Computers & Industrial Engineering
Volume 157, July 2021, 107371

Prediction of time series using an analysis filter bank of LSTM units

Jose Mejia [a], Liliana Avelar-Sosa [b], Boris Mederos [c], Everardo Santiago Ramírez [a], José David Díaz Roman [a]

Show more

Add to Mendeley Share Cite

https://doi.org/10.1016/j.cie.2021.107371 Get rights and content

Highlights

- A combination of LSTM and convolutional layers in an architecture similar to a filter bank is proposed.
- The proposed architecture performs better when the time series is noisy.

*Manuscript
Click here to view linked References

Prediction of time series using an analysis filter bank of LSTM units

__[1]

—-

Abstract

Time series emerge in various applications such as financial data and production data, however, most of the generated data exhibit nonlinear inter-dependency between samples and noise, making necessary the development of methods capable of handling such nonlinearities and other abnormalities. In this paper we present an architecture for prediction of time series embedded in noise. The proposed architecture combines a convolutional and long short term memory (LSTM) layers into a structure similar to an analysis filterbank of two channels. The first element of each channel is a convolutional layer followed by a LSTM, which is able to find temporal dependencies of the signal. Finally the channels are summed to obtain a prediction. We found that the frequency response of the filters resemble a complementary filter bank response, with each channel having a maximum at different bands which could suggest that it characterizes the incoming signal in frequency. Comparisons with other methods demonstrate that the proposed method offer much better results in terms of different error measures.

Keywords: time series, LSTM, filter bank

1. Introduction

Time series arises in several branches of science and technology such as: financial data, sensor networks [1], weather records, industrial observations, and

☆Fully documented templates are available in the elsarticle package on CTAN.
[1]_.

many other sources. However, most of the generated data by these applications exhibit nonlinear inter-dependency between samples, and measurements are often contaminated by noise coming from the sensor or the environment where the measurements were made. As a result, nonlinear approaches are often required for the analysis and forecasting. Therefore, the methods used for the analysis must be robust to noise or outliers that contaminate the data, thus, making crucial the development of methods capable of handling such ailments and nonlinearities.

Several algorithms for time series analysis and forecasting have been proposed in the literature as pointed out in [2, 3]; conventional approaches include autoregressive models such as ARMA, ARIMA [4], and hidden markov models [5]. However, with the recent interest in deep neural networks [6], the exploration of methods for time series forecast using new network architectures has been increased, especially the use of LSTM [7], because of their ability to capture the long-term and short-term dependencies in a sequence. This type of networks has been successfully applied to problems in natural language processing [8, 9], recognition of handwritten sequences [10], and electric power forecasting [11].

However, the modeling of long sequences, such as documents or physiological signals, requires that the LSTM network keeps dependencies between elements of the series for a long period of time and some important features could be lost in the process [9].

One approach to overcome this problem is through a multiscale analysis [12, 13, 14, 9], this allows the analysis of important features at multiple scales of time, and reduces dependencies intervals. For this end, the time series is decomposed in a hierarchy of new time series that are easier to model and predict, separating the fast dynamics from the slow ones and facilitating the analysis of long range correlations [15].

In this work we propose a multiscale network based on an LSTM architecture as the main prediction element, and preceded by convolutional layers. We hope that, in this configuration the filters of the convolutional layers make the network more immune to noise and to outliers in the data. In the literature there is the use of LSTM in conjunction with convolutional networks [16], however, the network topology is adapted to data in two dimensions or images. In contrast, in this work a convolutional network is introduced as a filter of adaptive coefficients to the bandwidth of the signal.

The rest of the document is organized as follows: section 2 offers an introduction of neural networks, section 3 explains the proposed methodology, section 4 shows the results obtained, and finally section 5 offers conclusions of this work and future directions.

2. Neural networks

This section offers a brief introduction to the network architectures used in the proposed scheme. For a more detailed treatment of this, the reader can consult [17] and [18].

2.1. Convolutional neural networks

Convolutional neural networks (CNN) were introduced in [19, 20] for the recognition of handwritten characters. These networks are fed with data organized as an uniform grid or matrix. Such a grid can be of one or several dimensions, and data can consist of signals in one dimension or higher dimension data such as images. CNNs are inspired by the visual cortex of the human brain, where there are specific areas that are only sensitive to certain features of the input, for example, some neurons in some specialized area are only excited by vertical edges. Architectures using CNNs, generally consist of one or more convolutional layers, followed by a pooling layer, which is used to compress data without loosing much information.

2.2. LSTM

The LSTM model proposed in [7] arises as a solution to the problem of decaying error during the training of existing recurrent neural networks (RNN). The problem was exposed in the works [21, 22], in which it was found that during training, the gradient dissipates exponentially or explodes. Thus, in practice, RNNs have difficulty learning long-term dependencies in the data. The LSTM, are a special type of RNN, capable of learning long term dependencies. This type of recurrent neural networks resolves the extreme values of the gradient experienced by classical RNNs, through the use of multiplicative gates that impose a constant error flowing through internal states in the cells that make up the LSTM network [23].

Unlike a conventional RNN neuron, the LSTM cell has a more complex mechanism with an additional state $C^{(t)}$, that keeps information at the time t. Such state can be modified through gates that interact linearly with the current state, these gates are composed of feedforward layers followed by nonlinear operations. There exist different types of gates to modify and control the state of the cell $C^{(t)}$: the forget gate, decides what information is discarded from the state of the cell, using the past output and the current input, this gate establishes or attenuates components of $C^{(t)}$ by multiplying them with values obtained through its activation function; another gate, the external input gate, decides what new information to store in the cell's state; finally a output gate controls the output of the LSTM cell, using the input, past output, and the current state $C^{(t)}$, see [18] for details.

3. Proposed scheme

Given a time series, represented by $X = \{x(1), x(2), ..., x(n)\}$, the prediction problem consists in obtaining a future value $x(n+1)$, a challenging task, due to undesirable disturbances in the data, such as noise or outliers. A common way of dealing with such unwanted local fluctuations is through the application of filters, particularly finite-response filters of linear phase [4]. The classic filters used are generally of the moving averaging type.

In this work, we propose a network that uses LSTM units as main prediction elements, and temporal convolutional layers as filters for the rejection of noise and outliers in the data. Temporal convolutions have proven to be effective as feature extractors[24, 25].

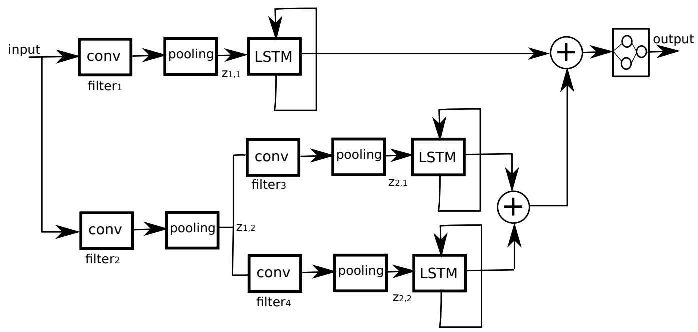


Figure 1: Diagram of a memory cell of an LSTM network

In the literature, it is common to use LSTM units in conjunction with convolutional networks such as the works in [16], where the network topology is adapted to two-dimensional data and embedded in the LSTM units, and in [25], where the scheme is used for classification tasks. In contrast, in this work the convolutional networks are introduced as processing filters of adaptive coefficients to the bandwidth of the signal, as preprocessing elements and not just for extracting features that are feed to the LSTM units.

In addition, it is expected that, by placing several filters and operators of pooling in a topology similar to the filter banks, this scheme allows to the different LSTM units to analyze different aspects of the signal.

The proposed scheme is show in Figure 1, where a decomposition of the signal through a network architecture similar to a bank of filters is introduced [26]. The filters of each phase are implemented through a temporal convolutional network. For operation, the convolutional network contains a single filter, in this case the filter is not restricted to a linear phase filter and it is expected that the training process determines the optimum filter coefficients, so that for the filter to be capable of eliminating the noise and anomalous values, and at the same time retain essential information for the prediction. Down sampling operations are implemented by pooling layers. After passing through a filter and a pooling stage, the signal is analyzed by an LSTM structure. We expected that this configuration makes the network more robust to noise and outliers in the data. We also expected that each filter is, in some way, complementary in the cases of the filters at the same decomposition level. This allows each LSTM unit to analyze a different aspect of the signal to be predicted. It is also expected that at each subsequent level, the signal will be easier to analyze and predict as in [26]. For this study, an architecture that breaks down the signal into two levels was used, implementing a poliphase scheme by down-sampling the even components of the input signal for the first phase, and the odd components for the other phase, similar to [27]. The first level of the proposed architecture is given by

$$Z_{1,1} = (\downarrow 2)_{even}(X * filter_1), \tag{1}$$

and

$$Z_{1,2} = (\downarrow 2)_{odd}(X * filter_2), \tag{2}$$

where $filter_1$ and $filter_2$ are the filters or kernels of their respective convolution layer, and $(\downarrow 2)_{odd}$ is the downsampling by two operation on odd coefficients of the input. The second level is given by the following expressions

$$Z_{2,1} = (\downarrow 2)_{even}(Z_{1,2} * Filter_3), \tag{3}$$

and

$$Z_{2,2} = (\downarrow 2)_{odd}(Z_{1,2} * Filter_4), \tag{4}$$

where $filter_3$ and $filter_4$ are the filters in the second level. The signals $Z_{1,1}$,$Z_{2,1}$, and $Z_{2,2}$ are fed to the LSTM units, expecting that each filter to be sensitive only to certain characteristics in the input and that the LSTMs can more easily find time dependencies in the time series. Finally, the network has a dense layer with linear activation as output, which is used to output the predicted value based in the LSTM outputs.

3.1. Performance measures

To evaluate the adaptability of the proposed model, the method used in [28] is followed, we used one-step-ahead forecasting for experimentation. One-step-ahead predictions, consisted in the forecasting of values one step at a time and then, for the next prediction, the actual and past values are used. The criteria used to evaluate the performance of the proposed scheme consist of the following metrics: The root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), direction accuracy (DA). These metrics are described in [28, 29].

4. Results

In this section, the results of a series of experiments for validation and comparison of the proposed algorithm are presented. The methods evaluated are ARIMA, a dense feed forward network (NN) and a simple LSTM network. The used ARIMA model consists of a four order AR term and a seven order AM term, two non-seasonal differences were used to approximate stationarity, the coefficients for the term were determined by a conjugate direction method [30]. The NN consists of three layers, the first two of them have three units with ReLu, and the output layer one linear unit. The LSTM consists of a LSTM layer with four neurons and a dense output layer with one linear unit. For the proposed architecture implementation, the number of filters of each convolutional layer was fixed to one; we used for $filter_1$ a size of 23, for $filter_2$ a size of 10, and a size of 2 for $filter_3$ and $filter_4$; at the output we used two dense layers of five and one neurons respectively. The first dense layer used a ReLu activation function while the last layer used a linear activation. All models, except ARIMA, were trained with the method of [31], with a mean squared error loss function, the number of epochs and number of units were determined by grid search. We perform three experiments: the first of these is a comparison between the methods operating on clean data, the second is aimed to verify the performance of the proposed method under samples corrupted by noise, and the third is to observe the response of the model when a variable number of samples is used to make predictions. The experiments were conducted in a computer with Pentium core i7 Dual-Core CPU T4400 @2.20GHz, with 5.8 GB of RAM and all methods were implemented using the Keras [32] library and python.

4.1. Comparisons on clean data

For this experiment, a database of samples of air quality used in [33] was employed. This database contains 9358 samples of responses averaged by time of five metal oxide chemical sensors embedded in an air quality sensing device. The device was located in a significantly contaminated area of an Italian city. The registration of the data started in March 2004 until February 2005. For more details regarding the database, the reader can consult [33].

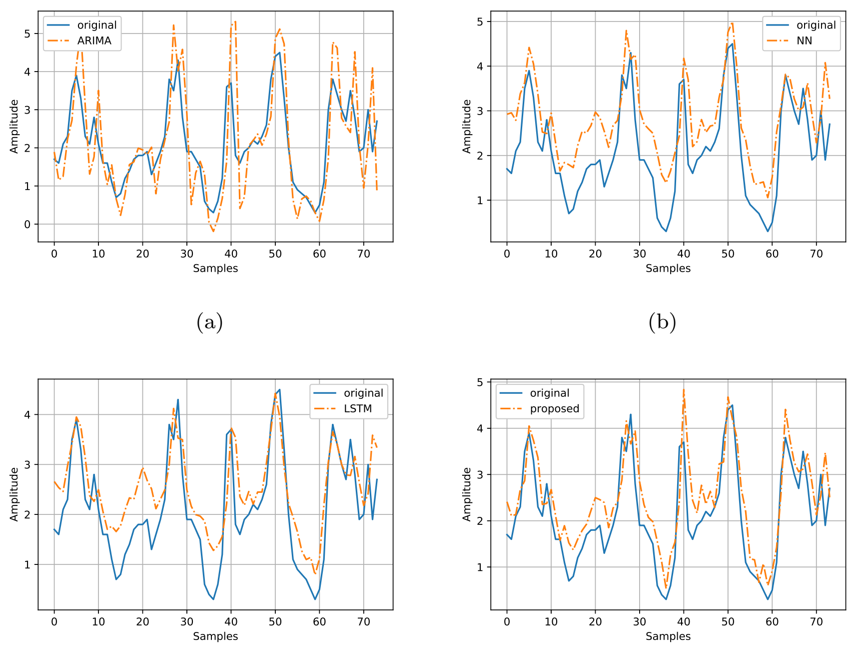
In all the experiments we take a set of 24 samples to make predictions, this value may seen somewhat random, however, a value not too far from a real application if we take it as if it were a prediction after 24 hours of sampling or a 24 day. In this section, we conduct a comparative study among simple LSTMs, conventional methods ARIMA, neural networks (NN), and the proposed model. The experiment consists of out-of-sample forecasting over an interval of 74 air quality samples, where each input sample consists of a fixed 24 samples-length mobile window along the time series. Previously, the methods were fit in a set of 300 samples of the air quality data.

The results for out-of-sample forecasting performance are summarized in Table 1, we can see that proposed method obtains better predictive accuracy than the other methods in most of the criteria evaluated, except in the fifth column where prediction with NN obtains a better result when the DA metric is used.

In Figure 2 we show the forecasting curves for each of the evaluated methods, it can be seen that globally all methods evaluated follow the actual data, although none of the methods has an exact prediction in the whole interval, also, it can be seen that the proposed method is able to follow more accurately the original curve, this observation confirms the quantitative results in Table 1. In Figure 3, we present another segment of the time series but with about half of the samples shown in the figure 2, in order to show more detail of the predictions attained by each method, once again it can be seen that the proposed method follows more accurately the original time series.

Table 1: Results of forecasting performance

	RMSE	MAE	MAPE	DA
ARIMA	2.11	1.88	89.37	0.62
NN	0.67	0.55	41.69	**0.69**
LSTM	0.72	0.60	51.63	0.68
proposed	**0.59**	**0.49**	**34.34**	0.67

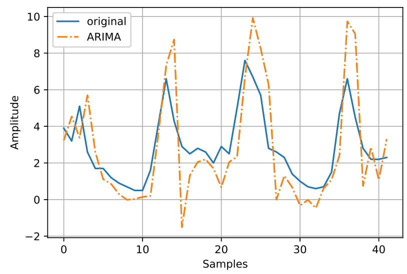


(a) (b) (c) (d)

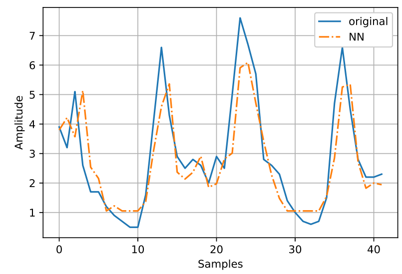
Figure 2: Results with the different methods on a set of the air quality database a) ARIMA b) NN, c) LSTM, and d) proposed.

4.2. Comparisons with noisy data

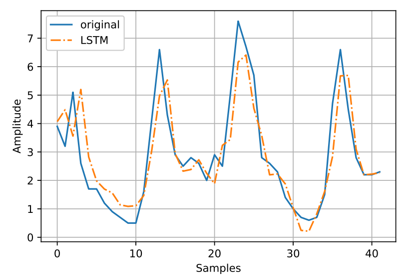
The second experiment is focused on knowing the response of the algorithm when there is a noisy signal. For this end, we add different levels of noise to the time series used in section 4.1. Gaussian noise with known variances



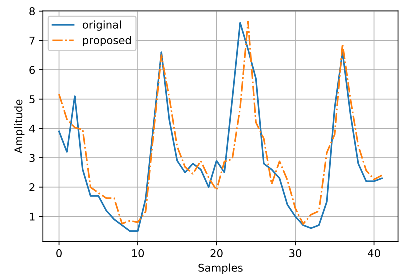
(a)



(b)



(c)



(d)

Figure 3: Results with the different methods on a set of the air quality database a) ARIMA b) NN, c) LSTM, and d) proposed.

of 0.3 to 1.0 were used; we begin with a variance of 0.3 since less amplitudes make no much different in the results as compared with zero noise. Figure 4 depicts the time series with added noise of 0.75. Predictions with the different methods where taken using a 24 samples-length mobile window of the noisy time series. Then, the results were evaluated using the metrics. Figure 5 shows how each method behaves as the noise increases, it can be seen that the LSTM and NN approaches loss accuracy as the noise increases, this could happen because an overfitting to the actual series. Surprisingly the ARIMA method has better performance than LSTM and NN, since there is not overfitting and it just follows the trend of the current series. In general, as the noise amplitude is increased, the error of prediction, RMSE, MAE, and MAPE, on all methods also increase. However, the error rate of the proposed method increases slower than in the

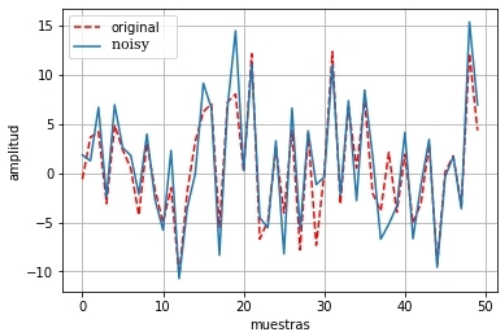
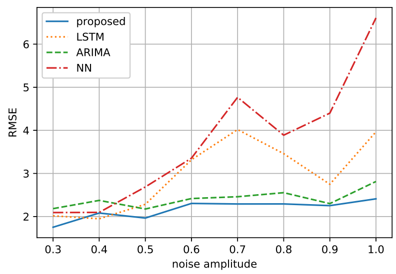


Figure 4: Actual time series show in red, while the solid blue line represents the time series with noised of 0.75 (only a 50 samples interval is show).

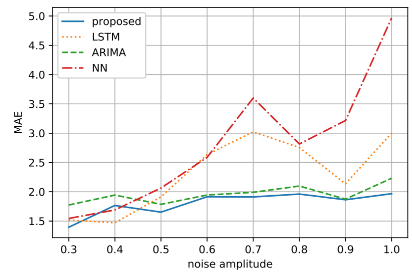
other methods, which implies that the proposed method is less affected when making predictions in the presence of noise. This behavior, could be explained because of the convolutional input layer which behaves as a filter to the noise, as we noted in Section 4.4.

4.3. Variable mobile window length

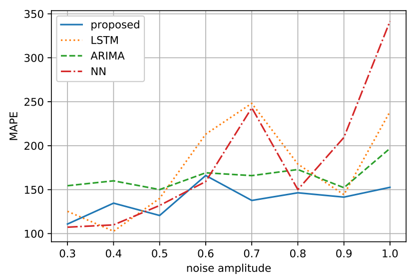
This section presents the sensitivity of the model to the number of samples in the mobile window used to predict. For the experiments we used the data from a time series representing the hourly German electricity spot prices, this data was taken from the Open power system data platform (*http://open-power-system-data.org*). For evaluation and comparisons we employ the same methods and performance metrics used in Section 4.2. The samples in the mobile window were chosen as 10, 20, 35, 40, 55, 60, 72, 100. The results obtained are shown in Table 2, it can be seen that in all the evaluated methods, there is a tendency to increase the prediction error as the size of the window gets larger. This may be due to the great variability of the data, thus, when taking a smaller sample the process is more local and with fewer variations, while for a longer window, the variability increases. Despite this, the proposed method performs better than the rest of the methods, this may be due to the fact that the different LSTM networks, because of their long-term memory, tend to adapt better than simple networks.



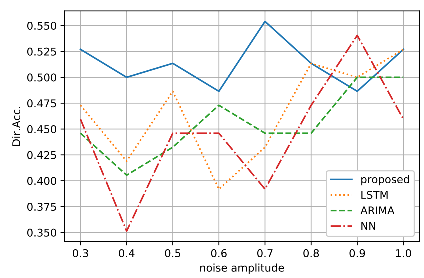
(a)



(b)



(c)



(d)

Figure 5: Results with the different methods on a noisy time series a) RMSE b) MAE, c) MAPE, and d) DA.

The prediction of the time series for the different methods, using a mobile window length of 55 samples, is shown in Figure 6.

4.4. Frequency response of the model

It is interesting to observe the frequency response of the filters. Figure 7 depicts the frequency response of the convolution filters. Filters 1 and 2 have a quasi complementary response, where filter 2 has a response resembling a stopband at 1.5 radians, while filter 1 has its peak at that frequency. The same effect occurs in filters 3 and 4. This could explain why the propose scheme has a better performance predicting from a noisy time series, since the signal could be denoised before arriving to the LSTM layers. Thus some channels seem to process characteristics with different frequency components. In Figure 8 we show the frequency response of a portion of 24 samples of the generated noise

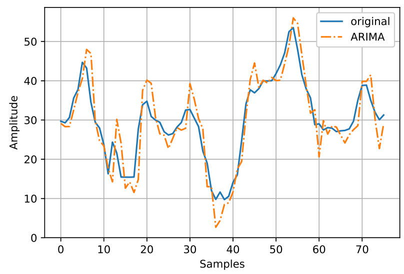
Table 2: Results of forecasting performance for different mobile window lengths on the electricity prices series.

		mobile window length							
Algorithm	Metric	10	20	35	40	55	60	72	100
ARIMA	RMSE	5.42	4.63	9.93	9.91	9.96	10.19	10.19	10.92
	MAE	4.17	3.48	7.70	7.61	7.58	7.80	7.62	8.18
	MAPE	17.48	14.49	34.77	35.36	32.62	34.03	32.80	19.64
	DA	0.66	0.68	0.02	0.01	0.01	0.01	0.01	0.00
NN	RMSE	3.96	4.14	3.55	**3.56**	3.23	3.97	3.45	13.28
	MAE	2.93	3.30	2.70	**2.72**	**2.52**	3.09	2.90	10.92
	MAPE	12.02	14.55	**10.30**	**10.54**	**9.10**	11.52	10.85	27.34
	DA	0.68	**0.65**	0.64	0.66	0.71	0.72	0.66	**0.00**
LSTM	RMSE	4.57	4.39	**3.47**	4.66	4.66	3.83	4.64	10.82
	MAE	3.64	3.60	**2.69**	3.72	3.77	3.29	3.85	9.85
	MAPE	14.69	15.59	10.82	13.45	15.62	12.96	13.32	26.65
	DA	0.69	0.68	0.70	0.68	0.71	0.73	0.71	0.58
Proposed	RMSE	**3.91**	**3.87**	5.10	5.15	**3.19**	**3.76**	**3.40**	**6.04**
	MAE	**2.82**	**3.00**	4.13	4.35	2.61	**3.02**	**2.76**	**5.05**
	MAPE	**11.82**	**12.79**	15.21	16.28	9.73	**11.49**	**10.18**	**13.52**
	DA	0.67	0.68	0.69	0.71	0.66	0.77	0.71	0.58

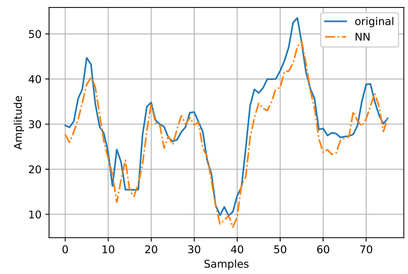
(variance of 0.4), it can be seen that there are frequency bands less affected by noise than other bands in the response.

5. Conclusions

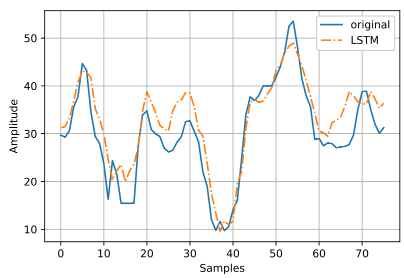
An algorithm for the prediction of time series was presented through a combination of LSTM and a convolutional networks. The architecture of the proposed network behaves similar to a filter bank. In this case, the filters adapt to the signals according to the training set by adjusting the coeficients of the convolution layers. It is expected that each of the filters that make up the network extracts different characteristics of interest of the signal, so that later these characteristics can be used by the LSTM network to predict the signal. Noise-contaminated data sets were used to train the network and compared with the prediction obtained by a pure LSTM network. When the original signal does not contain noise, both networks, LSTM and our proposed approach, behave in a similar way, however, as the original signal becomes more contaminated, the proposed network is capable of making a better prediction.



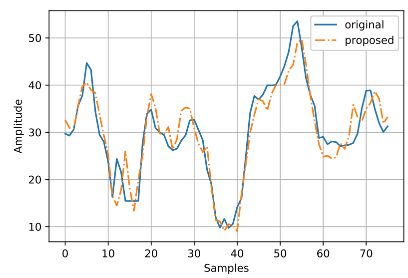
(a)



(b)

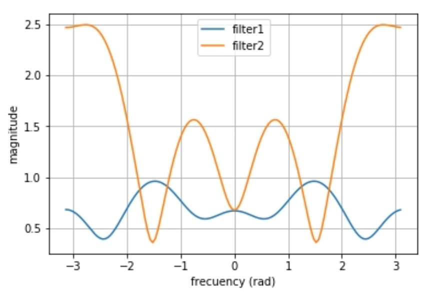


(c)

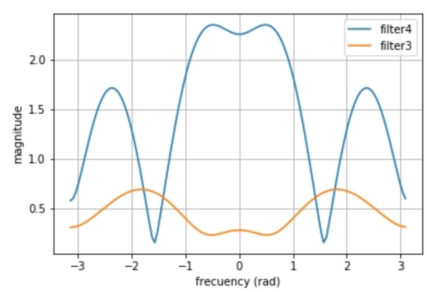


(d)

Figure 6: Results with the different methods on the data of the hourly German electricity spot prices a) ARIMA b) NN, c) LSTM, and d) proposed.



(a)



(b)

Figure 7: Frequency response of the convolutional filters in the proposed architecture. a) response of the input filters (*1 and 2 from Figure 1*) and b) response of the second channel filters (*3 and 4 from Figure 1*).

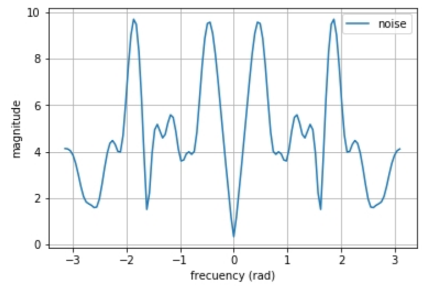


Figure 8: Frequency response of a realization of the noise generated. It can be seen that there are frequency bands, like the band from 0.7 to 1.0 rads and the DC band, which are less affected by noise

Thus the proposed model has several attributes that make it suitable for the analysis and subsequent prediction of series over time:

- A multiscale decomposition in terms of the characteristics of the signal. This is possible using several channels for the input signal, in each channel a filter of different size is used, thus each filter obtain characteristics of the signal at different sizes, a channel with a larger filter size will obtain global trends in the signal, while a smaller filter will look for more local characteristics.

- Rejection of noise. The use of filters before each LSTM layer could filter some of the noise of the signal, as observed in the frequency analysis section (4.4). This also agrees with the experiments carried out, when the noise was gradually increased, the proposed model obtained the least error with respect to the other methods presented.

- The inherited properties of the recurrent models, which permits to represent the sequences with hidden states and nonlinear functions, allows to the proposed model to represent complex sequences without the necessity to compensate for non stationarity.

- Forecasts accuracy is maintained even when few observations are used to output a prediction.

As future work, it is planned to analyze the behavior of the network using more subband filters, in addition, to explore using convolutional filters to add a synthesis stage and exploring deeper architectures.

References

References

[1] D. Tulone, S. Madden, Paq: Time series forecasting for approximate query answering in sensor networks, in: European Workshop on Wireless Sensor Networks, Springer, 2006, pp. 21–37.

[2] M. Längkvist, L. Karlsson, A. Loutfi, A review of unsupervised feature learning and deep learning for time-series modeling, Pattern Recognition Letters 42 (2014) 11–24.

[3] Z. H. Rahimi, M. Khashei, A least squares-based parallel hybridization of statistical and intelligent models for time series forecasting, Computers & Industrial Engineering 118 (2018) 44–53.

[4] C. Chatfield, The analysis of time series: an introduction, CRC press, 2016.

[5] B. H. Juang, L. R. Rabiner, Hidden markov models for speech recognition, Technometrics 33 (3) (1991) 251–272.

[6] A. Krizhevsky, I. Sutskever, G. E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in neural information processing systems, 2012, pp. 1097–1105.

[7] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[8] F. A. Gers, E. Schmidhuber, Lstm recurrent networks learn simple context-free and context-sensitive languages, IEEE Transactions on Neural Networks 12 (6) (2001) 1333–1340.

[9] P. Liu, X. Qiu, X. Chen, S. Wu, X. Huang, Multi-timescale long short-term memory neural network for modelling sentences and documents, in: Proceedings of the 2015 conference on empirical methods in natural language processing, 2015, pp. 2326–2335.

[10] A. Graves, M. Liwicki, S. Fernández, R. Bertolami, H. Bunke, J. Schmidhuber, A novel connectionist system for unconstrained handwriting recognition, IEEE transactions on pattern analysis and machine intelligence 31 (5) (2009) 855–868.

[11] A. Gensler, J. Henze, B. Sick, N. Raabe, Deep learning for solar power forecastingan approach using autoencoder and lstm neural networks, in: Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on, IEEE, 2016, pp. 002858–002865.

[12] S. Soltani, On the use of the wavelet decomposition for time series prediction, Neurocomputing 48 (1-4) (2002) 267–277.

[13] M. Costa, A. L. Goldberger, C.-K. Peng, Multiscale entropy analysis of complex physiologic time series, Physical review letters 89 (6) (2002) 068102.

[14] M. A. Ferreira, M. West, H. K. Lee, D. M. Higdon, et al., Multi-scale and hidden resolution time series models, Bayesian Analysis 1 (4) (2006) 947–967.

[15] G. Ercolano, D. Riccio, S. Rossi, Two deep approaches for adl recognition: A multi-scale lstm and a cnn-lstm with a 3d matrix skeleton representation, in: Robot and Human Interactive Communication (RO-MAN), 2017 26th IEEE International Symposium on, IEEE, 2017, pp. 877–882.

[16] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, W.-c. Woo, Convolutional lstm network: A machine learning approach for precipitation nowcasting, in: Advances in neural information processing systems, 2015, pp. 802–810.

[17] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436.

[18] I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, Deep learning, Vol. 1, MIT press Cambridge, 2016.

[19] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, L. D. Jackel, Handwritten digit recognition with a back-propagation network, in: Advances in neural information processing systems, 1990, pp. 396–404.

[20] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.

[21] S. Hochreiter, Untersuchungen zu dynamischen neuronalen netzen, Diploma, Technische Universität München 91 (1991) 1.

[22] Y. Bengio, P. Simard, P. Frasconi, Learning long-term dependencies with gradient descent is difficult, IEEE transactions on neural networks 5 (2) (1994) 157–166.

[23] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, Long short term memory networks for anomaly detection in time series, in: Proceedings, Presses universitaires de Louvain, 2015, p. 89.

[24] Z. Wang, W. Yan, T. Oates, Time series classification from scratch with deep neural networks: A strong baseline, in: Neural Networks (IJCNN), 2017 International Joint Conference on, IEEE, 2017, pp. 1578–1585.

[25] F. Karim, S. Majumdar, H. Darabi, S. Chen, Lstm fully convolutional networks for time series classification, IEEE Access 6 (2018) 1662–1669.

[26] P. P. Vaidyanathan, Multirate digital filters, filter banks, polyphase networks, and applications: a tutorial, Proceedings of the IEEE 78 (1) (1990) 56–93.

[27] G. Strang, T. Nguyen, Wavelets and filter banks, SIAM, 1996.

[28] F. Shen, J. Chao, J. Zhao, Forecasting exchange rate using deep belief networks and conjugate gradient method, Neurocomputing 167 (2015) 243–253.

[29] H. Y. Kim, C. H. Won, Forecasting the volatility of stock price index: A hybrid model integrating lstm with multiple garch-type models, Expert Systems with Applications.

[30] M. J. Powell, An efficient method for finding the minimum of a function of several variables without calculating derivatives, The computer journal 7 (2) (1964) 155–162.

[31] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[32] F. Chollet, et al., Keras, `https://keras.io` (2015).

[33] S. De Vito, E. Massera, M. Piga, L. Martinotto, G. Di Francia, On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario, Sensors and Actuators B: Chemical 129 (2) (2008) 750–757.