Deep Learning and Gradient Boosting for Urban Environmental Noise Monitoring in Smart Cities

Jérémy Renaud, Ralph Karam, Michel Salomon[0000−0002−1119−2760], and Raphaël Couturier[0000−0003−1490−9592]

FEMTO-ST Institute, CNRS, Univ. Bourgogne Franche-Comté (UBFC), Belfort, France
{ralph.karam,michel.salomon,raphael.couturier}@univ-fcomte.fr

Abstract. Every day the innovative IoT technology is further expanding in our environment, with applications deployed in various contexts including cities. Communities can indeed address problems linked to urbanization thanks to this technology through the Smart City concept and thus support a sustainable development of their cities. Artificial intelligence and namely its machine learning branch is expected to reinforce this trend by making smart cities smarter and better. However, smart cities can only be successful if they can be trusted and therefore machine learning is also a possible candidate for mitigating cyberattacks. In this paper we follow two directions. First, the ability of Gradient Boosting and Deep Learning to make long-term predictions of noise level is studied considering noise data collected in a suburb of an English city. Second, the possibility of detecting anomalous data resulting from malicious injections is also investigated. The obtained results show the relevance of a deep learning architecture.

Keywords: Deep Learning · Gradient Boosting · Prediction and anomaly detection · Noise monitoring · Smart City.

1 Introduction

With the advent of the Internet of Things (IoT), a system interconnecting computing devices, digital machines, and objects through a network, many innovative solutions based on it arise in order to solve problems touching a wide range of domains. Indeed, the deployment of such IoT devices can make our environment smarter. Thus the IoT can for example help us in our daily life at home by monitoring the lighting, media or security systems and so on. It can also provide better transportation conditions or improve manufacturing processes.

Among the possible applications of the IoT, the concept of Smart City addresses urban problems in order to have cities more citizen friendly [15]. Today, more than half of the world's population lives in cities and according to a United Nations forecast this number will continue to increase greatly in the future, reaching an expected proportion of 68% by 2050 [3]. Obviously, as the resident population of a city increases, many aspects of the city are impacted, namely

the energy and water distribution, the transportation system, environment, etc. Therefore municipal governments need a framework to help them face all these urbanization challenges, to take the best decisions for a sustainable development of the city and for improving the quality of life. Fortunately, the combination of IoT technology and artificial intelligence allows to develop a fully functional Smart City providing such a framework. On the one hand the IoT infrastructure collects data from the connected objects and machines, while on the other hand artificial intelligence extracts information from the large database.

Many cities throughout the world have already successfully deployed the IoT technology, giving an open access to data to promote innovative developments of solutions by public or private parties. For example, regarding urban mobility challenges, a congestion management system called Midtown in Motion [17] was deployed in New York City, while real-time traffic and transportation data are free of access in several cities like Amsterdam or Copenhagen [10]. Artificial intelligence and more particularly the machine learning technology can convert these data into insights and is thus a key to expand capabilities of smart cities [12, 5]. Typically, machine learning allows to make predictions useful for taking decisions or detecting problems. As an illustration, considering another urban challenge, air and noise pollution control, long-term predictions can give city administrators knowledge to choose the best transportation investments to possibly tackle these problems in the future, while real-time short-term predictions could enable an immediate action on the motor vehicle traffic congestion. In this work we focus on noise pollution, which according to European Environment Agency is a growing problem [1] and one which many people may not be aware of the impacts it has on their health. A first contribution of this work is thus a study of the ability of different machine learning models, and more specifically deep learning architectures, to provide relevant long-term predictions using as a case study real noise data collected in a suburb of an English city.

Together IoT technology and artificial intelligence can make cities smarter and better, but that requires that we can trust the data. However, like other systems, smart cities are prone to security threats and thus need solutions to mitigate potential cyberattacks and in particular False Data Injection Attacks (FDIA) [11]. These latter are difficult to detect as they alter the semantics of the data while preserving their syntactic correctness and logical consistency. Let us note that this security issue is not specific to Smart Cities, various systems are exposed to similar attacks (Smart Grid, Robotics 4.0, and so on). To detect security anomalies, machine learning is currently a hot topic in both research and cybersecurity industry. Therefore, the second contribution is to check whether an injection of false noise data can be detected.

The twofold contributions are presented throughout the following sections. Related works are discussed in Section 2. Section 3 deals with the noise data we used. The nature of the IoT infrastructure, the monitored area, and both acquisition and preprocessing steps are detailed. The next section briefly describes the machine learning models we studied. Experimental results are summarized and discussed in Section 5, while concluding remarks are drawn in the final one.

2 Related Works

In the literature, many articles touched the subject of noise prediction using deep learning [14],[2],[6],[8]. In [14], an LSTM was used to forecast the sound pressure and loudness levels for periods of 1 min, 5 min, 15 min, 30 min and 60 min. The data used for training and testing was issued from sensors located in an open office room where the majority of the noise is produced by human activities and speech. In [2],[6],[8] vanilla neural networks were used for noise prediction. In [2], the architecture used is made of a learning vector quantization network (LVQ) for discarding wrong input data and a feed forward neural network for predicting the noise levels. In [6],[8] a feedforward network is used for predicting noise level. The number of features used for the input is 25 (such as traffic flow type, average speed of vehicles...) in [6] and 5 (such as the traffic flow, percentage of heavy vehicles...) in [8]. In addition to deep learning some techniques use empirical models for noise prediction [16].

3 Urban Environmental Noise Data

To obtain noise pollution data, it is necessary to have access to an acoustic sensor network that consists of nodes capturing sound in near real-time. Such nodes can take measurements over time and preproccess them, in order to send sound pressure levels to a gateway that will further transmit these data to a storage platform. Many working smart cities have already deployed sensor networks allowing to collect and publish data through cloud platform. For example, the Smart Santander facility provide access to data captured by noise sensors [13] thanks to the open source platform called FIWARE [4].

In this work, we take benefit from a collaboration with the Flowbird company[1], world leader in parking and transport ticketing solutions, to get access to data issued from their Park&Breathe solution. This latter allows to create a network of noise and pollution sensors by using existing Strada parking terminals on which multi-sensor kits are integrated. Figure 1(a) displays a terminal with an integrated Park&Breath kit on its top. Let us notice that this solution is very interesting as it operates on existing urban furniture and can take advantage of the fine-meshed network built by parking terminals in many cities. However, not all terminals are usually upgraded with a sensing kit, it depends on the coverage of the area of interest to be monitored.

3.1 Monitored Area

Practically, the noise data are issued from parking terminals deployed in the city of Solihull, in the county of West Midlands in western-central England, and more precisely in one of its suburb called Shirley. Many terminals cover this area among which three of them are fitted with the Park&Breathe solution. The

[1] flowbird.group

map of Fig. 1(b) displays the locations of these terminals, where each of them is spotted by a red mark and a code. As can be seen on the map they monitor a part of the Stratford Road, with two that are almost opposite to each other.

An idea of the traffic flow that can be observed on the considered road is given by the data corresponding to the manual count point 46367 (flagged by the blue mark on the map) managed by the English Department of Transport. According to the most recent available data, an average daily flow of about 31,500 motor vehicles uses this road. Moreover, as this road is also numbered A34, it means that it is a trunk road of the English network. The monitored area is thus a good representative of urban areas where citizens are exposed to large noise nuisances.

3.2 Data Acquisition and Format

Noise data are available in the form of a file per terminal. A file in which a line represents a histogram composed of 77 sound levels as abscissa, while the number of times a particular sound level occurred during a time period is plotted as ordinate. For this work, we used noise levels of a time frame ranging from May 2019 up to end January 2020, where each line in a data file of a terminal corresponds to sound levels collected during 15 minutes.

However these data were not used directly. In fact, they were processed in order to rather work with average values over periods of 1 hour. Therefore, the initial values of the three files were replaced by average values following two methods. First method consists in replacing a set of 77 sound level values obtained at a time step by a single one computed as follows:

$$L(t) = \frac{\sum_{l=0}^{76} (N_l(t) \times (l + 35))}{\sum_{l=0}^{76} N_l(t)} \tag{1}$$

(a) Park&Breathe kit (b) Measurement points along Stratford Road

Fig. 1. Multi-sensor kit integration on a terminal (a) and (b) positions of the three upgraded terminals (red marks) as well as the manual count point (blue mark).

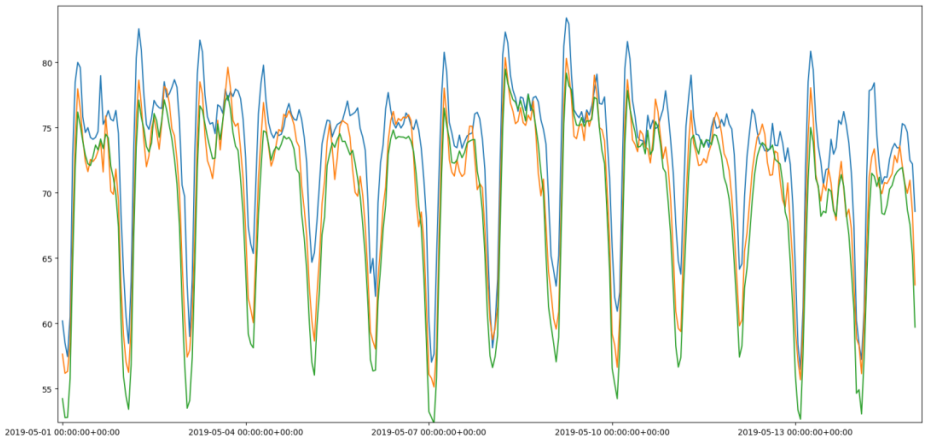

Fig. 2. Average sound levels observed between May 1st 2019 and mid may.

where $L(t)$ is the average value and $N_l(t)$ the number of times sound level l occurred during time step t. Notice that one needs to add 35 to l to obtain the true measure in decibels, because l is actually a sound level index. Second method keeps the histogram, but computes for each sound level its average value over a period of 1 hour. Since both methods consider a same period duration, they result in a same data set size of $19,443$ samples ($6,570$ samples for each of the terminals 500 and 515, $6,303$ for terminal 521). Figure 2 shows the curves obtained with the first method for a time range of two weeks going from May 1st 2019 until mid May. The three curves present the same evolution, with extreme values reflecting the proximity of the terminals to the road and the distance between them. Thus, the orange and green curves are the most similar as they belong to the two terminals almost in front of each other, additionally the blue one exhibits higher noise levels because it is closer to the road. One can see a periodic daily evolution where the lowest average noise level is captured early in the morning. The days of the weekend and more particularly the Sundays are also easily recognizable (May 5th and 12th), as well as the first Monday in May corresponding to the Early May Bank Holiday (May 6th). In the following only data issued from first method are used.

4 Studied Machine Learning models

As we deal with data that are time series, the most obvious deep learning architecture to investigate is the well-known Long Short-Term Memory (LSTM). Indeed, for most of the deep learning practitioners, sequence modeling usually means recurrent networks. An alternative to a pure recurrent neural network is to combine aspects of recurrent and convolutional architectures, using convolutions to change the representation of the original data. Deep learning is very powerful, but gradient tree boosting is also able to model relationship in the data

and thus seem worthy of investigation. In this work we conduct an evaluation of a representative of each of these machine learning models.

4.1 Stacked Long Short-Term Memory (Stacked-LSTM)

The LSTM architecture is a special type of recurrent neural networks based on memory blocks [7]. Previous architectures such as vanilla recurrent neural networks can be used for time series as well, however they are only valid for sequential data with short-term dependencies. The LSTM architecture advantage is its ability to make use of both short-term and long-term dependencies to make predictions. The LSTM has a hidden state as well as a cell state which plays the role of memory. The content of the memory as well as the output of the network is controlled using pointwise operators called gates (forget gate, input gate, and output gate). We did not make use of a Gated Recurrent Unit network (GRU), a more recent architecture that often provides comparable performance while being faster to train, because we did not obtain satisfactory predictions during some preliminary experiments. A possible explanation is that in theory LSTMs can remember longer sequences than GRU networks.

4.2 Combination of Convolutional Neural Network and LSTM (CNN-LSTM)

The idea in this kind of deep learning architecture is to take benefit from convolutional layers to extract relevant features from input data in order to be able to make easier predictions with the LSTM. For the temporal one-dimensional noise data considered in this work, we used convolutional layers that rely on 1D convolutions. Such layers can extract features from or transform one dimensional data such as multivariate time series. The use of convolutional layers to pretreat sequential data before feeding in a LSTM architecture is usual as it can easily improve forecasts made by the LSTM with a low computing cost.

4.3 Gradient Boosting (LightGBM)

LightGBM is a popular gradient boosting framework created by Microsoft. It is a tree based learning algorithm designed to have a faster training speed, a lower memory usage and a better accuracy [9]. It supports execution with GPU and parallelization, has a function for testing and keeps the best set of parameters. Compared to other boosting algorithms, LightGBM differs in the approach used for tree growth that is leaf-wise (best-fiirst) and not level-wise (depth-first).

5 Experimental results

The deep learning architectures have been implemented and evaluated with the easy-to-use Keras python library. This library is user-friendly and allows to perform the computations with different backends such as TensorFlow. It is particularly suited for the implementation of preexisting deep learning architectures,

when there is no need for low level operations design. In the case of gradient boosting, many models are already available in the python scikit-learn package, namely the LightGBM model.

5.1 Setup of the Deep Learning Architectures

- The Stacked-LSTM architecture we retained is composed of a first layer of 150 LSTM units, followed by a second layer of 100 units and a final dense layer reduced to a single neuron. The LSTM layers use both a ReLU activation function and the dense layer uses a linear activation function.
- The CNN-LSTM is made up of two parts. The first one is a fully convolutional neural network that consists of three layers of 16 neurons where each neuron computes one-dimensional convolutions. The convolution kernels are initialized using the Glorot uniform initializer available in Keras. The kernels have a size of 3. The shape of the data in input of the convolutional part is a vector whose size corresponds to the look_back value explained thereafter. At the output, the data have the same shape as at the input. The second part is a LSTM whose architecture is similar to the Stacked-LSTM.

A key parameter when working with time series is the look_back value. It corresponds to the amount of previous data that we will provide to the neural network so that it can predict the next value. For example, as can be seen in Fig. 3, with a look_back of 3 the network will have access to data $H0$, $H1$ and $H2$ to predict $H3$. Both architectures are first trained with Nadam optimizer, the version of Adam that uses the Nesterov momentum instead of regular momentum.

5.2 Average Noise Level Prediction

For these experiments, we used only the data samples from terminal 515, using 90% of them for training and the remaining 10% for testing. In practice, it means that 5,912 samples were used for training and 657 samples for testing. The prediction results obtained with the two deep learning architectures, considering different combinations of maximum training epochs and look back values, are shown in Table 1. To assess the quality of the predictions, we chose the Root Mean Square Error (RMSE) that is a standard way to measure the error of a model in its prediction ability. As can be seen, for the combinations with a low number of training epochs the best results are always provided by the Stacked-LSTM, while the CNN-LSTM gives results which are not bad at all, being of the same order or only slightly worse for a same look_back value. However, the the gap between the two architectures reduces as the number of epochs and the look_back value increase and finally it is the CNN-LSTM that takes the lead for the configuration with 50 epochs and a look_back value equal to 36. It can also be noticed that the CNN-LSTM appears to be more sensitive to the look_back value than the Stacked-LSTM. A possible explanation of these observations is that obviously the CNN-LSTM is deeper than the Stacked-LSTM, thus it needs

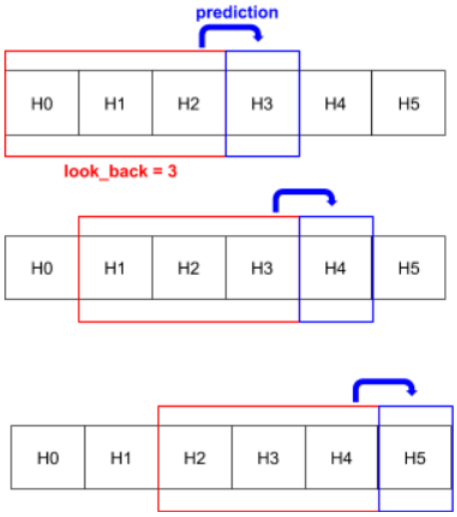

Fig. 3. Look back principle for time series forecasting.

a larger number of epochs and enough input data to take benefit from the the CNN part (to be trained to extract relevant features).

For the LightGBM gradient boosting method, the parameters controlling the learning process are different. num_leaves defines the maximum number of leaves in one tree, while the number of trees is set with the parameter denoted num_estimator. The learning rate is the step size at each iteration to reach the minimum of the loss function. Table 2 presents the prediction errors for two configurations of tree growth and learning rate. When comparing them to the errors shown in Table 1, LightGBM seems to be a very good competitor for the deep learning architectures, giving lower training and testing errors, even if in this latter case the improvement is very tiny. However, LightGBM exhibits large differences between the training and testing errors that might be a first evidence of overfitting to which leaf-wise growth is known to be more sensitive than its level-wise counterpart.

5.3 Optimization of the Deep Learning Architectures

To increase the accuracy of the predictions given by the deep networks, the data used for training have been shuffled (activation of the shuffle option in time series generator provided by Keras). This improves a neural network ability to generalize and prevent overfitting. Moreover, different optimizers have been evaluated with an early stopping when the loss stops decreasing. With that, the analysis of the convergence could be done for each optimizer. As highlighted by Table 3, from RMSE values' point of view, Adam is slightly outperformed by Nadam, which is itself overrun by RMSprop that is thus the optimizer giving the

Table 1. Evaluation of the deep learning architectures for different values of training epochs and look_back.

Deep learning architectures				
Type	Epochs	Look_back	RMSE on train	RMSE on test
Stacked-LSTM	25	12	1.35	1.29
	50	3	1.43	1.48
	50	12	1.22	1.34
	50	24	1.19	1.17
	100	24	0.96	1.01
	100	36	0.99	1.05
CNN-LSTM	25	12	1.37	1.48
	50	3	4.28	4.32
	50	12	1.27	1.52
	50	24	1.09	1.16
	100	24	1.03	1.17
	100	36	0.98	1.05

Table 2. Evaluation of LightGBM for different configurations of tree growth and learning rate.

LightGBM				
Num_leaves	Learning rate	N_estimator	RMSE on train	RMSE on test
11	0.05	500	0.78	1.00
11	0.04	1000	0.71	0.99

lowest errors. Besides, RMSprop exhibits the same behavior (almost the same number of epochs to reach convergence) for the two deep learning architectures, while Adam and Nadam get stuck much earlier in the case of Stacked-LSTM.

5.4 6 Day Forecasts

Up to now we focused on forecasting the average noise level one hour ahead (H+1), but this is not enough. Therefore, we have taken the two machine learning models that have achieved the lowest errors (CNN-LSTM and LightGBM) to make predictions for the next 6 days (H+144). We can observe on Fig. 4 that both models learned the underlying pattern of the average noise level evolution, with a small preference for the deep neural network as it predictions (the green curve) seem to be nearer to the real measurements (the blue curve which stops shortly before the x-axis value is equal to 6,600). Therefore we have retained this architecture to assess its ability to detect anomalies.

5.5 Detection of False Data Injection Attacks

Since we have the intuition that the CNN-LSTM architecture can predict values that correspond to reality, we are able to use it to detect potential anomalies that may appear in the data collected by the sensors. If a measured data is too far

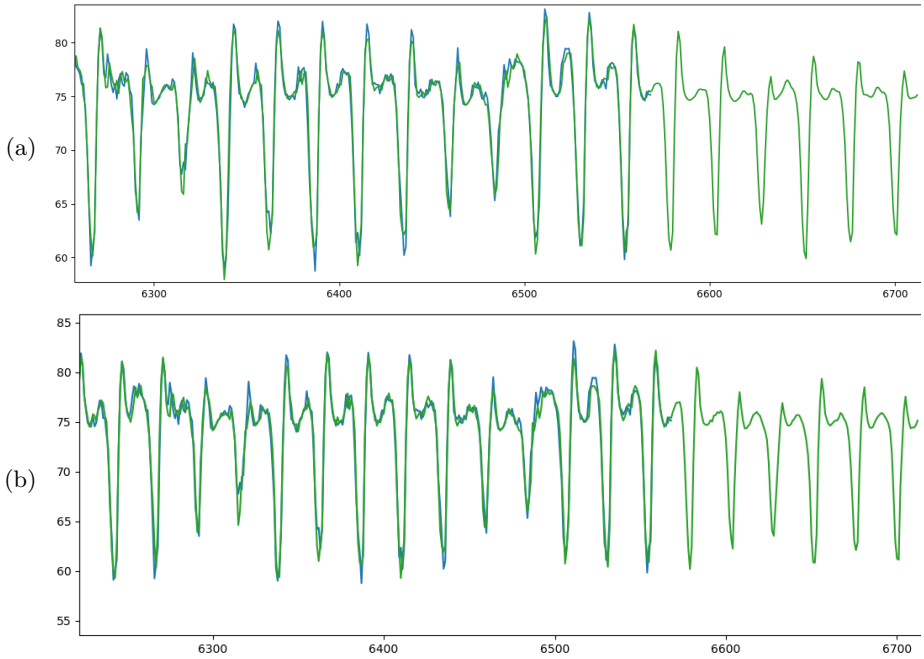

Fig. 4. 6 days forecasting with CNN-LSTM (a) and LightGBM (b).

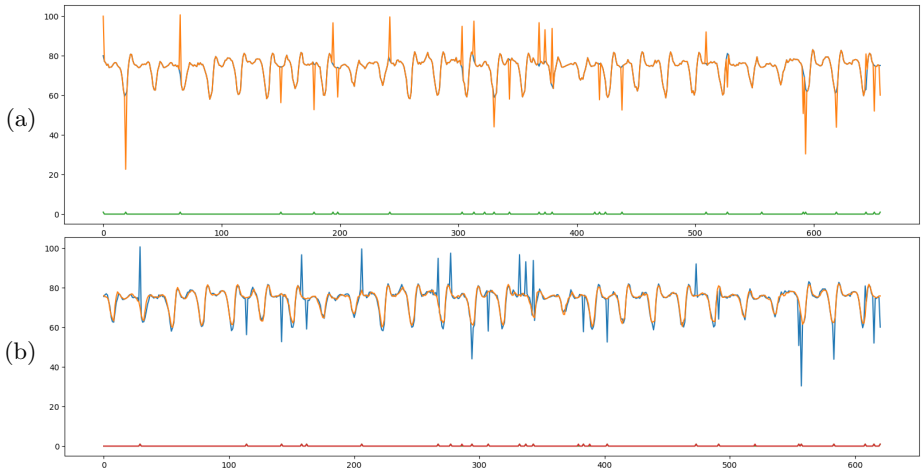

Fig. 5. Dataset with attacks (a) and detection of anomalies on average noise curve (b).

Table 3. Evaluation of deep learning architectures trained with different optimizers.

Deep learning architectures				
Type	Optimizer	Number of epochs for convergence	RMSE on train	RMSE on test
Stacked-LSTM	Nadam	134	1.05	1.04
	Adam	95	1.09	1.06
	RMSprop	259	0.96	0.97
CNN-LSTM	Nadam	223	0.94	0.94
	Adam	187	0.99	0.99
	RMSprop	265	0.92	0.95

from the corresponding forecast we can consider this data as an anomaly which might result from an FDIA attack. The first step is the creation of a dataset with a few anomalies. For that, we just randomly disturbed the input data by adding / subtracting values to some of the original average noise level values (see Fig. 5). The blue curve corresponds to the original data, the orange curve represents the data after injection, and the green curve shows where the data was attacked (0 for nothing and 1 for injection).

As explained above, since the CNN-LSTM neural network makes satisfactory predictions, a big difference between real measured data and the forecast implies the presence of an anomaly. In Fig. 5(a), the blue curve corresponds to the original data, the orange curve represents the data obtained after the false data injections, the green curve shows where data have been attacked (0 for normal data and 1 for anomalous data). In Fig. 5(b), the blue curve corresponds to the real data which underwent false data injections (it is equivalent to the orange curve in Fig. 5(a)), the orange curve represents the prediction of the deep neural network, and the red curve shows where an anomaly can be detected using the CNN-LSTM's predictions (0 for normal data and 1 for anomalous data). In fact, a false data injection attack is detected as soon as the difference between a real measured data and its counterpart predicted by the neural network is larger than a given threshold. Using this technique all of the anomalies were detected using a threshold of 5 db. This high detection rate can be explained by the ability of the deep learning architecture to make accurate forecasts. Note that lowering the detection threshold results in some false positive cases. To increase the detection accuracy, the prediction model must be improved.

6 Conclusion

In this paper, we have investigated the ability of deep learning architectures and gradient boosting to predict the real average urban noise level in the context of smart cities. The sensing platform used to collect the data has been presented, in particular the monitored area and the way the data were obtained and formatted. The experimental results show that relevant short-term predictions can be provided as well as 6 day forecasts capturing the daily pattern exhibited in

the data. Thanks to these results we have also checked whether anomalous data might be detected. Being able to detect such problems is of great interest for making smart cities more robust regarding possible false data injection attacks.

Acknowledgments

This work was supported by the DGA (French defence procurement agency) in the context of the GeLeaD project (project number ANR-18-ASTR-0011) related to the ANR ASTRID research program (specific support scheme for research works and innovation defence). It was also partially supported by the EIPHI Graduate School (contract ANR-17-EURE-0002). Computations have been performed on the supercomputer facilities of the "Mésocentre de Franche-Comté".

References

1. European Environment Agency: Environmental noise in Europe. Publications Office of the European Union, (2020). https://doi.org/10.2800/686249
2. Cammarata, G., Cavalieri, S., Fichera, A.: A neural network architecture for noise prediction. Neural Networks **8**(6), 963–973 (1995)
3. United Nations Department of Economic, Social Affairs: World Urbanization Prospects: The 2018 Revision. United Nations (2019). https://doi.org/10.18356/b9e995fe-en
4. Foundation, F.: FIWARE: The open source platform for our smart digital future, fiware.org - https://data.lab.fiware.org
5. Gautam, K., Puri, V., Tromp, J.G., Nguyen, N.G., Van Le, C.: Internet of things (iot) and deep neural network-based intelligent and conceptual model for smart city. In: Satapathy, S.C., Bhateja, V., Nguyen, B.L., Nguyen, N.G., Le, D.N. (eds.) Frontiers in Intelligent Computing: Theory and Applications. pp. 287–300. Springer Singapore, Singapore (2020)
6. Genaro, N., Torija, A., Ramos-Ridao, A., Requena, I., Ruiz, D., Zamorano, M.: A neural network based model for urban noise prediction. The journal of the Acoustical Society of America **128**(4), 1738–1746 (2010)
7. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with lstm. In: 1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470). vol. 2, pp. 850–855 vol.2 (1999)
8. Givargis, S., Karimi, H.: A basic neural traffic noise prediction model for tehran's roads. Journal of Environmental Management **91**(12), 2529–2534 (2010)
9. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. In: Advances in neural information processing systems. pp. 3146–3154 (2017)
10. Kosowatz, J.: 10 Smart Cities. Mechanical Engineering **142**(02), 32–37 (02 2020). https://doi.org/10.1115/1.2020-FEB1
11. Mode, G.R., Calyam, P., Hoque, K.A.: False data injection attacks in internet of things and deep learning enabled predictive analytics (2019)
12. Navarathna, P.J., Malagi, V.P.: Artificial intelligence in smart city analysis. In: 2018 International Conference on Smart Systems and Inventive Technology (ICSSIT). pp. 44–47 (2018). https://doi.org/10.1109/ICSSIT.2018.8748476

13. Navarro, J., TomasGabarron, J., Escolano, J.: On the application of big data techniques to noise monitoring of smart cities. In: Euroregio 2016 Oporto. Portuguese Acoustical Society (SPA) and Spanish Acoustics Society (SEA) (2016)
14. Navarro, J.M., Martínez-España, R., Bueno-Crespo, A., Martínez, R., Cecilia, J.M.: Sound levels forecasting in an acoustic sensor network using a deep neural network. Sensors **20**(3), 903 (2020)
15. Silva, B.N., Khan, M., Han, K.: Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities. Sustainable Cities and Society **38**, 697 – 713 (2018). https://doi.org/10.1016/j.scs.2018.01.053
16. Steele, C.: A critical review of some traffic noise prediction models. Applied acoustics **62**(3), 271–287 (2001)
17. Xin, W., Chang, J., Muthuswamy, S., Talas, M.: "Midtown in Motion": A new active traffic management methodology and its implementation in New York City. In: Transportation Research Board 92nd Annual Meeting (2013), https://trid.trb.org/view/1242412