Enhancing the Effectiveness of Neural Networks in Predicting Railway Track Degradation

Mahdieh Sedghi

Abstract With the advancements in artificial intelligence and the emergence of shallow and deep learning algorithms, there is a growing demand for precise and efficient methods of predicting asset degradation across various industries. This has led to a resurgence of interest in artificial neural networks (ANNs) as a solution. In this study, the aim is to evaluate the potential of using ANNs, as well as specific types of ANNs that are equipped to handle sequential data, such as Recurrent Neural Networks (RNNs), Long Short-Term Memory (LSTMs), and Gated Recurrent Units (GRUs), for predicting individual track geometry degradation in railway systems. The performances of these ANNs were evaluated by comparing their ability to predict degradation patterns for 110 segments obtained from a 30-km track section in Northern Sweden. Hyperparameters, which include the number of hidden layers, the number of neurons per layer, the learning rate, the activation function, batch size, and the optimizer, play a crucial role in defining the architecture and behaviour of a neural network. Hyperparameter tuning can significantly impact the accuracy and generalization ability of the ANNs. Therefore, the impact of hyperparameter tuning on the performance of each algorithm was also explored. The results indicated that GRU outperformed simple RNN, LSTM, and feedforward ANN in terms of prediction accuracy in predicting track geometry degradation. The results provide insights into using different ANN algorithms to predict asset degradation, emphasizing the importance of proper hyperparameter tuning in achieving optimal performance.

Keywords Neural networks · Recurrent neural network (RNN) · Long short-term memory (LSTM) · Gated recurrent unit (GRU) · Railway · Track geometry

M. Sedghi (✉)
Luleå University of Technology, Luleå, Sweden
e-mail: mahdieh.sedghi@ltu.se

U. Kumar et al. (eds.), *International Congress and Workshop on Industrial AI and eMaintenance 2023*, Lecture Notes in Mechanical Engineering,
https://doi.org/10.1007/978-3-031-39619-9_48

1 Introduction

Accurate prediction of track geometry degradation is critical for efficient railway operations, as it can aid in accident prevention and improve maintenance planning [1]. However, predicting track degradation is challenging due to the complexity of factors that affect it, including traffic, load, maintenance, and environmental factors [2]. Predicting track degradation involves analyzing how these factors interact and affect the degradation rate or level at a specific point in time. Artificial Neural Networks (ANNs) have been suggested as a potential solution for predicting track degradation, considering multiple input features, including traffic patterns, load conditions, maintenance schedules, and environmental factors [3]. NNs are a machine learning model that mimics the information processing mechanism of the human brain. There are various types of ANNs, including Feedforward-ANNs (FF-ANN), Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNNs), and Deep Belief Networks (DBNs), among others. It is a common approach to use historical degradation data to train a NN model, and the resulting model is applied to forecast degradation in unseen data [4]. The degradation rate or level at a specific point in time can be the output variable of the ANN [4].

The degradation pattern of a railway track can also be treated as a time series, and predicting its future degradation can be approached as a time series forecasting problem [5]. Historically, there has been a common belief that ANNs are not ideal for time series, primarily due to the typically short length of most time series [3, 6]. However, in the recent large-scale forecasting competition organized by the International Institute of Forecasters (IIF), called the M4 competition, RNN achieved impressive performance and a hybrid model combining exponential smoothing and RNN emerged as the winner [7]. Given the increasing use of sensors and condition monitoring tools for the predictive maintenance of railway tracks, a vast amount of data can be leveraged to predict degradation patterns and optimize track maintenance. RNNs have shown great potential in dealing with sequential data [3, 8], such as degradation patterns, and can be used to forecast future trends in track conditions. LSTM and GRU are two variants of RNNs that have demonstrated strong performance in learning long-term dependencies in sequential data [9]. However, most previous studies [4, 10–13] have neglected the high capabilities of RNNs in dealing with sequential data, such as degradation patterns.

In addition, an essential aspect of utilizing ANNs for degradation analysis is the selection of suitable structures and hyperparameters. Hyperparameter tuning involves selecting the best values for NN model parameters, such as the number of hidden layers, the number of neurons per layer, the learning rate, the activation function, batch size, and the optimizer [14]. Finding the optimal combination of hyperparameters is essential to enhance model accuracy [15]. However, in the reviewed literature on applying ANNs for railway track maintenance [4, 10–13, 16–18] in this paper, the issue of hyperparameter tuning has not received sufficient attention, and ANNs have typically been constructed using a manual search approach. To enhance the performance of ANNs in predicting degradation patterns, it may be necessary to

further investigate the potential impact of advanced techniques, such as Bayesian optimization, for optimizing hyperparameters.

This paper aims to investigate the performance of different ANNs, i.e., FF-ANN, Simple RNN, LSTM, and GRU, in predicting track geometry degradation. This study is the first of its kind to provide a comprehensive overview of the predictive capabilities of these neural networks for track geometry predictions while also addressing the issue of hyperparameter tuning.

The second section of the paper provides a literature review on the use of ANNs for predicting railway track geometry. In section three, the ANNs used in this study are explained. The fourth section elaborates on the hyperparameter tuning process adopted to optimize the performance of the neural network models. Section five presents a detailed analysis of the neural network models' performance based on data from a case study on a track section with 110 segments. Finally, section six highlights the findings' conclusions and implications for the railway track maintenance field.

2 Review of the Literature

ANNs have been suggested as a method for analyzing degradation in railway tracks, aiming to predict the degradation using diverse input features. [4, 10–12]. Several studies have used ANN to predict different aspects of railway track degradation. Guler [4] used ANN to predict track geometry degradation by considering track structure, traffic characteristics, layout, environmental factors, geometry, and maintenance and renewal data. Moridpour et al. [11] explored the impact of increased tram traffic on rail infrastructure and presented an ANN model to predict tram track degradation using existing data to reduce maintenance costs and improve system performance. Lee et al. [12] used an ANN and support vector regression to predict track geometry degradation based on several input variables, such as the track quality index value, curvature, velocity, and million gross tonnages. Ali et al. [13] applied a backpropagation-ANN to construct a deterioration model for railway tracks in the UK, using factors such as track geometry, ballast fouling index, train speed, catch pits, ballast age, and sleeper age. Gerum et al. [19] used RNN to predict the track defects and classify them into two groups of red and yellow defects. Falamarzi et al. [20] used a regression model and an ANN model to predict tram track gauge deviation, and both showed good performance with determination coefficients above 0.7. Finally, Khajehei et al. [10] used track geometry measurements, asset information, and maintenance history to predict track geometry degradation by ANN.

In other related studies, but not necessarily focused on degradation prediction, Bruin et al. [16] proposed using LSTM for fault detection and identification in railway track circuits based on commonly available measurement signals, achieving a correct classification rate of 99.7% and outperforming a convolutional network. Popov et al. [17] used ANN on data from a high-speed line in the UK to assess the efficiency of railway track maintenance.

The review of the literature revealed two gaps in the research on using ANNs to predict railway track degradation patterns:

- Firstly, none of the reviewed articles addressed the issue of hyperparameter tuning. The ANN structure was constructed using a manual search approach, which lacks a sophisticated hyperparameter tuning method. This could potentially limit the accuracy and efficiency of the model.
- Secondly, all the reviewed articles used only one type of ANNs, mostly FF-ANN, while neglecting the high capabilities of RNN in dealing with sequential data as degradation patterns. Therefore, incorporating RNN into the degradation analysis could provide more precise track degradation predictions and help enhance the decision support system for railway track management.

To address the above mentioned gaps, four ANNs, i.e., FF-ANN, RNN, LSTM, and GRU, are used for predicting track geometry degradation while using Bayesian optimization for hyperparameter optimization.

3 Neural Network Methods

3.1 FF-ANN

The most frequently used form of ANNs is the FF-ANN model, which comprises three types of layers: input, output, and hidden. In this model, each output layer node is linked to a target variable, while the input layer nodes are associated with predictor variables, as shown in Fig. 1 [18, 21]. The number of hidden layers and the number of nodes (neurons) in each layer together determine how complex the FF-ANN model is. FF-ANNs with multiple non-linear hidden layers can capture complex relationships between input and output variables, but limited training data may lead to overfitting due to sampling noise creating a complex relationship that does not exist in the test data [18, 21].

The number of hidden layers in an FF-ANN model is proportional to the complexity of the research object, and experiments are used to determine the optimal number of hidden layers [18].

3.2 RNN

RNNs are designed to handle sequential data, often used for time series analysis [3, 8]. Therefore, RNNs can be used for degradation analysis, which aims to monitor and predict the system's condition over time The RNN's structure resembles a multilayer perceptron but with time-delay connections between hidden units to retain information about the past and discover temporal correlations between distant events in the

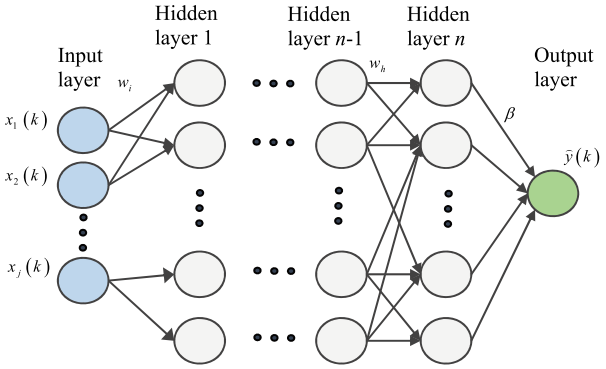
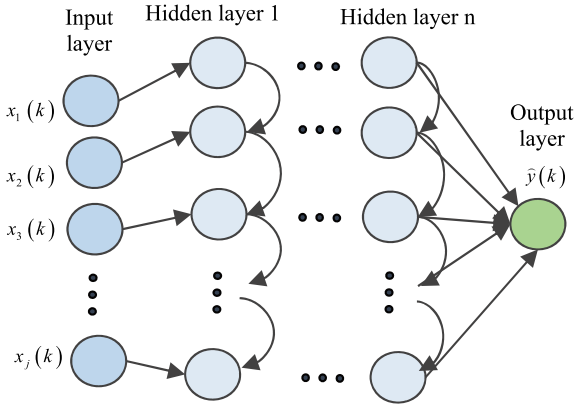


Fig. 1 FF-ANN structure

Fig. 2 RNN structure



data. [22, 23], as shown in Fig. 2. This approach allows building ANNs to process and analyze data effectively over time.

Even though RNNs can handle sequential data, such as time series, RNNs cannot learn long-term dependencies due to the vanishing-gradient problem [9, 24, 25]. The vanishing-gradient problem refers is a challenge encountered in RNNs that arises when the gradients computed for each time step are multiplied by the recurrent weight matrix, causing them to diminish in magnitude over time [9]. This problem causing current information to be prioritized over past events and hindering the learning of long-term dependencies [9]. As a result, LSTM and GRU models were developed to address these issues [24]. LSTM addresses this by controlling the flow of information within neurons using a gating mechanism that regulates the addition and deletion of information from an iteratively propagated cell state [9].

LSTM cells have three gates–input, forget, and output–to modify a cell state vector, which is iteratively propagated to capture long-term dependencies [9, 24, 25]. The controlled information flow within the cell helps the network remember

multiple time dependencies with varying characteristics [24]. GRU has a simpler cell structure than LSTM and uses a gating system with only an update and reset gate [3, 24]. Hewamalage et al. [3] provided further details on the mathematical models and structure of RNN, GRU, and LSTM.

4 Hyperparameter Tuning

Choosing the right architecture and hyperparameters is crucial for implementing ANNs in degradation analysis, as they heavily impact the behaviour of training algorithms and model performance [14]. The main hyperparameters of a NN include [14, 26]:

- The number of hidden layers: ANNs can have one or more hidden layers.
- The number of neurons per layer: An appropriate number of neurons should be selected to prevent overfitting or underfitting.
- Activation functions: Different activation functions, such as ReLU, sigmoid, and Tanh, introduce nonlinearity into the NN. The choice of activation function can significantly impact the model's performance.
- Learning rate: The learning rate controls the step size during optimization and determines how quickly the model converges.
- The number of epochs: The number of epochs determines how many times the training process will iterate over the entire training set.
- Batch size: During training, data is processed in batches. The batch size is a hyperparameter that determines the number of samples in each batch.
- Optimizer: The optimizer is used to update the model parameters during training. Different optimizers, such as stochastic gradient descent (SGD) and Adam, are available.

4.1 Bayesian Optimization

Bayesian optimization is a powerful and effective method for hyperparameter tuning in ANNs [14, 26]. A search space for hyperparameters is defined to implement Bayesian optimization, along with an acquisition function that balances exploitation and exploration [27, 28]. The acquisition function determines where to sample next based on the current state of the model and the target function [27]. The process is repeated until the optimal hyperparameters are found, or a stopping criterion is met [27].

5 Results and Discussion

Historical data from track Sect. 119 (TN-HO19), which spans 30 kms along the Swedish Iron ore line between Boden and Luleå, is utilized to assess the performance of investigated ANNs.

5.1 Data Preparation

5.1.1 Data Collections

The data used in the study included the shortwave measure of the rail's longitudinal level, which is an important track geometry variable. The shortwave measurement is defined as the amplitude of longitudinal waves with wavelengths between 3 and 25 m, as per the EN 13,848–1:2017 standard [29]. The study utilizes data gathered between 2007 and 2022 using a measurement train that records shortwave amplitude for each 25 cm of both rails. This results in 800 measurements per 200 m segment.

5.1.2 Data Cleaning

To make the data less sensitive to errors, the measurement data was segmented into 200-m lengths, and segment statistics were calculated. Missing observations were removed, and segments with less than 50% complete observations were considered missing. Data for objects such as switches, crossings, or platforms were removed as they were not relevant to the study.

5.1.3 Data Scaling

A standard scaling procedure is used to normalize the features of the dataset. Standard scaling involves subtracting the mean of each feature from its values and dividing it by its standard deviation.

5.2 Application and Evaluation of ANNs

5.2.1 Input Features

For the input feature, the standard deviation of longitudinal level and history of maintenance actions for 110 segments of track Sect. 119 are used to train the ANNs.

Table 1 Performance indicators

Performance indicators	Mathematical formulas
MAE	$\sum_{i=1}^{N} (y_i - \widehat{y}_i) \Big/ n$
MSE	$\sum_{i=1}^{N} (y_i - \widehat{y}_i)^2 \Big/ n$
MAPE	$(100/n) \sum_{i=1}^{N} \left\lvert \left(y_i - \widehat{y}_i\right) \Big/ y_i \right\rvert$
Prediction accuracy	$1 - (100/n) \sum_{i=1}^{N} \left\lvert \left(y_i - \widehat{y}_i\right) \Big/ y_i \right\rvert$

5.2.2 Performance Indicators

Various performance indicators, including Mean Absolute Error (MAE), Mean Squared Error (MSE), Mean Absolute Percentage Error (MAPE), and prediction accuracy based on MAPE, are used to evaluate the performance of ANNs, as shown in Table 1.

5.2.3 Pseudocode of ANNs

Two scenarios are being considered in this paper. The first scenario is called N-HO and involves no hyperparameter optimization. In N-HO, no hyperparameter optimization is performed, meaning the hyperparameters are set to values found by manual search or suggested values in previous studies. The second scenario, HO, involves hyperparameter optimization by Bayesian optimization. Pseudocodes for both scenarios are presented in Tables 2 and 3. The hyperparameters tested in HO are the number of neurons per second and third layers, activation functions, and the number of epochs. In both scenarios, the Python programming language version 3.10 is used, along with the TensorFlow library [30]. Additionally, the BayesianOptimization package from the Bayes_opt library is utilized for hyperparameter optimization in HO.

The development of a NN model with TensorFlow in Python involves the following steps:

- Feature scaling: A standard feature scaling is applied.
- Splitting data into train and test sets: 80% of the data are used as training set.
- Building the NN model: The next step is identifying the number of layers, the activation functions, the number of neurons in each layer, and other hyperparameters. This paper chooses a NN model with one input layer, two hidden layers, and one output layer. In N-HO, the ReLU activation function is selected for hidden layers, and the Linear activation function is selected for the output layer.
- Training the model: Once the NN model has been defined, the next step is to train it on the data. Training the model involves optimizing the model's parameters

Table 2 Pseudocode for N-HO

1	**For** s=1: Segments
N	**Predict track geometries by ANNs based on hyperparameters space (Q)**
	Input (Hyper parameters space *Q*)
2.1	**Scale** *data by Standard Scaler liberary*
2.2	**Split** *the historical degradation data for each segment into training and test data*
2.3	**Add** *one input layer, two hidden layers, and one output layer for ANNs (FF-ANN, simple RNN, LSTM, GRU) structure based on Q with activation functions "ReLU" for hidden layers and "Linear" for the output layer.*
2.4	**Compile** *ANNs with TensorFlow (loss=Mean Squared Error (), optimizer=Adam (), metrics=Root Mean Squared Error ())*
2.5	**Fit** ANNs *(FF-ANN, simple RNN, LSTM, GRU)* with TensorFlow
2.6	**Output** *MSE, MAE, and MAPE for test data and predicted values by the models*

(weights and biases) using an Adam optimization algorithm. During training, the model is iteratively updated based on the mean squared error (loss) between its predictions and the true values in the training data.

- Evaluating the model: After training, various performance metrics explained in Table 1 are used.
- Tuning the hyperparameters: In HO, Bayesian Optimization is used to find the optimal hyperparameters and run the model again based on the optimized values of hyperparameters.

5.2.4 Comparing the Performance of ANNs

Table 4 presents performance indicators for four different ANNs: FF-ANN, simple RNN, GRU, and LSTM. For each NN, two scenarios are considered: one without hyperparameter optimization (N-HO) and one with hyperparameter optimization (HO). For each performance indicator, the table presents the minimum, mean, and maximum values across all 110 segments. Based on the minimum, mean, and maximum values, the performance of the different ANNs and the impact of hyperparameter optimization are compared.

The range of values for hyperparameters is as follows:

- Number of neurons per second layer $\in$ [1, 100]
- Number of neurons per third layer $\in$ [1, 100]

Table 3 Pseudocode for HO

1	**For** s=1: Segments
	Hyperparameter optimization
2	**Input** (Hyper parameters space *Q*, Target score function *H*(*Q*), number of initial iterations *n0*, max number of iterations *n*)
2.1	**Define** *Target score function H(Q)*
2.2	**Scale** *data by Standard Scaler*
2.3	**Split** *the historical degradation data for each segment into training and test data*
2.4	**Add** *one input layer, two hidden layers, and one output layer to ANNs (FF-ANN, simple RNN, LSTM, GRU) structure based on Q*
2.5	**Compile** *NN model with TensorFlow (loss=Mean Squared Error (), optimizer=Adam (), metrics=Root Mean Squared Error())*
2.6	**Fit** NN model *(FF-ANN, simple RNN, LSTM, GRU)* with TensorFlow
2.7	**Calculate** *MSE for predicted values vs test set*
2.8	**Return** *H(Q)=1- MSE*
2.9	**Optimize** *H(Q) with Bayesian Optimization based on n0 and n values*
2.10	**Output** *Optimal values of Q, Q**
	Predict track geometries by ANNs based on *Q**
3	**Input** (Optimal Hyper parameters space *Q**)
3.1	**Scale** *data by Standard Scaler* **Split** *the historical degradation data for each segment into training and test data*
3.2	**Add** *one input layer, two hidden layers, and one output layer to ANNs (FF-ANN, simple RNN, LSTM, GRU) structure based on Q*
3.3	**Compile** *ANNs with TensorFlow (loss=Mean Squared Error (), optimizer=Adam (), metrics=Root Mean Squared Error())*
3.4	**Fit** ANNs *(FF-ANN, simple RNN, LSTM, GRU)* with TensorFlow
3.5	**Output** *MSE, MAE, and MAPE for test data and predicted values by the models*

Table 4 Comparison of ANNs' performance

Performance indicators		FF-ANN		Simple RNN		GRU		LSTM	
		N-HO	HO	N-HO	HO	N-HO	HO	N-HO	HO
MSE	Min	0.002	0.002	0.006	0.002	0.002	**0.002**	0.003	0.003
	Mean	0.01	0.01	0.01	0.01	0.01	**0.01**	0.01	0.01
	Max	0.03	0.02	0.03	0.02	0.03	**0.02**	0.03	0.02
MAE	Min	**0.03**	0.04	0.06	0.04	0.04	0.04	0.04	0.04
	Mean	0.09	0.08	0.09	0.08	0.09	**0.07**	0.09	0.09
	Max	0.16	0.13	0.14	0.12	0.14	**0.11**	0.16	0.13
MAPE	Min	0.03	0.03	0.05	0.04	0.03	**0.03**	0.04	0.04
	Mean	0.08	0.08	0.08	0.07	0.08	**0.06**	0.08	0.07
	Max	0.13	0.12	0.12	0.10	0.13	**0.08**	0.14	0.10
Prediction accuracy (1-MAPE)	Min	86%	88%	87%	90%	87%	**92%**	86%	90%
	Mean	91%	92%	91%	93%	91%	**93%**	91%	92%
	Max	96%	96%	94%	95%	96%	**96%**	95%	95%
The computational time for 110 segments (sec)		**81**	1692	211	1750	268	1895	315	2106

- Activation functions ∈ {ReLU (Rectified Linear Unit), Sigmoid, Tanh (Hyperbolic Tangent), Softmax, ELU (Exponential Linear Unit), SELU (Scaled Exponential Linear Unit), and Swish}
- Number of epochs ∈ [1, 200].

The results presented in Table 4 show that the hyperparameter optimization process, HO, leads to better performance compared to using default or suggested by experts hyperparameters, N-HO. The improvement is particularly evident in the minimum and maximum values of the performance indicators, such as MSE, MAE, and MAPE, as well as in the prediction accuracy. These results suggest that the hyperparameter optimization process can help identify better NN configurations that result in improved performance.

Furthermore, with hyperparameter optimization (HO), the GRU model achieved the best overall performance among the four ANNs. In general, all variations of RNN performed better than FF-ANN in HO. However, in N-HO, LSTM and RNNs performed worse than FF-ANN. It can be concluded that hyperparameter optimization has a higher impact on the performance of RNN, LSTM, and GRU compared to FF-ANN.

It is also important to note that the computational time required to train the models varies depending on the scenario (N-HO or HO) and the ANNs model. Generally, the HO scenarios require more computational time than N-HO due to the additional hyperparameter optimization process. Additionally, the LSTM and GRU models typically require more computational time than ANN and RNNs.

The choice between N-HO and HO depends on the specific problem and available resources. While HO can result in better performance, it also requires additional computational time. Therefore, it is important to consider the benefits of hyperparameter optimization against the costs before deciding on a particular approach. For practical implications, the computational time may not be critical when the real time prediction is not necessary.

6 Conclusion

The prediction of track geometry degradation poses significant challenges due to many factors influencing it. Developing advanced, efficient, and effective prediction models are imperative to ensure accurate prediction of track geometry degradation. The present study underscores the criticality of selecting appropriate NN architectures capable of capturing the temporal dependencies inherent in the track geometry degradation process. This study provides insights into the effectiveness of different ANNs (FF-ANN, simple RNN, LSTM, and GRU) for predicting track geometry degradation. The GRU model exhibited the most promising overall performance of the four ANNs evaluated in this research. As such, it is recommended that future research endeavors prioritize the exploration and optimization of GRU and LSTM as RNN variants when developing prediction models for track geometry degradation.

In addition, this study investigated the importance of hyperparameter tuning in improving predictive performance. While hyperparameter optimization can enhance the performance of ANNs, it is also important to consider the computational time required for this process. Therefore, future research should focus on developing more efficient hyperparameter optimization processes to achieve better performance while reducing computational time.

Overall, the accurate prediction of track geometry degradation can significantly improve maintenance planning and prevent accidents in railway operations. The use of advanced sensors and condition monitoring tools, and machine learning models, such as ANNs, can help facilitate this task. Further research in this area can lead to the development of more accurate and efficient predictive models that can be used in practice to enhance railway maintenance and safety. track geometry degradation. A limitation of this study is using only track geometry data and previous maintenance history for training ANNs. In future studies, additional input features can be included in the model to improve the predictive performance of ANNs.

Acknowledgements The author would like to express her sincere gratitude to her supervisors, Bjarne Bergquist, Osmo Kauppila, and Erik Vanhatalo, for their invaluable guidance and feedback throughout the research project. The author expresses gratitude for the financial and other support from the Luleå Railway Research Centre (JVTC) through the "Statistical Methods in Railway Maintenance" project and the Swedish Transportation Administration for providing the necessary data and practical insights.

References

1. Sedghi M, Kauppila O, Bergquist B, Vanhatalo E, Kulahci M (2021) A taxonomy of railway track maintenance planning and scheduling: a review and research trends. Reliab Eng Syst Saf. https://doi.org/10.1016/j.ress.2021.107827
2. Soleimanmeigouni I, Ahmadi A, Kumar U (2018) Track geometry degradation and maintenance modelling: a review. Proc Inst Mech Eng Part F J Rail Rapid Transit. https://doi.org/10.1177/0954409716657849
3. Hewamalage H, Bergmeir C, Bandara K (2021) Recurrent neural networks for time series forecasting: current status and future directions. Int J Forecast 37. https://doi.org/10.1016/j.ijforecast.2020.06.008
4. Guler H (2014) Prediction of railway track geometry deterioration using artificial neural networks: a case study for Turkish state railways. Struct Infrastruct Eng 10. https://doi.org/10.1080/15732479.2012.757791
5. Falamarzi A, Moridpour S, Nazem M (2019) A review of rail track degradation prediction models. Aust J Civ Eng. https://doi.org/10.1080/14488353.2019.1667710
6. Hyndman R (2018) A brief history of time series forecasting competitions 2018. https://robjhyndman.com/hyndsight/forecasting-competitions/
7. Smyl S (2020) A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting. Int J Forecast 36:75–85. https://doi.org/10.1016/j.ijforecast.2019.03.017
8. Connor JT, Martin RD, Atlas LE (1994) Recurrent neural networks and robust time series prediction. IEEE Trans Neural Netw 5:240–254. https://doi.org/10.1109/72.279188
9. Lindemann B, Maschler B, Sahlab N, Weyrich M (2021) A survey on anomaly detection for technical systems using LSTM networks. Comput Ind 131:103498. https://doi.org/10.1016/J.COMPIND.2021.103498
10. Khajehei H, Ahmadi A, Soleimanmeigouni I, Haddadzade M, Nissen A, Latifi Jebelli MJ (2022) Prediction of track geometry degradation using artificial neural network: a case study. Int J Rail Transp 10. https://doi.org/10.1080/23248378.2021.1875065
11. Moridpour S, Mazloumi E, Hesami R (2016) Application of artificial neural networks in predicting the degradation of tram tracks using maintenance data. Appl Big Data Anal Oper Manag. https://doi.org/10.4018/978-1-5225-0886-1.ch002
12. Lee JS, Hwang SH, Choi IY, Kim IK (2018) Prediction of track deterioration using maintenance data and machine learning schemes. J Transp Eng Part A Syst 144. https://doi.org/10.1061/jtepbs.0000173
13. Ali L, Amin S, Wehbi M (2021) Backpropagation algorithms of neural networks to construct the railway track deterioration model. In: 2021 7th International conference on models and technologies for intelligent transportation systems MT-ITS 2021. https://doi.org/10.1109/MT-ITS49943.2021.9529272
14. Wu J, Chen XY, Zhang H, Xiong LD, Lei H, Deng SH (2019) Hyperparameter optimization for machine learning models based on bayesian optimization. J Electron Sci Technol 17:26–40. https://doi.org/10.11989/JEST.1674-862X.80904120
15. Yang L, Shami A (2020) On hyperparameter optimization of machine learning algorithms: theory and practice. Neurocomputing 415. https://doi.org/10.1016/j.neucom.2020.07.061
16. De Bruin T, Verbert K, Babuska R (2017) Railway track circuit fault diagnosis using recurrent neural networks. IEEE Trans Neural Netw Learn Syst 28. https://doi.org/10.1109/TNNLS.2016.2551940
17. Popov K, De Bold R, Chai HK, Forde MC, Ho CL, Hyslip JP et al (2022) Big-data driven assessment of railway track and maintenance efficiency using artificial neural networks. Constr Build Mater 349:128786. https://doi.org/10.1016/J.CONBUILDMAT.2022.128786
18. Liao Y, Han L, Wang H, Zhang H (2022) Prediction models for railway track geometry degradation using machine learning methods: a review. Sensors 22:7275. https://doi.org/10.3390/HO2197275

19. Lopes Gerum PC, Altay A, Baykal-Gürsoy M (2019) Data-driven predictive maintenance scheduling policies for railways. Transp Res Part C Emerg Technol. https://doi.org/10.1016/j.trc.2019.07.020
20. Falamarzi A, Moridpour S, Nazem M, Hesami R (2018) Rail degradation prediction models for tram system: Melbourne case study. J Adv Transp 2018. https://doi.org/10.1155/2018/6340504
21. Khashei M, Bijari M (2010) An artificial neural network (p, d, q) model for timeseries forecasting. Expert Syst Appl 37. https://doi.org/10.1016/j.eswa.2009.05.044
22. Hewamalage H, Bergmeir C, Bandara K (2021) Recurrent neural networks for time series forecasting: current status and future directions. Int J Forecast 37:388–427. https://doi.org/10.1016/j.ijforecast.2020.06.008
23. Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: 30th international conference on machine learning ICML 2013
24. Lindemann B, Müller T, Vietz H, Jazdi N, Weyrich M (2021) A survey on long short-term memory networks for time series prediction. Procedia CIRP, vol 99. https://doi.org/10.1016/j.procir.2021.03.088
25. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
26. Cho H, Kim Y, Lee E, Choi D, Lee Y, Rhee W (2020) Basic enhancement strategies when using bayesian optimization for hyperparameter tuning of deep neural networks. IEEE Access 8. https://doi.org/10.1109/ACCESS.2020.2981072
27. Bischl B, Binder M, Lang M, Pielok T, Richter J, Coors S et al (2023) Hyperparameter optimization: foundations, algorithms, best practices, and open challenges. WIREs Data Min Knowl Discov n/a:e1484. https://doi.org/10.1002/widm.1484
28. Victoria AH, Maragatham G (2021) Automatic tuning of hyperparameters using Bayesian optimization. Evol Syst 12. https://doi.org/10.1007/N-HO2530-020-09345-2
29. 13848–5:2017 E (2017) Railway applications-track-track geometry quality-Part 5: geometric quality levels-plain line, switches and crossings. Brussels Eur Comm Stand
30. Developers T (2023). TensorFlow. https://doi.org/10.5281/ZENODO.7764425