Proceedings of the Federated Conference on Computer Science and Information Systems pp. 15–22
DOI: 10.15439/2020F211
ISSN 2300-5963 ACSIS, Vol. 21

Short-term air pollution forecasting based on environmental factors and deep learning models

Mirche Arsov, Eftim Zdravevski, Petre Lameski,
Ss Cyril and Methodius University,
Faculty of Computer Science and Engineering,
Skopje, North Macedonia
Email: mirche.arsov@gmail.com,
eftim.zdravevski@finki.ukim.mk,
petre.lameski@finki.ukim.mk

Roberto Corizzo,
Department of Computer Science
American University, Washington DC, USA
Email: rcorizzo@american.edu

Nikola Koteli, Kosta Mitreski, Vladimir Trajkovik
Ss Cyril and Methodius University,
Faculty of Computer Science and Engineering,
Skopje, North Macedonia
Email: nikola.koteli@hotmail.com,
kosta.mitreski@finki.ukim.mk,
vladimir.trajkovik@finki.ukim.mk

***Abstract*—The effects of air pollution on people, the environment, and the global economy are profound - and often under-recognized. Air pollution is becoming a global problem. Urban areas have dense populations and a high concentration of emission sources: vehicles, buildings, industrial activity, waste, and wastewater. Tackling air pollution is an immediate problem in developing countries, such as North Macedonia, especially in larger urban areas. This paper exploits Recurrent Neural Network (RNN) models with Long Short-Term Memory units to predict the level of PM10 particles in the near future (+3 hours), measured with sensors deployed in different locations in the city of Skopje. Historical air quality measurements data were used to train the models. In order to capture the relation of air pollution and seasonal changes in meteorological conditions, we introduced temperature and humidity data to improve the performance. The accuracy of the models is compared to PM10 concentration forecast using an Autoregressive Integrated Moving Average (ARIMA) model. The obtained results show that specific deep learning models consistently outperform the ARIMA model, particularly when combining meteorological and air pollution historical data. The benefit of the proposed models for reliable predictions of only 0.01 MSE could facilitate preemptive actions to reduce air pollution, such as temporarily shutting main polluters, or issuing warnings so the citizens can go to a safer environment and minimize exposure.**

***Index Terms*—RNN, LSTM, deep learning, air pollution**

I. Introduction

AIR pollutants exert a wide range of impacts on biological and socio-economic systems. Their effects on human health are of great interest. In particular, PM2.5 and PM10 (Particulate Matter) have been proven to have a significant impact on human respiratory efficiency. A number of studies have shown that the increase of respiratory diseases is correlated by a high concentration of air pollutants [1]. [2] represents a useful review of the emerging challenges and requirements for understanding adverse health outcomes from ambient particles. Consequently, it has become an important task to accurately track and analyze ambient air pollution in order to adjust public policies and health protection measures. The ability to predict exceeding of critical air quality thresholds is of particular importance. The potential for alert management systems that will provide warning communication to authorities and the population of health and environmental risks is high. Such systems have already been developed and deployed [3]. Studies such as [4], have shown that the data of ambient air quality can be modelled as stochastic time series, thereby making it possible to make a short-term forecast based on historical data. There are successful approaches relying on successful forecasting models over large multi-sensor data sets, based on sliding-window-based feature extraction and feature subset ensemble selection [5]–[9]. The latter approaches also show that it is feasible to use short-term predictions of dangerous concentrations in coal mines to reduce the workload, so preventing to reach the dangerous thresholds. The air pollution forecast in cities can be employed in a similar manner, such as temporarily shutting main polluters, or issuing warnings so the citizens can go to a safer environment and minimize exposure.

Long short-term memory (LSTM) [10] is an artificial recurrent neural network (RNN) [11] architecture used in the field of deep learning. Due to their chain-like nature, LSTMs are considered to be the typical architecture of neural networks to be used with sequences and lists. LSTM networks have already been used for time series multistep forecasting in multiple studies [12] [13] [14] [15] [16]. In [17] a similar approach is described, where convolutional neural networks are combined with LSTM to classify PM10 levels. In [18], an approach for air pollution forecasting using RNN with LSTM is presented. Alternative studies in the literature exploit feature extraction as a pre-processing step for the predictive task [19] [20] [21] [22] [23].

The widespread adoption of LSTM across different domains shows the effectiveness and reliability of this model in multistep forecasting task. The reason for their effectiveness is the ability to extract time-variant dependencies and correlations that are inherently present in real-life scenarios, and exploit them to predict future time steps. Differently than ARIMA models, which are autoregressive and capable to analyze exclusively univariate time series, LSTM models can exploit multiple time series in a combined manner. Potentially, leveraging the existing correlations between them can lead to obtain more accurate predictions.

In this paper, we performed experiments with air quality measurements data from the area of Skopje, North Macedonia. We used PM10 level measurements of the pollution combined with meteorological parameters to predict the PM10 level at point +3 hours in the future. The main contribution of this paper is that it combines different data sources to perform forecasting and compares the results to predictions when only air quality data is used. Our approach was to train the models with a data set from a single sensor, then gradually increase the number of air quality sensors used. A graphical representation of the workflow is shown in Figure 1. The results were then compared with the performance of the models trained using air quality and meteorological sensor data combined. Additionally, we used the data to examine the performance of different RNN architectures. For this purpose, we used the Keras framework, a high-level neural networks API capable of running on top of Tensorflow.

II. Methods

A. Dataset

The dataset consists of air quality sensor measurements from sensors deployed to different locations in the city of Skopje. Variety of parameters are monitored by the sensors, including PM10 and PM2.5 particles, as well as the presence of NO2, CO, O3, and SO2. Measurements are done in intervals of one hour. This dataset was also enriched with meteorological parameters, namely temperature and atmospheric pressure, measured at the Skopje-Petrovec meteorological station. For the purpose of this research, 12 consecutive measurements from the air pollution measurement points and from the meteorological station are used. In some of the models, we used a set of 24 consecutive measurements, but no considerable improvement was observed.

The dataset in this form has not been studied and published before. However, a subset of the data has been used in a previous study. In [17] a similar approach is described, where subset of the data is used to classify future values using a combination of LSTM and convolutional neural networks. Repository with the source code used, as well as the preprocessed dataset, is available at https://gitlab.com/magix.ai/air-pollution-skopje.

Fig. 2 shows the seasonality and trend in the data set. It is clearly noticeable that disturbances and irregularities are present in the air quality sensor data. Due to these reasons, to train the recurrent neural network models, we used data in the range from December 2011 to December 2019. In order to model possible malfunctions in the sensors, we introduced a Dropout layer in some of the architectures.

The used measurements are listed bellow, grouped by location:

- Municipality of Karposh, North Macedonia
  - PM10 concentration
- Municipality of Centar, North Macedonia
  - Measurement station Centar - PM10 concentration
  - Measurement station Rektorat - PM10 concentration
- Municipality of Miladinovci, North Macedonia
  - PM10 concentration
- Municipality of Petrovec, North Macedonia
  - Temperature (in Degrees Celsius)
  - Atmospheric Pressure at station level

The sampling frequency of the meteorological parameters differs from the one used in the air quality sensors. A pre-processing phase was needed to fit the data set for training and validation purposes. The pre-processing consists of the following steps:

- Missing data interpolation
- Min-Max normalization
- 12 samples data window preparation

The data was divided into train, validation and test data sets. For training, we used data in the time interval 01.12.2011 - 31.12.2019. This dataset consists of 70129 samples. Validation samples were taken dynamically as 1 per cent from the training data points (709 samples). Before the training process, we used a smaller two-year subset of the data for hyperparameter optimization. Data points for optimization were taken from the interval from 01.08.2014 to 01.08.2016 (17534 samples). Hyper-parameter tuning was validated using a small two-months data set in the time frame 01.11.2016 - 31.12.2016 (1430 samples).

For testing the performance of the different architectures, a test data set was used. This data set consists of the data points in January 2020.

B. Baseline model: ARIMA

Among many available methods for time series regression, one of the most popular and broadly used are Autoregressive integrated moving average (ARIMA) model [24]. Results obtained in this study confirmed that the ARIMA has a strong potential for short-term spot prediction. ARIMA form a class of time series models that are widely applicable in the field of time series forecasting. In the ARIMA model, the future value of a variable is a linear combination of past values and errors after removing the trend – by differencing.

C. Deep learning models architecture

In this paper, we wanted to compare several different architectures and see how they perform in comparison to the ARIMA model. We used LSTM, SimpleRNN and GRU layers. In some of the architectures, we added a dropout layer to mitigate temporary failures of some sensors.

RNN mainly deals with the processing of sequence data, such as text, speech, and time series. This type of data exists in an orderly relationship with each other; each piece of data is associated with the previous piece. Another example is climate data, where, for example, the temperature of a day is related to the temperature of the previous day. Therefore, we can form many sets of sequences from the data using time from a set of continuous data, and the correlation between sequences can be observed from multiple sets of sequences.

Our approach was to build a simple model using LSTM and Dense layers and then gradually increase the complexity of the

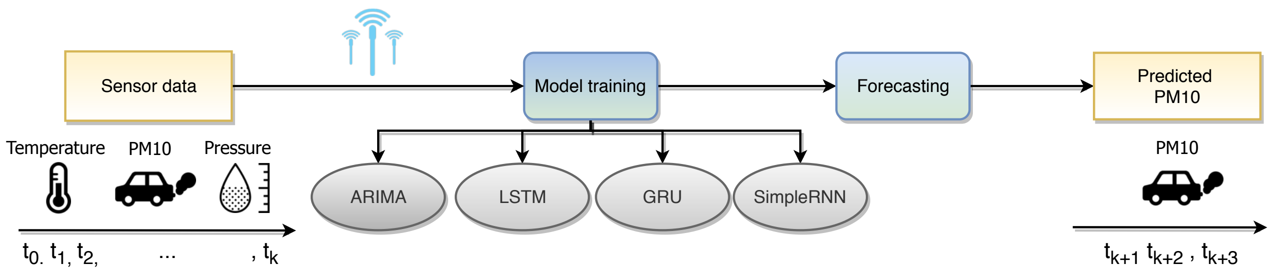

Fig. 1: Graphical representation of the proposed method for pollution forecasting.

architecture. An overview of the models is given in Table I-II. We started with the simplest architecture, one LSTM layers with optimized number of units and one Dense layer. Then, we trained the network using a univariate data set, values from one air quality sensor, and we validated the ability to make short-term predictions, +3 hours in the future.

This simple architecture performed slightly better than the ARIMA model. 3 shows the loss function for the training and validation phases while 4 show the performance of this model compared to the performance of the ARIMA model.

In order to improve the performance, we added data from a second sensor to treat the problem as a multivariate series and compared the performance of the architecture. Then we increased the data set to include data from 4 sensors on different locations (Table I, approach number 3 and 4). This caused a small decrease in the performance of the LSTM model. One of the main ideas of this paper was to investigate the influence of meteorological data. Therefore as a final test, we added temperature and air pressure parameters to the training data set (Table I, approach number 5). The result was an increase in accuracy and performance. Figure 4 shows the performance compared to the ARIMA model in the first week of the test data. This architecture showed the best performance when we compared the models using MSE as a performance metric. The simple LSTM model seems to outperform all other models described in this paper.

As a second approach in this paper, we increased the complexity of the architecture and the number of hidden layers. A second layer of optimized LSTM was added with a Dropout layer in between (Table I-II, approach number 6). This model showed a slightly decreased accuracy.

As a next step, we introduced a SimpleRNN layer in two variants. In [25], it is shown that RNN can be used for time series forecasting. SimpleRNN is a fully-connected RNN layer where the output is to be fed back to the input. As a first experiment, we replaced the first LSTM layer with a SimpleRNN, and in the second the RNN layer was added as an input to the first LSTM followed by a Dropout layer (Table I approach number 8 and 9). The latter approach exhibited a better performance than the first.

As an additional attempt to improve the results, the data set was extended to 24 hours history data window (Table I approach number 10). This did not bring any significant improvement. Gated Recurrent Unit or GRU [26] modifies the LSTM by fusing the forget and input gates into an update gate. Additionally, the cell states and hidden states are merged. The resulting model is simpler than standard LSTM models, and has been growing increasingly popular in the past few years. Due to sensor failures, the data set, as most of the real-life time series data, is characterized by a variety of missing values. It has been noted that missing values and their missing patterns are often correlated with the target labels. There are studies [27] that examine architectures based on GRU to time series data analysis. An experiment with a GRU layer was made. We extended the RNN architecture with a GRU layer followed by a SimpleRNN + LSTM, and a Dense layer as an output. This architecture showed a decreased performance in comparison to ARIMA and the previous architectures (Table I, approach number 7). During training and validation, the model showed improvements, such as faster learning (steeper decline in the loss function in the first couple of epochs). For the test data set, the accuracy decreased, which was an indicator that this architecture was overfitting to the training data set.

Due to the small number of features, we expected that additional layers could only increase the probability of overfitting on the data, although further research is experimentation is necessary to prove this.

For this particular experiment, we use mean squared error loss function, and for the model optimization, we used the Adam optimizer [28]. The implementation is done with Keras [29].

D. *Parameter tuning*

We used parameter tuning in order to obtain the best predictive model. For hyperparameter optimization, a smaller subset of the training data was used. Data points were taken in the interval from 01.08.2014 to 01.08.2016 (17534 samples). Hyper-parameter tuning was validated using a two-months validation data set in the time frame 01.11.2016 - 31.12.2016 (1430 samples). Table III presents the parameters that are tuned with the ranging values. Optimization was done using the Keras-Tuner library[1].

The following parameters were tuned in order to obtain the best architecture:

- *Dropout* - Deep neural networks with a large number of parameters can be powerful tools. However, overfitting

[1]https://keras-team.github.io/keras-tuner/

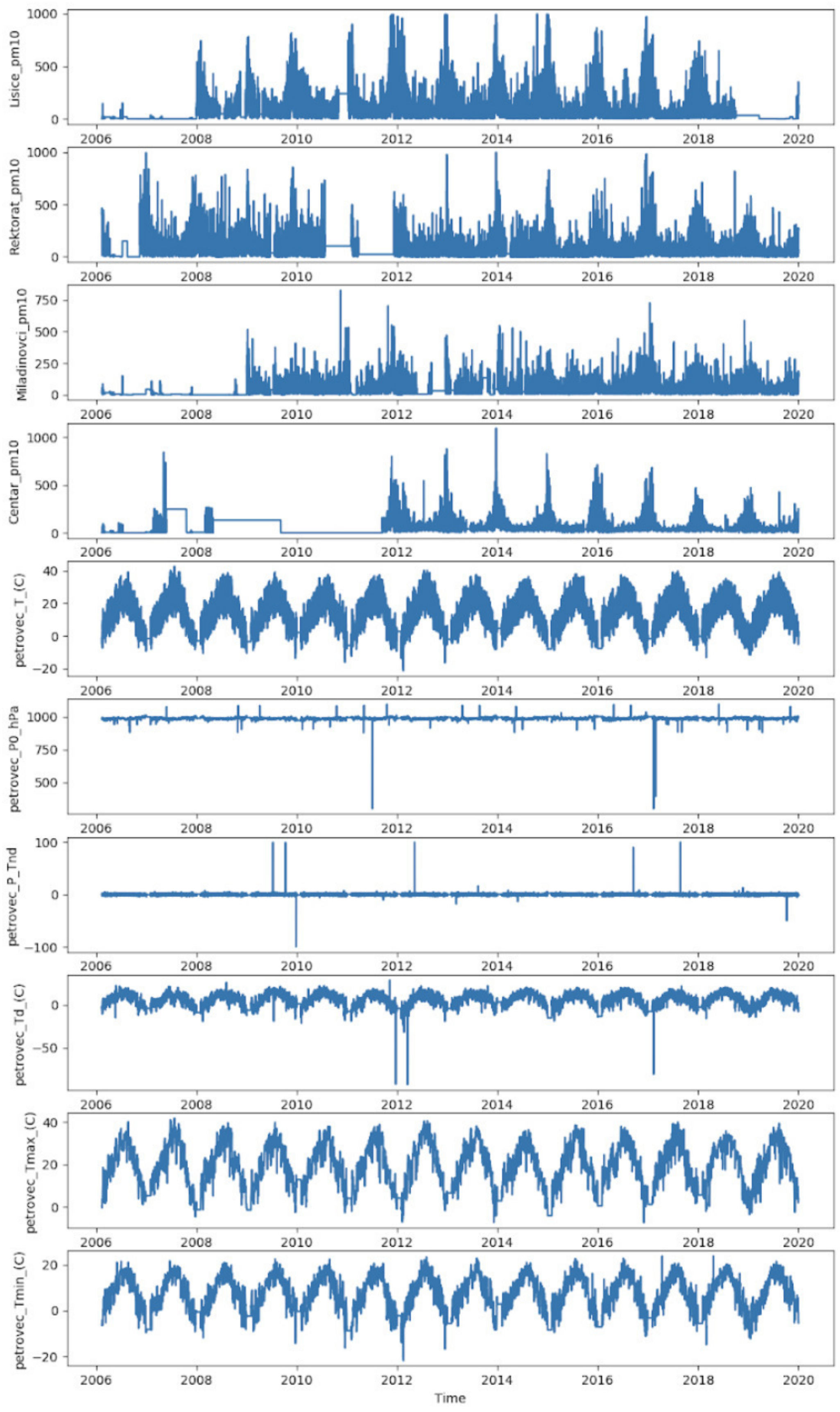

Fig. 2: Exploration of seasonality and trend in the dataset

TABLE I: Summary of evaluated approaches with achieved average forecasting performance in terms of Mean Square Error (MSE), Root Mean Square Error (RMSE), and percentage of improvement in terms of reduction in RMSE with respect to ARIMA.

#	Input	Data	Architecture	MSE	RMSE	% Impr.
1	1 x 12	PM10	LSTM + Dense	0.0140	0.1185	6.02
2	1 x 12	PM10	ARIMA (12)	0.0159	0.1261	/
3	2 x 12	2 x PM10	LSTM + Dense	0.0143	0.1198	4.90
4	4 x 12	4 x PM10	LSTM + Dense	0.0124	0.1114	11.60
5	6 x 12	4 x PM10 + Temp. + Pressure	LSTM + Dense	0.0109	0.1043	17.28
6	6 x 12	4 x PM10 + Temp. + Pressure	LSTM + Dropout + LSTM + Dense	0.0115	0.1072	14.90
7	6 x 12	4 x PM10 + Temp. + Pressure	GRU + SimpleRNN + LSTM + Dense	0.0428	0.2069	-6.40
8	6 x 12	4 x PM10 + Temp. + Pressure	SimpleRNN + LSTM + Dense	0.0150	0.1224	2.93
9	6 x 12	4 x PM10 + Temp. + Pressure	SimpleRNN + LSTM + Dropout + LSTM + Dense	0.0125	0.1118	11.34
10	6 x 24	4 x PM10 + Temp. + Pressure	SimpleRNN + LSTM + Dropout + LSTM + Dense	0.0127	0.1126	10.70

TABLE II: Summary of evaluated approaches with different configurations in terms of number of units in the LSTM layer (U), Learning Rate (LR), Dropout rates (D).

#	Architecture	Parameters optimized
1	LSTM + Dense	U: 2-128 + LR [0.01, 01]
2	ARIMA (12)	None
3	LSTM + Dense	U: 2-124 + Learning rates [0.01, 01]
4	LSTM + Dense	U: 2-124 + LR [0.01, 01]
5	LSTM + Dense	U: 2-124 + LR [0.01, 01]
6	LSTM + Dropout + LSTM + Dense	LSTM (2-24) + D [0.3, 0.2, 0.1] + LSTM (2-124) + LR [0.01, 01]
7	GRU + SimpleRNN + LSTM + Dense	GRU (12-256) + RNN (1-128) + LSTM (2-124) + LR [0.01, 01]
8	SimpleRNN + LSTM + Dense	RNN (1-128) + LSTM (2-124) + LR [0.01, 01]
9	SimpleRNN + LSTM + Dropout + LSTM + Dense	RNN (1-128) + LSTM (2-24) + D [0.3, 0.2, 0.1] + LSTM (2-124) + LR [0.01, 01]
10	SimpleRNN + LSTM + Dropout + LSTM + Dense	RNN (1-128) + LSTM (2-24) + D [0.3, 0.2, 0.1] + LSTM (2-124) + LR [0.01, 01]

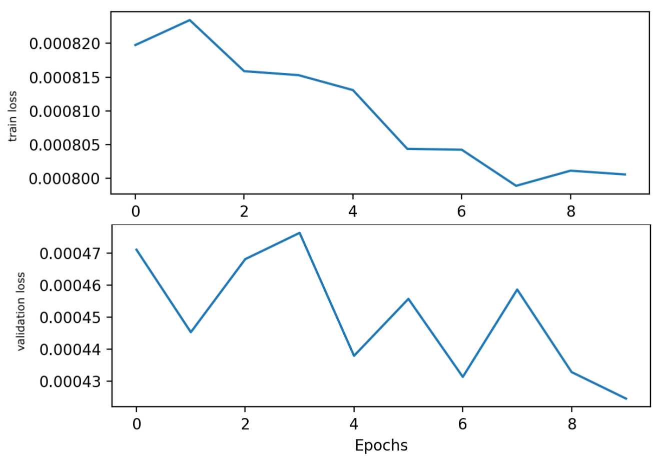

Fig. 3: Train and validation MSE of the single layer LSTM model tested

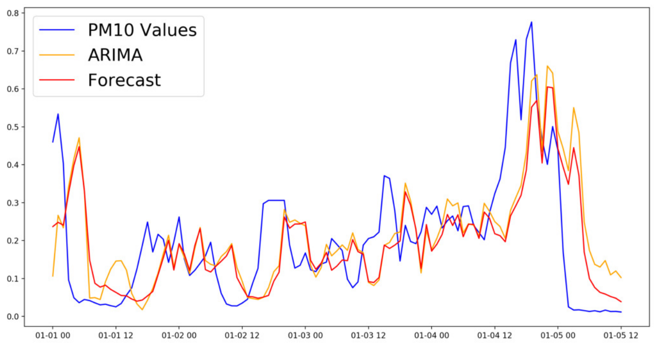

Fig. 4: Performance comparison to ARIMA model in the first week of the test data set

can be a problem in such networks. This often happens when neural nets are trained on relatively small datasets. The lack of control over the learning process often leads to cases where the neural network can not generalize and make forecasts for new data. Dropout is a technique for addressing this problem. The idea is to randomly drop units from the neural network in the training phase in order to prevent units from co-adapting too much.

- *Learning rate* - The learning rate is a hyperparameter that controls how much to change the model in response to the estimated error each time the model weights are updated. Choosing the learning rate is challenging as a value too small may result in a lengthy training process that could get stuck, whereas a value too large may result in learning a sub-optimal set of weights too fast or an unstable training process. The learning rate controls how

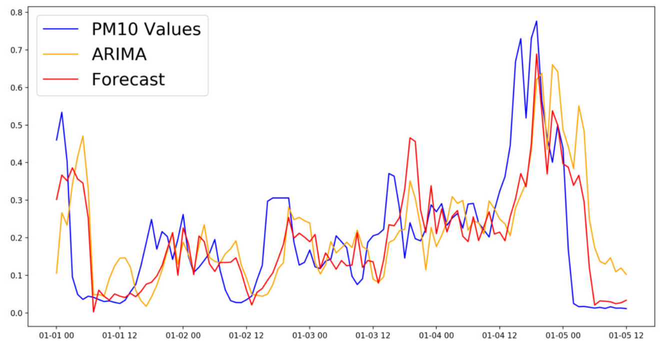

Fig. 5: Performance comparison to ARIMA model in the first week of the test data set. Training data includes meteorological parameters

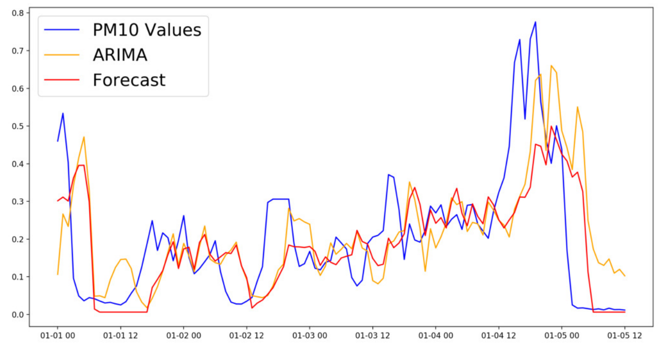

Fig. 6: Performance comparison of the SimpleRNN+LSTM architecture to the ARIMA model in the first week of the test data set. Training data includes meteorological parameters

quickly the model is adapted to the problem.

- *LSTM layer units* - the number of LSTM cells in the layer is a parameter that we used in our model optimization. The number of units determines the dimensionality of the output space.
- *RNN units* - the number of RNN cells in the layer. By default, the output of an RNN layer contains a single vector per sample. This vector is the RNN cell output corresponding to the last timestep, containing information about the entire input sequence. The units parameter determines the shape of this output. A RNN layer can also return the entire sequence of outputs for each sample (one vector per timestep per sample).
- *GRU units* - parameter that determines the dimensionality of the output vector.

TABLE III: Parameters used for tuning the neural network

Parameter name	Min Value	Max Value	Step
Learning rate	32	128	2
Dropout rate	0.1	0.3	0.1
LSTM 1 layer units	2	128	2
LSTM 2 layer units	2	124	4
RNN layer units	1	128	4
GRU layer units	12	256	4

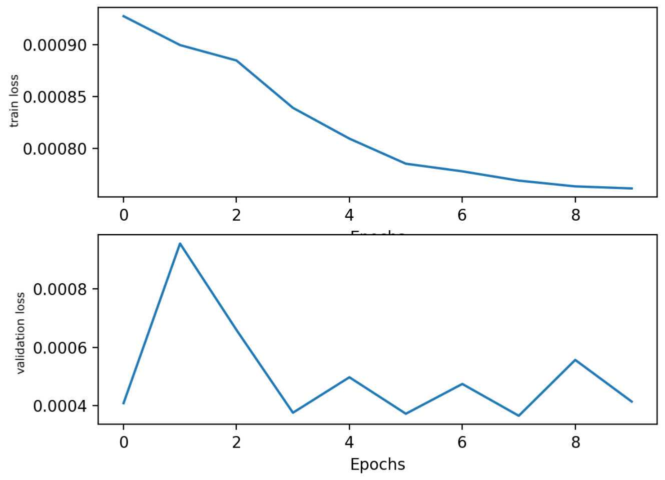

Fig. 7: Train and validation MSE of the RNN+LSTM model with 24 hours training data samples

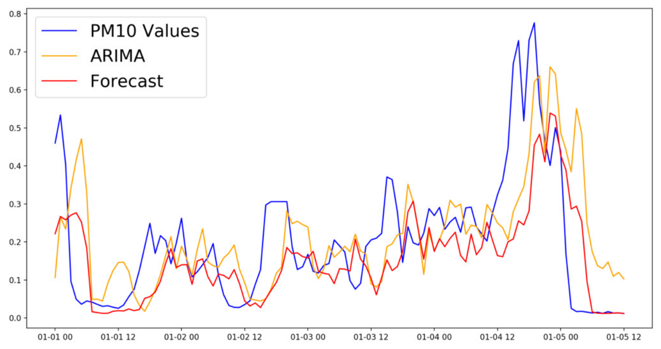

Fig. 8: Performance comparison of the RNN+LSTM architecture trained with extended 24 hours data sequence

We performed a grid search through the parameter space, trying every possible combination of the parameters.

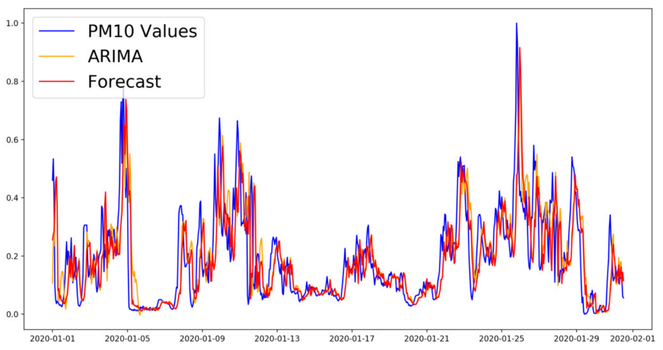

Fig. 9: Performance of the LSTM model with data from two sensors

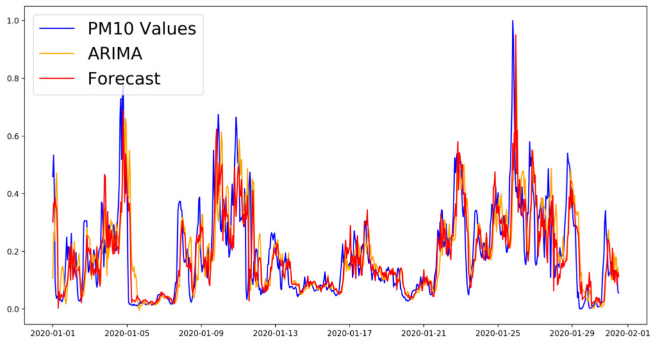

Fig. 10: Performance of the LSTM model with data extended with meteorological parameters

III. Conclusion

All architectures, except for the model with GRU layer, outperformed the ARIMA model in forecasting near-term future values (+3 hours). Models based on simple LSTM architecture exhibited the best results, leading to an improvement of up to 17.28% in terms of RMSE reduction with respect to ARIMA. Table I shows a summary of the results obtained in the experiments. More complex architectures can lead to overfitting. Further research is needed to solve this problem. We concluded that the combination of meteorological and air pollution measurements data improves the performance of LSTM and RNN+LSTM neural networks as a near-term predictive models over the performance of the same architectures used with air quality data alone. Additionally, the results show that the combination of meteorological and air pollution measurements data with LSTM and RNN + LSTM neural networks leads to good short-term predictive models.

It is important to note that our approach can be used to analyze different pollution datasets, as well as time series data in other domains. In fact, the possibility to query weather stations through web services allows to easily complement on-site sensor measurements of pollutants with historical and predicted weather data.

Further experiments are needed to examine the existing models by introducing additional data of air quality as well as of meteorological nature. Other experiments should also be performed to examine the model's accuracy for extended near-term future forecasts, for example, +6 and +9 hours in the future.

Acknowledgment

The work presented in this paper was partially funded by the Ss. Cyril and Methodius University in Skopje, Faculty of Computer Science and Engineering. We also acknowledge the support of NVIDIA through a donation of a Titan V GPU.

References

[1] A. S. Whittemore, "Air pollution and respiratory disease," *Annual review of public health*, vol. 2, no. 1, pp. 397–429, 1981.

[2] M. R. Heal, P. Kumar, and R. M. Harrison, "Particles, air quality, policy and health," *Chemical Society Reviews*, vol. 41, no. 19, pp. 6606–6630, 2012.

[3] R. Arasa, M. Picanyol, and J. Solé, "Analysis of the integrated environmental and meteorological forecasting and alert system (siam) for air quality applications over different regions of the iberian peninsula," in *Proceedings of HARMO15 Congress. Madrid. http://www. harmo. org/Conferences/Proceedings/_Madrid/publishedSections/H15-70. pdf*, 2013.

[4] G. Fronza and P. Melli, *Mathematical Models for Planning and Controlling Air Quality: Proceedings of an October 1979 IIASA Workshop*. Elsevier, 2014.

[5] D. Slezak, M. Grzegorowski, A. Janusz, M. Kozielski, S. H. Nguyen, M. Sikora, S. Stawicki, and L. Wrobel, "A framework for learning and embedding multi-sensor forecasting models into a decision support system: A case study of methane concentration in coal mines," *Information Sciences*, vol. 451-452, pp. 112 – 133, 2018.

[6] A. Janusz, M. Grzegorowski, M. Michalak, L. Wrobel, M. Sikora, and D. Slezak, "Predicting seismic events in coal mines based on underground sensor measurements," *Engineering Applications of Artificial Intelligence*, vol. 64, pp. 83–94, 2017.

[7] E. Zdravevski, P. Lameski, R. Mingov, A. Kulakov, and D. Gjorgjevikj, "Robust histogram-based feature engineering of time series data," in *2015 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2015, pp. 381–388.

[8] A. Janusz, D. Slezak, M. Sikora, and L. Wrobel, "Predicting dangerous seismic events: Aaia'16 data mining challenge," in *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2016, pp. 205–211.

[9] E. Zdravevski, P. Lameski, and A. Kulakov, "Automatic feature engineering for prediction of dangerous seismic activities in coal mines," in *2016 Federated Conference on Computer Science and Information Systems (FedCSIS)*, 2016, pp. 245–248.

[10] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] A. C. Tsoi and A. Back, "Discrete time recurrent neural network architectures: A unifying review," *Neurocomputing*, vol. 15, no. 3-4, pp. 183–223, 1997.

[12] L. Yunpeng, H. Di, B. Junpeng, and Q. Yong, "Multi-step ahead time series forecasting for different data patterns based on lstm recurrent neural network," in *2017 14th Web Information Systems and Applications Conference (WISA)*. IEEE, 2017, pp. 305–310.

[13] M. Ceci, R. Corizzo, D. Malerba, and A. Rashkovska, "Spatial autocorrelation and entropy for renewable energy forecasting," *Data Mining and Knowledge Discovery*, vol. 33, no. 3, pp. 698–729, 2019.

[14] A. Tokgöz and G. Ünal, "A rnn based time series approach for forecasting turkish electricity load," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018, pp. 1–4.

[15] B. B. Sahoo, R. Jha, A. Singh, and D. Kumar, "Long short-term memory (lstm) recurrent neural network for low-flow hydrological time series forecasting," *Acta Geophysica*, vol. 67, no. 5, pp. 1471–1481, 2019.

[16] R. Corizzo, M. Ceci, H. Fanaee-T, and J. Gama, "Multi-aspect renewable energy forecasting," *Information Sciences*, 2020.

[17] V. Stojov, N. Koteli, P. Lameski, and E. Zdravevski, "Application of machine learning and time-series analysis for air pollution prediction," in *CIIT 2018*, 2018.

[18] Y.-T. Tsai, Y.-R. Zeng, and Y.-S. Chang, "Air pollution forecasting using rnn with lstm," in *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech)*. IEEE, 2018, pp. 1074–1079.

[19] R. Corizzo, M. Ceci, E. Zdravevski, and N. Japkowicz, "Scalable autoencoders for gravitational waves detection from time series data," *Expert Systems with Applications*, vol. 151, p. 113378, 2020.

[20] A. Zhao, L. Qi, J. Dong, and H. Yu, "Dual channel lstm based multi-feature extraction in gait for diagnosis of neurodegenerative diseases," *Knowledge-Based Systems*, vol. 145, pp. 91–97, 2018.

[21] B. Petrovska, E. Zdravevski, P. Lameski, R. Corizzo, I. Štajduhar, and J. Lerga, "Deep learning for feature extraction in remote sensing: A case-study of aerial scene classification," *Sensors*, vol. 20, no. 14, p. 3906, 2020.

[22] B. Petrovska, T. Atanasova-Pacemska, R. Corizzo, P. Mignone, P. Lameski, and E. Zdravevski, "Aerial scene classification through fine-tuning with adaptive learning rates and label smoothing," *Applied Sciences*, 2020.

[23] S. Ryan, R. Corizzo, I. Kiringa, and N. Japkowicz, “Pattern and anomaly localization in complex and dynamic data,” in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, 2019, pp. 1756–1763.

[24] U. Kumar and V. Jain, “Arima forecasting of ambient air pollutants (o 3, no, no 2 and co),” *Stochastic Environmental Research and Risk Assessment*, vol. 24, no. 5, pp. 751–760, 2010.

[25] J. Zhang and K. Man, “Time series prediction using rnn in multi-dimension embedding phase space,” in *SMC’98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No. 98CH36218)*, vol. 2. IEEE, 1998, pp. 1868–1873.

[26] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation,” *arXiv preprint arXiv:1406.1078*, 2014.

[27] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, “Recurrent neural networks for multivariate time series with missing values,” *Scientific reports*, vol. 8, no. 1, pp. 1–12, 2018.

[28] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[29] F. Chollet *et al.*, “Keras,” https://github.com/fchollet/keras, 2015.