# Automatic estimation of the wind turbine noise with recurrent neural networks

Abdelazyz Rkhiss, Arthur Finez, Jean-Rémy Gloaguen, Colin Le Bourdat, Gabriel Vasile, Julien Maillard

25-29 August 2024
inter.noise
NANTES FRANCE

# Automatic estimation of the wind turbine noise with recurrent neural networks

Abdelazyz RKHISS[1]
GIPSA-Lab, Grenoble INP
11 Rue des Mathématiques, 38400 Saint-Martin-d'Hères, FRANCE

Arthur FINEZ
ENGIE Green
Urban Garden, 6 Rue Alexander Fleming, 69007 Lyon, FRANCE

Jean Rémy GLOAGUEN
ENGIE Green
INSULA, 11 rue Arthur III, 44232 Nantes, FRANCE

Colin LE BOURDAT
ENGIE Green
INSULA, 11 rue Arthur III, 44232 Nantes, FRANCE

Gabriel VASILE
GIPSA-Lab, Grenoble INP
11 Rue des Mathématiques, 38400 Saint-Martin-d'Hères, FRANCE

Julien MAILLARD
CSTB
24 Rue Joseph Fourier, 38400 Saint-Martin-d'Hères, FRANCE

**ABSTRACT**

*There is growing interest in the development of renewable energies, particularly wind power. However, wind turbines generate noise that can affect the sound environment of nearby residents. This study focuses on the isolation of wind turbine noise (WTN) level from the surrounding total noise. Our method is based on a Recurrent Neural Network (RNN) Architecture that captures temporal dependencies in the acoustic signal. This proposal is compared to Non-Negative Matrix Factorization (NMF) that has shown first promising results on a previous study on simulated sound scenes.*

*Our approach relies on simple RNN Vanilla conducted using an end-to-end trained model, Gated Recurrent Network (GRU), and a Long Short Term Memory (LSTM). The training and testing dataset is constructed by superimposing measured background noise with synthesized specific noise; wind turbine noise (WTN), effectively simulating realistic environmental conditions. This study aims to assess the acoustic impact of wind turbines on communities and attempt to control turbine operation using advanced machine learning techniques.*

[1] abdelazyz.rkhiss@grenoble-inp.fr

## 1. INTRODUCTION

Residents living near wind farms have occasionally reported concerns about the noise generated by wind turbines, which may adversely affect their sleep quality and general health condition. In [1], Freiberg et al. provides a detailed study on the potential health effects of wind turbine noise in residential areas.

To address these concerns, the French regulatory standards for onshore wind turbines include specific criteria aimed to control the impact of the WTN. Today (national) standards set the actual limits on the Emergence $E$ noise level. $E$ is defined as the difference between the total noise level $L_{TN}$ (sound level during wind turbines operation) and the background noise level $L_{BN}$ (when the wind turbines are inactive):

$$E = L_{TN} - L_{BN}. \tag{1}$$

The regulation standard states: 1) the critical $L_{TN}$ threshold is set to 35 dB(A); 2) if the measured $L_{TN}$ exceeds this critical value the current emergence $E$ is evaluated; 3) on a 24h cycle, the tolerated Emergence level must not surpass 5 dB(A) during the day (7 a.m. to 10 p.m.), and it is limited to 3 dB(A) at night (10 p.m. to 7 a.m.).

A common strategy for enforcing this standard is to implement a curtailment plan aimed at reducing the wind turbine acoustic noise emission. Nevertheless, the validity of the adopted curtailment plan is frequently limited to a specific time duration in which $L_{BN}$ is assumed stationary. This approach, based on directly measuring $L_{BN}$, has a significant impact on the electricity production. For the site in question, it may become either too restrictive, resulting in electricity production below potential, while noise remains below the regulatory limits, or insufficiently restrictive, with noise pressure levels exceeding the regulatory thresholds. The $L_{BN}$ can be altered by the environmental variables and other intermittent noise sources, leading to discrepancies in noise level estimates between the active and inactive phases of wind turbine operation, other strategies can be employed in the design and operation of wind turbines to minimize their noise [2].

To successfully wind turbine monitoring, to limit economic losses and to ensure more regular noise controls, an automatic tool to estimate the emergence $E$ without stopping the turbines would be beneficial to wind farm operators and neighbour residents. Currently, a NMF-based approach was used by Gloaguen et al [3]. NMF has attracted attention also for its usefulness in sound classification [4].

Related to NMF limits, as the error is high when the Specific Noise $L_{SN}$ is dominant, we are moving towards the use of deep neural networks, given their success in the field of source separation [5], sound classification [6] and sound event detection [7].

In other works, we find the use of random forest regression [8] to predict the measurements of noise emitted by a wind farm using the data recorded by the supervisory control and data acquisition (SCADA), neuro-fuzzy methodology to estimate noise level of wind turbines [9].

Our paper proposes a new approach focusing on the use of RNN for the estimation of the $L_{SN}$ level. We begin by construct the learning dataset in Section 2, followed by the method in Section 3. Subsequently, we delve into the experimental results in 4, and finally, we compare our approach with NMF in Section 5.

## 2. LEARNING DATA SET

To construct the data-set, $L_{SN}$ and $L_{BN}$ are needed, the $L_{BN}$ noise can be obtained using sonometers during periods when the turbines are not operational, while the $L_{SN}$ is inherently mixed with the surrounding background noise. To overcome this limitation a specific noise synthesis method is employed. This method allows to generate synthetic turbine noise samples

that mimic the characteristics of the machine noise is able to simulate the real-world noise conditions.

### 2.1. Background noise

Measurements of background noise, in one-third-octave per second, at a local resident's home near a future wind farm, are collected together with meteorological data (wind speed and direction) obtained by a ground LIDAR near the fictive turbine position. The figure below is a sample from the measured $L_{BN}$.

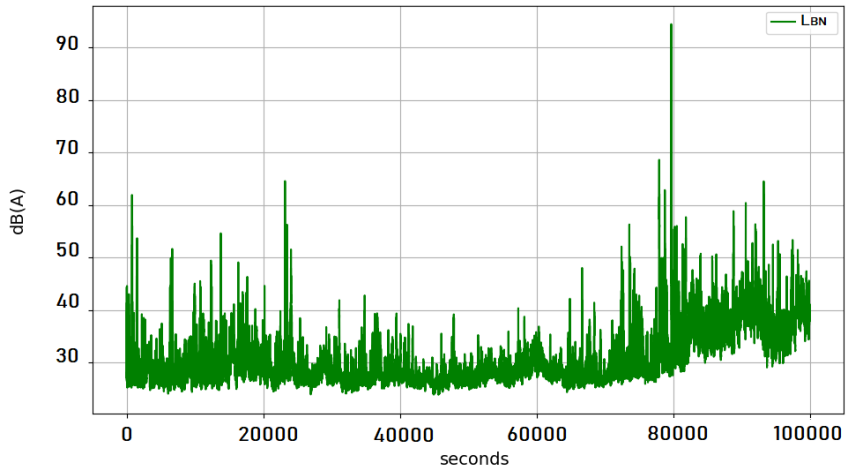


Figure 1: Measured background noise

### 2.2. Specific noise

Every wind turbine comes with a Machine Specifications (MS), it describes the sound Power Level (Lw) in dB in each octave band, according to wind speed. $L_{SN}$ can be interpolated using the MS and wind speed, recorded at the same moment when the sonometer capture $L_{BN}$.

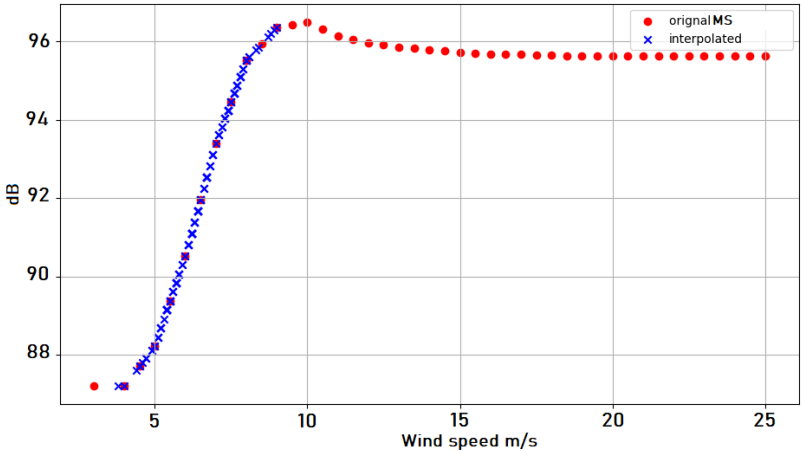


Figure 2: Sound Power Level Interpolated for the 1 kHz Frequency Band Across Various measured Wind speeds

To combine the $L_{SN}$, which is in the octave bands, with the $L_{BN}$ in the third-octave bands, we need to switch from the octave band to the third-octave band. Switching from octave band

to one-third octave band enables more precise resolution of the spectral characteristics of noise, and is commonly used in acoustic analysis and measurement standards for applications such as environmental noise assessment, building acoustics and hearing protection.

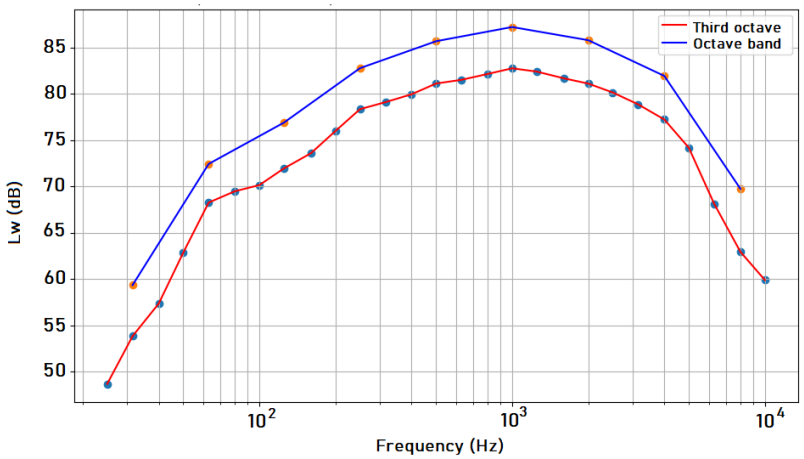


Figure 3: Sound power levels in octave band and translated third octave bands

The next step is to change the sampling frequency of the specific noise signal from 10 minutes to one second, using the Fourier transform, adapting to the desired sampling frequency with an upsampling ratio of 1/600.

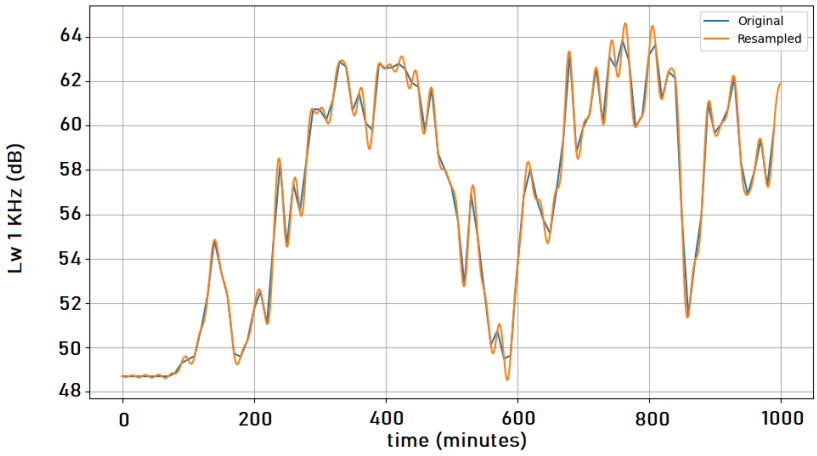


Figure 4: Original acoustic signal per 10 minutes and the interpolated per second

Having obtained a specific noise level in third-octave bands per second, it now becomes important to account for the sound propagation of the wind turbine through the environment. To achieve this, we use attenuation filters designed to be applied to the sound power level. These filters are widely utilized to simulate how sound intensity diminishes as it moves through the environment conditions, geometrical divergence, atmospheric absorption or ground effect. This sound propagation model defines the celerity gradient that defines the acoustic sound propagation (favorable or unfavorable) [10].

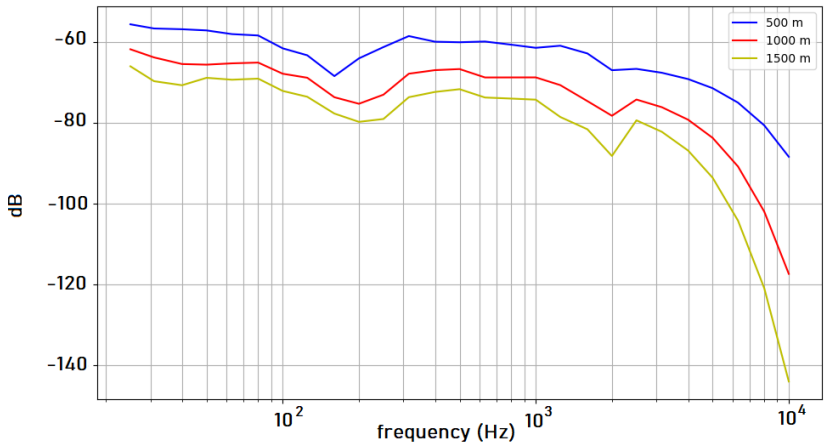


Figure 5: attenuation filters for distances 500 m, 1000 m and 1500 m between the ground and the wind turbine

### 2.3. Total noise

Finally the total noise $L_{TN}$ is obtained by summing the specific noise $L_{SN}$ with the background noise $L_{BN}$. It is necessary to apply an A-weighting filter (dB(A)), for the both $L_{BN}$ and $L_{SN}$:

$$L_{TN} = L_{BN} \oplus L_{SN}, \tag{2}$$

with the $\oplus$ symbol representing the energetic summation. Finally we have a Data Set ($DS1$) as a nearly one-month time series of total noise and specific noise for each measuring point at 500 m, 1000 m and 1500 m, i.e. a total of 3 months of time series.

## 3. METHODS

### 3.1. Recurrent neural networks

Given the temporal nature of $DB1$, where each line represents the sound measurements for a specific second $t$, and the subsequent line represents the following second $t+1$, it becomes imperative to adopt RNN [11] architectures that effectively capture the temporal dependencies within the time series data, At each instant t, the forward pass is modelled by the following equations:

$$a(t) = f(W_{aa} \cdot a(t-1) + W_{ax} \cdot x(t) + b_a), \tag{3a}$$

$$y(t) = g(W_{ya} \cdot a(t) + b_y), \tag{3b}$$

where $x(t)$ and $a(t)$ are respectively the network input and the activation at time t, $f$, $g$ are activation functions, $W$ and $b$ are respectively the weights and biases to be learned during network training to compute $a(t)$ the hidden state and $y(t)$ the output at instant $t$.

To address this, the constructed architectures are leveraging both the basic RNN "Vanilla" model, as well as specialized cells/units such as Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU). These architectures are well-suited for capturing long-term dependencies and mitigating issues like vanishing gradients [12] commonly encountered in traditional RNNs, a gradient explosion can also occur, but very rarely. To overcome these shortcomings, new RNN variants have been introduced into the literature using the LSTM to overcome the problem of gradient vanishing, it was initially proposed by Hochreiter et al [13], then improved in [14]. Using the state of the cell in the final calculation makes LSTM very powerful in tasks where information needs to be stored and used later (long-term dependency).

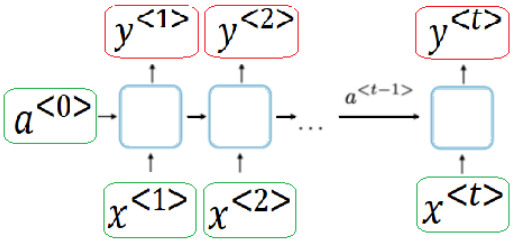


Figure 6: Traditional RNN architecture with many-to-many mapping

The GRU was introduced in 2014 by Cho et Al [15] to solve the gradient limits encountered by classical recurrent networks but also to propose an architecture with fewer parameters to train compared to an LSTM, which can make GRU more powerful in some uses where long-term memory is not required.

### 3.2. Architecture

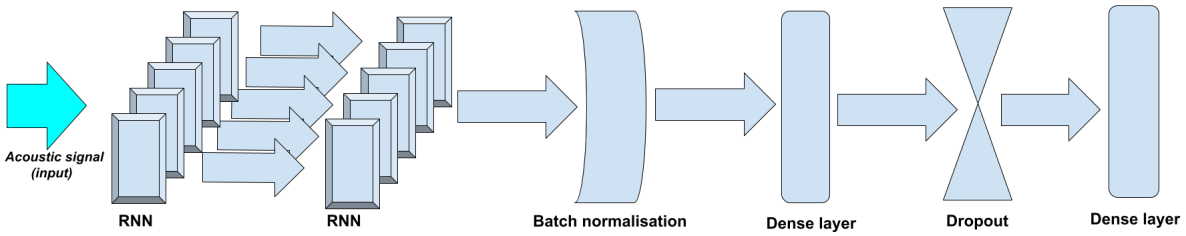


Figure 7: The proposed architecture

Concerning the network architecture, a comparison of RNN cell type is proposed between Vanilla, LSTM and GRU, using a common network design..

RNN is selected as the foundational layer of the architecture with return sequences. This setting allows the RNN to output sequences instead of a single output, which is essential for capturing sequential dependencies in data. A secondary layer without return sequences, which aggregates the sequential outputs of the RNN and feeds them into subsequent layers. This layer helps in extracting high-level features from the sequential outputs generated by the first RNN layer.

To improve the stability and convergence of our model a batch normalization layer is added, after the secondary layer by normalizing the activations of each layer across the mini-batch, reducing internal covariate shift and accelerating training. A Dense layer follows the batch normalization step, serving as a fully connected layer that learns complex mappings between the features extracted by the RNN and the target output.

To prevent over-fitting and improve the generalization ability of our model, a Dropout regularization layer used after the Dense layer. The architecture concludes with a final Dense layer with a linear function for sound power level predictions $L_{SN}$.

## 4. EXPERIMENTS

### 4.1. Setting and metrics

The three RNN models with the architecture mentioned in Figure 7 are learned with $DS1$ (Section 2) to provide comparison data. These models are designed to take as input a sequence

of 60 seconds and 19 features, where each feature represents one-third octave band covers a frequency range from 31.5 Hz to 2 kHz, and predict the $L_{SN}$ value using 60 neurons as output.

For an input sample, the output of the network is a value representing the estimated $L_{SN}$ in dB(A), with the Mean Squared Error (MSE) serving as the loss function between the estimated value $\hat{y}$ and the target $y$, Loss = $(y - \hat{y})^2$, so we compute the cost as the mean loss:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2, \tag{4}$$

along with the evaluation metric Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|. \tag{5}$$

To minimize the cost function MSE, we chose the Adam optimizer, proposed by Kingma and Ba [16], with a learning rate of $10^{-3}$ chosen through the experimentation as illustrated in Figure 8. This choice ensures stable fitting. It minimizes the evaluation metric MAE as well.

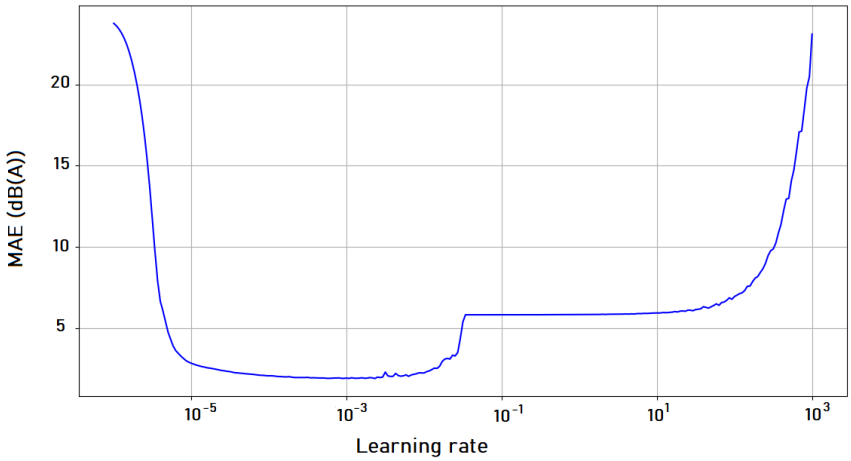


Figure 8: Evolution of the MAE according to the change in the learning rate

Since the model is ready with metrics and hyperparameters, the input data $L_{TN}$ and target $L_{SN}$ consist of 115,398 sequences obtained by dividing the number of seconds of $DS1$ by 60 as the input sequence length. Notably, acoustic data in the various frequency bands include both negative and positive values, so we decided not to normalize the input data.

### 4.2. Experimental results

This subsection presents the experimental results obtained from proposed RNN architecture for the three models: Vanilla, LSTM, and GRU, by dividing $DS1$ into 70 % for training and 30 % for testing. Within the 70 % training set, further partitioned 20 % as a validation subset. This partitioning allowed to refine the models during training while maintaining a separate $DS1$ for validation to monitor performance and prevent overfitting.

During the training phase, all models converged successfully, with a slight overfitting observed for the Vanilla model training, indicated by a marginal difference of 0.4 dB(A) in MAE between the training and validation sets. Notably, the Vanilla model required significantly more evaluation time for the test set, 19 seconds against 5 seconds for the LSTM model and 4 seconds for the GRU model. Both the GRU and LSTM models showed promising performance in the tests, as shown in the following figures.

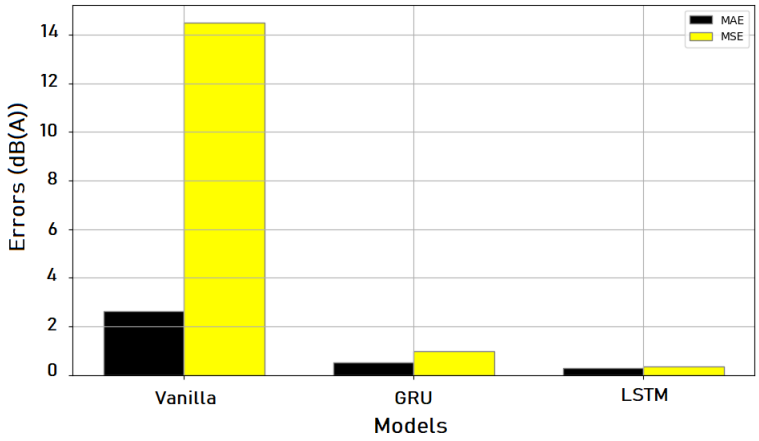


Figure 9: Comparison of Mean Absolute Error (MAE) and Mean Squared Error (MSE) among Models

Figure 9 demonstrates the cost function for the test data and the MAE metric for all three models. Figure 10 illustrates the temporal evolution of the target and predictions across the three models on an example extract.

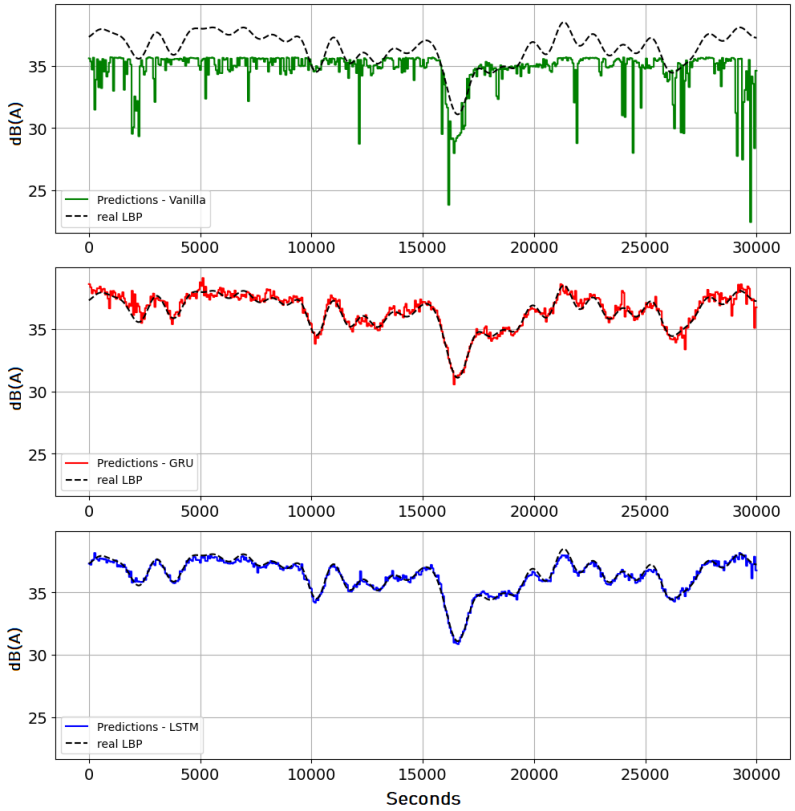


Figure 10: Temporal evolution of earget $L_{SN}$ and predictions across models

The simple RNN "Vanilla," is not suitable to our problem with higher MAE errors and weaker MSE cost function minimization. Therefore, we turned to more advanced RNN cells. While GRU is faster in execution, and LSTM, which performs the best on the test set, as shown in the figures 9 and 10. For test set of $DS1$ results are promising, indicating good performance of the models.

## 5. NMF COMPARISON

### 5.1. Data set

The dataset used in [3] on which NMF has been tested is now used to assess the performance of the proposed approach, with pre-processing of this Data Set ($DS2$).

$DS2$ contains 30 independent wind turbines noise scenes, expressed in third octave bands from 31,5 Hz to 2 kHz. As in [3], the scenes are expressed in dB(A) and propagated at 500 m, 1000 m and 1500 m with the same attenuation filters for 3 different propagation conditions (homogeneous, favorable and highly favorable). Each of this propagated scene are merged with a background noise measured in two different locations following a SNR such as : $SNR = L_{SN} - L_{BN}$ with $SNR = \{-9, -6, -3, 3, 6, 9\}$ dB(A). A complete description of this dataset can be found in [3]. In all it is 3780 scenes of 600 seconds expressed in third octave band that are available.

With this formula, we calculate the new level of $L_{SN}$ according to a given SNR:

$$\mathrm{SN}' = \mathrm{SN} \times \left( \frac{E_{\mathrm{BN}}}{E_{\mathrm{SN}}} \right) \times 10^{0.1 \times \mathrm{SNR}}, \tag{6}$$

where:
- SN represents the specific noise sound pressure in Pascal.
- $E_{\mathrm{BN}}$ and $E_{\mathrm{SN}}$ represents respectively the median of background noise and specific noise for a given scene.

### 5.2. Fine tuning the models

Given that the $DS2$ is small for training and testing our models and also contains a specific measured noise, while the $DS1$ consists of synthesized specific noise, we opted for fine tuning from pre-trained models on the $DS1$. The procedure consists in selecting $SNR = \{-9, 0, 9\}$ dB(A) from the $DS2$ for a single distance of $d = 500$ m, then test the models on the remaining portion of the dataset and also on another dataset with a completely different background noise.

The pre-trained RNN models on $DS1$ are fitting well to the selected portion of $DS2$. Evaluation of their performance MAE between the target $L_{SN}$ and the estimated $\hat{L}_{SN}$ for an other portion with a distance $d = 1000$ m, for various SNR values, as shown in the figure 11:

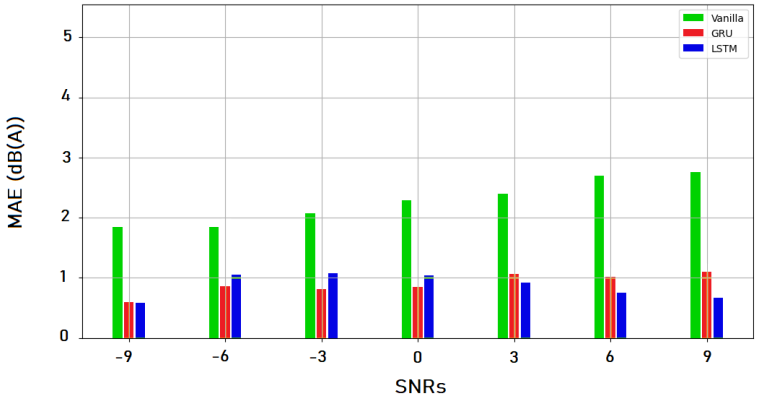


Figure 11: MAE error for different SNR values for each model

For Vanilla, it takes 2 seconds per epoch to fine-tune on a portion of $DS2$, exhibiting weak convergence that may requires additional epochs to more converge. However, both GRU and LSTM models, trained for the same number of epochs as Vanilla, require only 12 milliseconds with robust convergence. Notice that GRU performs well under negative SNRs with additional errors for positive SNRs. LSTM maintains consistent error levels across positive SNRs. The results across

distances and propagation conditions show that, even with fine-tuning on a single specific case, the models still perform better in estimating levels at distances of 1000 m and 1500 m.

### 5.3. Comparison

To compare RNN models with NMF, the actual emergence $E$ and the estimated emergence $\hat{E}$ are computed for each scene, then we calculate the mean absolute error (MAE) of emergence using the formula cited for $M = 30$ scenes in the article [3]:

$$\text{MAE} = \frac{\sum_{m=1}^{M} |E_m - \hat{E}_m|}{M}, \tag{7}$$

Finally, the models are tested with another dataset containing a different background where the NMF was tested, for $d = 500$ m and various SNR.

| Models/SNRs | -9 | -6 | -3 | 0 | 3 | 6 | 9 |
|---|---|---|---|---|---|---|---|
| NMF | 0,790 | 0,819 | 0,664 | 0,334 | 1,289 | 3,319 | 5,672 |
| Vanilla | 0,162 | 0,344 | 0,554 | 0,736 | 0,927 | 1,292 | 1,090 |
| GRU | 0,117 | 0,273 | 0,406 | 0,392 | 0,538 | 0,609 | 0,648 |
| LSTM | 0,098 | 0,223 | 0,368 | 0,407 | 0,597 | 0,605 | 0,596 |

Table 1: Comparison of emergence MAE for different methods

With this dataset which is different from the one used for fine-tuning models, the results are very promising compared with those of NMF, especially for highly positive SNRs that was a problem for NMF method. At high SNRs, the estimation of the emergence is much more sensitive to the error made on the background noise: an underestimation of 0.5 dB(A) on the $L_{BSN}$ at a SNR of 9 dB(A) leads to an overestimation of the emergence of 2 dB(A). The NMF's initial error at these SNRs prevented it from being applicable in cases where wind noise is predominant (e.g. night period).

## 6. CONCLUSION

This paper proposes the utilization of deep learning models for estimating wind turbine noise from total noise. Specifically, we employ RNNs with a fixed architecture for each RNN unit/cell and we compare with NMF method on synthesized scenes combining measured background noise and wind noise synthesized from weather and sound power data. RNNs all have fewer MAEs than NMF, in particular LSTM is the most stable model at different SNRs.

To be sure of the results and enrich the work, there are avenues to be explored, working on the generalization of the method, testing the method with several types of wind turbine and several wind turbines in the same scene.

There are several ways for extending deep learning models, for example with two different RNN units in the same model, implementing normalization and preprocessing techniques with a larger and more realistic dataset, investigating convolutional neural networks CNNs, which have shown promising initial results in our parallel work, and considering the integration of attention mechanisms if necessary.

**REFERENCES**


1. Alice Freiberg, Christiane Schefter, Maria Girbig, Vanise C. Murta, and Andreas Seidler. Health effects of wind turbines on humans in residential settings: Results of a scoping review. *Environmental Research*, 169:446–463, February 2019.
2. Stefan Oerlemans, Murray Fisher, Thierry Maeder, and Klaus Koegler. Reduction of Wind Turbine Noise Using Optimized Airfoils and Trailing-Edge Serrations. *Aiaa Journal - AIAA J*, 47:1470–1481, June 2009.
3. Jean-Rémy Gloaguen, David Ecotière, Benoit Gauvreau, Arthur Finez, Arthur Petit, and Colin Bourdat. Automatic estimation of the sound emergence of wind turbine noise with nonnegative matrix factorization. *The Journal of the Acoustical Society of America*, 150:3127–3138, October 2021.
4. Emmanouil Benetos, Margarita Kotti, and C. Kotropoulos. Application of Nonnegative Matrix Factorization to Musical Instrument Classification. May 2012.
5. Po-Sen Huang, Minje Kim, Mark Hasegawa-Johnson, and Paris Smaragdis. Deep learning for monaural speech separation. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1562–1566, May 2014. ISSN: 2379-190X.
6. Chengyun Zhang, Haisong Zhan, Zezhou Hao, and Xinghui Gao. Classification of Complicated Urban Forest Acoustic Scenes with Deep Learning Models. *Forests*, 14:206, January 2023.
7. Arseniy Gorin, Nurtas Makhazhanov, and Nickolay Shmyrev. *DCASE 2016 Sound Event Detection System Based on Convolutional Neural Network*. September 2016.
8. Gino Iannace, Giuseppe Ciaburro, and Amelia Trematerra. Wind Turbine Noise Prediction Using Random Forest Regression. *Machines*, 7:69, November 2019.
9. Vlastimir Nikolić, Dalibor Petković, Lip Yee Por, Shahaboddin Shamshirband, Mazdak Zamani, Žarko Ćojbašić, and Shervin Motamedi. Potential of neuro-fuzzy methodology to estimate noise level of wind turbines. *Mechanical Systems and Signal Processing*, 66-67:715–722, January 2016.
10. Benjamin Cotté. Extended source models for wind turbine noise propagation. *The Journal of the Acoustical Society of America*, 145:1363–1371, March 2019.
11. David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. Publisher: Nature Publishing Group.
12. Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, March 1994. Conference Name: IEEE Transactions on Neural Networks.
13. Sepp Hochreiter and Jürgen Schmidhuber. Long Short-term Memory. *Neural computation*, 9:1735–80, December 1997.
14. Felix A. Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to Forget: Continual Prediction with LSTM. *Neural Computation*, 12(10):2451–2471, October 2000. Conference Name: Neural Computation.
15. Kyunghyun Cho, Bart Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Y. Bengio. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. June 2014.
16. Diederik Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, December 2014.