Elevator Breakdown Prediction using LSTM Analysis of Monitoring Data

1st Gabriele Iuculano
School of Computing
University of Leeds
Leeds, England
gabriele.iuculano@hotmail.it

2nd Muhammad Zeeshan Babar
School of Computing
University of Leeds
Leeds, England
M.Z.Babar@leeds.ac.uk

***Abstract*—Elevator malfunctions represent significant challenges in modern building infrastructures due to their resulting inconvenience and downtime, impacting transportation quality and reliability. To address these issues, the current study proposes to utilize Long Short-Term Memory (LSTM) networks to predict safety-related breakdowns that lead to immediate elevator stops. The methodology combines monitoring data analysis with LSTM networks to forecast elevator safety chain breakdowns. In response to the often-encountered problem of unbalanced datasets, a real balanced dataset is constructed using monitoring data from operating elevators worldwide. This dataset is formed using language modeling techniques, identifying behavioral patterns across various time horizons, and mapping these to a classification problem. Further steps involve setting temporal boundaries, embedding data within these boundaries, deploying an LSTM neural network, and subsequently fine-tuning hyperparameters. Experiment results indicate that an LSTM neural network can predict elevator safety chain breakdowns with an F1-score of 85% on average across multiple time windows.**

***Index Terms*—Deep Learning, RNN, LSTM, Elevator, Embedding, Predictive Maintenance**

I. Introduction

Elevators play a crucial role in modern urban infrastructure, especially in high-rise buildings where they facilitate vertical transportation of people and goods while assuring the safety and comfort of the building's occupants. Malfunctions can cause significant inconvenience, result in financial and reputational loss, and in some cases pose safety concerns. Therefore, it is essential to establish effective strategies to predict elevator malfunctions and intervene prior to a breakdown. In recent years, predictive maintenance techniques using data to predict in advance when a machine is likely to malfunction have garnered much attention. Indeed, predicting malfunctions enables efficient scheduling the machine maintenance, reducing downtime and associated costs. Predictive maintenance in the context of elevators may help facility operation teams better allocate resources and optimize maintenance efforts, resulting in improved elevator performance and higher customer satisfaction.

A. Background and problem statement

In the context of elevator breakdown prediction, it is critical to note that not all elevator failures can be effectively predicted. Certain breakdowns occur without a defined pattern, such as when external actions cause damage to the system. Conversely, other types of breakdowns may be predicted because the affected units follow an unusual behavioral pattern, which can be observed through their monitoring data entries. This research focuses on the second premise. Within the company, a subset of breakdowns has been identified as potentially linked to specific patterns. From this list, a particular elevator safety chain breakdown at the door level has been selected. This breakdown is among the most common ones which can happen spontaneously and without external causes. Therefore, it represents a good candidate for breakdown prediction. The selected safety chain breakdown is associated with elevator system malfunctions that prevent the safety chain at the door level from closing properly, constituting an unsafe condition that hinders elevator operation. Hence, when the malfunction occurs, the elevator immediately becomes out-of-service. This work aims to establish a prediction scheme specifically targeting the aforementioned safety-related

B. Literature Review

In recent years, several studies have examined the application of machine learning algorithms for predictive maintenance purposes. In particular, log analysis has garnered extensive coverage in this context. For instance, the authors of [3] proposed a novel algorithm architected to amass and scrutinize logs, extract patterns, and identify anomalous content. They leveraged an innovative conceptualization of log similarity, based on the Longest Common Subsequence (LCS).

The concept of predictive maintenance has also been widely addressed in academic literature. In [2], authors offer a comprehensive survey on data-driven methods, in the context of artificial intelligence. Their work posits that both Machine Learning and Deep Learning can execute predictive maintenance tasks remarkably well, with the average prediction accuracy reaching 95.06%. The insights provided by this work helped to shape our understanding of the landscape of predictive maintenance techniques that could be relevant to this thesis.

Specifically in the domain of elevators, the authors of [1] ventured into the degradation assessment and fault modes classification using logistic regression on elevator door systems. They employed a dataset consisting of vibration and current data, which were transformed into wavelet packet energies and used as features. Furthermore, many researchers have acknowledged the potential of LSTM networks in anomaly detection and predictive maintenance. In particular, the authors of [4] developed an LSTM-based anomaly detection model for log analysis. Their work underscores the efficacy of LSTM in handling sequence data, outshining other popular anomaly detection algorithms and achieving the highest AUC (Area Under the Curve) of 0.913. This finding is consistent with our choice of utilizing LSTM networks for predicting elevator malfunctions.

Despite these valuable contributions, there appears to be limited specific literature on the application of LSTM networks to predict elevator malfunctions. Hence, this present study aims to fill this void by introducing an innovative approach and an operational dataset specifically tailored for elevator breakdown prediction.

C. Contributions

The objective of this work is to develop predictive models that can foresee 15, 30, 45, and 60 days in advance when the elevator safety chain breakdown may occur, allowing a more efficient maintenance plan, and ultimately enhancing elevator reliability. The proposed methodology consists of first collecting real monitoring data to construct a balanced language-based dataset (Section II-A). Then, this data is used to train, configure, and test an LTSM network targeting elevator breakdown prediction (Section II-D). In particular, the contributions of this thesis pertain to several key factors:

- Opposite to existing studies that rely on publicly available academic datasets, the proposed work uses real operating data that reflect the condition and behavior of a real-world system. Thus, this thesis addresses the critical issue of *data validity*, essential to ensure efficacy in predictive maintenance systems.
- This work specifically focuses on the construction of a *consistent dataset*, overcoming the challenges related to the large volume of operational data, which often constrains malfunction analysis.
- This work uses *comprehensive real-world data* to build a consistent dataset. It includes monitoring messages collected from an internal corporate database that records entries from internet-connected installations worldwide.

II. Methodology

A. Data collection and transformation

This research uses a dataset from 700 elevator installations across Europe, North America, and Asia that covers two years of monitoring, from January 2021 to January 2023. It contains multiple regular events and malfunctions spanning five classes: Breakdown, Error, Warning, Status, and Diagnostic. Figure 1 shows the structure of the elevator monitoring events, as communicated from connected elevators. Each event entry is uniquely defined by the *class-subcode* pair, and linked to a specific installation identified by its *equipment number* and address. In order to streamline the dataset and make it more

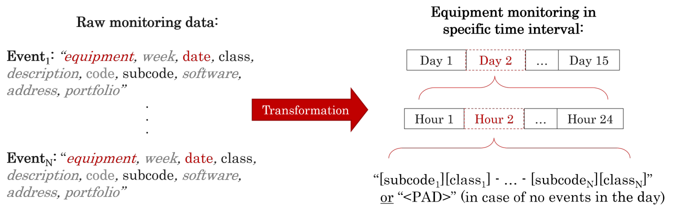

Raw monitoring data:
Event$_1$: "equipment, week, date, class, description, code, subcode, software, address, portfolio"
Event$_N$: "equipment, week, date, class, description, code, subcode, software, address, portfolio"
Transformation
Equipment monitoring in specific time interval:
Day 1
Day 2
...
Day 15
Hour 1
Hour 2
...
Hour 24
"[subcode$_1$][class$_1$] - ... - [subcode$_N$][class$_N$]"
or "<PAD>" (in case of no events in the day)

Fig. 1: Monitoring data transformation

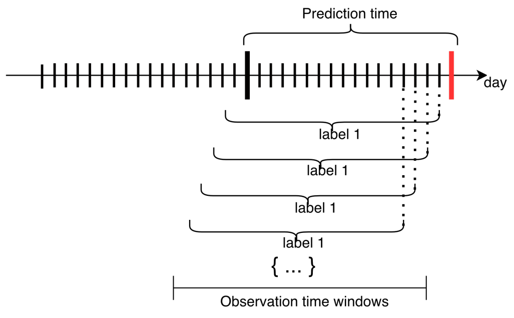
conducive to our research objectives, several modifications are applied. Firstly, irrelevant data elements such as *week*, *description* and *code* (in grey in Figure 1) are omitted as they do not directly pertain to the primary focus of investigating elevator performance and maintenance trends. Next, the *date*, initially containing date and time information, is transformed into a continuous integer sequence in the form of year-month-day-hour-minute-second (e.g., 20230101010101). This transformation simplifies the handling of date information and enables more straightforward computation of time intervals between events. Finally, if multiple events of the same *class-subcode* occur during a day, it is represented only once in the new proposed monitoring data format, thus avoiding any bias of the prediction model with the count of events. To account for days with no entries, a special marker <PAD>was introduced in the aggregated sequence of days. This approach ensures that the delta time between data points is preserved, allowing for a more accurate representation of the temporal relationships among events within the sampling period. These dataset modifications streamline the data structure and enable a more efficient and focused analysis of elevator behavioral trends. By preserving the temporal relationships between events and removing less pertinent information, the breakdown prediction algorithm concentrates on the key variables indicating the overall elevator performance, ultimately contributing to the improvement of elevator installations worldwide.

B. Time window creation

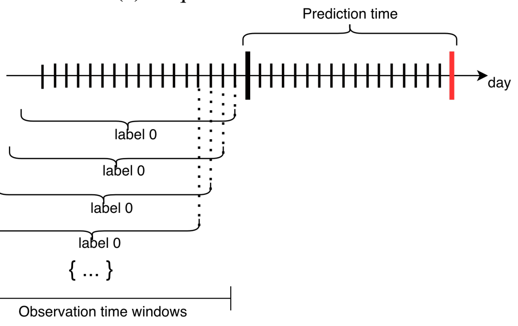
The research necessitates the definition of three key terms:

- A *window* is a time interval defined by two times (t_1, t_2) with $t_1 < t_2$ and the *dimension* d of the window $d = t_2 - t_1$. Windows of fixed dimension $d \in \mathcal{D} = \{15, 30, 45, 60\}$ are used. A generic window w_i can be associated with a *label* $\ell(w_i) \in \{0, 1\}$ (with 0 indicating no breakdown and 1 indicating a breakdown).
- A *prediction time* is a window (t_1, t_2) in which the safety chain breakdown occurs at time t_2. It constitutes the timeframe within which an issue is sought to be predicted, based on the patterns discerned during the observation times.
- An *observation time* is a window (t_1, t_2) during which data is observed and analysed. It is a subset of the prediction time.

To train distinct LSTM-based models, separate datasets were created for different combinations of observation and prediction times in $\mathcal{D}$. A labeled dataset named $d_1 - d_2$, where



(a) Sequences labeled as 1



(b) Sequences labeled as 0

Fig. 2: Time window creation: The red markers indicate the day when the breakdown occurs, while the black markers denote the days with no breakdowns.

$d_1 \geq d_2$ and $d_1, d_2 \in \mathcal{D}$, is created in the following manner: let $w = (t_1, t_2)$ be a prediction window of dimension d_2. The dataset considers all the observation windows $w_i = (t_{i_1}, t_{i_2})$ of dimension d_1, such that $t_1 - 2d_2 \leq t_{i_2} < t_2$. For each such window, the label $\ell(w_i) = 1$ is assigned if $t_1 \leq t_{i_2} \leq t_2$ and $\ell(w_i) = 0$ if $t_{i_2} < t_1$, as illustrated in Figure 2.

For example, in the dataset $30 - 15$ the prediction window has a dimension of 15 days, and all the observation windows have a dimension of 30 days. Suppose that the prediction window is $w = (50, 75)$ (i.e., the elevator breakdown occurs on day 75), the dataset contains all the sequences of 30-days contained in a windows $w_i = (t_{i_1}, t_{i_2})$ of dimension $d_i = 30$ starting from day $t_1 - 2 \cdot 15 = 20$. This dataset is composed of 30 sequences. The label of each sequence is the label of the corresponding window. In this case, there are 15 sequences labeled 1 and 15 sequences labeled 0. Additionally, each dataset is also enriched with random sequences from installations that have never experienced the issue in question, with these sequences assigned a label of '0'. This approach is intended to boost the model's capability to distinguish between situations leading to the breakdown and those that do not. By incorporating these sequences into the dataset, the aim is for the model to be able to generalize and accurately predict outcomes on new installations it has yet to encounter.

C. Conversion into embeddings

The further step in the methodology involves transforming the sequence monitoring data into a format more suitable as input to the LSTM-based model. To accomplish this, a technique called embedding is employed. This technique enables to representation of the monitoring data sequence in a fixed-dimensional continuous vector space. It facilitates extracting meaningful relationships from the dataset and enhances prediction capabilities by converting the prediction-time sequences dataset into 100-dimensional embeddings using Doc2Vec. This approach offers several advantages, with the top two being:

- It reduces data dimension and improves computational efficiency, allowing us to decrease the model training time.
- Considering that the dataset is highly susceptible to noise, as a breakdown can be generated by external sources (e.g., field technician, user mishandling, vandalism), embeddings help make the model more noise-resilient. Continuous representations enable the model to learn from subtle similarities and differences between entities, yielding better performance.

D. Model construction

In the context of elevator breakdown prediction, a specific type of recurrent neural network (RNN) called Long Short-Term Memory (LSTM) is utilized. Introduced by Hochreiter and Schmidhuber in their groundbreaking work [5], this type of recurrent neural network (RNN) is explicitly designed to manage sequence data containing long-term dependencies. Given its ability to capture the temporal patterns and dependencies in the monitoring data, the LSTM network is well-suited for accurate breakdown prediction. This is made possible by their structure, which includes a cell state alongside three "gates" or "ports": an *input gate*, *output gate*, and *forget gate* [8]. Figure 3 offers a visual representation of the LSTM network structure. At each step, the cell state (also known as the memory cell) is potentially updated based on the interaction between the new incoming data and the gate mechanisms. In particular:

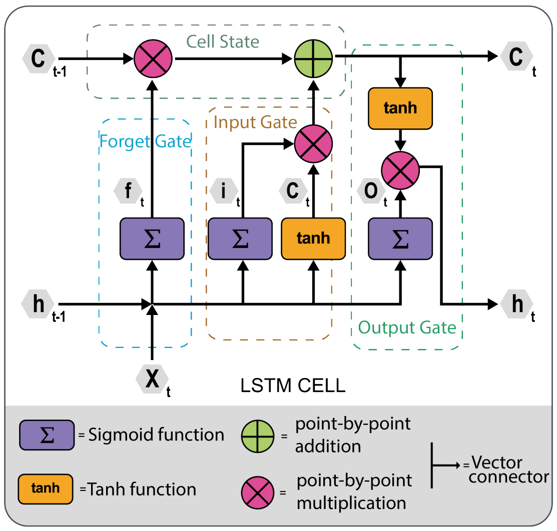


Fig. 3: LSTM structure.

- The *Forget Gate* has the task of identifying which pieces of information within the cell state are no longer relevant and therefore can be eliminated. After the forget gate operates, the information deemed not important is discarded from the cell state.
- After the forget gate operation, the *Input Gate* is responsible for determining what new pieces of information will be updated in the cell state. It first creates a vector of new candidate values that could be added to the state, and then it makes a decision about which values to update, producing a new state.
- The *Output Gate* decides which information will be used in the output, based on the current state of the cell state and the current input. It looks at the current input and the updated cell state to decide which information from the cell state is relevant and should be used to generate the next hidden state.

The decision to use LSTM networks is motivated by the sequential nature of the data. Specifically, temporal windows can be viewed as corpora, with each corpus representing a set of temporally ordered observations. Among the various types of RNN, LSTMs have proven to be particularly effective in dealing with complex temporal sequences, such as those found in machine translation tasks [6] [7]. Drawing parallels to how LSTM networks discern long-term patterns within a text corpus to perform accurate translations, these networks might discover long-term patterns in temporal windows, thereby enhancing the accuracy of breakdown predictions. The architecture designed for the LSTM network in this research includes an input layer, followed by a hidden layer, and concludes with an output layer. The input layer receives the 100-dimensional embeddings of the data, while the hidden LSTM layer is dedicated to learning the temporal patterns and dependencies inherent in the data. The output layer is composed of a single sigmoid activation function, generating a singular value representing the probability of a breakdown occurring within the defined time window.

E. Hyperparameters configuration and training

The dataset is divided into 72% for training, 8% for validation, and 20% for testing. To optimize the hyperparameters of the model, a Random Search is employed on the training dataset. The hyperparameters under consideration are:

- The number of units in the input LSTM layer: ranges from 32 to 128 units, with a step of 8.
- The number of units in the hidden layer: ranges from 16 to 64 units, with a step of 8.
- The learning rate: varying between a minimum value of 10^{-5} and a maximum value of 10^{-2}.

Random Search randomly selects combinations of hyperparameters from a specified range or distribution. As it examines only a random subset of the hyperparameter space, Random Search is significantly less computationally intensive compared to Grid Search. Although Random Search does not guarantee the identification of the optimal combination of hyperparameters when the hyperparameter space is large and/or some hyperparameters have a minor impact on the model's performance, it can yield similar or even superior performance to Grid Search [10]. During Random Search, the model is trained using the training data, and the performance is evaluated on the validation set. The binary cross-entropy loss function is used to measure the difference between the predicted probabilities and the true labels, while the Adam optimizer [9] is employed to minimize this loss by updating the model's weights. The validation set, being distinct from the training set, provides an unbiased estimate of the model's performance on unseen data, thereby enabling the monitoring of the model's ability to generalize. The combination of hyperparameters that yield the best performance on the validation set is considered optimal. Following the identification of the optimal hyperparameters, the LSTM network is trained once again, this time employing the identified parameters and using the union of training and validation datasets. The performance of this final model is then evaluated on the test set.

III. Experimental Results

This section discusses the outcomes of the experiments, covering the evaluation of the LSTM network's performance and the analysis of breakdown predictions at 15, 30, 45, and 60-day time horizons. To facilitate the discussion, true negative, false negative, false positive, and true positive are referred to as TN, FN, FP, and TP, respectively.

A. Model Performance

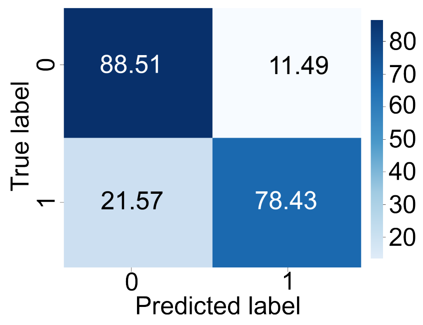
The performance of the LSTM network is gauged through various metrics such as *precision*, *recall*, *F1-score*, and the *Area Under the Receiver Operating Characteristic curve* (*AUROC*). These metrics collectively offer a comprehensive evaluation of the model's ability to accurately predict elevator breakdowns within the designated time windows. From a business perspective, minimizing the number of FP is crucial. This is because each maintenance visit inconveniently interrupts the client's business. To address this, all classification thresholds are incorporated into the performance evaluation with the objective of minimizing the following cost function:

$$F1_{weight} * (1 - F1) + FP_{weight} * FP \qquad (1)$$

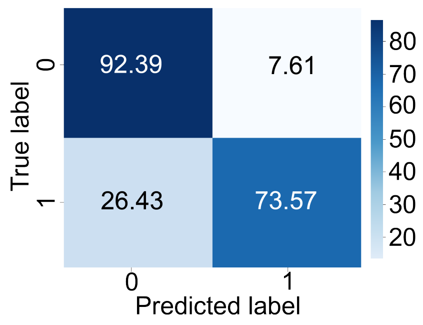
In the above equation, $F1_{weight}$ and FP_{weight} are weights assigned to the F1-score and FP rate, respectively. These weights can be adjusted to reflect the importance or cost associated with each component in the specific business context. In the following experiments, they are set to 0.5, ensuring equal importance for maximizing the F1-score and minimizing the FP. Thus, the cost function favors solutions with high F1-scores and penalizes those with high FP rates. The objective is to obtain a model that optimizes its predictive accuracy while minimizing any potential disruption to the client's operation.

B. Experiment 1: 15-day Safety Prediction

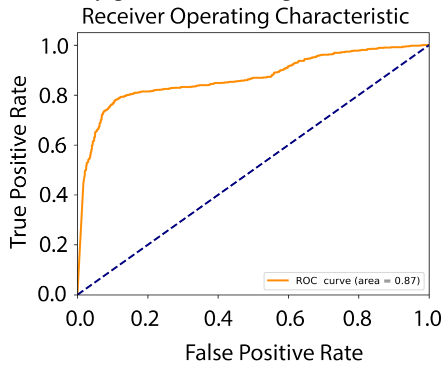
The first experiment focuses on the prediction of Safety breakdown within a 15-day window. The models used for this



(a) Confusion Matrix: 15-day observation, 15-day prediction.



(b) Confusion Matrix: 15-day observation, 15-day prediction with opt threshold.



(c) ROC curve: 15-day observation, 15-day prediction.

Fig. 4: Confusion Matrices and ROC Curve for the 15-15 model at 0.5 and Optimized Threshold.

experiment include the 15-15, 30-15, 45-15, and 60-15 models. The numbers represent the observation and prediction times, respectively. Model performance is evaluated based on two different thresholds. The first threshold is a conventional classification threshold of 0.5. The second threshold is determined through an optimization process aimed at minimizing the cost function. For this second threshold, all the thresholds utilized to construct the ROC curve are assessed, and the value that minimizes the defined cost function (Section 3.1) is selected, thereby aiming to improve model performance.

The results are summarized in Table I for the 0.5 thresholds and the optimized threshold. Note that in the table, the values of TN, FN, FP, and TP are expressed as percentages, normalized for each row.

TABLE I: Comparison of model performance using standard and optimized thresholds for 15, 30, 45, and 60-day observations with 15-day prediction.

Model	Threshold	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
15-15	0.5	0.8454	0.8447	0.8454	0.8449	88.51	11.49	21.57	78.43
30-15	0.5	0.8171	0.8170	0.8172	0.8171	84.12	15.88	21.29	78.71
45-15	0.5	0.8340	0.8351	0.8340	0.8332	88.32	11.68	22.37	77.63
60-15	0.5	0.8285	0.8286	0.8285	0.8284	85.10	14.90	19.63	80.37
15-15	0.9460 (opt)	0.8498	0.8509	0.8498	0.8473	92.39	7.61	26.43	73.57
30-15	0.8938 (opt)	0.8267	0.8291	0.8267	0.8248	89.78	10.22	26.26	73.74
45-15	0.9229 (opt)	0.8378	0.8447	0.8378	0.8357	92.37	7.63	26.30	73.70
60-15	0.9088 (opt)	0.8330	0.8374	0.8330	0.8318	90.10	9.90	24.19	75.81

In terms of accuracy, precision, recall, and F1-score, the 15-15 model provides the best results when using a 0.5 classification threshold. This suggests that using 15 days of observation time for predicting the next 15 days provides the most accurate results under these conditions. The values of FP deserve careful consideration, as they represent a tangible financial loss for a company. In the case of the 15-15 model, the FP rate stands at 11%, when normalized per row. With the threshold optimized, we observe performances similar to the 0.5 threshold, with the 15-15 model once again outperforming the others. However, this time there is a significant reduction in FP to 7.61%, albeit at the cost of a 5% decrease in TP. From 78.43% to 73.57% (Figure 4). The other models (30-15, 45-15, and 60-15) show a decrease in performance as the observation period increases, both under the standard 0.5 threshold and the optimized threshold conditions. This suggests that longer observation periods may not necessarily improve the predictive performance for the 15-day prediction task. In conclusion, for a 15-day prediction of Safety breakdown, the 15-15 model using an optimized threshold is preferable over the other models due to its performance and its ability to minimize FP.

C. *Experiment 2: 30-day Safety Prediction*

The second experiment aims to predict Safety breakdown over a period of 30 days. The models used in this experiment are the 30-30, 45-30, and 60-30 models. The results are summarized in Table II for the 0.5 thresholds and the optimized threshold. Regarding model performance, the 30-30 model shows the best results with the 0.5 threshold as well as the optimized threshold.

This suggests that a 30-day observation period for predicting the following 30 days provides the most accurate results under these circumstances. With the optimized threshold, the 30-30 model demonstrates a significant reduction in FP down to 6.29%, albeit at the cost of a slight decrease in TP from 81.81% to 80.97%.

In conclusion, for a 30-day prediction of Safety breakdown, the 30-30 model using an optimized threshold is preferable over the other models due to its performance and its ability to minimize FP.

D. *Experiment 3: 45-day Safety Prediction*

In Experiment 3, 45-45 and 60-45 models are utilized for 45-day Safety predictions. The results are summarized in Table

TABLE II: Comparison of model performance using standard and optimized thresholds for 30, 45, and 60-day observations with 30-day prediction.

Model	Threshold	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
30-30	0.5	0.8790	0.8800	0.8790	0.8782	92.60	7.40	18.19	81.81
45-30	0.5	0.8714	0.8722	0.8714	0.8710	90.68	9.32	16.89	83.11
60-30	0.5	0.8634	0.8646	0.8635	0.8632	89.84	10.16	17.40	82.60
30-30	0.6781 (opt)	0.8816	0.8837	0.8816	0.8806	93.71	6.29	19.03	80.97
45-30	0.9766 (opt)	0.8709	0.8789	0.8709	0.8694	95.17	4.83	22.15	77.85
60-30	0.9031 (opt)	0.8640	0.8705	0.8640	0.8629	93.59	6.41	21.33	78.67

TABLE III: Comparison of model performance using standard and optimized thresholds for 45, and 60-day observations with 45-day prediction.

Model	Threshold	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
45-45	0.5	0.8306	0.8348	0.8306	0.8298	89.21	10.79	23.40	76.60
60-45	0.5	0.8265	0.8287	0.8265	0.8260	87.28	12.72	22.16	77.84
45-45	0.7958 (opt)	0.8279	0.8396	0.8279	0.8259	92.55	7.45	27.49	72.51
60-45	0.9524 (opt)	0.8243	0.8389	0.8243	0.8219	93.15	6.85	28.72	71.28

III for the 0.5 thresholds and the optimized threshold. Between the two models, 45-45 appears to be slightly more promising for both thresholds. Regarding FP, in the 45-45 model with the optimized threshold, there is a reduction in FP from 10.79% to 7.45%, at the cost of a reduction in TP from 76.60% to 72.51%.

Interestingly, the TN increases from 89.21% to 92.55%. In essence, with metric values that are very close to those of the 45-45 model at a 0.5 threshold, this confirms the effectiveness of the model with the optimized threshold.

E. Experiment 4: 60-day Safety Prediction

In Experiment 4, only the 60-60 model is utilized for 60-day Safety predictions. The results for the 0.5 thresholds and the optimized threshold are summarized in Table IV.

The overall performance of the model with the two thresholds is quite similar, although it is necessary to point out that in the model with the optimized threshold, there is a strong degradation of TP. In fact, there is a reduction in FP from 14.45% to 7.67%, at the non-negligible cost of a reduction of almost 7% in TP (73.89% to 66.13%).

IV. Conclusion and Future Work

The findings of this research illustrate the potential of LSTM neural networks in forecasting elevator failures using monitoring data up to 60 days in advance. The models were developed to offer a performance that is not only promising across all investigated time horizons (15, 30, 45, and 60 days) but also significantly superior to traditional maintenance techniques. The capacity to anticipate breakdowns, evident in the models, equips facilities management and maintenance teams with the means to make more efficient planning and resource allocation for maintenance tasks, thereby mitigating downtime and correlated expenses. The strength of this approach lies in the conversion of data into embedding, supplemented by the LSTM network's proficiency in discerning long-term dependencies. This dual strategy proves to be effective in detecting breakdown patterns. This work further introduces a cost function that rephrases predictive maintenance as an optimization problem, for instance, minimizing warehouse costs, thereby providing a more comprehensive perspective on handling maintenance tasks. While a natural decline in performance is observed with the extension of the prediction horizon, this is a predictable outcome given the complexities associated with long-range predictions. Despite this, the models continue to perform robustly even at the challenging 60-day horizon, underlining the potential of this methodology for long-term breakdown prediction.

TABLE IV: Comparison of model performance using standard and optimized thresholds for 60-day observation with 60-day prediction.

Model	Threshold	Accuracy	Precision	Recall	F1-score	TN	FP	FN	TP
60-60	0.5	0.7984	0.8015	0.7984	0.7976	85.55	14.45	26.11	73.89
60-60	0.9340 (opt)	0.7951	0.8144	0.7951	0.7911	92.33	7.67	33.87	66.13

An inherent challenge of LSTM-based predictive models is their lack of explainability. Understanding why they make a certain prediction can be difficult, especially crucial in a field like predictive maintenance, where comprehending the cause of a potential breakdown can facilitate future system development. A future research direction could be the integration of attention mechanisms to visualize which parts of the input are deemed most significant during prediction, potentially increasing the models' interpretability.

In summary, this research evidences the capacity of LSTM-based predictive models in revolutionizing maintenance strategies for complex systems such as elevators. Additionally, the introduction of a cost function offers a tangible way to transition predictive maintenance into an optimization problem, for example in the optimization of warehouse costs.

References

[1] J. Yan and J. Lee, "Degradation assessment and fault modes classification using logistic regression," Journal of Manufacturing Science and Engineering, vol. 127, no. 4, pp. 912–914, 2004.

[2] W. Zhang, D. Yang, and H. Wang, "Data-driven methods for predictive maintenance of Industrial Equipment: A Survey," IEEE Systems Journal, vol. 13, no. 3, pp. 2213–2227, 2019.

[3] Y. Zhao and others, "Improvement of the log pattern extracting algorithm using text similarity," in IEEE International Parallel and Distributed Processing Symposium Workshops, 2018.

[4] Z. Zhao, C. Xu, and B. Li, "A LSTM-based anomaly detection model for log analysis," Journal of Signal Processing Systems, vol. 93, no. 7, pp. 745–751, 2021.

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

[6] I. Sutskever, O. Vinyals, and Q.V. Le, "Sequence to sequence learning with Neural Networks," arXiv.org, 2014.

[7] K. Cho and others, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," arXiv.org, 2014.

[8] F.A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in International Conference on Artificial Neural Networks, 1999.

[9] D.P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv preprint arXiv:1412.6980, 2015.

[10] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization," Journal of Machine Learning Research, 2012.