e.g.* low, medium, high ...) for an urban area, accurate predictions can be used as an indication to swap the current plan with a more appropriate plan. Although our focus was on the forecasting of urban traffic we believe that the methodology we have devised for

| Training data | Testing data | Moving average | No timestamp input | Shuffled order | Window size [min] | RMSE | MRE | Correlation |
|---|---|---|---|---|---|---|---|---|
| 2 weekdays | 1 weekday | | | | 55 | 134.50 | 24.50% | 93.60% |
| | | ✓ | | | 45 | 47.67 | 7.36% | 99.33% |
| | | ✓ | ✓ | | 60 | 40.41 | 11.00% | 99.53% |
| | | ✓ | ✓ | ✓ | 15 | 45.73 | 8.92% | 99.24% |
| | | | ✓ | | 20 | 126.51 | 28.52% | 94.31% |
| | | | ✓ | ✓ | 60 | 135.29 | 28.85% | 94.27% |
| 2 weekends | 1 weekend | | | | 30 | 122.93 | 27.22% | 91.81% |
| | | ✓ | | | 30 | 26.69 | 9.05% | 99.42% |
| | | ✓ | ✓ | | 25 | 24.22 | 5.39% | 99.53% |
| | | ✓ | ✓ | ✓ | 35 | 35.01 | 8.23% | 98.99% |
| | | | ✓ | | 15 | 100.11 | 27.27% | 92.60% |
| | | | ✓ | ✓ | 20 | 97.66 | 27.75% | 92.71% |
| 2 weeks | 1 week | | | | 40 | 129.99 | 29.03% | 93.54% |
| | | ✓ | | | 40 | 21.05 | 5.80% | 99.84% |
| | | ✓ | ✓ | | 25 | 43.50 | 8.79% | 99.27% |
| | | ✓ | ✓ | ✓ | 30 | 42.34 | 8.82% | 99.27% |
| | | | ✓ | | 35 | 118.40 | 27.60% | 94.56% |
| | | | ✓ | ✓ | 60 | 121.98 | 28.47% | 94.23% |

Table 3. Summary of vehicular flow prediction errors on different datasets using Deep Learning with different pre-processing schemes

pre-processing and prediction of traffic intensity time series can be easily adapted to other time series with minor modifications as per the domain of prediction.

## ACKNOWLEDGEMENTS

The data contained in the 5T Turin open data datasets and the information presented are the property of the City of Turin and are made available with IODL v2.0 - Italian Open Data License (http://www.dati.gov.it/iodl/2.0/).

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper.

## REFERENCES

5T srl, 2011. City of Turin - Traffic flows (Real time). 5T srl http://opendata.5t.torino.it/get_fdt (04 Feb 2011).

Bolshinsky, E. and Freidman, R., 2012. Traffic Flow Forecast Survey. Technical report, Computer Science Department, Technion.

Box, G. E. P. and Pierce, D. A., 1970. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association* 65(332), pp. 1509–1526.

Castro-Neto, M., Jeong, Y. S., Jeong, M. K. and Han, L. D., 2009. Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions. *Expert Systems with Applications* 36, pp. 6164–6173.

Chan, K. Y., Dillon, T. S., Singh, J. and Chang, E., 2012. Neural-network-based models for short-term traffic flow forecasting using a hybrid exponential smoothing and levenberg-marquardt algorithm. *IEEE Transactions on Intelligent Transportation Systems* 13(2), pp. 644–654.

Clark, S., 2003. Traffic Prediction Using Multivariate Nonparametric Regression. *Journal of Transportation Engineering* 129(2), pp. 161–168.

Fox, K. and Clark, S., 1998. Evaluating the benefits of a responsive UTC system using microsimulation. *Institute for Transport Studies, University of Leeds, Leeds, UK.(Working Paper).*

Ghosh, B., Basu, B., Asce, M. and Mahony, M. O., 2007. Bayesian Time-Series Model for Short-Term Traffic Flow Forecasting. *Journal of transportation engineering* 133(March), pp. 180–189.

H2O.ai, 2016. H2O Open-source Library Software, Version 3.8.2.6. H2O.ai https://www.h2o.ai/h2o/ (24 May 2016).

Hong, W. C., Dong, Y., Zheng, F. and Wei, S. Y., 2011. Hybrid evolutionary algorithms in a SVR traffic flow forecasting model. *Applied Mathematics and Computation* 217(15), pp. 6733–6747.

Huang, W., Song, G., Hong, H. and Xie, K., 2014. Deep architecture for traffic flow prediction: Deep belief networks with multitask learning. *IEEE Transactions on Intelligent Transportation Systems* 15(5), pp. 2191–2201.

Krizhevsky, A., Sutskever, I. and Hinton, G. E., 2017. ImageNet classification with deep convolutional neural networks. In: F. Pereira, C. J. C. Burges, L. Bottou and K. Q. Weinberger (eds), *Communications of the ACM*, Vol. 60, Curran Associates, Inc., pp. 84–90.

Lv, Y., Duan, Y., Kang, W., Li, Z. and Wang, F.-Y., 2015. Traffic Flow Prediction with Big Data: A Deep Learning Approach. *IEEE Transactions on Intelligent Transportation Systems* 16(2), pp. 865–873.

Mauro, V. and Taranto, C. D., 1990. UTOPIA. In: J.-P. PERRIN (ed.), *Control, Computers, Communications in Transportation*, IFAC Symposia Series, Pergamon, Oxford, pp. 245–252.

Okutani, I. and Stephanedes, Y. J., 1984. Dynamic prediction of traffic volume through Kalman filtering theory. *Transportation Research Part B* 18(1), pp. 1–11.

Qiao, F., Yang, H. and Lam, W. H., 2001. Intelligent simulation and prediction of traffic flow dispersion. *Transportation Research Part B: Methodological* 35(9), pp. 843–863.

RapidMiner, 2017. RapidMiner Studio Software, Version 7.6.0. RapidMiner https://rapidminer.com/products/studio/ (17 Aug 2017).

Shuai, M., Xie, K., Pu, W., Song, G. and Ma, X., 2008. An online approach based on locally weighted learning for short-term traffic flow prediction. In: *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems - GIS '08*, ACM, p. 1.

Skehan, S., 1996. Adaptive Traffic Control System. In: *Compendium of Technical Papers for the 66th ITE Annual Meeting*, Institute of Transportation Engineers, pp. 203–207.

Smith, B. L. and Demetsky, M. J., 1994. Short-term traffic flow prediction models-a comparison of neural network and nonparametric regression approaches. In: *Systems, Man, and Cybernetics, 1994.'Humans, Information and Technology'., 1994 IEEE International Conference on*, Vol. 2, IEEE, pp. 1706–1709.

Sun, S., Zhang, C. and Yu, G., 2006. A Bayesian Network Approach to Traffic Flow Forecasting. *IEEE Transactions on Intelligent Tra* 7(1), pp. 124–132.

van Hinsbergen, J. W. C. and Sanders, F. M., 2007. Short Term Traffic Prediction Models.

Vlahogianni, E. and Karlaftis, M., 2011. Temporal aggregation in traffic data: implications for statistical characteristics and model choice. *Transportation Letters* 3(1), pp. 37–49.

Vlahogianni, E. I., Karlaftis, M. G. and Golias, J. C., 2014. Short-term traffic forecasting: Where we are and where we're going. *Transportation Research Part C: Emerging Technologies* 43, pp. 3–19.

Walker, M. J., 2017. Hype Cycle for Emerging Technologies, 2017. Insight report, Gartner, Inc.

Williams, B. M. and Hoel, L. A., 2003. Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results. *Journal of Transportation Engineering* 129(6), pp. 664–672.

Wood and K, 1993. Urban traffic control, systems review. *Urban traffic control, systems review* 1(1), pp. 1–53.

Zhang, J., Wang, F.-Y., Wang, K., Lin, W.-H., Xu, X. and Chen, C., 2011. Data-Driven Intelligent Transportation Systems: A Survey. *IEEE Transactions on Intelligent Transportation Systems* 12(4), pp. 1624–1639.

*Revised July 2018*