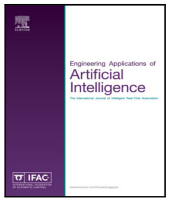
Engineering Applications of Artificial Intelligence 121 (2023) 106041

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Long-term traffic flow forecasting using a hybrid CNN-BiLSTM model ☆

Manuel Méndez, Mercedes G. Merayo, Manuel Núñez *

Design and Testing of Reliable Systems Research Group, Complutense University of Madrid, 28040, Madrid, Spain

ARTICLE INFO

Keywords:
Deep learning
Hybrid models
Traffic flow forecasting

ABSTRACT

The increase of road traffic in large cities during the last years has produced that long and short-term traffic flow forecasting is a critical need for the authorities. The availability of good traffic flow prediction methods is a must to make informed decisions concerning (punctual) traffic congestions. Previous work has shown that the accuracy of these methods decreases if we consider urban traffic and long-term predictions. In this paper we present a hybrid model, combining a Convolutional Neural Network and a Bidirectional Long–Short-Term Memory network, and apply it to long-term traffic flow prediction in urban routes. This model combines the capability of CNN to extract hidden valuable features from the input model and the capability of BiLSTM to understand the temporal context. In order to assess the usefulness of our model, we considered four streets of the city of Madrid with different characteristics and compared the results of our proposed model with the ones obtained by eight widely used baseline models. The results show that our hybrid model outperforms the baseline models with respect to three metrics commonly used in regression: mean absolute error, root mean squared error and accuracy.

1. Introduction

Urban sprawl implies a high growth of the traffic flow in big cities. Because of this, traffic congestion is one of the main problems of most big cities. Traffic flow forecasting is an essential tool for traffic management, specially, in the most congested and frequented roadways. The high presence of signalisation in cities makes urban traffic flow more unpredictable and random than traffic flow in freeways. In most cases, temporal trends (monthly, daily or hourly) are insufficient to optimally model traffic flow. Other factors such as meteorological conditions, traffic flow in nearby roadways and type of day (working day, public holiday or weekend) are also essential to model traffic flow.

Traffic flow forecasting can be classified in *short-term* and *long-term*. On the one hand, short-term models are able to predict traffic flow from a few minutes to one hour in advance, that is, in a very close future. On the other hand, long-term models are able to predict traffic flow from two hours to several days in advance. It seems reasonable that in short-term models, temporal trends have a very high influence in the traffic flow. However, in long-term models, the influence of temporal trends decreases and we have to look for other factors, such as the ones mentioned above, whose influence will be more significant.

By reviewing the existing literature on traffic flow prediction, we noticed that the majority of the approaches consider predictions in the short-term. In these approaches, high accuracy is usually achieved only when predicting the values a few minutes in advance. Unfortunately, this is not a very useful information to manage the traffic flow because it will be too late to hint drivers about bad conditions. In order to provide advanced trip plannings to users, we think that it is more important to develop models that can make accurate predictions in longer terms.

By focusing on the architectures used to forecast traffic flow in recent work, we can mention the frequent usage of Long–Short-Term Memory (LSTM) networks. These models can easily learn and understand the sequence information. However, they are adapted to deal with temporal correlations and they can only use the features given by the dataset. In order to improve its performance, some authors have suggested the hybridisation between LSTMs and Convolutional Neural Networks (CNNs). CNNs have the capability to filter out noise and extract new and valuables features from the input matrix. Thus, an LSTM will use the new information given by a CNN to identify temporal patterns and make the corresponding forecasts. Therefore, a hybrid CNN-LSTM model usually will outperform the accuracy of a single LSTM model.

Bidirectional LSTM (BiLSTM) networks are an extension of LSTM networks in which two LSTMs are applied: one is applied to the input data (forward) and the other one is applied to the reverse of the input data (backward). Experimental evidence (Siami-Namini et al., 2019) shows that in tasks related to time series, BiLSTM models offer better

☆ This work has been supported by the State Research Agency (AEI) of the Spanish Ministry of Science and Innovation under grant PID2021-122215NB-C31 (AwESOMe); the Comunidad de Madrid, Spain under grant S2018/TCS-4314 (FORTE-CM) co-funded by EIE Funds of the European Union.

* Corresponding author.
E-mail addresses: manumend@ucm.es (M. Méndez), mgmerayo@fdi.ucm.es (M.G. Merayo), mn@sip.ucm.es (M. Núñez).

https://doi.org/10.1016/j.engappai.2023.106041
Received 1 July 2022; Received in revised form 21 December 2022; Accepted 21 February 2023
Available online 1 March 2023

predictions than LSTMs, although LSTMs reach the equilibrium faster than BiLSTMs. As the granularity increases, the temporal dependencies are fewer. Because of this, it is needed to find new relationships between variables that do not necessarily have to be temporal. In this line, the application of convolutional operations to the input matrix before the application of the BiLSTM model is a promising approach that we will explore in this paper.

In this work we propose a novel hybrid CNN-BiLSTM architecture to forecast long-term traffic flow. The goal is to leverage the potential of a CNN block to extract complex characteristics from the input matrix and the capability of a BiLSTM block to find temporal dependencies between variables by understanding doubly (forward and backward) the context in each situation. In order to evaluate the performance of our CNN-BiLSTM model, we compare it with eight baseline models. In particular, we compare it with CNN-LSTM, in order to determine whether it is worth to use BiLSTM rather than LSTM, and with a BiLSTM, in order to determine whether it is worth to use a hybrid model or not.

As a case study, we will apply our model and we will compare it with the baselines in four main roadways of Madrid. We take data from the official website of Madrid City Council, being data originally extracted from sensors that are part of a network of permanent traffic stations.[1]

1.1. Contributions

The main contributions of the paper are summarised as follows:

1. We developed a novel architecture that can be used in time series forecasting. This architecture is based on the hybridisation of a CNN and a BiLSTM neural network.
2. We applied the new model to forecast the traffic flow in four stations of Madrid. The model showed a very good performance with respect to three metrics.
3. A side result of our experiments was the creation of a historical database on traffic flow measured in four stations of Madrid and a procedure to built such a database in other stations. The database currently contains flow traffic information (of the considered station), temporal information, nearest stations information and context information.
4. With the goal of optimising the model, we performed a careful fine-tuning of the considered hyperparameters.
5. In order to show the usefulness of our model, we compared it with several powerful baseline models typically used in time series forecasting.
6. The code of our proposed method and the datasets used in this paper are freely available at https://github.com/MMH1997/CNN-BiLSTM-network.

1.2. Outline

The rest of the paper is organised as follows. In Section 2, we briefly review related work. In Section 3 we present the main characteristics of the problem that we will confront and will describe the data used in it. In addition, we will explain the data collecting and data pre-processing phases. In Section 4, we describe the proposed hybrid model. In Section 5, we briefly present the main characteristics of each baseline model. In Section 6, we present our experiments and discuss the quality/accuracy of the proposed model, in particular, in comparison with baseline models. Finally, in Section 7, we give our conclusions and outline some directions for future work.

[1] https://datos.gob.es/es/catalogo/l01280796-aforos-de-trafico-en-la-ciudad-de-madrid-permanentes1.

2. Related work

Short-term traffic flow forecasting is a recurrent case study in order to show the quality of forecasting models. In fact, many machine learning models have been developed to tackle this issue. We can classify them in machine learning models and deep learning models. Machine learning models such as auto-regressive models (ARIMA or similar) (Lin and Huang, 2021; Chen et al., 2011), K-nearest neighbours (KNN) (Hong et al., 2015), linear regression (Li, 2020) or Support Vector Regression (SVR) (Soon et al., 2019) are usually applied. In recent years, deep learning models such as, long–short term memory networks (LSTM) (Chu et al., 2021; Kang and Zhang, 2020; Poonia and Jain, 2020), multi layer perceptron networks (Shubhangi and Pratibha, 2021), bidirectional LSTM (BiLSTM) networks (Abduljabbar et al., 2021) and radial basis function networks (RBF) (Zhu et al., 2014; Abdi et al., 2010) have also been used.

Work in the scope of long-term traffic flow forecasting is less common. First, statistical models are usually not used in recent work. We can mention the use of a two context-aware Random Forest model (Zarei et al., 2013) and the hybridisation of an ARIMA model and a support vector machine model (Wang et al., 2017). In contrast, artificial neural networks are commonly used. For example, a recurrent neural network (RNN) with a GPU-based implementation has been used to forecast long-term traffic flow in the cities of Odense and Beijing, obtaining a high accuracy (Belhadi et al., 2020). Some authors have used graph neural networks (GNN) to forecast traffic flow. Very recent work presents a spatial–temporal graph attention network that learns the dynamic graph structure and spatial–temporal dependency of the data (Kong et al., 2022). This model is evaluated in two public datasets collected in California. Also very recently, an overview of recent spatio-temporal GNN models for traffic forecasting has been presented (Bui Khac Hoai et al., 2022). However, recent approaches tend to use hybrid neural networks, combining at least two different models, and are able to obtain better accuracy. For example, the combination of an LSTM model and an encoder–decoder architecture using a hard attention mechanism (Wang et al., 2021) was developed and applied to data from Los Angeles. Following a similar line, a hybrid model which combines wavelet networks, LSTM and CNN has been used to forecast next day traffic flow in England Li et al. (2021) while a hybrid model, which combines a graph convolutional network and an LSTM, was proposed to forecast long-term traffic flow in New York (Peng et al., 2021). A CNN auto-encoder and a BiLSTM auto-encoder have been used to detect anomalies in traffic flow (Méndez et al., 2022a). Finally, it is worth to mention that multi-task learning has also been used in traffic flow forecasting (Zhang et al., 2021). The idea is to combine the data of different cities (in this case, Xi'an and Chengdu) to obtain a single model.

Now, we will focus on models similar to our proposal. First, it is worth pointing out that BiLSTM models are not commonly used to work with time series, although recent studies show that BiLSTM reaches higher accuracy than LSTM due to their capability to use input data twice for training (Siami-Namini et al., 2019). However, in other topics, mostly related to natural language processing, BiLSTMs have been successfully applied. For example, in mention extraction tasks (Lin et al., 2019) or to write support systems (Makarenkov et al., 2019). Moreover, a single CNN very rarely has been employed in time series problems because its nature is not adequate for it. However, specially in recent years, it has been often combined with classical time series models to extract hidden information and, thus, increase the performance of the model. For example, a hybrid model composed by an LSTM, a CNN and an auto-encoder has been applied to forecast electricity consumption in Brazil (Rick and Berton, 2022) while the combination of a DNN and an LSTM has been used to detect phishing URLs (Ozcan et al., 2023). Similarly, models which combine CNNs and BiLSTMs are not frequent in the literature and most of them are tailored to solve problems related to natural language processing

Table 1
Target stations and their auxiliary stations.

Target station: location (direction)	Aux. station 1: location (direction)	Aux. station 2: location (direction)
P/Castellana (N-S)	C/José Abascal (W-E)	A/ General Perón (E-W)
C/Arturo Soria (N-S)	C/Alcalá (W-E)	C/Costa Rica (W-E)
P/Sta María (S-N)	P/del Prado (N-S)	P/Infanta Isabel (E-W)
C/Gran Vía (W-E)	C/Hortaleza (S-N)	C/Alcalá (E-W)

such as automatic speech recognition (Passricha and Aggarwal, 2020), text classification (Li et al., 2018), sentiment analysis (Abu Kwaik et al., 2019; Rhanoui et al., 2019) and fake news detection (Balwant, 2019). Other environments where these hybrid models have been recently used are video summarising (Hussain et al., 2020), DNA-protein binding predicting (Zhang et al., 2020) and remaining useful life estimation (Song et al., 2021).

An interesting line of work, where hybrid models are also present, takes genetic programming as *base* model. These approaches have been use to forecast the stock market (Ari and Alagoz, 2023) and electricity power generation (Peiris et al., 2022). In addition, hybrid genetic programming and neural network models have been used to predict wind power (Zameer et al., 2017), forecast pollutants concentration (Ari and Alagoz, 2022) and suspended sediment concentration in rivers (Li et al., 2022).

To the best of our knowledge, a CNN-BiLSTM hybrid model has not been used to deal neither with time series nor with traffic flow forecasting.

3. Problem description, data collection and processing

In this section we briefly describe the problem that we confront and the data that we will use. In particular, we will sketch the process to transform data into the format used by our model.

3.1. Problem description

Our problem is to forecast long-term traffic flow in major roadways of big cities. Specifically, we will forecast the flow 12, 24, 48 and 72 h in advance. We will use data of the four last available timesteps. Let h be a given timestep and $h+k$ be the timestep in which we desire to make the forecasting: we will use nine predictor variables, corresponding to timesteps ranging from $h-3$ to h, to forecast the target variable in the $h+k$ timestep. In order to provide a solution to this problem, our final goal is to find a model which explains, with an error as small as possible, the behaviour of the target variable over time. Let $x_1, \ldots, x_9$ be the nine predictor variables (we will describe them in the next section), y be the target variable (we will also describe it in the next section) and $\epsilon \in \mathbb{R}_+$ be an error. Our model can be expressed by the following equation:

$$y(h+k) = f\begin{pmatrix} x_1(h-3), \ldots, x_1(h), \\ x_2(h-3), \ldots, x_2(h), \\ \ldots \\ x_9(h-3), \ldots, x_9(h) \end{pmatrix} + \epsilon \tag{1}$$

3.2. Data description

In this section we will identify the predictor variables and the target variable. We will also indicate the sources of the data that we used for our experiments.

The target variable is the hourly number of vehicles passing through a traffic station in a certain direction. In order to perform experiments that could validate the usefulness of our model, we selected four traffic stations in the city of Madrid, targeting streets with big traffic flows but having different characteristics. Three of them are located in core city areas and one of them is a main entry/exit roadway of the city. The stations are:

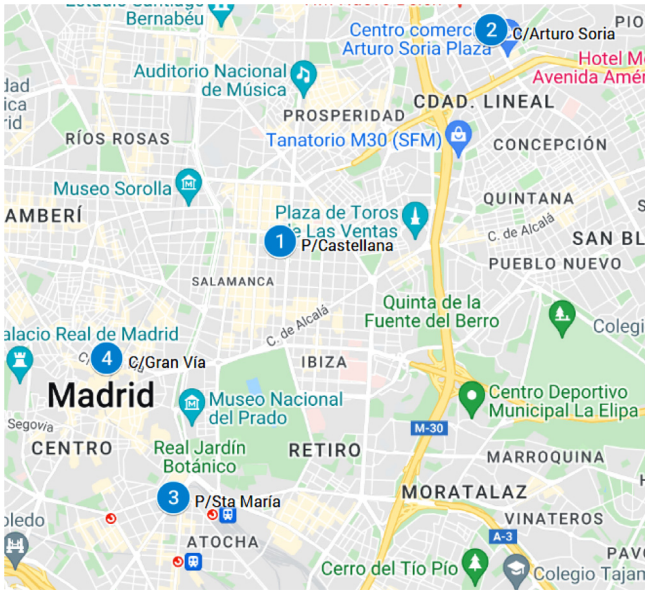


Fig. 1. Location of the target stations in a map of Madrid.

- *Paseo de la Castellana* (P/Castellana) taking data in the North–South direction. This is a major street that crosses Madrid from downtown to North.
- *Calle Arturo Soria* (C/Arturo Soria) taking data in the North–South direction. This is a long street that crosses Madrid from North to South in the East part of the city.
- *Paseo de Santa María de la Cabeza* (P/Sta María) taking data in the South–North direction. This is an important communication channel from downtown to the South of the city.
- *Calle Gran Vía* (C/Gran Vía) taking data in the West–East direction. This is a shopping and touristic centrally located street.

Fig. 1 shows the distribution of the four target stations in a map of Madrid.

The first group of predictor variables are: type of day (public holiday, working day or weekend), daily average temperature, daily minimum temperature, daily maximum temperature and daily rain. In addition, we also consider as predictor variables the hourly amount of vehicles in the target station and in two auxiliary stations (two different for each target station). We selected the auxiliary stations by taking into account the traffic intensity (we prefer stations in roadways with high traffic intensity) and the proximity to the target station. Table 1 shows the corresponding auxiliary stations for each target station. Finally, we also use the hour in the timestep as a predictor variable. Therefore, we have nine predictor variables. For each prediction, we collect input data from four different hours: for each prediction we use 36 predictor variables. Fig. 2 shows an extract of the initial dataset (before pre-processing) where we can see, for each hour value, the values corresponding to the predictor and target variables.

For our experiments, data have been extracted from the websites of different institutions. Meteorological data have been obtained from

	aux1(t-12)	aux2(t-12)	original(t-12)	tmean(t-12)	rainfall(t-12)	tmin(t-12)	tmax(t-12)	typeday(t-12)	hour(t)	obj(t)
0	799	1442.0	658.0	76.0	0.0	36.0	116.0	festivo	13	468
1	894	903.0	458.0	76.0	0.0	36.0	116.0	festivo	14	201
2	760	397.0	190.0	76.0	0.0	36.0	116.0	festivo	15	243
3	669	312.0	147.0	76.0	0.0	36.0	116.0	festivo	16	349
4	665	299.0	124.0	76.0	0.0	36.0	116.0	festivo	17	430

Fig. 2. Extract of initial data (Arturo Soria station).

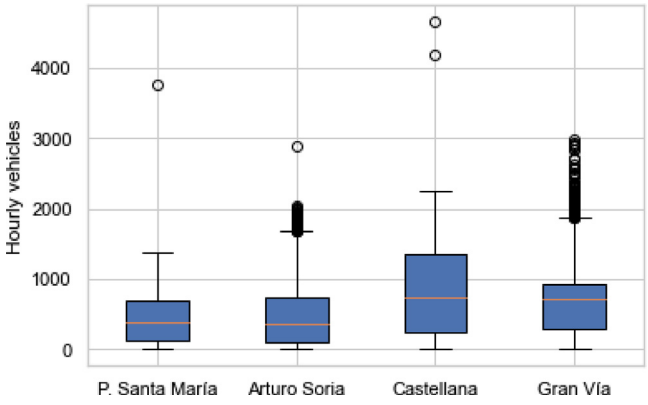


Fig. 3. Target data in each station (box-plot).

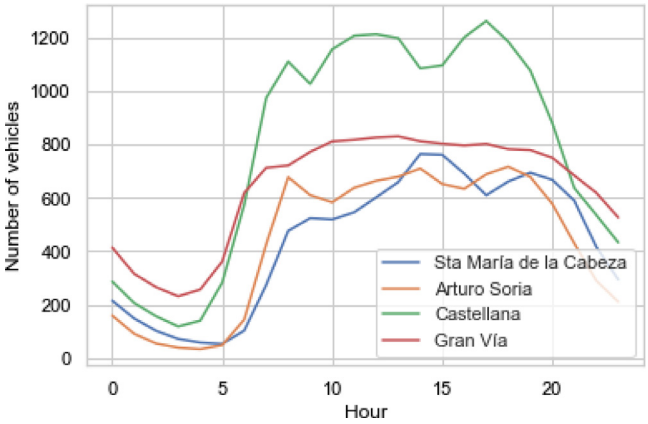


Fig. 4. Mean of vehicles by hour in each target station.

AEMET[2] and correspond to measures in El Retiro meteorological station, located in the centre of Madrid, while type of day and traffic data have been extracted from the Madrid City Hall website.[3]

3.3. Data analysis

Next, we will make a brief descriptive analysis of the hourly traffic flow in the target stations that we will consider in our experiments. In Fig. 3 we observe similar ranges in three of the stations. The range of the fourth one, P/Castellana, is wider. Moreover, we appreciate a very small number of atypical values which will be considered actual values and not caused by a human error, as we will discuss in the next section. In Fig. 4 we can observe that the hourly evolution of the traffic flow follows a similar pattern in all the stations (each of them in its corresponding range). Summarising, this evolution shows an increase on the number of vehicles from 5 a.m. until approximately 9/10 a.m. Then, there is a valley with small oscillations until approximately 6/7 p.m. At this time, a big decrease in the traffic flow begins until 5 a.m. of the next day. We note that the variation between day and night in C/Gran Vía is smaller than the one happening in the rest of stations (in relative terms). This implies that traffic flow in C/Gran Vía is significantly less dependent on temporal variables compared to the rest of stations.

3.4. Data pre-processing

Data pre-processing has not been an easy task. In this section, we will describe step by step the whole process performed so that users making use of our work can try and mimic these steps to forecast traffic flow in other cities.

- In the first step, we downloaded the different data-sets from the corresponding websites. Once all data-sets were downloaded, we unified the format of the date column (the junction column) in all data-sets. Then, we transformed daily data into hourly data and joined all data-sets.
- In the second step, we added non available (NA) data by using the mean between previous and next values (all NA data are numerical). We also categorised data which required it and scaled all data between 0 and 1.
- Next, we normalised all the variables (predictor variables and target variable) between 0 and 1 by using the well-known *MinMaxScaler* method. This step was applied in order to transform all the numerical variables to the same scale.
- Subsequently, we transformed data from a sequence to pairs of input (predictor variables) and output (target variable), that is, we transformed a time series problem into a supervised machine learning problem.
- Finally, we reshaped data from a flatten vector to a matrix with nine columns (one by predictor variable) and four rows (one by timestep). The reshape is needed because the proposed model requires data in matrix form as input.

It should be mentioned that we did not remove *outliers*. We manually analysed outlier data and concluded that, in all cases, they corresponded to a notorious event happening in Madrid and they were not due to a human error.

4. CNN-BiLSTM architecture

In this section, we will describe the proposed model: a hybrid CNN-BiLSTM. First, this architecture is built to leverage CNN advantages for filtering and removing noise data, obtaining substantial internal valuable information of the time series. The removal of the noise data is performed by applying dimension reduction. Therefore, non-relevant information, that is, the noise, is not included in the reduced matrix. The hidden internal valuable information is obtained by applying the

[2] AEMET is the state meteorological agency of Spain: https://opendata.aemet.es/centrodedescargas/productosAEMET?.

[3] https://datos.madrid.es/portal/site/egob/.

convolutional operation: the filters or kernel matrix pass through the input matrix, highlighting different characteristics depending on the kernel and, thus, obtaining hidden information. Second, this architecture is built to leverage the capability of a BiLSTM network to model and forecast both in short-term and long-term cases of temporal data sequences. The power of BiLSTM models lies in the training process. In this model, input data is trained twice, from left to right and from right to left, in parallel. Both context interpretations are combined in the output, providing more comprehensive information of the data context in this model than, for example, in a unidirectional LSTM model. We might say that the CNN acts as an encoder layer mechanism that learns new features from the input data sequences, while the BiLSTM acts as a decoder that learns the temporal relations in data stream and outputs the traffic flow prediction.

The proposed model has two differentiated blocks. The *CNN block* has a CNN layer, a pooling layer and a flatten operator. The CNN block output will be used as input by the *BiLSTM block*. This second block includes a BiLSTM layer, a dropout layer and a dense layer. Since convolutional and pooling layers were originally developed to extract features from an image, they only work with matrix data. Therefore, we will need to format input data to be in matrix form as we explained in Section 3.4. Next, we will describe in detail the CNN and BiLSTM blocks.

4.1. CNN block

As usual, convolutional and pooling layers aim to filter incoming data to extract important information from the matrix. Convolutional operations are performed by the convolutional layer between input data and smaller matrices called kernels or filters. Usually, several filters are considered and they can be seen as windows with a given height and width that slide a certain number of cells, usually called stride, across the input matrix by applying the convolution operation on each sub-region of the matrix that it intersects. This process obtains a convoluted matrix, which represents a singular feature of the original input matrix. Obviously, the application of several filters will carry out that more different convoluted features will be obtained. A model that uses several convoluted features will more accurately represent the input matrix than a model that only uses the original input matrix. Therefore, the performance of a given model will be improved by using more filters. The convolutional operation can be formally expressed as follows. Let H be the input matrix, I be the kernel matrix and m and n be the indexes of rows and columns of the result matrix (R). We have

$$\begin{aligned} R[m,n] &= (H \cdot I)[m,n] \\ &= \sum_j \sum_k I[j,k] \cdot H[m-j, n-k] \end{aligned} \tag{2}$$

The convolutional layer uses the $ReLU$ activation function. This is the most widely used activation function applied in CNN models. Its main advantage is that it does not activate every neuron at the same time because it transforms all negative values into 0. Due to this, $ReLU$ has a big computational efficiency. Actually, $ReLU$ is six times faster to train than other activation functions such as *tanh* (Chou et al., 2017). $ReLU$ is defined as follows:

$$f(x) = x^+ = \max(0, x) \tag{3}$$

The convolutional layer is followed by a max pooling layer. A pooling layer is a sub-sampling method that aims to reduce the dimension of the convolutional matrix by removing values but keeping the most important features detected by each filter. There are several types of pooling layers, but the most widely used, being the one used in our model, is the *max pooling layer*. Similarly to the previous description of filter, we use a window to traverse the matrix. We take the values of each patch of the matrix as input and return the maximum of each patch. This layer produces new matrices that can be considered as summarised versions of the convolutional matrices. The pooling operation also induces an increase of the model robustness. That is, small modifications in the input will not affect the output. Finally, the pooling layer is followed by a flatten operator, which unrolls the values at the last dimension in order to reshape the next layer input.

4.2. BiLSTM block

The second component of the model is a BiLSTM block composed by a BiLSTM network, a dropout layer and a dense layer.

In order to understand the performance and architecture of a BiLSTM network, we will first describe an (unidirectional) LSTM network. An LSTM network is a type of recurrent neural network (RNN). Traditional RNNs do not have a previously defined structure of layers. They use cyclic connections in their hidden layers that provide short-term memory to the model and, therefore, the capability of working with sequence data (as time series). However, traditional RNNs have the issue known as *long-term dependencies problem*: a gradual forgetfulness of past information as the number of neurons increases. LSTMs solve this problem by saving relevant information through all LSTM units in a kind of conveyor belt known as *memory cell*.

Each LSTM unit consists of a memory cell and three gates that regulate the information flow by deciding which information will be forgotten and which one will remain in the model. This is the key to learn long-term dependencies by the LSTM and solve the previously mentioned issue. These three gates are called *forget gate*, *input gate* and *output gate*. Next, we briefly describe their features.

In the forget gate, a sigmoid function is applied to the information included in the current input (X_t) and the previous hidden state ($h-1$). This function, denoted by f_t, returns a value between 0 and 1 that indicates the percentage of the information that will be kept.

The input gate takes the information from the current input and from the previous hidden state and passes them through a second sigmoid function, transforming this information into values between 0 and 1. The same information also passes through a *tanh* function, which helps to regulate the network, returning a value between -1 and 1. Then, the sigmoid output (i_t) is multiplied by the *tanh* output ($\tilde{C}_t$) to decide which information is important to keep.

Now, we have enough information to compute the cell state. First, the previous cell state (C_{t-1}) is multiplied by the forget output. Then, the output of the input gate is added, updating the cell state with new values considered relevant by the network. The result of these two operations is the new cell state (C_t).

Finally, we have the output gate. It decides the value of the next hidden state. First, the information from the current input and from the previous hidden state passes through a third sigmoid function. Then, the new cell state passes through a *tanh* function. Both of these outputs are multiplied point-by-point. The output of the output gate (o_t) will be the new hidden state (h_t). Therefore, the new cell state and the new hidden state will be carried to the next timestep.

Formally, the operations that happen in an LSTM cell are given by the following expressions:

$$\begin{aligned} f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= tanh(W_c \cdot [h_{t-1}, x_t] + b_C) \\ C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \\ o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t &= o_t \cdot tanh(C_t) \end{aligned} \tag{4}$$

where σ represents a sigmoid function, *tanh* represents a hyperbolic tangent function, x_t represents the input data in time t, h_t represents the hidden state in time t, each W_x represents a weight matrix and each b_x represents a bias vector.

Once we have explained the behaviour of an LSTM network, we can describe how BiLSTM networks work. They are an evolution of LSTM networks based on Bidirectional RNNs, an extension of RNNs. A BiLSTM network contains a forward LSTM network and a backward LSTM network. The forward LSTM uses as input a sequence of values ranging from $t-k$ to t while the backward LSTM uses as input a sequence ranging from t to $t-k$. The outputs of the forward ($\overrightarrow{h}$) and backward ($\overleftarrow{h}$) networks are computed by using the previously described

mechanism of a single LSTM. The BiLSTM layer outputs a vector (Y_t) that is obtained by using the following equation:

$$Y_t = \sigma(\overrightarrow{h_t}, \overleftarrow{h_t}) \tag{5}$$

In this case, a sigmoid function unifies the outputs of the single LSTM networks.

Note that BiLSTMs use the so-called *dropout technique* in order to regulate over-fitting. The dropout rate is indeed a hyperparameter to consider in the definition of the model.

Finally, we stack a fully-connected layer that returns the prediction output.

4.3. Tuning parameters

First, we set in advance the number of epochs (100) and the batch size (32). Then, we applied an exhaustive search algorithm in order to fine-tune the hyperparameters of the model. The ranges of the possible values of hyperparameters were selected by ensuring to use a reasonable computational time in the execution of the model (at most, 70 seconds per epochs). The obtained hyperparameters were as follows.

The convolutional layer contains 256 filters. Our experiments showed that having a higher number of filters achieved a very modest decrease of the error while considerably increasing the computation time. Besides, using a lower number of filters considerably increased the error. As kernel, we use a 2×2 matrix. Although this kernel dimension might look too small, we must take into account that the input matrix has also a relatively small dimension (9×4). By using a higher dimension kernel, a small number of hidden features would be extracted by the convolutional operation. In the max pooling layer, we use a 2×2 window with a stride of 2 to apply the max pooling operation. Again, we use a small window dimension because of the small dimension of the input matrix. We fixed the stride value after several experiments revealing that for higher values, the dimension reduction implies a significant information loss. This pooling operation is made by using a *same padding* technique, which means that output and input sizes would be the same with stride equal to 1. Therefore, it requires the filter window to slip outside the input matrix. For this, it is necessary to add either a row or a column of zeros. Our experiments showed that the usual *no padding* technique, which means that only the input matrix will be covered by the filter window with the given stride, obtained slightly worse results than same padding.

Focusing now on the BiLSTM block, the BiLSTM layer contains 500 BiLSTM units (each of them is composed by a forward LSTM unit and a backward LSTM unit). We considered 500 BiLSTM units because this number provided a good trade-off between observed error and computational time. The initial weights of the BiLSTM layer were set from a random normal distribution. We also made preliminary experiments with other distributions such as random uniform, truncated normal, zeros distribution and ones distribution. After that, we conclude that random normal gave the best results. Then, we added a dropout layer with a rate of 0.5. As the dropout rate did not seem to have influence in the final results, we decided to use the usually considered best dropout rate value for hidden layers (Baldi and Sadowski, 2013). The fully connected layer has only 1 output neuron with a linear activation function because we predict only one value.

We used the Adam optimiser as optimisation algorithm for stochastic gradient descent on the network by using as loss function the *mean absolute error*.

Finally, we would like to explain the order applied to the 36 predictor variables (nine variables by each of the four considered timesteps). It is natural to consider a 9×4 input matrix because the relationship between variables with a difference of one timestep will be bigger than the relationship between variables with a bigger difference. For the same reason, we considered natural that the rows should be in ascending time order. However, there is not a by default order for the nine columns corresponding to the predictor variables. We considered a grouping by *affinity* (see Fig. 2) but it should be analysed whether this order makes sense. In order to show that we are choosing the *best* order, it would be necessary to produce the $9! = 362.880$ different combinations and perform the experiments for each of these combinations. Even if this would be feasible, this approach does not scale with the number of predictor variables. Therefore, we decided to perform a different experiment with the goal of checking whether the specific order between columns have a relevant impact. We produced eight random combinations, took one traffic station and one prediction and compared the training curves to see whether there were significant differences between them. As can be seen in Fig. 5, the curves of all the groupings are quite similar, so that we have some evidence to claim that the order of the predictor variables does not significantly affect either the training process or the final loss.

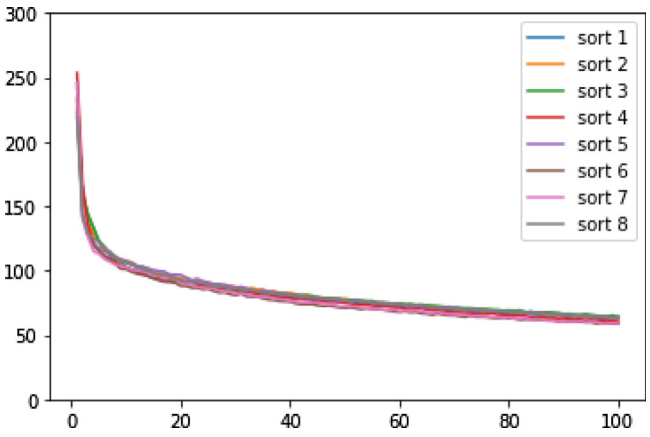


Fig. 5. Loss function value (y axis) during the epochs (x axis) of the training process, using eight different combinations of the same predictor variables (Arturo Soria, 12 h in advance).

We have used *Netron* (Netron, 2022) to provide a graphical representation of our hybrid model, including the values of the relevant hyperparameters (see Fig. 6). The interested reader is referred to https://github.com/MMH1997/CNN-BiLSTM-network for the complete code of the model.

5. Baseline models

In this section we will briefly describe the baseline models whose performance will be compared with the one of the proposed model. We will specify the hyperparameter values for each baseline model.

5.1. Random forest (RF) regressor

The RF Regressor algorithm is a statistical machine learning model whose prediction is based on the output of combined regression trees. These regression trees are trained by randomly sampled subsets of the original training set. The final output value of the RF model will be the mean of the regression trees outputs. The use of several regression trees considerably improves the use of a single one. Specifically, it avoids original handicaps of regression trees such as a high susceptibility to a brief data modification and over-fitting.

We will use the following hyperparameters in this algorithm: 100 regression trees, a maximum depth of 10 for each tree and a minimum number of samples required in a node before splitting it of 20. These values were set after performing preliminary experiments. These experiments showed that for more than 100 regression trees, the mean absolute error converges to a similar value. Similarly, maximum depth and minimum number of samples required in a node have been selected after a fine-tuning process.

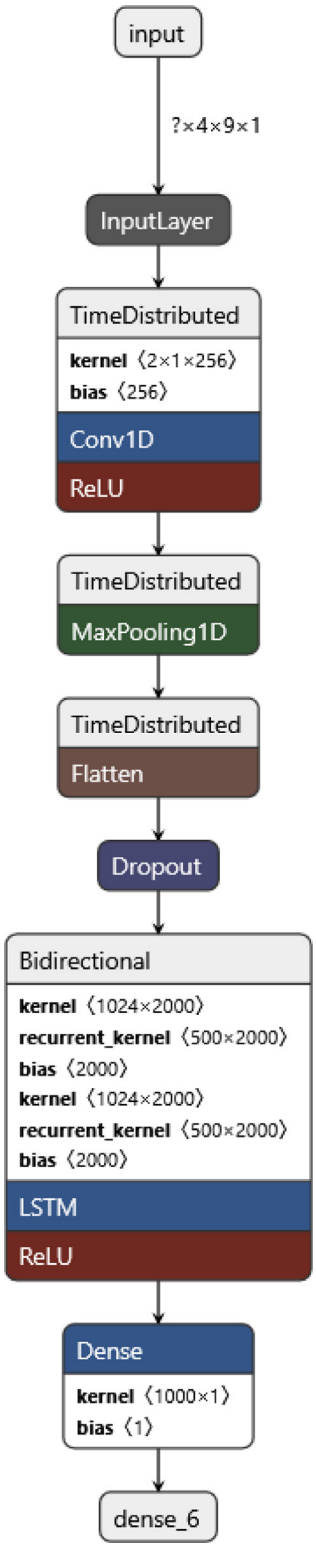


Fig. 6. Our proposed hybrid model.

5.2. Linear regression (LR)

LR is a statistical machine learning algorithm used to find a linear relationship between a dependent or target variable and several independent or predictor variables. In our case, we do not use any variant such as Ridge or Lasso.

5.3. k-Nearest neighbours algorithm (KNN)

KNN is a non-parametric and statistical algorithm that bases its output on the mean of the target variable of the k most similar observations. KNN does not have a training phase, that is, it does not learn a specific model. It just makes predictions computing the similarity between observations.

After a fine-tuning process, we decided to use the following hyperparameters: 5 as k value, *euclidean metric* as similarity distance and the weight of each neighbour will be inversely proportional to its distance to the target observation.

5.4. Simple recurrent neural networks (SRNN)

SRNNs were the first recurrent network. They carry out a recurrent connection between the output of the previous and current inputs. This process develops a temporal feedback in the network. SRNNs were the prelude of the more powerful and accurate LSTM networks and derivatives.

In our experiments, after a fine-tuning process, we used an SRNN with 500 units, followed by a dense layer with 1 neuron. In this case, we train the model using a Stochastic Gradient Descent as optimiser, with a learning rate of 0.1.

5.5. Convolutional neural network (CNN)

CNN is commonly used to work with images and matrices. Its architecture is described in Section 4. In this case, we use the matrix 9×4 as input to obtain the corresponding target variable as output. We use a $2D$ convolutional layer with 256 filters, a kernel size of 2×2 and a $relu$ as activation function. This is followed by a MaxPooling layer with a pool size 2×2. Then, we add a flatten layer and finally a dense layer with 1 neuron and a linear function as activation.

5.6. LSTM network

LSTM network is the most widely used deep learning model in regression and time series tasks. Its architecture has been described in Section 4. In our experiments, we use a unidirectional LSTM layer with 500 units followed by a dropout layer with a rate of 0.5 and by a dense layer with 1 neuron. We use these hyperparameters in order to obtain a model with a similar complexity to our proposed model.

5.7. BiLSTM network

BiLSTM is a decisive baseline model. It will show us whether it is worth to add the CNN block to the proposed model. Its architecture is the same as the one described in Section 4, but without including the CNN block. As happens with the previously described baseline, we will use similar hyperparameters to the proposal with the same purpose. The BiLSTM contains 500 BiLSTM units followed by a dropout layer with a rate of 0.5 and by a dense layer with 1 neuron.

5.8. CNN-LSTM network

This is also a decisive baseline model. It will allow us to determine whether it is worth to use a BiLSTM rather than an LSTM in the proposed model. Its architecture is similar to the one described in Section 4. The only difference is that we will use here a unidirectional LSTM network rather than a BiLSTM network. The hyperparameters in the CNN block will be exactly the same as in the proposed model. The LSTM block contains 500 LSTM units followed by a dropout layer with a rate of 0.5 and by a dense layer with 1 neuron.

5.9. Discarded alternative baseline models

In addition to the considered baseline models, we analysed other possibilities such as multi-layer perceptron neural network (MLP). However, this type of neural networks does not have a temporal component. For example, in order to use MLP it would be necessary a data flattened, which would completely eliminate the temporal dependencies, so that data can be taken by the model as a single vector.

6. Experiments

In this section, we evaluate the proposed model in the four data-sets corresponding to each target station. We also compare it with the eight baseline models previously described. For each experiment, we use 2/3 of data to train and 1/3 of data to test. We randomly chose the training and testing sets. In order to evaluate the performance of the models, we considered three metrics.

- *Mean absolute error (MAE)*. This metric is one of the most used in regression to evaluate the quality of a model predictions. Its definition is:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i| \tag{6}$$

  being n the total number of evaluated observations, $\hat{y}_i$ the predicted value of the ith observation and y_i the actual value of the ith observation.
- *Accuracy (Acc)*. This is a classification metric that can be very useful to implement measures by the authorities. For each station, we will classify the values (actual and predicted) in three categories (low volume of traffic, medium volume of traffic and high volume of traffic). Let P_k be the kth percentile of the target variable. The first category includes the observations which are between P_0 and P_{15}, the second category includes the observations which are between P_{15} and P_{85} and the third category includes the observations which are between P_{85} and P_{100}. Acc is defined as the ratio between the number of well-classified observations and the total number of observations.
- *Root mean square error ($RMSE$)*. This metric is also widely used in regression. We considered it just in case that it could reveal results different from the ones corresponding to MAE. Its definition is:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{7}$$

  being n the total number of evaluated observations, $\hat{y}_i$ the predicted value of the ith observation and y_i the actual value of the ith observation.

It should be mentioned that we considered another typical metric in regression: *mean absolute percentage error* ($MAPE$). However, $MAPE$ has some disadvantages (Goodwin and Lawton, 1999) that hinder its usefulness. For example, when the actual value is close to 0, it yields extremely large percentage errors. In case that actual value is 0 and the predicted value is not 0, it produces undefined errors. We have this situation in our data, particularly, early morning data in working days, when the hourly traffic flow can be close to 0 or even 0. Therefore, we discarded it.

Concerning some of the models that we use, note that neural networks and RF models present a random component. Because of this, we evaluated each of these models five times in each station, using different seeds, in order to give consistency to our experiments. Then, the average of the five evaluations by model and station was considered.[4] The rest of models (KNN and LR) do not have a random component and, therefore, only one evaluation in each station was performed.

We will show and comment the results of the experiments in each target station and from a broad perspective. In the discussion of the results, we will emphasise more the values corresponding to MAE. The rationale is that models are trained to optimise MAE and that our observations show that results corresponding to the three metrics are quite similar.

Finally, one remark about the presentation of the results corresponding to the experiments. MAE values obtained by the LR and CNN models are much higher than the ones corresponding to the rest of the models. Therefore, we have decided to remove the LR and CNN models in the graphs because, using a linear scale, it would condition the plot ranges and, therefore, the graphical displays. Nevertheless, these values can be found in the different tables presenting the numeric results.

6.1. P/Castellana station results

In this station, as we can see in Fig. 7 and Table 2, the best performance is obtained in all time granularities by the proposed model except when predicting results 12 h in advance. The MAE of the proposed model is 1.2% higher than the MAE of BiLSTM in the 12 h granularity. However, the MAE of the proposed model is 3.6% smaller than the MAE of the second best model (CNN-LSTM) in 24 h granularity; 3.2% smaller than the MAE of the second best model (BiLSTM) in 48 h granularity and 1.8% smaller than the MAE of the second best model (BiLSTM) in 72 h granularity. By taking into account the average of all granularities, the MAE of the proposed model is 2.2% smaller than the MAE of the second best model (BiLSTM). Acc behaves similar to MAE: the proposed model obtains the best performance in all granularities except when working 12 h in advance (in this case, BiLSTM obtains the best performance). Considering the $RMSE$, in all granularities our proposal obtains the best results except when working 24 h in advance. By taking into account the average of all granularities, the $RMSE$ of the proposed model is 2.3% smaller than the $RMSE$ of the second best model (RF). Interestingly enough, in this station, all the models show better results, both for MAE and for Acc (not for $RMSE$), in 24 h granularity than in 12 h granularity. This phenomenon appears sometimes in some models applied to other stations.

6.2. C/Arturo Soria station results

In this station, as we can see in Fig. 8 and Table 3, the best results are obtained by the proposed model in all granularities. The MAE of the proposed model is 2.5% smaller than the MAE of the second best model (BiLSTM) in 12 h granularity; the MAE of the proposed model is 13.5% smaller than the MAE of the second best model (BiLSTM) in 24 h granularity; the MAE of the proposed model is 5.6% smaller than the MAE of the second best model (BiLSTM) in 48 h granularity and, finally, 7.7% smaller than the MAE of the second best model (BiLSTM) in 72 h granularity. On average, the MAE of the proposed model is 7.4% smaller than the MAE of the second best model (BiLSTM). The Acc and $RMSE$ of the proposed model also obtain the best performance in all granularities.

6.3. P/Sta María station results

In this station, as we can see in Fig. 9 and Table 4, the proposed model obtains again the best performance in all granularities. The MAE of the proposed model is 2.8% smaller than the MAE of the second best model (BiLSTM) in 12 h granularity; the MAE of the proposed model is 5.3% smaller than the MAE of the second best model (RF) in 24 h granularity; the MAE of the proposed model is 2.4% smaller than the MAE of the second best model (BiLSTM) in 48 h granularity and 1.7% smaller than the MAE of the second best model (BiLSTM) in 72 h granularity. On average, the MAE of the

[4] We think that repeating the experiment five times was enough because the results were very stable and there was a small variance. In fact, average and best results were very close, as can be appreciated in Tables 2–5.

Table 2
MAE (top), Acc (centre) and $RMSE$ (bottom) average values by granularity and model (P/Castellana).

Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	176.23	153.44	183.29	196.42	175.44	153.37	182.56	196.01
LR	395.33	228.38	300.63	319.03	395.33	228.38	300.63	319.03
KNN	179.99	167.45	197.78	209.18	179.99	167.45	197.78	209.18
SRNN	165,44	161.61	213.52	208.77	161.04	153.41	203.75	205.14
CNN	353.44	269.79	326.07	338.02	350.68	262.14	320.86	337.11
LSTM	157.61	153.67	187.74	197.81	152.52	151.86	182.31	192.67
CNN-LSTM	151.57	141.55	178.63	185.65	146.49	**135.00**	174.62	182.83
BiLSTM	**142.85**	142.75	174.55	187.32	141.09	139.86	171.60	182.89
Proposed model	144.22	**136.51**	**168.92**	**183.95**	**138.62**	135.09	**166.83**	**180.38**
Granularity	Mean				Max			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	0.79	0.829	0.792	0.778	0.792	0.831	0.792	0.779
LR	0.707	0.75	0.72	0.712	0.707	0.75	0.72	0.712
KNN	0.803	0.813	0.789	0.768	0.803	0.813	0.789	0.768
SRNN	0.807	0.813	0.782	0.773	0.817	0.827	0.786	0.777
CNN	0.687	0.743	0.72	0.711	0.692	0.747	0.724	0.714
LSTM	0.826	0.84	0.805	0.794	0.834	0.846	0.814	0.798
CNN-LSTM	0.834	0.846	0.808	0.807	0.838	0.852	0.817	0.811
BiLSTM	**0.84**	0.847	0.817	0.807	0.846	0.853	**0.824**	**0.816**
Proposed model	0.839	**0.851**	**0.822**	**0.808**	**0.848**	**0.856**	0.823	0.812
Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	253.95	**233.04**	**264.73**	293.57	253.02	232.82	**263.91**	292.9
LR	475.13	321.28	393.11	408.41	475.13	321.28	393.11	408.41
KNN	269.28	256.02	291	304.78	269.28	256.02	291	304.78
SRNN	263.98	264.76	322.02	317.52	258.03	253.57	307.1	306.58
CNN	458.45	354.49	429.96	443.1	455.09	350.72	423.65	438.9
LSTM	257.14	256.76	294.7	320.66	249.4	253.79	289.78	311.89
CNN-LSTM	249.96	244.93	287.29	299.11	238.66	235.92	281.15	295.82
BiLSTM	230.42	243.28	283.49	300.93	227.97	239.9	280.84	291.76
Proposed model	**224.46**	241.47	277.24	**291.11**	**219.37**	**226.54**	273.74	**285**

Table 3
MAE (top), Acc (centre) and $RMSE$ (bottom) average values by granularity and model (C/Arturo Soria).

Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	111.45	99.00	125.45	129.76	111.27	98.93	124.76	129.60
LR	242.80	150.72	198.54	204.11	242.80	150.72	198.54	204.11
KNN	107.85	108.59	129.25	134.47	107.85	108.59	129.25	134.47
SRNN	88.36	93.74	134.43	128.04	85.10	96.20	126.72	124.10
CNN	174.79	163.33	195.42	201.25	170.30	158.47	192.90	196.61
LSTM	95.36	106.03	123.90	131.42	93.07	103.72	121.46	127.74
CNN-LSTM	100.02	108.57	128.73	135.03	98.35	104.60	127.65	128.79
BiLSTM	84.13	98.68	117.97	128.14	81.81	97.08	116.60	126.44
Proposed model	**82.06**	**85.41**	**111.43**	**118.21**	**79.87**	**84.02**	**108.32**	**117.4**
Granularity	Mean				Max			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	0.835	0.842	0.802	0.795	0.833	0.84	0.801	0.794
LR	0.715	0.75	0.718	0.715	0.715	0.75	0.718	0.715
KNN	0.815	0.816	0.792	0.784	0.815	0.816	0.792	0.784
SRNN	0.837	0.829	0.787	0.8	0.844	0.834	0.797	0.815
CNN	0.714	0.724	0.737	0.733	0.722	0.73	0.741	0.737
LSTM	0.828	0.816	0.798	0.793	0.804	0.807	0.796	0.788
CNN-LSTM	0.82	0.814	0.787	0.79	0.817	0.81	0.782	0.781
BiLSTM	0.837	0.827	0.81	0.802	0.831	0.822	0.804	0.793
Proposed model	**0.853**	**0.852**	**0.825**	**0.82**	**0.848**	**0.847**	**0.815**	**0.814**
Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	168.31	161.36	192.76	196.78	168.22	161.18	191.75	196.38
LR	299.36	218.53	264.21	266.91	299.36	218.53	264.21	266.91
KNN	174.38	177.06	201.65	206.95	174.38	177.06	201.65	206.95
SRNN	165.50	179.94	219.09	210.16	162.25	176.24	215.93	203.41
CNN	253.35	231.67	271.94	278.38	247.84	229.15	268.80	271.25
LSTM	154.90	172.33	193.57	200.00	152.19	169.43	192.55	196.23
CNN-LSTM	152.70	174.14	196.78	203.02	150.33	168.60	195.82	198.03
BiLSTM	137.93	162.37	185.80	195.51	135.30	161.00	183.42	194.46
Proposed model	**137.05**	**158.12**	**181.02**	**190.34**	**133.08**	**157.44**	**176.24**	**188.04**

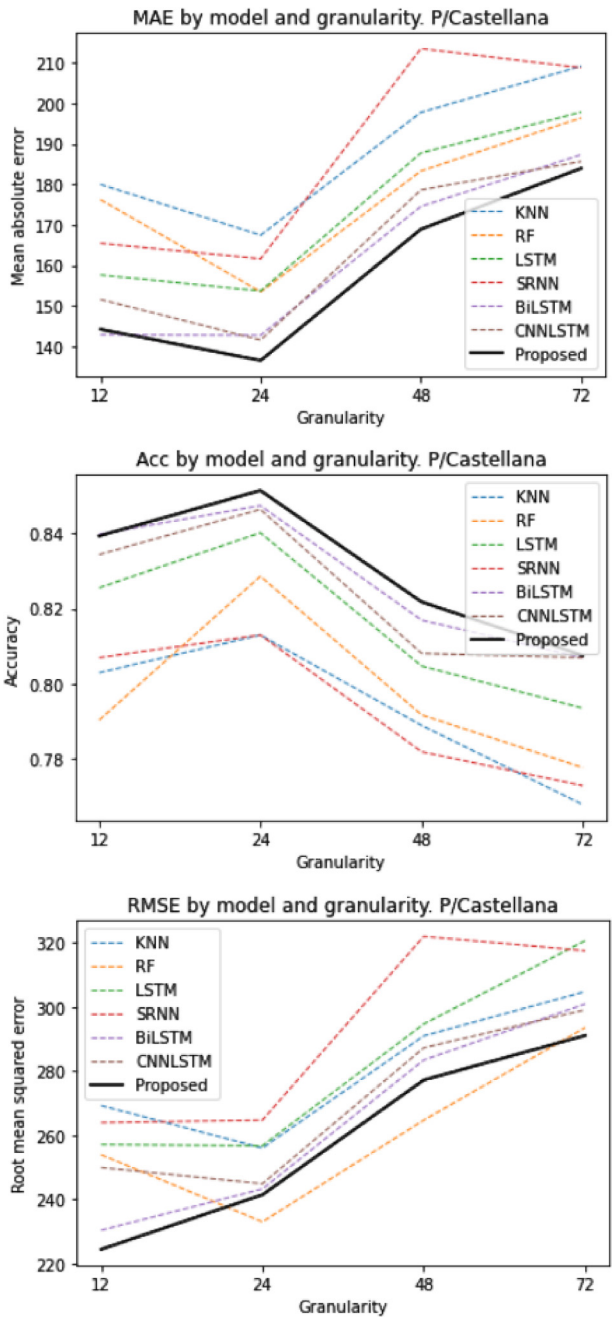


Fig. 7. Line charts of MAE (top), Acc (centre) and $RMSE$ (bottom) by model and granularity (P/Castellana).

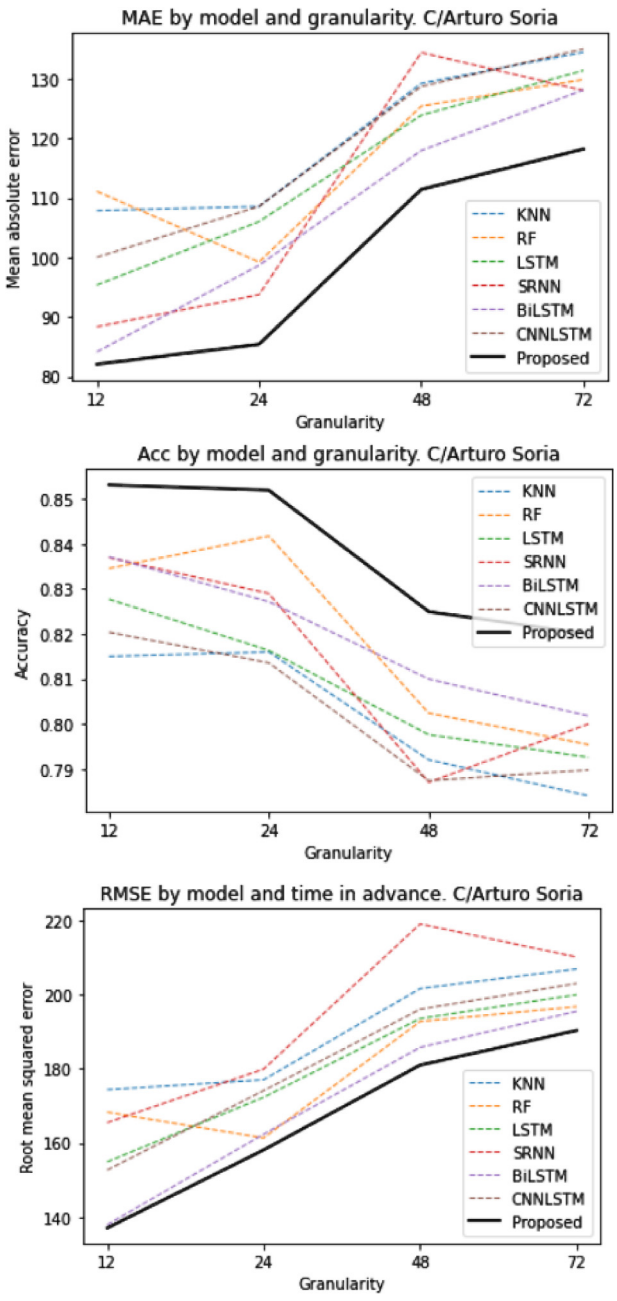


Fig. 8. Line charts of MAE (top), Acc (centre) and $RMSE$ (bottom) by model and granularity (C/Arturo Soria).

proposed model is 3.6% smaller than the MAE of the second best model (BiLSTM). The Acc of the proposed model also obtains the best performance in all granularities. However, the $RMSE$ of the proposed model obtains the best performance in all cases except in 24 h in advance. On average, the $RMSE$ of the proposed model is 2.2% smaller than the $RMSE$ of the second best model (BiLSTM).

6.4. C/Gran Vía station results

Finally, the MAE results corresponding to C/Gran Vía station show again that our proposed model outperforms all the baseline models for all granularities (see Fig. 10 and Table 5). The MAE of the proposed model is 7.1% smaller than the MAE of the second best model (BiLSTM) in 12 h granularity; the MAE of the proposed model is 4.3% smaller than the MAE of the second best model (RF) in 24 h granularity; the MAE of the proposed model is 7.3% smaller than the MAE of the second best model (RF) in 48 h granularity and 3.1% smaller than the MAE of the second best model (BiLSTM) in 72 h granularity. On average, the MAE of the proposed model is a 6.7% smaller than the MAE of the second best model (BiLSTM). In this station, the Acc of the proposed model is the best in all granularities except in 12 h in advance. In this case, BiLSTM is slightly better than the proposed model. However, the $RMSE$ of the proposed model obtains the best performance in only one granularity (48 h in advance). On the average, the $RMSE$ of the proposed model is 0.8% higher than the obtained in the best model (BiLSTM).

We assume that anomalies seen in these results can be explained by considering the particularities of the C/Gran Via data mentioned in data analysis Section 3.3.

Table 4
MAE (top), Acc (centre) and $RMSE$ (bottom) average values by granularity and model (P/Sta María).

Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	95.44	100.46	118.01	122.18	95.20	100.19	117.72	122.00
LR	212.83	138.61	175.71	185.15	212.83	138.61	175.71	185.15
KNN	124.31	107.45	120.12	127.39	124.31	107.45	120.12	127.39
SRNN	98.32	103.50	135.83	124.14	95.02	99.12	124.72	122.70
CNN	165.34	157.29	182.85	189.79	129.64	155.41	181.70	185.53
LSTM	96.05	107.29	120.77	122.44	94.75	104.43	117.60	121.69
CNN-LSTM	99.49	103.65	118.86	122.44	95.48	103.21	116.54	121.14
BiLSTM	89.75	104.11	112.97	119.17	88.13	101.64	110.97	117.83
Proposed model	**87.30**	**96.12**	**110.25**	**117.20**	**85.05**	**93.68**	**105.76**	**113.49**
Granularity	Mean				Max			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	0.828	0.836	0.789	0.791	0.83	0.838	0.789	0.792
LR	0.7	0.736	0.71	0.706	0.7	0.736	0.71	0.706
KNN	0.783	0.82	0.795	0.781	0.783	0.82	0.795	0.781
SRNN	0.82	0.819	0.77	0.79	0.829	0.823	0.793	0.795
CNN	0.707	0.729	0.725	0.721	0.713	0.73	0.727	0.723
LSTM	0.828	0.816	0.791	0.788	0.837	0.821	0.796	0.792
CNN-LSTM	0.827	0.83	0.805	0.797	0.839	0.836	0.809	0.8
BiLSTM	0.838	0.819	0.811	0.8	0.846	0.83	0.817	0.806
Proposed model	**0.847**	**0.839**	**0.821**	**0.811**	**0.851**	**0.846**	**0.827**	**0.818**
Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	138.84	**148.12**	165.65	170.90	138.49	147.91	165.23	170.66
LR	250.55	189.85	223.74	235.18	250.55	189.85	223.74	235.18
KNN	174.61	160.42	170.67	267.84	174.61	160.42	170.67	267.84
SRNN	154.09	160.69	200.05	186.55	149.65	155.05	184	182.92
CNN	233.27	210.59	241.12	248.36	229.8	210.13	237.39	242.29
LSTM	141.19	155.72	169.13	169.71	136.77	153.11	167.2	168.99
CNN-LSTM	152.16	161.58	176.07	179.77	147.48	160.58	173.77	178.64
BiLSTM	131.13	151.53	162.66	167.73	130.04	148.91	160.07	166.2
Proposed model	**126.78**	151.36	**162.44**	**159.12**	**124.23**	**147.42**	**159.34**	**155.02**

Table 5
MAE (top), Acc (centre) and $RMSE$ (bottom) average values by granularity and model (C/Gran Vía).

Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	153.26	154.36	175.40	173.94	153.02	154.06	175.08	173.86
LR	259.88	196.55	243.91	251.27	259.88	196.55	243.91	251.27
KNN	154.19	163.22	176.77	182.84	154.19	163.22	176.77	182.84
SRNN	146.56	151.16	197.96	189.87	145.33	149.03	187.62	186.94
CNN	251.39	210.4	243.94	254.32	241.22	206.02	242.07	248.64
LSTM	151.75	164.61	185.41	186.13	150.02	163.19	181.75	182.19
CNN-LSTM	148.40	156.94	175.86	179.07	146.42	156.31	174.44	176.53
BiLSTM	143.43	161.29	175.69	175.20	140.86	156.36	171.38	172.29
Proposed model	**133.22**	**147.7**	**162.66**	**168.61**	**131.38**	**143.91**	**159.29**	**164.86**
Granularity	Mean				Max			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	0.774	0.782	0.764	0.756	0.775	0.783	0.765	0.757
LR	0.7	0.73	0.705	0.698	0.7	0.73	0.705	0.698
KNN	0.774	0.774	0.756	0.746	0.774	0.774	0.756	0.746
SRNN	0.782	0.783	0.739	0.739	0.784	0.787	0.745	0.743
CNN	0.696	0.707	0.699	0.689	0.701	0.71	0.7	0.691
LSTM	0.78	0.767	0.752	0.739	0.787	0.771	0.761	0.746
CNN-LSTM	0.787	0.783	0.764	0.755	0.793	0.786	0.767	0.761
BiLSTM	**0.805**	0.78	0.758	0.743	**0.809**	0.783	0.764	0.756
Proposed model	0.793	**0.794**	**0.77**	**0.761**	0.803	**0.8**	**0.783**	**0.765**
Granularity	Mean				Min			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	227.05	**237.78**	264.29	263.39	226.95	237	263.79	263.23
LR	330.7	283.97	333.29	338.74	330.7	283.97	333.29	338.74
KNN	231.73	250.37	265.98	267.84	231.73	250.37	265.98	267.84
SRNN	241.14	256.25	308.66	303.99	243.79	252.88	296.72	301.48
CNN	339.19	304.66	340.29	352.37	327.48	300.05	337.17	346.3
LSTM	230.6	248.07	272.23	267.54	227.86	243.76	266.6	262.43
CNN-LSTM	242.61	266.54	288.49	291.55	240.53	266.54	288.5	291.55
BiLSTM	**220.92**	243.05	261.21	**257.62**	219.13	237.2	**255.77**	**250.2**
Proposed model	224.55	239.19	**261.11**	266.24	**217.7**	**230.73**	258.2	261.17

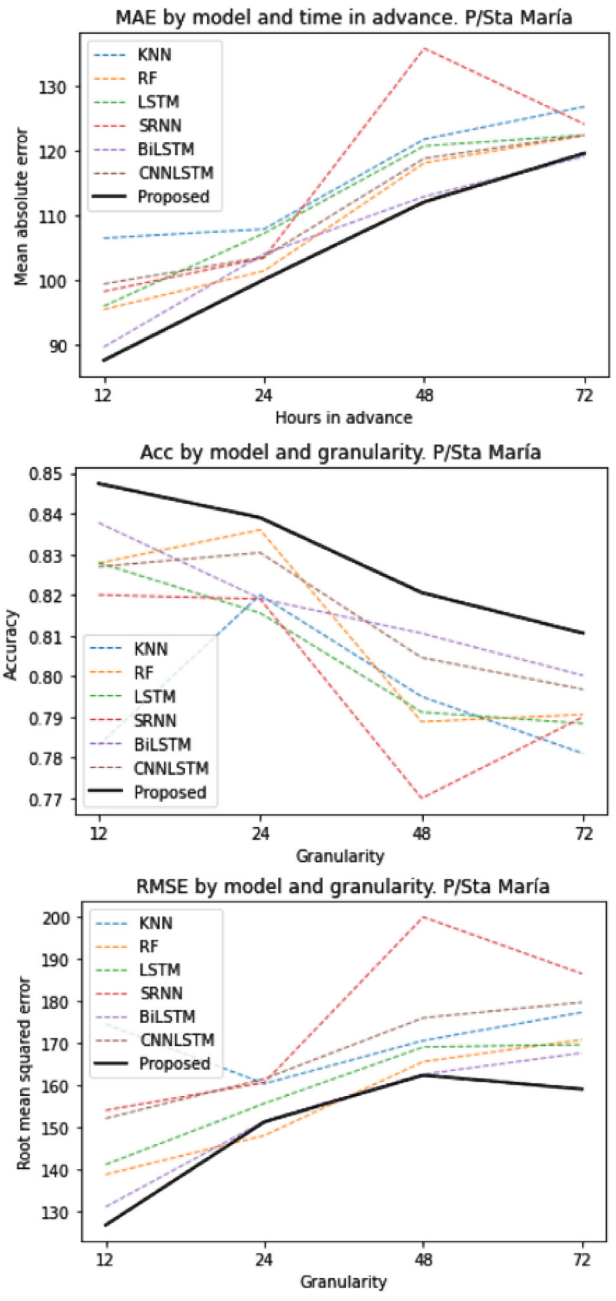


Fig. 9. Line charts of MAE (top), Acc (centre) and $RMSE$ (bottom) by model and granularity (P/Sta María).

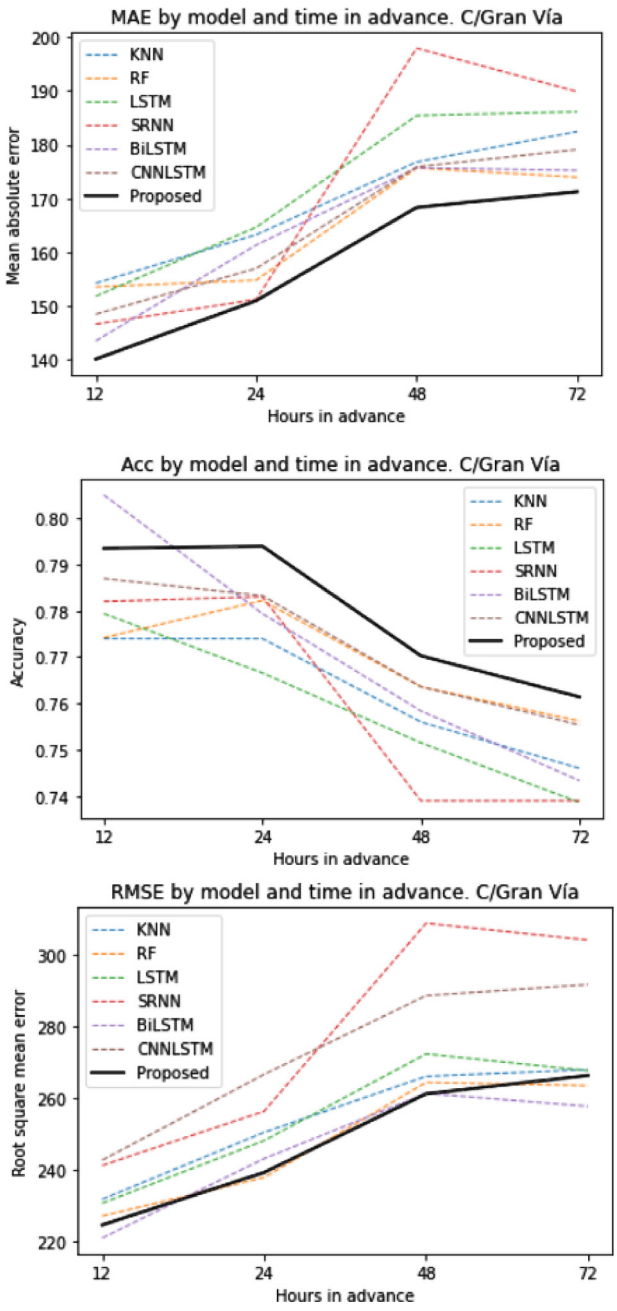


Fig. 10. Line charts of MAE (top), Acc (centre) and $RMSE$ (bottom) by model and granularity (C/Gran Vía).

6.5. Summary of the results

Finally, we will analyse the general behaviour of each model by taking into account the average value of all the stations. As we can see in Fig. 11 and Table 6, the proposed model obtains the best results in all the granularities and metrics except considering $RMSE$ in 24 h in advance. By granularity, the MAE of the proposed model is 3% smaller than the MAE of the second best model (BiLSTM) in 12 h granularity; 8.1% smaller than the MAE of the second best models (RF and BiLSTM) in 24 h granularity; 4.8% smaller than the MAE of the second best model (BiLSTM) in 48 h granularity and 3.6% smaller than the MAE of the second best model (BiLSTM) in 72 h granularity. On average, the MAE of the proposed model is 4.8% smaller than the MAE of second best model, BiLSTM. Now, we can analyse the impact of the components that we apply in the proposed model. On average, to use a BiLSTM block rather than an LSTM block implies an 8.1% decrease of the MAE while adding a CNN block rather than not adding it implies a decrease of the MAE by 4.8%. Finally, the performance of both improvements implies a decrease of the MAE by 10.3% with respect to a single LSTM model.

The station in which the proposed model obtains the biggest difference with respect to the baseline models is C/Gran Vía. This may be in part due to the smaller temporal dependence in this station, as we mentioned in Section 3.3. This result shows that the hidden characteristics detected by the CNN block of the proposed model are not only temporal dependencies: it is also able to identify relationships between different predictor variables. On the contrary, the baseline models are not so well suited to detect these relationships. These non-temporal relations are present to a greater or lesser extent in all stations data. Because of this, the proposed model obtains the best results in fifteen out of the sixteen performed experiments.

Table 6
MAE, Acc and $RMSE$ values by granularity and model (average of the four stations).

Granularity	MAE				Acc				$RMSE$			
	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h	12 h	24 h	48 h	72 h
RF	134.1	126.817	150.542	155.575	0.807	0.822	0.786	0.78	197.04	**195.09**	221.86	231.16
KNN	141.585	136.678	155.98	163.47	0.794	0.806	0.783	0.77	212.5	210.97	232.33	39.35
SRNN	124.64	127.5	140.44	162.7	0.8115	0.811	0.7695	0.775	206.18	215.51	265.45	254.33
LSTM	125.197	132.897	154.455	159.45	0.815	0.81	0.786	0.778	195.96	208.22	232.41	239.48
CNN-LSTM	124.87	127.677	150.52	155.547	0.817	0.818	0.791	0.787	199.36	211.79	237.16	243.36
BiLSTM	115.04	126.713	145.295	152.458	0.83	0.818	0.8	0.788	180.1	200.06	223.29	230.45
Proposed model	**111.69**	**116.43**	**138.32**	**146.99**	**0.833**	**0.834**	**0.809**	**0.8**	**178.21**	197.53	**220.45**	**226.7**

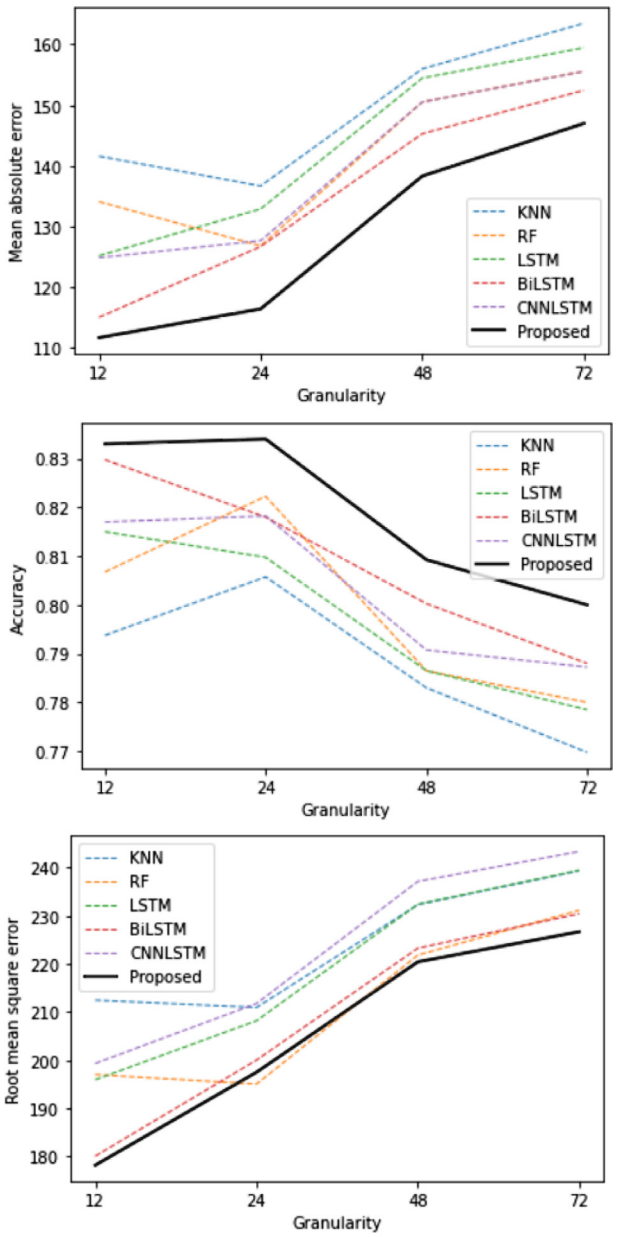


Fig. 11. Line charts of MAE (top), Acc (centre) and $RMSE$ (bottom) by model and granularity (average of four stations).

Briefly discussing the Acc values, we can mention that it commonly behaves as MAE. The Acc of the proposed model obtains the best results in all granularities. We can highlight that there is a low increase of the Acc in the proposed model (and in others such as RF, CNN-LSTM and KNN) from the 12 to the 24 h granularity. For this metric, the proposed model obtains the best performance in fourteen out of the sixteen experiments.

Finally, briefly discussing the $RMSE$ values, we note that, in general, our proposal obtains the best performance in all the granularities except in 24 h in advance. In this case, RF obtains the best results in three out of four stations. However, RF obtains poorer $RMSE$ than our proposal when predicting 12 h in advance, while our proposal obtains similar $RMSE$ to RF in 24 h in advance. Moreover, our proposal obtains the best performance in ten out of the sixteen experiments while the second best model (RF) obtains only the best results in four out of the sixteen experiments. Therefore, our model does not only stand out for its best performance, but also by its higher consistency with respect to baseline models.

7. Conclusions and future work

In this paper we presented the development of a hybrid model between two types of neural networks (CNN and BiLSTM) for long-term traffic flow forecasting. We have evaluated the usefulness of our model in four traffic stations of Madrid. In order to assess the quality and performance of the proposed model, we compared it with eight baseline models typically used in time series problems. The results show that, in general, the proposed model obtains the lowest error in all granularities. By granularity, and considering MAE, the proposal takes a higher difference respect to the best baseline models in 24 h granularity, obtaining an MAE 8.1% smaller than the one obtained by the second best model. On the contrary, the lower difference with respect to the best baseline model is achieved in 12 h granularity, obtaining an MAE 3.3% smaller than the one obtained by the second best model. We would also like to point out that, on average, the accuracy obtained in 24 h granularity (0.834) is slightly higher than the one obtained in 12 h granularity (0.833). By station, and considering Acc, the proposal obtains the higher accuracy in C/Arturo Soria station, obtaining an average accuracy of 0.837, while the lower is obtained in C/Gran Vía station (0.78). Our proposal obtains the best $RMSE$ in 10 out of the sixteen experiments. It also confirms the best performance of our proposal with respect to the baseline models. Moreover, the consistency of the proposed model in all the granularities is another advantage to take into account. We can bring out two main conclusions of our work. First, the extraction of characteristics (CNN layer) from a time series matrix is relevant to obtain a better quality of the forecasts. Second, in light of the obtained results, we can claim that a BiLSTM network obtains better results than an LSTM network in time series tasks. By extension, the hybridisation of BiLSTM and CNN is more accurate than the hybridisation between an LSTM and CNN.

We consider some lines for future work. First, in order to decrease the error of the model, we would like to perform a detailed analysis about the behaviour of different hyperparameters in the CNN-BiLSTM hybrid model. Second, we would like to export the model to other traffic stations in Madrid and to other cities that could require it. In the second case, we will establish a criterion for processing data as general as possible. Third, we would like to introduce new predictor variables.

For example, it would be useful to have a variable that determines the state of traffic restrictions under stressful and unusual conditions such as a pandemic. In this line, we would like to use air quality prediction models (Méndez et al., 2023), in particular, our previous work on ozone concentration (Méndez et al., 2022b), to enhance traffic flow predictions. Finally, we would like to integrate our work with current research on Complex Event Processing (CEP) (Díaz et al., 2020; Roldán et al., 2020; Semlali et al., 2021; Brazález et al., 2022) to implement a system that could automatically raise alarms by taking into account *complex events* generated from different contexts, specifically, traffic flow and air quality.

CRediT authorship contribution statement

Manuel Méndez: Conceptualization, Software, Validation, Data curation, Writing – original draft, Writing – review & editing, Visualization. **Mercedes G. Merayo:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition. **Manuel Núñez:** Conceptualization, Methodology, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

We would like to thank the anonymous reviewers of this paper for the careful reading and their constructive comments, which have helped us to further strengthen the paper.

References

Abdi, J., Moshiri, B., Sedigh, A.K., 2010. Comparison of RBF and MLP neural networks in short-term traffic flow forecasting. In: 2010 International Conference on Power, Control and Embedded Systems. pp. 1–4. http://dx.doi.org/10.1109/ICPCES.2010.5698623.

Abduljabbar, R., Dia, H., Tsai, P.-W., 2021. Unidirectional and bidirectional LSTM models for short-term traffic prediction. J. Adv. Transp. 2021, 1–16. http://dx.doi.org/10.1155/2021/5589075.

Abu Kwaik, K., Saad, M., Chatzikyriakidis, S., Dobnik, S., 2019. LSTM-CNN deep learning model for sentiment analysis of dialectal arabic. In: Smaïli, K. (Ed.), Arabic Language Processing: From Theory to Practice. Springer International Publishing, Cham, ISBN: 978-3-030-32959-4, pp. 108–121. http://dx.doi.org/10.1007/978-3-030-32959-4_8.

Ari, D., Alagoz, B.B., 2022. An effective integrated genetic programming and neural network model for electronic nose calibration of air pollution monitoring application. Neural Comput. Appl. 34, 12633–12652. http://dx.doi.org/10.1007/s00521-022-07129-0.

Ari, D., Alagoz, B.B., 2023. DEHypGpOls: a genetic programming with evolutionary hyperparameter optimization and its application for stock market trend prediction. Soft Comput. 27, 2553–2574. http://dx.doi.org/10.1007/s00500-022-07571-1.

Baldi, P., Sadowski, P.J., 2013. Understanding dropout. In: Burges, C.J.C., Bottou, L., Ghahramani, Z., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a Meeting Held December 5-8, 2013, Lake Tahoe, Nevada, United States. pp. 2814–2822.

Balwant, M.K., 2019. Bidirectional LSTM based on POS tags and CNN architecture for fake news detection. In: 2019 10th International Conference on Computing, Communication and Networking Technologies. ICCCNT, pp. 1–6. http://dx.doi.org/10.1109/ICCCNT45670.2019.8944460.

Belhadi, A., Djenouri, Y., Djenouri, D., Lin, C.-W., 2020. A recurrent neural network for urban long-term traffic flow forecasting. Appl. Intell. 50, http://dx.doi.org/10.1007/s10489-020-01716-1.

Brazález, E., Macià, H., Díaz, G., Valero, V., Boubeta-Puig, J., 2022. PITS: an intelligent transportation system in pandemic times. Eng. Appl. Artif. Intell. 114, 105154. http://dx.doi.org/10.1016/j.engappai.2022.105154.

Bui Khac Hoai, N., Cho, J., Yi, H., 2022. Spatial-temporal graph neural network for traffic forecasting: An overview and open research issues. Appl. Intell. 52, http://dx.doi.org/10.1007/s10489-021-02587-w.

Chen, C., Hu, J., Meng, Q., Zhang, Y., 2011. Short-time traffic flow prediction with ARIMA-GARCH model. In: IEEE Intelligent Vehicles Symposium (IV), 2011, Baden-Baden, Germany, June 5-9, 2011. IEEE, pp. 607–612. http://dx.doi.org/10.1109/IVS.2011.5940418.

Chou, C., Shie, C., Chang, F., Chang, J., Chang, E.Y., 2017. Representation learning on large and small data. http://dx.doi.org/10.48550/arXiv.1707.09873.

Chu, Q., Li, G., Zhou, R., Ping, Z., 2021. Traffic flow prediction model based on LSTM with finnish dataset. In: 2021 6th International Conference on Intelligent Computing and Signal Processing. ICSP, pp. 389–392. http://dx.doi.org/10.1109/ICSP51882.2021.9408888.

Díaz, G., Macià, H., Valero, V., Boubeta-Puig, J., Cuartero, F., 2020. An intelligent transportation system to control air pollution and road traffic in cities integrating CEP and colored Petri nets. Neural Comput. Appl. 32 (2), 405–426. http://dx.doi.org/10.1007/s00521-018-3850-1.

Goodwin, P., Lawton, R., 1999. On the asymmetry of the symmetric MAPE. Int. J. Forecast. (ISSN: 0169-2070) 15 (4), 405–408. http://dx.doi.org/10.1016/S0169-2070(99)00007-2.

Hong, H., Huang, W., Zhou, X., Du, S., Bian, K., Xie, K., 2015. Short-term traffic flow forecasting: Multi-metric KNN with related station discovery. In: 12th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2015, Zhangjiajie, China, August 15-17, 2015. IEEE, pp. 1670–1675. http://dx.doi.org/10.1109/FSKD.2015.7382196.

Hussain, T., Muhammad, K., Ullah, A., Cao, Z., Baik, S.W., de Albuquerque, V.H.C., 2020. Cloud-assisted multiview video summarization using CNN and bidirectional LSTM. IEEE Trans. Ind. Inform. 16 (1), 77–86. http://dx.doi.org/10.1109/TII.2019.2929228.

Kang, C., Zhang, Z., 2020. Application of LSTM in short-term traffic flow prediction. In: 2020 IEEE 5th International Conference on Intelligent Transportation Engineering. ICITE, pp. 98–101. http://dx.doi.org/10.1109/ICITE50838.2020.9231500.

Kong, X., Zhang, J., Wei, X., Xing, W., Lu, W., 2022. Adaptive spatial-temporal graph attention networks for traffic flow forecasting. Appl. Intell. 52, 1–17. http://dx.doi.org/10.1007/s10489-021-02648-0.

Li, D., 2020. Predicting short-term traffic flow in urban based on multivariate linear regression model. J. Intell. Fuzzy Systems 39 (2), 1417–1427. http://dx.doi.org/10.3233/JIFS-179916.

Li, Y., Chai, S., Ma, Z., Wang, G., 2021. A hybrid deep learning framework for long-term traffic flow prediction. IEEE Access 9, 11264–11271. http://dx.doi.org/10.1109/ACCESS.2021.3050836.

Li, S., Xie, Q., Yang, J., 2022. Daily suspended sediment forecast by an integrated dynamic neural network. J. Hydrol. 604, 127258. http://dx.doi.org/10.1016/j.jhydrol.2021.127258.

Li, C., Zhan, G., Li, Z., 2018. News text classification based on improved Bi-LSTM-CNN. In: 2018 9th International Conference on Information Technology in Medicine and Education. ITME, pp. 890–893. http://dx.doi.org/10.1109/ITME.2018.00199.

Lin, X., Huang, Y., 2021. Short-term high-speed traffic flow prediction based on ARIMA-GARCH-M model. Wirel. Pers. Commun. 117 (4), 3421–3430. http://dx.doi.org/10.1007/s11277-021-08085-z.

Lin, J.C.-W., Shao, Y., Zhou, Y., Pirouz, M., Chen, H.-C., 2019. A Bi-LSTM mention hypergraph model with encoding schema for mention extraction. Eng. Appl. Artif. Intell. (ISSN: 0952-1976) 85, 175–181. http://dx.doi.org/10.1016/j.engappai.2019.06.005, URL https://www.sciencedirect.com/science/article/pii/S0952197619301447.

Makarenkov, V., Rokach, L., Shapira, B., 2019. Choosing the right word: Using bidirectional LSTM tagger for writing support systems. Eng. Appl. Artif. Intell. (ISSN: 0952-1976) 84, 1–10. http://dx.doi.org/10.1016/j.engappai.2019.05.003, URL https://www.sciencedirect.com/science/article/pii/S0952197619301034.

Méndez, M., Ibias, A., Núñez, M., 2022a. Using deep learning to detect anomalies in traffic flow. In: 14th Asian Conference on Intelligent Information and Database Systems, ACIIDS'22, LNCS 13757. Springer, pp. 299–312.

Méndez, M., Merayo, M.G., Núñez, M., 2023. Machine learning algorithms to forecast air quality: a survey. Artif. Intell. Rev. (in press).

Méndez, M., Montero, C., Núñez, M., 2022b. Using deep transformer based models to predict ozone levels. In: 14th Asian Conference on Intelligent Information and Database Systems, ACIIDS'22, LNAI 13757. Springer, pp. 169–182.

2022. Netron open source tool. https://netron.app/, accessed: 2022-06-30.

Ozcan, A., Catal, C., Donmez, E., Senturk, B., 2023. A hybrid DNN–LSTM model for detecting phishing URLs. Neural Comput. Appl. 35, 4957–4973. http://dx.doi.org/10.1007/s00521-021-06401-z.

Passricha, V., Aggarwal, R.K., 2020. A hybrid of deep CNN and bidirectional LSTM for automatic speech recognition. J. Intell. Syst. 29 (1), 1261–1274. http://dx.doi.org/10.1515/jisys-2018-0372.

Peiris, A.T., Jayasinghe, J., Rathnayake, U., 2022. Forecasting electricity power generation of Pawan Danawi wind farm, Sri Lanka, using Gene Expression Programming. Appl. Comput. Intell. Soft Comput. 2022, 7081444. http://dx.doi.org/10.1155/2022/7081444.

Peng, H., Du, B., Liu, M., Liu, M., Ji, S., Wang, S., Zhang, X., He, L., 2021. Dynamic graph convolutional network for long-term traffic flow prediction with reinforcement learning. Inform. Sci. (ISSN: 0020-0255) 578, 401–416. http://dx.doi.org/10.1016/j.ins.2021.07.007, URL https://www.sciencedirect.com/science/article/pii/S0020025521006976.

Poonia, P., Jain, V.K., 2020. Short-term traffic flow prediction: Using LSTM. In: 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3). pp. 1–4. http://dx.doi.org/10.1109/ICONC345789.2020.9117329.

Rhanoui, M., Mikram, M., Yousfi, S., Barzali, S., 2019. A CNN-BiLSTM model for document-level sentiment analysis. Mach. Learn. Knowl. Extract. (ISSN: 2504-4990) 1 (3), 832–847. http://dx.doi.org/10.3390/make1030048, URL https://www.mdpi.com/2504-4990/1/3/48.

Rick, R., Berton, L., 2022. Energy forecasting model based on CNN-LSTM-AE for many time series with unequal lengths. Eng. Appl. Artif. Intell. (ISSN: 0952-1976) 113, 104998. http://dx.doi.org/10.1016/j.engappai.2022.104998, URL https://www.sciencedirect.com/science/article/pii/S0952197622001889.

Roldán, J., Boubeta-Puig, J., Martínez, J.L., Ortiz, G., 2020. Integrating complex event processing and machine learning: An intelligent architecture for detecting IoT security attacks. Expert Syst. Appl. 149, 113251. http://dx.doi.org/10.1016/j.eswa.2020.113251, 1–22.

Semlali, B.B., Amrani, C.E., Ortiz, G., Boubeta-Puig, J., de Prado, A.G., 2021. SAT-CEP-monitor: An air quality monitoring software architecture combining complex event processing with satellite remote sensing. Comput. Electr. Eng. 93, 107257. http://dx.doi.org/10.1016/j.compeleceng.2021.107257.

Shubhangi, D., Pratibha, A., 2021. Asthma, alzheimer's and dementia disease detection based on voice recognition using multi-layer perceptron algorithm. In: 2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems. ICSES, pp. 1–7. http://dx.doi.org/10.1109/ICSES52305.2021.9633923.

Siami-Namini, S., Tavakoli, N., Namin, A.S., 2019. The performance of LSTM and BiLSTM in forecasting time series. In: 2019 IEEE International Conference on Big Data (Big Data). pp. 3285–3292. http://dx.doi.org/10.1109/BigData47090.2019.9005997.

Song, J.W., Park, Y.I., Hong, J.J., Kim, S.G., Kang, S.J., 2021. Attention-based bidirectional LSTM-CNN model for remaining useful life estimation. In: 53rd IEEE Int. Symposium on Circuits and Systems, ISCAS'21. IEEE, pp. 1–5. http://dx.doi.org/10.1109/ISCAS51556.2021.9401572.

Soon, K.L., Lim, J.M.-Y., Parthiban, R., 2019. Extended pheromone-based short-term traffic forecasting models for vehicular systems. Eng. Appl. Artif. Intell. (ISSN: 0952-1976) 82, 60–75. http://dx.doi.org/10.1016/j.engappai.2019.03.017, URL https://www.sciencedirect.com/science/article/pii/S0952197619300673.

Wang, Y., Li, L., Xu, X., 2017. A piecewise hybrid of ARIMA and SVMs for short-term traffic flow prediction. In: Liu, D., Xie, S., Li, Y., Zhao, D., El-Alfy, E.M. (Eds.), Neural Information Processing - 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part V. In: Lecture Notes in Computer Science, vol. 10638, Springer, pp. 493–502. http://dx.doi.org/10.1007/978-3-319-70139-4_50.

Wang, Z., Su, X., Ding, Z., 2021. Long-term traffic prediction based on LSTM encoder-decoder architecture. IEEE Trans. Intell. Transp. Syst. 22 (10), 6561–6571. http://dx.doi.org/10.1109/TITS.2020.2995546.

Zameer, A., Arshad, J., Khan, A., Raja, M.A.Z., 2017. Intelligent and robust prediction of short term wind power using genetic programming based ensemble of neural networks. Energy Convers. Manage. 134, 361–372. http://dx.doi.org/10.1016/j.enconman.2016.12.032.

Zarei, N., Ghayour, M.A., Hashemi, S., 2013. Road traffic prediction using context-aware random forest based on volatility nature of traffic flows. In: Selamat, A., Nguyen, N.T., Haron, H. (Eds.), Intelligent Information and Database Systems - 5th Asian Conference, ACIIDS 2013, Kuala Lumpur, Malaysia, March 18-20, 2013, Proceedings, Part I. In: Lecture Notes in Computer Science, vol. 7802, Springer, pp. 196–205. http://dx.doi.org/10.1007/978-3-642-36546-1_21.

Zhang, Y., Qiao, S., Ji, S., Li, Y., 2020. DeepSite: bidirectional LSTM and CNN models for predicting DNA–protein binding. Int. J. Mach. Learn. Cybern. 11, 841–851. http://dx.doi.org/10.1007/s13042-019-00990-x.

Zhang, Y., Yang, Y., Zhou, W., Wang, H., Ouyang, X., 2021. Multi-city traffic flow forecasting via multi-task learning. Appl. Intell. 51, 1–19. http://dx.doi.org/10.1007/s10489-020-02074-8.

Zhu, J.Z., Cao, J.X., Zhu, Y., 2014. Traffic volume forecasting based on radial basis function neural network with the consideration of traffic flows at the adjacent intersections. Transp. Res. C (ISSN: 0968-090X) 47, 139–154. http://dx.doi.org/10.1016/j.trc.2014.06.011, URL https://www.sciencedirect.com/science/article/pii/S0968090X14002010.