ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Constraint Guided Recurrent Convolutional AutoEncoders for Condition Indicator Estimation

Maarten Meire[1,2], Quinten van Baelen[1,2], Ted Ooijevaar[3] and Peter Karsmakers[1,2] *

1 - KU Leuven - Dept. of Computer Science; Leuven.AI
3000 Leuven - Belgium

2 - Flanders Make @ KU Leuven

3 - Flanders Make vzw - CoreLab MotionS
3001 Leuven - Belgium

Abstract. To effectively monitor industrial applications, an accurate estimate of their condition, or a Condition Indicator (CI), is required. Recently, a CI estimation method called Monotonically Constraint Guided Autoencoders (MCGAE) was introduced, which constrains the CI to remain within predefined ranges for both normal and anomalous data while also enforcing monotonic behavior over time. However, that work employed a Convolutional AutoEncoder (CAE) architecture that did not capture longer-term temporal dependencies. In this study, we evaluate a recurrent CAE variant that incorporates a Long Short-Term Memory (LSTM) layer and compare its performance against alternative architectures without a recurrent component. Experimental results on a bearing run-to-failure dataset, indicate that the addition of LSTM improves the monotonic behavior of the estimated CI.

1 Introduction

When considering the monitoring of industrial assets, one of the crucial parts is rotating machinery. The primary source of failure in such machinery is due to Rolling Element Bearings [1, 2]. The monitoring of REB is most commonly done through vibration analysis [2, 3]. Various methods, such as physical, statistical, and data-driven, have been used for the purpose of condition monitoring [2, 4, 5]. For the data-driven models, Deep Learning (DL) methods have been receiving more attention in recent years [4]. These will be the focus of this work.

A wide range of DL models have been used in the field of condition monitoring. As the vibration data is sequential, Recurrent Neural Networks (RNN), often Long Short-Term Memory (LSTM), are commonly used to analyze this data [5]. In [6] an LSTM was used to estimate the Remaining Useful Life (RUL) of the main bearing in a wind turbine using the sideband energy ratios and characteristic frequencies as input. The raw vibration data was provided to an LSTM in [7] and a Condition Indicator (CI) was estimated by comparing the model predictions with the actual data. These models often work on the raw

*This research received funding from the Flemish Government (AI Research Program). This research has received support of Flanders Make, the strategic research centre for the manufacturing industry.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

data or handcrafted features. However, other networks could be combined with RNN to possibly further enhance their performance [5]. An example of this is the combination of CNN and RNN in [8], where the convolutional layers are made recurrent to estimate the RUL of a bearing. A multistage Convolutional AutoEncoder (CAE) was used in [9] to estimate a CI, and an LSTM was used in combination with this CI for RUL estimation.

In [10], a constrained learning method called Monotonically Constraint Guided Autoencoder (MCGAE) was proposed to estimate a CI that satisfies an enforced fault-detection threshold and exhibits monotonic behavior to better represent the condition of an REB. However, this method was evaluated only using a CAE architecture. As discussed earlier, an RNN or a hybrid CRNN design, may offer advantages for MCGAE by better capturing temporal dependencies. Therefore, this work extends the CAE architecture with an LSTM layer to investigate whether incorporating recurrent dynamics can more effectively exploit the temporal information in the data than the existing CNN-based implementation.

The main contributions of this work are:

- A comparison of three architectures: (i) a 1D CAE (*Conv1D*), (ii) a 2D CAE (*Conv2D*), and (iii) a hybrid model combining *Conv1D* with an LSTM layer (*Conv1D-LSTM*).
- An evaluation of how different input sequence lengths (i.e., numbers of time steps) affect model performance.

The rest of the paper is organized as follows. In Section 2 the MCGAE algorithm will be explained. The dataset, preprocessing, experimental details and metrics are discussed in Section 3. This is followed by the performed experiments and results in Section 4. Finally, this work is concluded in Section 5.

2 Monotonically Constraint Guided Autoencoder

The goal of Monotonically Constraint Guided Autoencoder (MCGAE) [10] is to subject the learning objective of an AE to a set of constraints: (i) normal data should be mapped inside of a sphere, with radius R_1, around the origin, (ii) anomalous data should be mapped outside of a larger sphere, with radius R_2, around the origin, and (iii) data should be mapped further away from the origin in relation with time, in this case when in the life time of a machine the data was captured. It is important to note that the final constraint should be applied separately for the life time of each machine, hereafter called a run, as there is, in general, no relation between data from different machines at a certain point

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

in time. This results in the following constrained learning objective

$$\begin{aligned} &\underset{\boldsymbol{\theta}_{\mathcal{E}},\boldsymbol{\theta}_{\mathcal{D}}}{\operatorname{argmin}} \frac{1}{N} \sum_{\boldsymbol{x}\in\mathcal{N}} \left\| \boldsymbol{x} - \mathcal{D}\left(\mathcal{E}\left(\boldsymbol{x} \mid \boldsymbol{\theta}_{\mathcal{E}}\right) \mid \boldsymbol{\theta}_{\mathcal{D}}\right)\right\|^2, \\ &\text{s.t. } \forall \boldsymbol{x} \in \mathcal{N} : f(\mathcal{E}(\boldsymbol{x}|\boldsymbol{\theta}_{\mathcal{E}})) \in B[0, R_1], \\ &\quad \forall \boldsymbol{x} \in \mathcal{A} : f(\mathcal{E}(\boldsymbol{x}|\boldsymbol{\theta}_{\mathcal{E}})) \notin B[0, R_2], \\ &\quad \forall r, \forall \boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_2} \in \mathcal{S}^r : t_1 < t_2 < t_a^r \Rightarrow \|f(\mathcal{E}(\boldsymbol{x}_{t_1}|\boldsymbol{\theta}_{\mathcal{E}}))\| < \|f(\mathcal{E}(\boldsymbol{x}_{t_2}|\boldsymbol{\theta}_{\mathcal{E}}))\|, \end{aligned} \tag{1}$$

where $\mathcal{E}$ and $\mathcal{D}$ are the encoder and decoder, respectively, $\boldsymbol{\theta}$ represents the weights, $B[0, R_1]$ and $B[0, R_2]$ denote the closed balls, with radius R_1 and R_2, respectively, around the origin, r denotes the considered run, t_a^r is the time at which the first anomalous point occurs for run r, $\mathcal{A}$ the set of anomalous points, $\mathcal{S}^r = \mathcal{N}^r \cup \mathcal{U}^r$, with $\mathcal{N}^r$ the set of normal points of run r, $\mathcal{U}^r = \{u_i^r\}_{i=1}^{U^r}$ being the set of unlabeled points of run r, with U^r the amount of unlabeled points of run r, and $\boldsymbol{x}_{t_i}$ is the sample on time t_i of run r. $f()$ represents the recurrent LSTM layer. When no recurrent layer is used $f(\boldsymbol{x}) = \boldsymbol{x}$. To allow for better generalization, the assumption is made that $R_1 < R_2$, as this creates some separation between normal and anomalous data. CGGD [11] is used to optimize this optimization problem. Note that when $f()$ includes an LSTM layer, special care is required to update the encoder's weights so that both the learning objective and the satisfaction of constraints can be improved. To accommodate this, an implementation similar to that proposed in [12] is adopted.

3 Experimental setup

Dataset The Smart Maintenance (SM) dataset contains vibration data collected from 70 accelerated run-to-failure tests of bearings, hereafter called runs, using 7 bearing test rigs. This dataset was previously used in [10].

In this study data from all test rigs was used, with a subset of 8 runs being omitted due to different stopping conditions. This results in a dataset of 41 accelerated runs, that end with a fatigue fault, and 21 fully healthy runs, without an initial indentation. From each test rig the first 2 accelerated runs and the first fully healthy runs will be used for training, while the remaining runs will be used for evaluation purposes.

Preprocessing The preprocessing of the acceleration signals into log mel spectra is completely analogous to the one used in [10]. Depending on the model that will be considered, multiple consecutive samples, or time steps, will be combined before being provided to the model. This is done by using a sliding window with a varying size and a hop size of 8 time steps.

Experimental details In the experiments a total of 3 models are used: i) a 1D CAE (*Conv1D*), ii) a 2D CAE (*Conv2D*), and iii) a combination of *Conv1D* with an additional LSTM layer (*Conv1D-LSTM*). The same general architecture as in [10] was used, with the amount of filters and neurons in the layers being

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

empirically selected. The encoder is built using 3 convolutional blocks with 64, 64, and 32 filters, respectively, followed by a Fully Connected layer (FC), with 16 neurons. The decoder is built using a FC layer, with 1024 neurons, and 3 deconvolutional blocks, with 32, 64, and 64 filters, respectively. A final convolutional layer with 1 filter is used as the output layer. As mentioned earlier the *Conv1D-LSTM* combines *Conv1D* with an additional LSTM layer, consisting out of 16 cells.

The radii R_1 and R_2 were set to 0.25 and 0.5, respectively, due to a tanh activation in the LSTM layer limiting the upper bound of the norm in the latent space. The models were trained for 300 epochs with the Adam optimizer [13] and a learning rate of $1e^{-3}$. The model was saved each epoch if the performance improved. The performance is determined by the MCGAE learning objective, the reconstruction error of the AE, and the satisfaction ratio of the constraints, which is the percentage of points for which the constraints are satisfied. If the satisfaction ratio is above 95%, only the learning objective is considered.

Metrics A comparison of the models will be made in two ways: (i) the discriminative performance between normal and anomalous behavior, and (ii) the monotonicity of the CI as a measure of quality. The former will be evaluated using the balanced accuracy (BA), defined as the mean of the recall for the normal and anomalous classes. It was opted to use the BA instead of metrics such as F1 score, recall or precision, for consistency with the evaluation approach used in [10]. In this work anomalous data is considered as the positive class. For the latter evaluation, it is expected that a CI should show a monotonic trend, either increasing or decreasing, until a fault occurs. This can be evaluated using the Spearman's ρ [14], as was previously done in [10].

4 Experiments and results

Experiments To compare the different models a fixed training and testing set were created. More specifically, the first 2 accelerated runs and the first fully healthy run from each test rig of the SM dataset were used for training, and the remaining runs were used for testing. The training set is then further split 75% and 25% for training and validation, respectively. All experiments were repeated 4 times, using different seeds for model generation and batch shuffling during training. The amount of time steps that are provided to the models will be set to 8, 16, 32, and 64 to investigate the effect of more temporal information. The hop size was fixed to 8, as to retain more training data. A hop size of 1 is used for testing. As *Conv1D* uses 1D convolutions only a single time step can be used as input. It was opted to use only the final time step of the multiple time steps that are provided to the other models for *Conv1D*.

Results The results in terms of the BA for the different models and window sizes are shown in Table 1. It can be seen that the BA is quite similar for all models and window sizes. However, *Conv1D-LSTM* does show a slightly lower BA, with a slightly decreasing trend when more time steps are used. This lower

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Window size	8	16	32	64
Conv1D	$\mathbf{0.875 \pm 0.029}$	$\mathbf{0.876 \pm 0.040}$	$\mathbf{0.868 \pm 0.039}$	$\mathbf{0.880 \pm 0.027}$
Conv2D	0.869 ± 0.042	0.865 ± 0.057	$\mathbf{0.859 \pm 0.064}$	$\mathbf{0.864 \pm 0.051}$
Conv1D-LSTM	0.842 ± 0.080	0.834 ± 0.066	0.814 ± 0.095	0.816 ± 0.095

Table 1: The BA results for the different models and different window sizes. The significantly better results for each window size are in bold.

BA is likely due to *Conv1D-LSTM* showing a slower increase in the CI and also a lower CI estimation for anomalous data, resulting in the CI exceeding the threshold slightly later in comparison with the other models. However, the CI estimated by *Conv1D-LSTM* also tends to show less variability between consecutive time steps. This could be attributed to the tanh activation at the output of the LSTM layer resulting in non-linear behavior in the latent space, and thus altering the behavior of the CI. It could also be that the sequential processing of the time steps by the LSTM layer results in a more controlled CI. It can also be argued that this delay in exceeding the threshold might not be a problem in specific applications, as it is relatively small.

Next to the discriminative performance, the quality of the CI is evaluated using the Spearman's ρ, for which the results are shown in Table 2. These results show that *Conv1D-LSTM* achieves a noticeably better Spearman's ρ in comparison with the other models. This indicates the effectiveness of the LSTM layer in the model, compared to the other methods. The window size appears to only have a small impact on the performance of the models.

5 Conclusion

In this work the combination of a CAE and LSTM, to improve the usage of the temporal information, with the MCGAE algorithm was investigated. The effect of adding more time steps as input to the models was also investigated. The SM dataset, containing run-to-failure vibration data from REB, was used to compare *Conv1D*, *Conv2D*, and *Conv1D-LSTM*.

The results show a similar BA for all models, with *Conv1D-LSTM* performing slightly worse. Although this is mostly due to *Conv1D-LSTM* exceeding the threshold slightly later, which could be argued to not be a problem in specific applications. For the Spearman's ρ a different behavior can be seen, as *Conv1D-LSTM* performs noticeably better than the other models. The impact of window size also appears to only have a small impact on the performance.

Further research could include the use of a more elaborate LSTM network, which could further exploit the temporal information and improve generalization.

References

[1] Ahmed Nabhan, Nouby Ghazaly, Abdelhalim Samy, and MO Mousa. Bearing fault detection techniques-a review. *Turkish Journal of Engineering, Sciences and Technology*, 3(2), 2015.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Window size	8	16	32	64
Conv1D	0.356 ± 0.229	0.370 ± 0.210	0.333 ± 0.210	0.367 ± 0.221
Conv2D	0.269 ± 0.259	0.246 ± 0.299	0.266 ± 0.374	0.261 ± 0.349
Conv1D-LSTM	$\mathbf{0.510 \pm 0.301}$	$\mathbf{0.513 \pm 0.263}$	$\mathbf{0.507 \pm 0.299}$	$\mathbf{0.525 \pm 0.265}$

Table 2: The Spearman's ρ results for the different models and different window sizes. The significantly better results for each window size are in bold.

[2] Vigneshwar Kannan, Tieling Zhang, and Huaizhong Li. A review of the intelligent condition monitoring of rolling element bearings. *Machines*, 12(7), 2024.

[3] Duy-Tang Hoang and Hee-Jun Kang. A survey on deep learning based bearing fault diagnosis. *Neurocomputing*, 335, 2019.

[4] Krish K. Raj, Shahil Kumar, and Rahul R. Kumar. Systematic review of bearing component failure: Strategies for diagnosis and prognosis in rotating machinery. *Arabian Journal for Science and Engineering*, 50(8), December 2024.

[5] Liuyang Song, Tianjiao Lin, Ye Jin, Shengkai Zhao, Ye Li, and Huaqing Wang. Advancements in bearing remaining useful life prediction methods: a comprehensive review. *Measurement Science and Technology*, 35(9), June 2024.

[6] Linli Li and Qifei Jian. Remaining useful life prediction of wind turbine main-bearing based on lstm optimized network. *IEEE Sensors Journal*, 24(13):21143–21156, 2024.

[7] Yasir Saleem Afridi, Laiq Hasan, Rehmat Ullah, Zahoor Ahmad, and Jong-Myon Kim. Lstm-based condition monitoring and fault prognostics of rolling element bearings using raw vibrational data. *Machines*, 11(5), 2023.

[8] Biao Wang, Yaguo Lei, Tao Yan, Naipeng Li, and Liang Guo. Recurrent convolutional neural network: A new framework for remaining useful life prediction of machinery. *Neurocomputing*, 379:117–129, 2020.

[9] Zhaozong Wang, Jiangfeng Cheng, Hui Zheng, Xiaofu Zou, and Fei Tao. Multistage convolutional autoencoder and bcm-lstm networks for rul prediction of rolling bearings. *IEEE Transactions on Instrumentation and Measurement*, 72:1–13, 2023.

[10] Maarten Meire, Quinten Van Baelen, Ted Ooijevaar, and Peter Karsmakers. Constraint guided autoencoders for joint optimization of condition indicator estimation and anomaly detection in machine condition monitoring. *Machine Learning*, 114(7), May 2025.

[11] Quinten Van Baelen and Peter Karsmakers. Constraint guided gradient descent: Training with inequality constraints with applications in regression and semantic segmentation. *Neurocomputing*, 556:126636, 2023.

[12] Yonas Tefera, Quinten Van Baelen, Maarten Meire, Stijn Luca, and Peter Karsmakers. Constraint-guided learning of data-driven health indicator models: An application on bearings. *International Journal of Prognostics and Health Management*, 16(2), 2025.

[13] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, Conference Track Proceedings*, 2015.

[14] Stephen Kokoska and Daniel Zwillinger. Crc standard probability and statistics tables and formulae, student edition, March 2000.