sensors

MDPI

Article

Railway Track Inspection Using Deep Learning Based on Audio to Spectrogram Conversion: An on-the-Fly Approach

Muhammad Shadab Alam Hashmi [1], Muhammad Ibrahim [2], Imran Sarwar Bajwa [2], Hafeez-Ur-Rehman Siddiqui [1], Furqan Rustam [1], Ernesto Lee [3,*] and Imran Ashraf [4,*]

1 Faculty of Computer Science and Information Technology, Khawaja Fareed University of Engineering and Information Technology, Rahim Yar Khan 64200, Pakistan; shadab.alam@kfueit.edu.pk (M.S.A.H.); siddiqov@gmail.com (H.-U.-R.S.); furqan.rustam1@gmail.com (F.R.)
2 Department of Computer Science, The University of Bahawalpur, Bahawalpur 63100, Pakistan; muhammad.ibrahim@iub.edu.pk (M.I.); imran.sarwar@iub.edu.pk (I.S.B.)
3 Department of Computer Science, Broward College, Broward County, FL 33301, USA
4 Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Korea
* Correspondence: elee@broward.edu (E.L.); imranashraf@ynu.ac.kr (I.A.)

Abstract: The periodic inspection of railroad tracks is very important to find structural and geometrical problems that lead to railway accidents. Currently, in Pakistan, rail tracks are inspected by an acoustic-based manual system that requires a railway engineer as a domain expert to differentiate between different rail tracks' faults, which is cumbersome, laborious, and error-prone. This study proposes the use of traditional acoustic-based systems with deep learning models to increase performance and reduce train accidents. Two convolutional neural networks (CNN) models, convolutional 1D and convolutional 2D, and one recurrent neural network (RNN) model, a long short-term memory (LSTM) model, are used in this regard. Initially, three types of faults are considered, including superelevation, wheel burnt, and normal tracks. Contrary to traditional acoustic-based systems where the spectrogram dataset is generated before the model training, the proposed approach uses on-the-fly feature extraction by generating spectrograms as a deep learning model's layer. Different lengths of audio samples are used to analyze their performance with each model. Each audio sample of 17 s is split into 3 variations of 1.7, 3.4, and 8.5 s, and all 3 deep learning models are trained and tested against each split time. Various combinations of audio data augmentation are analyzed extensively to investigate models' performance. The results suggest that the LSTM with 8.5 split time gives the best results with the accuracy of 99.7%, the precision of 99.5%, recall of 99.5%, and F1 score of 99.5%.

Keywords: railway track inspection; spectrograms; acoustic signals; machine learning; deep convolution neural networks; LSTM

Citation: Hashmi, M.S.A.; Ibrahim, M.; Bajwa, I.S.; Siddiqui, H.-U.-R.; Rustam, F.; Lee, E.; Ashraf, I. Railway Track Inspection Using Deep Learning Based on Audio to Spectrogram Conversion: An on-the-Fly Approach. *Sensors* **2022**, *22*, 1983. https://doi.org/10.3390/s22051983

Academic Editor: Gwanggil Jeon

Received: 13 January 2022
Accepted: 28 February 2022
Published: 3 March 2022

1. Introduction

The railway network is an important transportation channel that serves as the backbone for many developing countries such as Pakistan. The railway system plays a vital role in a country's economy by moving people, as well as goods, efficiently and rapidly [1]. With the increase in the number of passengers, the railway network is becoming more sophisticated, burdened, and prone to tear and wear. At the same time, environmental conditions and mechanical forces are speeding up the degradation of railway tracks [2]. Rail tracks are one of the most essential and integral parts of the railway network, and the inspection of rail tracks is essential to prevent accidents and to reduce injuries and casualties [3]. Pakistan is a country where a large number of people use trains for traveling.

In Pakistan, 757 train accidents have been recorded from 2012 to 2017 [4], with an average of 125 accidents per year. Although the ratio of train accidents is higher for developing countries, the United States (US) also has a higher number of accidents. A

total of 11,434 rail accidents are reported in the year 2019, causing 7730 injuries and 937 fatalities [5]. A yearly summary of accidents in the US is provided in Table 1. The train accidents in Pakistan and the US show that the lives of hundreds of thousands of people are at risk, and negligence or human error in rail track inspection can increase fatalities and injuries. Proper inspection and timely detection of faults can save countless human lives, as well as reduce the financial losses of the railway network [6]. However, the inspection and maintenance of rail tracks are expensive and time-consuming activities.

Table 1. Railway accidents stats in Pakistan [7].

Year	No. of Deaths	No. of Injured
2015	40	No Reports
2016	28	150
2017	16	10
2018	No Reports	32
2019	97	100
2020	19	No Reports
2021	32	64

For rail track inspection, different non-destructive evaluation (NDE) techniques have been used such as Eddy current testing [8], magnetic flux leakage testing [9], ultrasonic testing [9,10], phased array detection [11], guided wave detection [12], and so on. More details about the tools and techniques used for rail track inspection can be found in [8,13]. In traditional systems, visual inspection and acoustic emission may be included [13]. In the last few years, the Internet of Things (IoT) and machine learning and deep learning networks are gaining large attention for railway track inspection. These approaches are used to develop novel, efficient and effective NDE systems. For this purpose, high-speed cameras [14] and acoustic transducers [15] are installed that provide a blend of traditional inspection methods with the machine learning models. The use of machine learning approaches has eased tasks for human beings with increased processing time and automatic feature extraction. Now, the analysis of long railways tracks is easy and fast when using automated approaches. Even though many techniques are working in rail track inspection, anomaly detection, and classification, numerous challenges remain unsolved, such as a lack of proper datasets, the effective detection of a variety of faults, testing at speeds over 80 km/h, and handling sensors producing big data [13,16].

Before using deep learning, hand-crafted feature engineering was used in applications related to computer vision and audio machine learning. Hand-crafted feature extraction required in-depth, domain-specific knowledge for problem-solving and tuning up the system for better performance [17]. Recently, the automatic feature extraction capability of deep learning has attracted many researchers to work with rail fault detection [18]. Railway track inspection and classification has three main steps. The first step is the preprocessing of the 'wav' files for eliminating undesired sounds. After preprocessing, feature extraction is performed by using spectrograms. The spectrograms are generated on-the-fly to make the system more flexible. Then, the classification model is trained for railway track fault detection.

Figure 1 shows steps for providing an understanding of the tasks carried out in this research. In this research, three different deep learning models are tested for their performance. This research focuses on convolutional 1D, convolutional 2D, and long short-term memory (LSTM). The dataset [19] used in this research is a balanced one with three classes including normal, superelevation, and wheel burnt. The first step of preprocessing is to remove the noise of air, rain, and all the dead space in the audio by using a signal envelope function with a threshold of 100 Hz. After cleaning all audio files, they are downsampled to 16 kHz and then split into 1.7 s, 3.4 s, and 8.5 s durations for the experiments. Later, spectrograms are generated on-the-fly. The on-the-fly approach

provides flexibility for performing different experiments and increases efficiency at the same time. The key contributions of the study are are follows:

- A novel approach is proposed that can work on the real acoustic dataset for finding railway track faults. The proposed approach uses an on-the-fly approach to generate spectrograms during the training process. It offers flexibility as compared to the traditional approach where the spectrogram dataset is generated before the model training.
- A comparative analysis of the impact of different time slots of the audio sample is carried out. This research also compared the impact of using Mel-Spectrogram and Log-Spectrogram for training purposes.
- Performance analysis is carried out using state-of-the-art techniques with respect to accuracy, precision, recall, and F1 score. In addition, its performance is validated using a statistical *t*-test.

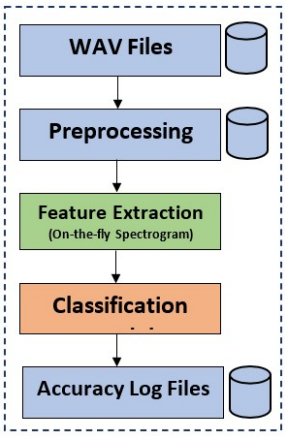


Figure 1. Architecture of the study.

The rest of the paper is organized in the following fashion. Important research works related to the current study are discussed in Section 2. The proposed methodology and related items are provided in Section 3. Section 4 provides the analysis and discussions of the experimental results. In the end, the conclusion is given in Section 5.

2. Related Work

Rail track inspections can be generally categorized into two classes: structural inspection and geometric inspection [20]. Structural inspections are conducted to find structural defects which, may include wheel burnt rails or other structural degradation. Geometric inspections are conducted to find geometric irregularities, which may include rail misalignments and other similar problems. Geometric irregularities may also cause structural defects and either of them can lead to train accidents. More information related to structural and geometric defects can be found in [21,22]. A large body of literature can be found that different techniques of shallow learning, deep learning, and transfer learning for rail tracks inspection. In shallow learning, Refs. [23–27] use support vector machine (SVM), Refs. [19,28] use random forest (RF), Ref. [29] uses Adaboost, and [30,31] use principal components analysis (PCA). Regarding the deep learning approach, Refs. [24,32–35] use convolutional neural networks (CNN), Refs. [36–38] use LSTM, and [39] uses a combination of both CNN and RNN called convolutional–recurrent neural network (CRNN). In transfer learning approaches, Refs. [40,41] uses ResNet, and [41] uses a visual geometry group (VGG).

Shafique et al. [19] use tree-based classification models, random forest (RF) and decision tree (DT), which performed well compared to deep learning models for rail track inspection. Jie Liu et al. [23] investigate different variants of SVM, such as twin SVM, LSSVM, etc. They used data from the braking system and tested the model on the KEEL data repository. They focused on the braking system of high-speed trains using an

imbalanced dataset and verified their work on publicly available datasets. The sensors are used to collect data related to the braking system. Xavier et al. [24] worked on the dataset collected from 85 miles of railway track. They collected an image dataset from a moving vehicle. They implemented both CNN and SVM techniques in their research. They classify rail fasteners as good, missing, or broken. Study [25] worked to detect geometric defects using the SVM model. The authors used the RAS problem-solving competition 2015 dataset and investigated less severe geometric defects that can develop into more severe geometric defects. Hajizadeh et al. [26] performed a structural inspection to detect structural defects using shallow machine learning techniques. He used SVM and introduced a new metric known as positive and unlabeled learning performance (PULP). The study uses PULP to access the working of classifiers on datasets containing only defective observations.

Along the same lines, Ref. [27] utilize an imbalanced dataset of images taken from the video cameras to detect squats defects. They used a semi-supervised technique to handle vast amounts of unbalanced data. The authors detect structural defects of railway tracks in [28]. The PCA, Kernal-PCA, and histogram match (HM) are used for feature extraction. The extracted features from a dataset comprising non-defective images are utilized to train RF to show the superiority of PCA features for the task at hand. The authors used Adaboost in [29] to detect structural damage and damage specifically related to broken rail fasteners. The study used a Haar-like feature set as the geometrical characteristics of fasteners. Famurewa et al. [30] worked on the geometric aspect of railway tracks. The study detects abnormal patterns in sharp curves by using the PCA algorithm. The experiments are carried out on the Swedish railway network using manually collected data. Lasisi et al. [31] also worked on geometric features of rail by using the PCA algorithm for dimensionality reduction. Kernal-PCA is used along with the SVM to classify four different types of defects in the US Class I railway network.

Dawei Li et al. [32] used a faster region-based CNN for the railway track's fault. The images are captured using 6 cameras at the speed of a maximum of 15 km/h to detect structural defects of the surface of the metro tunnel. The study suggests that better results can be obtained at a train speed of 5 to 10 km/h. Faghih-Roohi et al. [33] used the CNN model with a manually labeled image dataset collected from rail tracks of the Netherlands. They used three CNN architectures of different sizes and obtained the best result by the deepest architecture. Similarly, Giben et al. [34] suggested a four-layered CNN model on a manually labeled image dataset collected from the US Northeast Corridor for the classification of material used in the rail track. Santur et al. [35] used a 3D laser camera for a more accurate and fast inspection of the rail track. The study used CNN for the binary classification of rail tracks into 'healthy' or 'faulty' tracks. The study [36] used an ultrasonic vehicle for finding flaws in the railway track. The LSTM model is used for the detection which shows its feasibility of detection faults at the speed of 15 km/h. Fu et al. [37] worked to detect the structural damage. They used LSTM for classification. They worked on the dataset produced by SIMPACK simulation software.

Bruin et al. [38] used LSTM to detect the presence of trains on the track. They worked to detect different structural faults such as ballast degradation, insulated joint defect, conductive object, etc. Qin et al. [39] used a hybrid neural network with the data generated in SIM-PACK simulation software. Four classes are used for experiments, including normal, LD fault, AS fault, and AD fault. Chen et al. [41] worked on image data randomly selected from inspection videos. They took the images of current-carry rings and manually labeled them as 'normal' or 'faulted'. They used VGG, ResNet50, and Res2Net50 models for performance evaluation. They trained and tested the models to predict normal and defective current carry rings. Yang et al. [40] analyzed the structural defects, e.g., rail end better, localized surface collapse, and turning points, etc. They used ResNet and fully CNN (FCN) for the problem. The experimental results show that ResNet is slightly better than FCN with 0.77 and 0.76 precision scores for ResNet and FCN, respectively. Mahfuz et al. [42] worked on a dummy system using an Ultrasonic sensor and Arduino Mega. They worked to detect cracks on railway tracks without using machine learning or deep learning

models. The study [38] worked to find faults on railway track circuits. They used LSTM to determine different faults in circuits such as conductive objects, electrical disturbance, etc.

3. Materials and Methods

The architecture of the proposed approach is given in Figure 2. The main advantage of this research is generating the spectrograms at a run time that makes the system more flexible than the traditional approaches. Traditionally, spectrograms are generated and stored on a disk before feeding them into the machine and deep learning networks, which makes the network slower and requires more space. For example, MusicNet [43], a large labeled dataset publicly available to learn music features requires a substantial disk size. This dataset is 20 GB in size, and creating different types of spectrograms using different parameters, it can take up to 1TB of disk space.

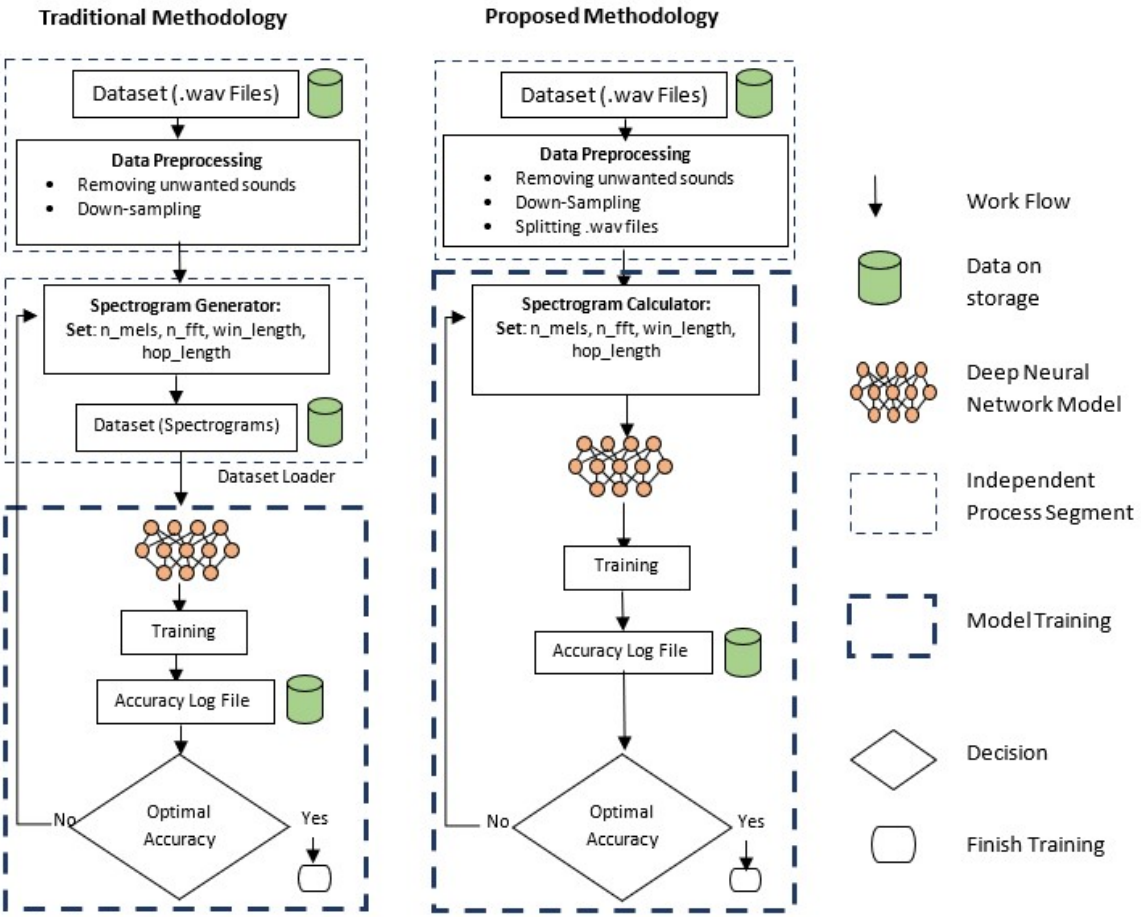


Figure 2. Proposed methodology in contrast with traditional acoustic-based research methodology.

Even if the aspect of high storage requirement is ignored, sound to spectrograms conversion requires a large amount of time. For the experiments, this study uses Librosa, a famous Python library for converting sound to spectrograms. On Google Colab, it took about 20 h and 48 min for converting sound to spectrogram for 1240 samples, and it took about 29 h and 40 min for converting 1850 sound to spectrogram conversion on PC. Librosa takes 1.0 min and 0.96 min on average for creating one spectrogram on both CPU and GPU, respectively. Optimizing the parameters for creating better spectrograms to obtain more accurate results can consume even larger time. So, in this research, instead of generating spectrograms as a separate and independent process, they are calculated on-the-fly as a layer of the neural network using Kapre [44].

3.1. Dataset

The dataset used in this research is collected by [19]. It has 720 mono channel audio 'wav' samples and is collected in the area of Sadiq Abad Junction, Pakistan. Figure 3 shows the setup used for the data collection. An onsite setup is used where 2 microphones were mounted at a distance of 1.75 inches from wheel and track contact point. A mechanical cart is used for data collection at an average speed of 35 km per hour. Two ECM-X7BMP Unidirectional electric condensers are used, which are supplied with a 3-pole locking mini plug. The sensitivity of the microphone is -44.0 ± 3 dB, while the output impedance is 1.2 k$\Omega \pm 30\%$. The dynamic range is 88 dB, the signal-to-noise ratio is 62 dB, and the operating voltage is 5.0 V. The microphones are connected through a wire and are unidirectional. For further details about the data collection process, the readers are referred to [19].

Figure 3. The railway cart equipped with microphones for the data collection [19].

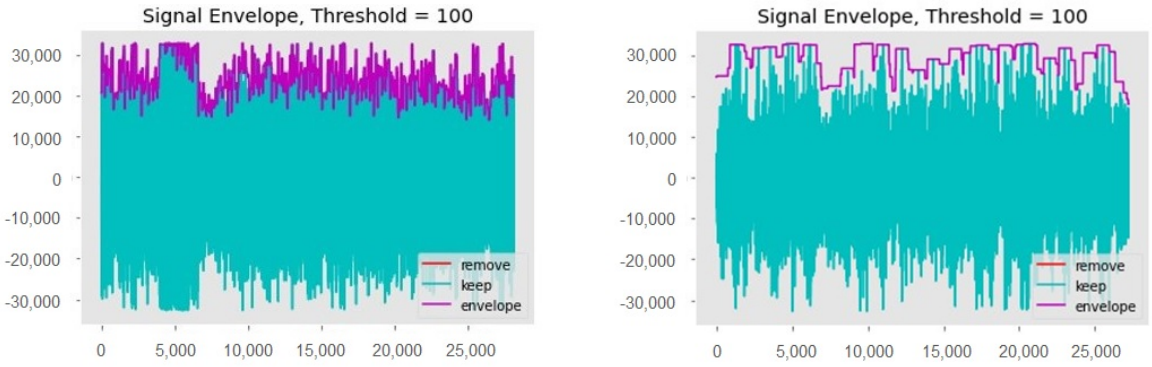
The dataset has three classes, i.e., normal, superelevation, and wheel burnt. It is a balanced dataset with each class containing 240 samples with a length of 17 s each. For experiments, every 17 s, the audio file is split into 1.7 s, 3.4 s, and 8.5 s time intervals (see Table 2).

Table 2. Number of samples in each time split, with and without augmentation.

Audio Length	**1.7 s**	**3.4 s**	**8.5 s**
Number of samples without augmentations	5800	2830	1160
Number of samples with augmentations	N/A	N/A	3480

3.2. Data Preprocessing

Data preprocessing is one of the most important steps required for machine learning models, as the learning process is affected by the preprocessing. For preprocessing, first of all, unwanted sounds are removed from all collected audio samples. These unwanted sounds include the sound of rain and wind, etc. Then, dead space from the audio is removed by using a threshold value, see Figure 4. A threshold value of 100 Hz is used to clean the audio samples, so any frequency below 100 Hz will be removed. Finally, the down-sampling technique is used at the rate of 16,000 [45]. This down-sampling is used during the splitting of all audio files for specific time intervals and saved back to the storage. For the research purpose, audio is split into 1.7 s, 3.4 s, and 8.5 s. Figure 4b, is the zoomed portion of the plot Figure 4a.



(**a**) (**b**)

Figure 4. Signal envelope function: (**a**) Signal envelope plot; (**b**) Zoomed portion of the plot.

3.3. Data Augmentation

Undoubtedly an appropriate artificial neural network (ANN) model and its tuned-up hyperparameters are essential factors for the development of a good model; however, appropriate data are equally important for testing and proving the efficacy of the ANN model.

If the dataset is inappropriate and small, even a good model's result will not be acceptable. So, to enhance the suitability of the data, different audio data augmentation or audio deformation techniques are applied without disturbing the semantic labeling of data, and these techniques are discussed as follows. The study uses audiomentations [46], a Python library for augmentation. Audio augmentations used in this research are as follows:

- **Shift**: In this technique, audio samples are shifted backward or forward. This shifting can be performed with or without rollover. Rollover is of Boolean type; if it is set to true, then the samples rolled beyond the last or the first position are re-injected into the audio. Parameters used for shift are min_fraction = -0.7, max_fraction = 0.7, rollover = 'True', $p = 0.7$. Implementation of shift can be seen in Figure 5d–f.
- **Gaussian Noise**: In this technique of audio data deformation, Gaussian noise is added into the raw audio before feeding it into the ANN. Parameters which we used to add Gaussian noise are min_aplitude = 0.003, max_amplitude = 0.022, and $p = 0.7$. Implementation of Gaussian noise can be seen in Figure 5g–i.

3.4. Proposed Methodology

This research uses three deep learning models for experiments including Convolutional 1D (Conv1D), Convolutional 2D (Conv2D), and LSTM, which belong to the RNN family [47]. RNN and LSTM have proved to be better models as compared to the hidden Markov Model in different domains, e.g., hand-writing recognition [48], emotion recognition [49], generating music composition [50], and speech recognition [51], etc. LSTM consists of a connected memory cell to form an artificial neural network. Each cell has three gates, an input gate, output gate, and forget gate, that are used to perform the read function, write function, and reset function, respectively. Precisely, the input of the input cell is multiplied by the input gate. The output of the output cell is multiplied by the output gate, and previous cell values are multiplied by the forget gate. These functions can be defined by the equations as follows:

$$i^t = \sigma(W_{xi}x^t + W_{hi}h^{t-1} + W_{ci}c^{t-1} + b_i) \quad (1)$$

$$f^t = \sigma(W_{xf}x^t + W_{hf}h^{t-1} + W_{cf}c^{t-1} + b_f) \quad (2)$$

$$a^t = tanh(W_{xc}x^t + W_{hc}h^{t-1} + b_c) \quad (3)$$

$$c^t = f^t \odot c^{t-1} + i^t \odot a^t \quad (4)$$

$$o^t = \sigma(W_{xo}x^t + W_{ho}h^{t-1} + W_{co}c^t + b_o) \quad (5)$$

$$h^t = o^t \odot tanh(c^t) \quad (6)$$

where i is the input gate, f is the forget gate, o is the output gate, a is the cell input activation vector, c is a self-connected state vector, and all have a size similar to h which is the hidden vector. W_{ci}, W_{co}, and W_{cf} are all peephole connection weights which are diagonal. So, element m in each gate vector only receives input from element m of the cell vector.

In this research, bidirectional LSTM (BiLSTM) is used, which increases predictions accuracy as compared to simple LSTM [52]. BiLSTM uses two independent LSTM networks: one network learns data from start to end, and the second network learns data from end to start. The following are the building blocks of our proposed state-of-the-art model.

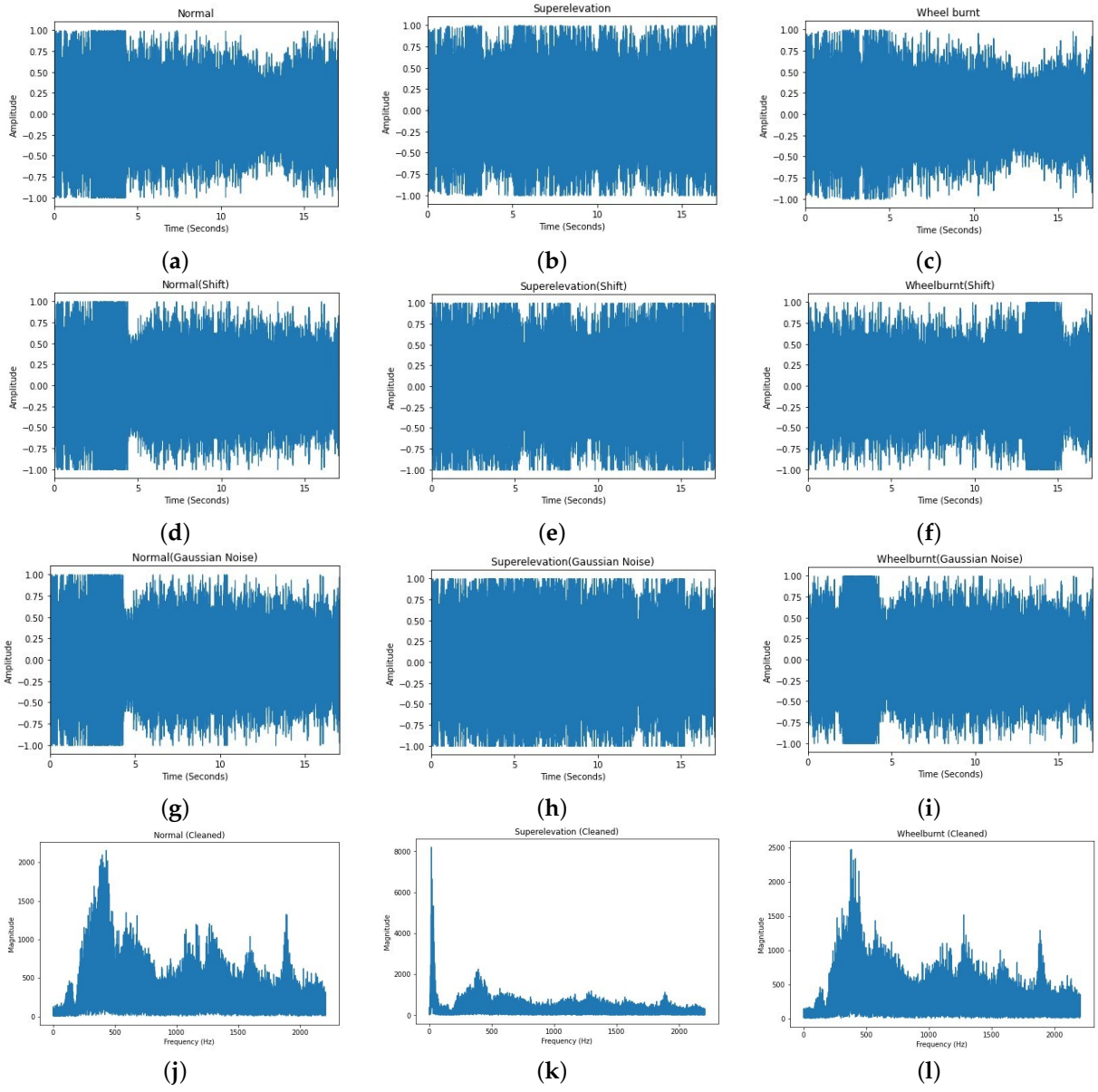


Figure 5. Time domain representation of different classes: (**a**) Normal class, (**b**) Superelevation class, and (**c**) Wheel burnt class; Time domain representation with shift augmentation: (**d**) Normal class, (**e**) Superelevation class, and (**f**) Wheel burnt class; Time domain representation with Gaussian noise augmentation: (**g**) Normal class, (**h**) Superelevation class, and (**i**) Wheel burnt class; Frequency representation of different classes, cleaned using signal envelope function and without augmentation: (**j**) Normal class, (**k**) Superelevation class, and (**l**) Wheel burnt class.

3.4.1. STFT Layer

During the audio recording analog signals are converted into digital signals and the number of samples, recorded during one second, becomes the sampling rate. To convert the time-domain signal into a frequency-domain, Discrete Fourier Transform (DFT) is used [53], which can be represented as:

$$S[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} \tag{7}$$

where $X[k]$ is the frequency-domain output, $x[n]$ is the nth sample in the time-domain audio input, N is the length of the window, n is the discrete time index, k is the discrete frequency index. For real-valued inputs, the frequency domain output $X[k]$ for $k \in [1, N/2]$ is equal to the output $X[k]$ for $k \in [N/2, N-1]$ in reverse order.

Short-Time Fourier Transform (STFT) is a sliding window concept of DFT. Instead of converting the whole time-domain signal into a frequency domain signal, we cut the signal into a small window and then perform transformation [54]. It is the standard way to perform audio analysis-based applications. These segmented signals can be represented as:

$$x_l = w[n]x[n + lL], 0 \leq n \leq N - 1 \tag{8}$$

where l is the index of the frame, N is the length of the window, L is the hop size, and n is index to local time. The parameters used for the model are ($n\_fft = 512$, win_length = 400, hop_length = 160), and the Hann Window function is used:

$$w(k) = 0.5\left(1 - cos\left(\frac{2\pi k}{k-1}\right)\right), \quad k = 1, 2, \ldots, K \tag{9}$$

By stacking up the STFTs, a spectrogram is formed, which is the representation of time-frequency intensity [55]. Fast Fourier transform (FFT) is an algorithm that is used as a quick resource for computing STFT in digital computers [56,57]. We usually obtain the log-spectrogram, which is not from a human perception, (see Figure 6a) because the human ear is more sensitive to low frequencies as compared to high frequencies. To convert log-spectrograms from a human perception, Stevens et al. introduced the Mel frequency scale in 1937 [58]. It was an attempt to quantify the pitch in a way that reflects the same differences between Mel-scale pitch and perceived pitch, irrespective of the frequency in Hertz [58]. Stevens et al. [59] and W. Koening [60] also tried to modify the original Mel-scale, so there is no single formula available in literature [61]. One of the formulas mentioned by O'Shaughnessy in his book [62] is given in Equation (10).

$$m = 2595 log_{10}\left(1 + \frac{f}{700}\right) \tag{10}$$

After obtaining the Mel scale conversion, Mel filter banks can be constructed, which are then multiplied with previously obtained spectrograms to obtain Mel-scale spectrograms [63]. The obtained Mel filter bank can be seen in Figure 7, and from this Mel filter bank, we obtain Mel-spectrograms as given in Figure 6.

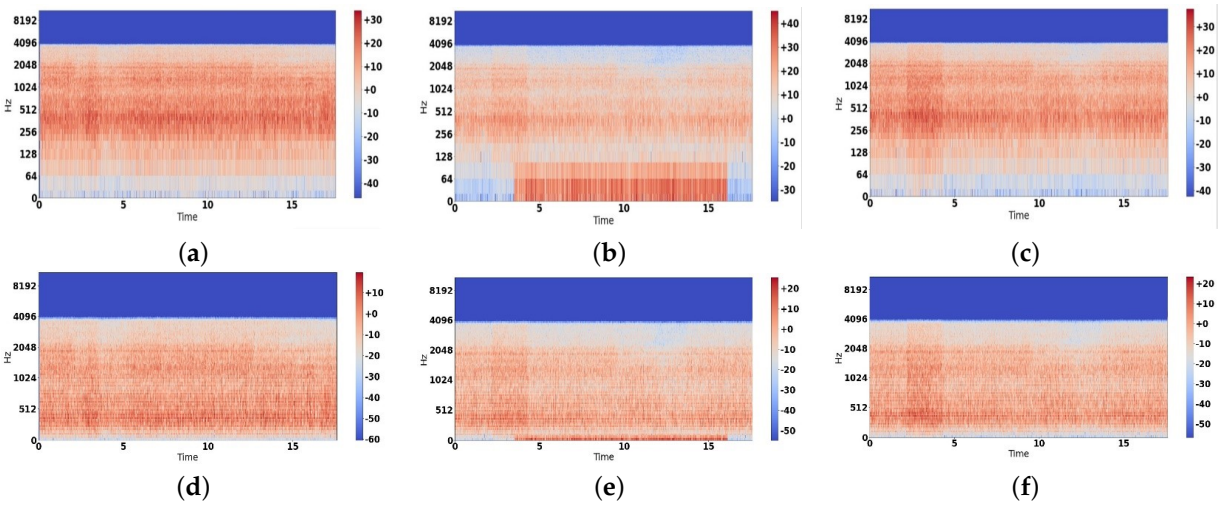


Figure 6. Log spectrogram representation of different classes: (**a**) Normal class, (**b**) Superelevation class, and (**c**) Wheel burnt class; Mel spectrograms after applying Mel filter bank: (**d**) Normal class, (**e**) Superelevation class, and (**f**) Wheel burnt class.

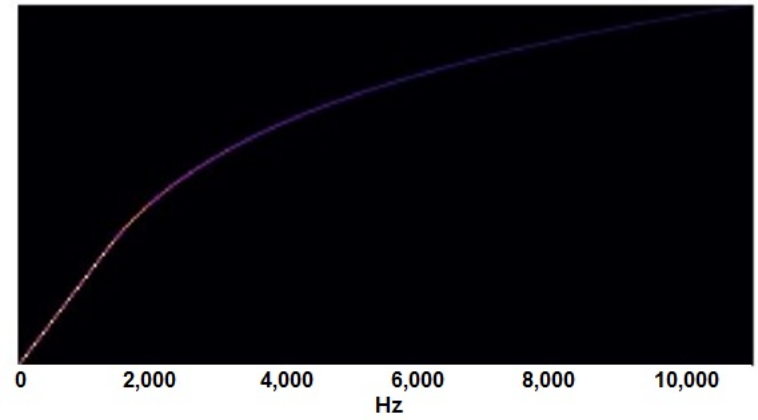


Figure 7. Mel filter bank used for creating Mel-spectrograms.

3.4.2. Batch Normalization Layer

A batch normalization layer is used in the start to normalize data before feeding them into the network [64]. Equation (11) normalizes the data by using a standard normal distribution with zero mean and one variance. This normalization is necessary because the activation function which is used at the first layer of all three models, Conv1D, Conv2D, and BiLSTM, is a hyperbolic tangent:

$$\alpha_i^{norm} = \mathrm{Y}_i \frac{\alpha_i - \mu}{\delta_i} + \beta_i \tag{11}$$

where α_i is the actual activation function of a specific neuron, μ is the mean value, δ_i is the standard deviation of the inputting neuron, Y_i is the expansion factor, β_i is the translation factor, and α_i^{norm} is the new normalized value.

3.4.3. Maximum (Max) Pooling Layer

The max pooling layer is used after every convolutional layer in the Conv1D and Conv2D models and the LSTM model to reduce the number of parameters. The total trainable parameters without the max-pooling layer were 131,092,339, and with yjr max-pooling layer, they were reduced to 544,627 in the Conv2D model.

3.4.4. Dense Layer

The dense layer consists of neurons, and the inputs of these neurons are associated weights; after performing some linear functions, they pass outputs to the next layer [65]. All the neurons of a dense layer are connected to the input and output layers. A dense layer can be presented as:

$$X = f(Y \times w + b) \tag{12}$$

where Y is the input layer, w is the weight, b is th ebias vector, f is the activation function, and X is the output layer [66].

3.4.5. Flatten Layer

The flatten layer is used to convert data into a 1D array, which is the output of convolutional layers. Finally, the output of the flatten layer is connected to a fully connected layer, which eventually performs the final classification.

3.4.6. Drop Out Layer

Overfitting is a severe issue in the field of machine learning and deep learning. Overfitting is a situation in which the model gives satisfying results during the training process, but in the testing process, it gives unsatisfactory results. Normally, it happens when multiple neurons detect the same results repeatedly [67] and neurons have to be dropped. The drop out layer is presented as follows:

$$z_i^{(l+1)} = w_i^{(l+1)} y^l + b_i^{(l+1)} \tag{13}$$

Consider a network of L hidden layers, where $l \in 1, \ldots, L, z^{(l)}$ denotes the vector of inputs into layer l, $y^{(l)}$ denotes the vector of outputs from layer l, $(y^{(0)} = x)$ is the input, and $w^{(l)}$ and $b^{(l)}$ are the weights and biases, respectively, at layer l.

3.4.7. Softmax Layer

Softmax is the last layer of the deep learning model. It is used for multiclass classification in which the output is characterized categorically [67]. It is an activation function and very much important in ANN. Moreover, it is used to decide whether neurons are active or otherwise. The primary objective of the Softmax layer is to highlight the maximum value in the neurons. It assigns one as the maximum weight of neurons and sets other neurons' weight to zero. The Softmax function can be presented as follows:

$$S(y_i) = \frac{e^{yi}}{\sum_k e^{y^k}}, \quad k = 1, 2, \ldots, k \tag{14}$$

where y is the input layer, and S is the output layer.

4. Results and Discussion

For experiments, this study uses the Intel Core i3-4010U@1.70 GHz CPU, with 8 GB RAM and a GTX 550 graphics card. Python 3.8 is selected as the development environment. Several detailed experiments are performed to find the best-performing deep learning model to classify the rail track into three different types, i.e., normal, wheel burnt, and superelevation. For experiments, three deep learning models, Conv1D, Conv2D, and LSTM, are used and all these models are trained and tested for three different lengths of audio samples. Detailed scenarios and their performances are compared as below:

- **Scenario 1**—each audio sample is split into 1.7 s, and Conv1D, Conv2D, and LSTM are trained and tested against all three classes.
- **Scenario 2**—each audio sample is split into 3.4 s, and Conv1D, Conv2D, and LSTM are trained and tested against all three classes.
- **Scenario 3**—each audio sample is split into 8.5 s, and Conv1D, Conv2D, and LSTM are trained and tested against all three classes.

Audio data augmentation is applied to the best-performing model. Augmentation is performed on both datasets, first on the training dataset, and then on testing data to validate the model's generalization. The basic reason for splitting audio into smaller chunks is not to overload the neural network model during training and to avoid overfitting.

4.1. Results for Scenario 1

By splitting a 17 s audio sample into 1.7 s, we have 10 files for each sample. A total of 5100 files are used for training in the first scenario. After performing training, the performance and training time of all three models can be observed in Table 3. All models are trained for 30 epochs and 10-fold cross-validation. In this scenario, Conv2D performed better than the other two models, but it took a significantly longer time than the Conv1D and LSTM models. LSTM took the least time for training.

Table 3. Performance evaluation and training time required by all three models with 1.7 s split time.

Model	Accuracy	Precision	Recall	F1	Training (Hours)
Conv1D	0.949	0.932	0.923	0.924	8.66
Conv2D	0.965	0.955	0.947	0.948	16.9
LSTM	0.933	0.911	0.899	0.900	6.80

4.2. Results for Scenario 2

For the second scenario, all 17 s audio samples are split into 3.4 s for the training purpose. In this scenario, we have 2550 files for training our deep learning models. In this split time, LSTM proves to be the most efficient model again, and Conv1D proves the most effective model, with 95% accuracy as shown in Table 4.

Table 4. Performance evaluation and training time required by all three models with 3.4 s split time.

Model	Accuracy	Precision	Recall	F1	Training (Hours)
Conv1D	0.959	0.939	0.938	0.938	7.5
Conv2D	0.952	0.930	0.929	0.928	15.08
LSTM	0.930	0.920	0.895	0.893	6.75

4.3. Results for Scenario 3

In this scenario, audio samples are split into 8.5 s in length. This split gives maximum accuracy among all three time-split experiments. LSTM performed better than Conv1D in terms of all performance evaluation parameters and almost equally well as Conv2D, but LSTM proved to be the most efficient model by taking the least time for training, as shown in Table 5.

Table 5. Performance evaluation and training time required by all three models with 8.5 s split time.

Model	Accuracy	Precision	Recall	F1	Training (Hours)
Conv1D	0.972	0.958	0.959	0.958	7.91
Conv2D	0.990	0.985	0.986	0.986	16.25
LSTM	0.990	0.986	0.986	0.986	7.5

So, among all three scenarios, the combination of 8.5 s split time with LSTM proves the best combination, as LSTM gives 99% accuracy, which is equivalent to the accuracy of the Conv2D model, but LSTM took about half the training time as compared to Conv2D, as shown in Tables 5 and 6 shown the results with each augmentation approach.

Table 6. Performance evaluation using all variants of LSTM with 8.5 s split time.

LSTM Model	Accuracy	Precision	Recall	F1
Training augmented, Testing un-augmented	0.997	0.995	0.995	0.995
Training augmented, Testing augmented	0.982	0.973	0.973	0.973
Training un-augmented, Testing augmented	0.933	0.912	0.90	0.900

Now, to deeply examine LSTM's effectiveness and efficiency, we performed data augmentation. We performed audio augmentation of the following types:

(**a**) Gaussian noise;
(**b**) Shift;

which enhanced the test dataset from 210 to 630 files. Tables 7 and 8 showing the confusion matrix values with proposed approaches.

Table 7. Confusion matrix of LSTM 8.5 s split, un-augmented training dataset and un-augmented test dataset.

Actual class	Normal	70	0	0
	Superelevation	0	70	0
	Wheel burnt	3	0	67
	Predicted class	Normal	Superelevation	Wheel burnt

Table 8. Confusion matrix of LSTM 8.5 s split, augmented training dataset and un-augmented test dataset.

Actual class	Normal	69	0	1
	Superelevation	0	70	0
	Wheel burnt	0	0	70
	Predicted class	Normal	Superelevation	Wheel burnt

To prove the effectiveness of our LSTM model, we performed testing on the augmented dataset using the model which was previously trained on an un-augmented training dataset. In this case, as the training was performed on an un-augmented dataset, our model made 63 wrong predictions out of 630, as shown in Tables 9 and 10 shown 17 wrong predictions using augmented training dataset and augmented test dataset.

Table 9. Confusion matrix of LSTM 8.5 s split, un-augmented training dataset and augmented test dataset.

Actual class	Normal	201	0	9
	Superelevation	9	201	0
	Wheel burnt	45	0	165
	Predicted class	Normal	Superelevation	Wheel burnt

Table 10. Confusion matrix of LSTM 8.5 s split, augmented training dataset and augmented test dataset.

Actual class	Normal	199	0	11
	Superelevation	0	210	0
	Wheel burnt	6	0	204
	Predicted class	Normal	Superelevation	Wheel burnt

4.4. Discussion

In this section, a detailed analysis of experiments is presented. Three deep learning models, Conv1D, Conv2D, and LSTM, have been thoroughly experimented with. Deep learning models allow the flexibility needed for creating an on-the-fly spectrogram as a layer of the neural network, which is not available in the shallow machine learning models. For experiments with shallow learning, first, an offline dataset of spectrograms must be created. Creating an offline dataset of spectrograms is time-consuming and requires large storage. For the current research, the estimated time required for creating spectrograms can be seen in Table 11, and tuning the parameters of spectrograms requires a large time. In Table 11, it can be observed that for all the cases, the time required for the generation of the Spectrogram dataset is much higher than the training time required by each model, except for the case of 8.5 s, where the Conv 2D's training time is very close to spectrogram generation time. However, due to the slowness of conv2D, it was dropped, and we selected the LSTM model instead.

Table 11. Time required for signal envelop function.

Split Time	Time Required to Generate Spectrogram Dataset (Hours)	Training Time for Conv 1D (Hours)	Training Time for Conv 2D (Hours)	Training Time for LSTM (hours)
1.7 s	85	8.66	16.9	6.8
3.4 s	42.5	7.5	15.08	6.75
8.5 s	16.66	7.91	16.25	7.5

It is, therefore, better to utilize deep learning models' flexibility to create on-the-fly spectrograms and feed them directly into models instead of creating them as a separate and independent process. Three deep learning models, i.e., Conv1D, Conv2D, and LSTM, are incorporated in this research, and each model is trained and tested for three different time splits of the audio dataset, i.e., 1.7 s, 3.4 s, and 8.5 s. LSTM gives the best accuracy using the 8.5 s split time with the least training time. The LSTM model consists of an LSTM layer, a dense layer, a dropout layer, and a softmax layer. LSTM memory cells remember the past data, and it is used for time-series analysis. The percentage of correct predictions made by different variants of LSTM can be seen in Figure 8.

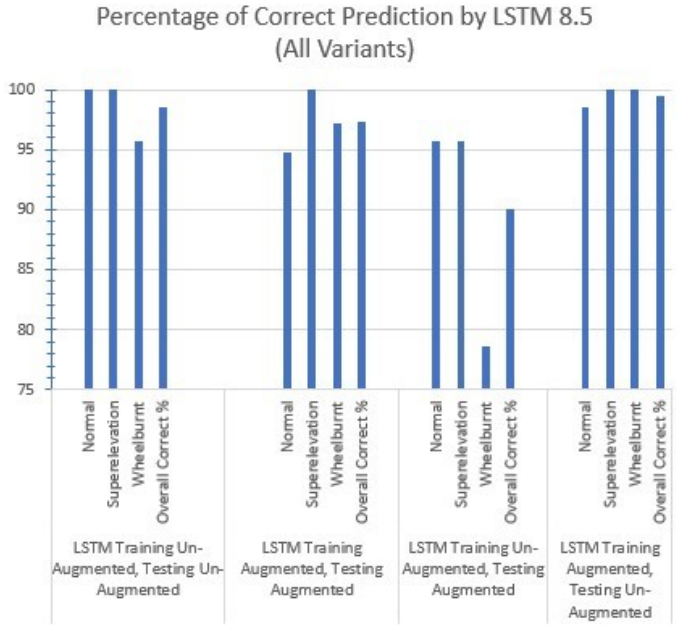


Figure 8. Percentage of correct predictions made by all variants of LSTM with 8.5 s split time.

Tables 12–14 show the architectural details of Conv1D, Conv2D, and LSTM models used for experiments.

Table 12. Layers detail of Conv1D model.

Layer	Values	Output Shape
get_melspectrogram_layer	n_mels = 128, n_fft = 512, win_length = 400, hop_length = 160, sample_rate = 16,000, return_decibel = True	(None, 500, 257, 1)
filterbank	n_mels = 128, n_fft = 512, win_length = 400, hop_length = 160	(None, 500, 128, 1)
batchNormalization	Axis = 2	(None, 500, 128, 1)
Conv1D	Activation = Hyperbolic tangent, Filters = 8, Kernel = 4	(None, 500, 125, 8)
maxPooling2D	2 × 2	(None, 250, 62, 8)
Conv1D	Activation = ReLU, Filters = 16, Kernel = 4	(None, 250, 59, 16)
maxPooling2D	2 × 2	(None, 125, 29, 16)
Conv1D	Activation = ReLU, Filters = 32, Kernel = 4	(None, 125, 26, 32)
maxPooling2D	2 × 2	(None, 62, 13, 32)
Conv1D	Activation = ReLU, Filters = 64, Kernel = 4	(None, 62, 10, 64)
maxPooling2D	2 × 2	(None, 31, 05, 64)
Conv1D	Activation = ReLU, Filters = 128, Kernel = 4	(None, 31, 02, 128)
globalMaxPooling2D	-	(None, 128)
Dropout	Rate = 0.1	(None, 128)
Dense	Activation = ReLU	(None, 64)
Dense	Activation = Softmax	(None, 3)

Table 13. Layers detail of Conv2D model.

Layer	Values	Output Shape
get_melspectrogram_layer	n_mels = 128, n_fft = 512, win_length = 400, hop_length = 160, sample_rate = 16,000, return_decibel = True	(None, 500, 257, 1)
filterbank	n_mels = 128, n_fft = 512, win_length = 400, hop_length = 160	(None, 500, 128, 1)
batchNormalization	Axis = 2	(None, 500, 128, 1)
Conv2D	Activation = Hyperbolic tangent, Filters = 8, Kernel = 7 × 7	(None, 500, 128, 8)
MaxPooling2D	2 × 2	(None, 250, 64, 8)
Conv2D	Activation = ReLU, Filters = 16, Kernel = 5 × 5	(None, 250, 64, 16)
MaxPooling2D	2 × 2	(None, 125, 32, 16)
Conv2D	Activation = ReLU, Filters = 16, Kernel = 5 × 5	(None, 125, 32, 16)
MaxPooling2D	2 × 2	(None, 63, 16, 16)
Conv2D	Activation = ReLU, Filters = 32, Kernel = 3 × 3	(None, 63, 16, 32)
MaxPooling2D	2 × 2	(None, 32, 08, 32)
Conv2D	Activation = ReLU, Filters = 32, Kernel = 3 × 3	(None, 32, 08, 32)
Flatten	-	(None, 8192)
Dropout	Rate = 0.2	(None, 8192)
Dense	Activation = ReLU	(None, 64)
Dense	Activation = Softmax	(None, 3)

Table 14. Layers detail of LSTM model.

Layer	Values	Output Shape
get_melspectrogram_layer	n_mels = 128, n_fft = 512, win_length = 400, hop_length = 160, sample_rate = 16,000, return_decibel = True	(None, 500, 257, 1)
filterbank	n_mels = 128, n_fft = 512, win_length = 400, hop_length = 160	(None, 500, 128, 1)
batchNormalization	Axis = 2	(None, 500, 128, 1)
Reshape	Value = −1	(None, 500, 128)
Dense	Activation = Hyperbolic tangent	(None, 500, 64)

Table 14. *Cont.*

Layer	Values	Output Shape
BiLSTM	Units = 64, return_sequences = True	(None, 500, 64)
Concatenate (Skip connection)	Axis = 2	(None, 500, 128)
Dense	Activation = ReLU	(None, 500, 64)
MaxPooling1D	-	(None, 250, 64)
Dense	Activation = ReLU	(None, 250, 32)
Flatten	-	(None, 8000)
Dropout	Rate = 0.2	(None, 8000)
Dense	Activation = ReLU	(None, 32)
Dense	Activation = Softmax	(None, 3)

4.5. Results Using Log Spectrogram

Experiments are performed using the log spectrogram and the LSTM model with an 8.5-second sub-sample. The performance of the LSTM model is significant with log spectrogram as compared to the Mel spectrogram, as it achieves the highest accuracy 100% with log spectrogram when we used augmentation on training but not on test data. Overall, the performance with the log spectrogram is better as compared to the Mel spectrogram in terms of all evaluation parameters, as shown in Table 15. Figure 9 shows the comparison between the Mel spectrogram (MS) and log spectrogram (LS), while TATU, TATA, and TUTA indicate training augmented, testing un-augmented; training augmented, testing augmented; and training un-augmented, testing augmented, respectively. Log spectrogram took 5.08 h of training, which is less than the Mel-scale spectrograms, which took 7.5 h of training.

Table 15. Results of LSTM model with log Spectrogram.

LSTM Model (8.5 s)	Accuracy	Precision	Recall	F1
Training augmented testing un-augmented (Log Spectrogram)	1	1	1	1
Training augmented testing augmented (Log Spectrogram)	0.995	0.992	0.992	0.992
Training un-augmented testing augmented (Log Spectrogram)	0.96	0.941	0.94	0.94

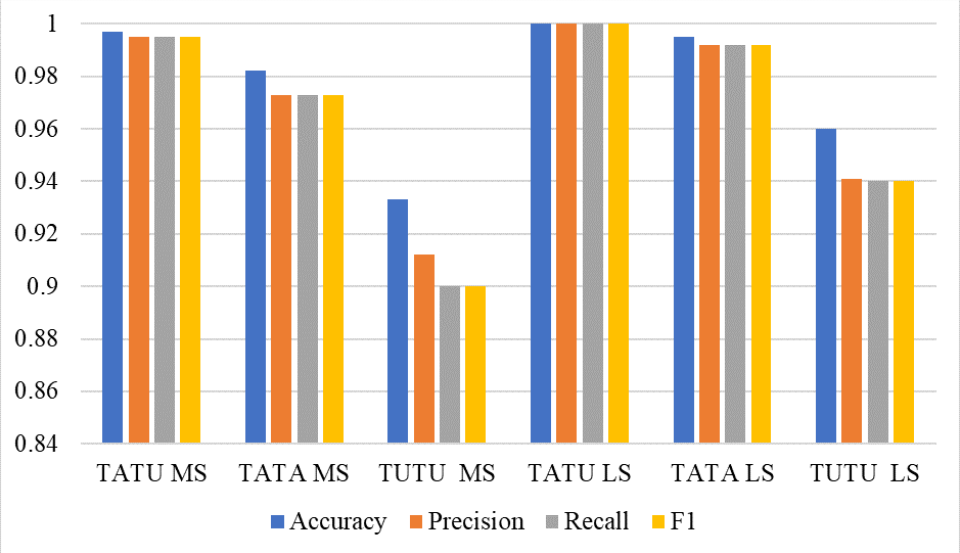


Figure 9. LSTM comparison using log spectrogram and Mel spectrogram.

We also deploy state-of-the-art deep learning models in comparison with the proposed approach for a fair comparison. We deploy three pre-trained models including VVG16, InceptionV3, and ResNet-50. The results given in Table 16 indicate that other deep learning

approaches do not provide high performance similar to the proposed approach. These models do not provide better results with our used dataset without an on-the-fly approach. Apparently, the reason is the complex architecture of the models that need large datasets to produce better results. The current dataset being with a small feature set is not able to provide high accuracy with these models.

Table 16. Results using pre-trained deep learning models on the original dataset.

Model	Accuracy	Precision	Recall	F1 Score
VVG-16	0.41	0.41	0.51	0.45
InceptionV3	0.35	0.35	0.35	0.35
ResNet-50	0.42	0.42	0.42	0.42

4.6. Statistical Significance Test

We carried out the statistical significance test to show the significance of the proposed approach LSTM with log spectrogram [68,69]. We deployed the T-test on the results of LSTM with both the Mel spectrogram and the log spectrogram. In the output of the *t*-test, we have null hypotheses and alternative hypotheses as follows:

- Null hypothesis (H_o): There is no significant difference between the result of LSTM using log spectrogram and LSTM using the Mel spectrogram.
- Alternative hypothesis (H_a): There is a significant difference between the result of LSTM using log spectrogram and LSTM using the Mel spectrogram.

If the *t*-test accepts the null hypothesis, it means there is no significant difference between the result of LSTM using a log spectrogram and LSTM using a Mel spectrogram and vice versa. The *t*-test rejects the null hypothesis and accepts the alternative hypothesis because the t statistic value is greater than the critical value. The t statistic = 4.889 and critical value = 0.540 on the results of LSTM. This *t*-test result shows that LSTM with log spectrogram achieves better results, which are statistically significant.

4.7. Significance of Using Automated Approach for Railway Track Inspection

The proposed approach has several advantages over the current manual inspection system in Pakistan.

- Railway tracks are vulnerable to damage and deterioration by several extreme events, such as buckling by heat [70]. Extreme heat in Pakistan leads to buckling, causing severe accidents. The manual railway track inspection system in Pakistan is unable to perform efficient and effective inspections, which can be augmented by the current proposed automated system.
- Every year, Pakistan faces flooding, which may cause cutting slope failures [71]. Additionally, ballasts may be washed away [72]. With an automated system, the fast and accurate detection of such defects is possible.
- Of the several sensors and methods used for railway track fault detection, such as cameras, radiography, thermal sensors, and optical-laser-based sensors, a microphone can provide fault detection at a higher speed [73]. Despite its shortcoming of being affected by noise, it is still possible to find surface defects, wheel defects, etc., using a microphone setup.
- This study used an on-the-fly approach, which is appropriate and effective as compared to previous studies on train track fault detection.
- This study outperforms the accuracy of previous study [19] in terms of accuracy. This study also takes the advantage of on-the-fly extraction of spectrograms, without saving spectrogram dataset on the disk. On-the-fly approach makes the system flexible, so we can perform experiments with different lengths of audio with different types of spectrograms such as the Mel-Spectrogram and the Log-Spectrogram.

We believe that these features make the current approach suitable enough to be utilized for real-world railway tracks inspection in Pakistan.

5. Conclusions

Keeping in view the importance of railway track inspection for fault detection in saving human lives, this study presents an automatic railway inspection approach using audio data with a novel deep learning LSTM model. In addition, Conv1D and Conv2D models are also tested for the same task. Each sample of 17 s is split into subsamples of 1.7 s, 3.4 s, and 8.5 s to reduce the processing time and computational complexity. Three deep learning models, Conv1D, Conv2D, and LSTM, are extensively studied for each variation of split time. Mel spectrograms and log spectrograms are used for feature extraction and spectrograms are generated on-the-fly as a layer of the deep learning model. This research provides a flexible approach compared to the traditional approach in which the audio dataset is converted into spectrograms' dataset and stored prior to models' training that requires substantial time and space. Several experiments are performed for in-depth investigation of models' performance where, firstly, the un-augmented dataset is used in several experiments with unique combinations of each model with each split time. Secondly, LSTM with 8.5 s split time proved to be the best performer, and it is further tested with augmented training and augmented testing datasets. Finally, after performing augmentation on both training and testing datasets, experiments are performed with 2850 and 630 samples for training and testing, respectively. The results show that LSTM provides an accuracy of 98.2%, a precision of 97.3%, a recall of 97.3%, and an F1 score of 97.3%. The model trained on the augmented dataset obtains an accuracy of 99.7% against the un-augmented test dataset.

Author Contributions: Conceptualization, M.S.A.H.; data curation, M.I.; formal analysis, M.S.A.H. and M.I.; funding acquisition, E.L.; investigation, M.I., I.S.B., F.R. and I.A.; methodology, M.S.A.H., I.S.B. and H.-U.-R.S.; project administration, H.-U.-R.S. and E.L.; resources, M.S.A.H. and I.S.B.; software, F.R. and M.S.A.H.; supervision, I.A.; validation, H.-U.-R.S. and F.R.; visualization, F.R. and E.L.; writing—review and editing, M.S.A.H. and I.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Florida Center for Advanced Analytics and Data Science funded by Ernesto.Net (under the Algorithms for Good Grant).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Railways. Available online: https://www.worldbank.org/en/topic/transport/brief/railways (accessed on 30 January 2022).
2. Nakhaee, M.C.; Hiemstra, D.; Stoelinga, M.; van Noort, M. The recent applications of machine learning in rail track maintenance: A survey. In *Reliability, Safety, and Security of Railway Systems. Modelling, Analysis, Verification, and Certification. RSSRail 2019*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; pp. 91–105.
3. Imdad, F.; Niaz, M.T.; Kim, H.S. Railway track structural health monitoring system. In Proceedings of the 2015 15th International Conference on Control, Automation and Systems (ICCAS), Busan, Korea, 13–16 October 2015; pp. 769–772.
4. British Broadcasting Corporation. Pakistan Train Fire: Are Accidents at a Record High? 2019. Available online: https://www.bbc.com/news/world-asia-50252409 (accessed on 15 February 2022).
5. Statista. Number of Rail Accidents and Incidents in the United States from 2013 to 2020. 2020. Available online: https://www.statista.com/statistics/204569/rail-accidents-in-the-us/ (accessed on 15 February 2022).
6. Audit Report on the Accounts of Pakistan Railways Audit Year 2019–2020. Available online: https://www.agp.gov.pk/SiteImage/Policy/Audit%20Report%202019-20%20Railways..pdf (accessed on 30 January 2022).

7. Timeline of Major Train Accidents in Pakistan in the Past Five Years. Available online: https://www.geo.tv/latest/296133-timeline-of-major-train-accidents-in-pakistan-during-past-five-years (accessed on 20 February 2022).
8. Ph Papaelias, M.; Roberts, C.; Davis, C.L. A review on non-destructive evaluation of rails: State-of-the-art and future development. *Proc. Inst. Mech. Eng. Part F J. Rail Rapid Transit.* **2008**, *222*, 367–384. [CrossRef]
9. Wu, F.; Li, Q.; Li, S.; Wu, T. Train rail defect classification detection and its parameters learning method. *Measurement* **2020**, *151*, 107246. [CrossRef]
10. Chen, W.; Liu, W.; Li, K.; Wang, P.; Zhu, H.; Zhang, Y.; Hang, C. Rail crack recognition based on adaptive weighting multi-classifier fusion decision. *Measurement* **2018**, *123*, 102–114. [CrossRef]
11. Pohl, R.; Erhard, A.; Montag, H.J.; Thomas, H.M.; Wüstenberg, H. NDT techniques for railroad wheel and gauge corner inspection. *NDT Int.* **2004**, *37*, 89–94. [CrossRef]
12. Rose, J.L.; Avioli, M.J.; Mudge, P.; Sanderson, R. Guided wave inspection potential of defects in rail. *NDT Int.* **2004**, *37*, 153–161. [CrossRef]
13. Li, Q.; Zhong, Z.; Liang, Z.; Liang, Y. Rail inspection meets big data: Methods and trends. In Proceedings of the 2015 18th International Conference on Network-Based Information Systems, Taipei, Taiwan, 2–4 September 2015; pp. 302–308.
14. Li, Q.; Ren, S. A real-time visual inspection system for discrete surface defects of rail heads. *IEEE Trans. Instrum. Meas.* **2012**, *61*, 2189–2199. [CrossRef]
15. Edwards, R.; Holmes, C.; Fan, Y.; Papaelias, M.; Dixon, S.; Davis, C.; Drinkwater, B.; Roberts, C. Ultrasonic detection of surface-breaking railhead defects. *Insight-Non-Destr. Test. Cond. Monit.* **2008**, *50*, 369–373.
16. Min, Y.; Xiao, B.; Dang, J.; Yue, B.; Cheng, T. Real time detection system for rail surface defects based on machine vision. *EURASIP J. Image Video Process.* **2018**, *2018*, 1–11. [CrossRef]
17. Doshi, K. Audio Deep Learning Made Simple (Part 1): State-of-the-Art Techniques. Available online: https://towardsdatascience.com/audio-deep-learning-made-simple-part-1-state-of-the-art-techniques-da1d3dff2504 (accessed on 20 February 2022).
18. Yuan, M.; Li, J.; Liu, Y.; Gao, X. Automatic recognition and positioning of wheel defects in ultrasonic B-Scan image using artificial neural network and image processing. *J. Test. Eval.* **2019**, *48*, 308–322. [CrossRef]
19. Shafique, R.; Siddiqui, H.U.R.; Rustam, F.; Ullah, S.; Siddique, M.A.; Lee, E.; Ashraf, I.; Dudley, S. A novel approach to railway track faults detection using acoustic analysis. *Sensors* **2021**, *21*, 6221. [CrossRef]
20. Sharma, S.; Cui, Y.; He, Q.; Mohammadi, R.; Li, Z. Data-driven optimization of railway maintenance for track geometry. *Transp. Res. Part C Emerg. Technol.* **2018**, *90*, 34–58. [CrossRef]
21. Alahakoon, S.; Sun, Y.Q.; Spiryagin, M.; Cole, C. Rail flaw detection technologies for safer, reliable transportation: A review. *J. Dyn. Syst. Meas. Control.* **2018**, *140*, 020801. [CrossRef]
22. Soleimanmeigouni, I.; Ahmadi, A.; Kumar, U. Track geometry degradation and maintenance modelling: A review. *Proc. Inst. Mech. Eng. Part F J. Rail Rapid Transit.* **2018**, *232*, 73–102. [CrossRef]
23. Liu, J.; Hu, Y.; Yang, S. A SVM-based framework for fault detection in high-speed trains. *Measurement* **2021**, *172*, 108779. [CrossRef]
24. Gibert, X.; Patel, V.M.; Chellappa, R. Deep multitask learning for railway track inspection. *IEEE Trans. Intell. Transp. Syst.* **2016**, *18*, 153–164. [CrossRef]
25. Hu, C.; Liu, X. Modeling track geometry degradation using support vector machine technique. *Am. Soc. Mech. Eng.* **2016**, *49675*, V001T01A011.
26. Hajizadeh, S.; Li, Z.; Dollevoet, R.P.; Tax, D.M. Evaluating classification performance with only positive and unlabeled samples. In Proceedings of the Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR), Joensuu, Finland, 20–22 August 2014; pp. 233–242.
27. Hajizadeh, S.; Núnez, A.; Tax, D.M. Semi-supervised rail defect detection from imbalanced image data. *IFAC-PapersOnLine* **2016**, *49*, 78–83. [CrossRef]
28. Santur, Y.; Karaköse, M.; Akin, E. Random forest based diagnosis approach for rail fault inspection in railways. In Proceedings of the 2016 National Conference on Electrical, Electronics and Biomedical Engineering (ELECO), Bursa, Turkey, 1–3 December 2016; pp. 745–750.
29. Xia, Y.; Xie, F.; Jiang, Z. Broken railway fastener detection based on adaboost algorithm. In Proceedings of the 2010 International Conference on Optoelectronics and Image Processing, Haikou, China, 11–12 November 2010; Volume 1, pp. 313–316.
30. Famurewa, S.M.; Zhang, L.; Asplund, M. Maintenance analytics for railway infrastructure decision support. *J. Qual. Maint. Eng.* **2017**, *23*, 310–325. [CrossRef]
31. Lasisi, A.; Attoh-Okine, N. Principal components analysis and track quality index: A machine learning approach. *Transp. Res. Part C Emerg. Technol.* **2018**, *91*, 230–248. [CrossRef]
32. Li, D.; Xie, Q.; Gong, X.; Yu, Z.; Xu, J.; Sun, Y.; Wang, J. Automatic defect detection of metro tunnel surfaces using a vision-based inspection system. *Adv. Eng. Informat.* **2021**, *47*, 101206. [CrossRef]
33. Faghih-Roohi, S.; Hajizadeh, S.; Núnez, A.; Babuska, R.; De Schutter, B. Deep convolutional neural networks for detection of rail surface defects. In Proceedings of the 2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, Canada, 24–29 July 2016; pp. 2584–2589.

34. Giben, X.; Patel, V.M.; Chellappa, R. Material classification and semantic segmentation of railway track images with deep convolutional neural networks. In Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP), Quebec City, QC, Canada, 27–30 September 2015; pp. 621–625.
35. Santur, Y.; Karaköse, M.; Akin, E. A new rail inspection method based on deep learning using laser cameras. In Proceedings of the 2017 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 16–17 September 2017; pp. 1–6.
36. Xu, Q.; Zhao, Q.; Yu, G.; Wang, L.; Shen, T. Rail defect detection method based on recurrent neural network. In Proceedings of the 2020 39th Chinese Control Conference (CCC), Shenyang, China, 27–29 July 2020; pp. 6486–6490.
37. Fu, Y.; Huang, D.; Qin, N.; Liang, K.; Yang, Y. High-speed railway bogie fault diagnosis using LSTM neural network. In Proceedings of the 2018 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 5848–5852.
38. De Bruin, T.; Verbert, K.; Babuška, R. Railway track circuit fault diagnosis using recurrent neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2016**, *28*, 523–533. [CrossRef] [PubMed]
39. Qin, N.; Liang, K.; Huang, D.; Ma, L.; Kemp, A.H. Multiple convolutional recurrent neural networks for fault identification and performance degradation evaluation of high-speed train bogie. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 5363–5376. [CrossRef]
40. Yang, C.; Sun, Y.; Ladubec, C.; Liu, Y. Developing machine learning-based models for railway inspection. *Appl. Sci.* **2021**, *11*, 13. [CrossRef]
41. Chen, Y.; Song, B.; Zeng, Y.; Du, X.; Guizani, M. Fault diagnosis based on deep learning for current-carrying ring of catenary system in sustainable railway transportation. *Appl. Soft Comput.* **2021**, *100*, 106907. [CrossRef]
42. Mahfuz, N.; Dhali, O.A.; Ahmed, S.; Nigar, M. Autonomous railway crack detector robot for bangladesh: SCANOBOT. In Proceedings of the 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Dhaka, Bangladesh, 21–23 December 2017; pp. 524–527.
43. Cheuk, K.W.; Anderson, H.; Agres, K.; Herremans, D. nnaudio: An on-the-fly gpu audio to spectrogram conversion toolbox using 1d convolutional neural networks. *IEEE Access* **2020**, *8*, 161981–162003. [CrossRef]
44. Choi, K.; Joo, D.; Kim, J. Kapre: On-gpu audio preprocessing layers for a quick implementation of deep neural network models with keras. *arXiv* **2017**, arXiv:1706.05781.
45. Genussov, M.; Cohen, I. Musical genre classification of audio signals using geometric methods. In Proceedings of the 2010 18th European Signal Processing Conference, Aalborg, Denmark, 23–27 August 2010; pp. 497–501.
46. Jordal, I. Audiomentations: A Python Library for Audio Data Augmentation. 2021. Available online: https://zenodo.org/record/5470338#.YaMxwFVByUk (accessed on 20 Febuary 2022).
47. Graves, A.; Schmidhuber, J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Netw.* **2005**, *18*, 602–610. [CrossRef] [PubMed]
48. Graves, A.; Liwicki, M.; Fernández, S.; Bertolami, R.; Bunke, H.; Schmidhuber, J. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *31*, 855–868. [CrossRef] [PubMed]
49. Wöllmer, M.; Metallinou, A.; Eyben, F.; Schuller, B.; Narayanan, S. Context-sensitive multimodal emotion recognition from speech and facial expression using bidirectional lstm modeling. In Proceedings of the INTERSPEECH 2010, Makuhari, Japan, 26–30 September 2010; pp. 2362–2365.
50. Liu, I.; Ramakrishnan, B. Bach in 2014: Music composition with recurrent neural network. *arXiv* **2014**, arXiv:1412.3191.
51. Sak, H.; Senior, A.; Beaufays, F. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. *arXiv* **2014**, arXiv:1402.1128.
52. Ganegedara, T. *Natural Language Processing with TensorFlow: Teach Language to Machines Using Python's Deep Learning Library*; Packt Publishing Ltd.: Birmingham, UK, 2018.
53. Wang, Z. Fast algorithms for the discrete W transform and for the discrete Fourier transform. *IEEE Trans. Acoust. Speech Signal Process.* **1984**, *32*, 803–816. [CrossRef]
54. Nawab, S.H. Short-time Fourier transform. In *Advanced Topics in Signal Processing*; Prentice Hall: Hoboken, NJ, USA, 1988.
55. Kersta, L. Amplitude Cross-Section Representation with the Sound Spectrograph. *J. Acoust. Soc. Am.* **1948**, *20*, 796–801. [CrossRef]
56. Cooley, J.W.; Tukey, J.W. An algorithm for the machine calculation of complex Fourier series. *Math. Comput.* **1965**, *19*, 297–301. [CrossRef]
57. Bergland, G.D. A guided tour of the fast Fourier transform. *IEEE Spectr.* **1969**, *6*, 41–52. [CrossRef]
58. Stevens, S.S.; Volkmann, J.; Newman, E.B. A scale for the measurement of the psychological magnitude pitch. *J. Acoust. Soc. Am.* **1937**, *8*, 185–190. [CrossRef]
59. Stevens, S.S.; Volkmann, J. The relation of pitch to frequency: A revised scale. *Am. J. Psychol.* **1940**, *53*, 329–353. [CrossRef]
60. Koening, W. A New Frequency Scala for Acoustic Measurements. *Bell Lab Rec.* **1949**, 299–301. Availavle online: https://ci.nii.ac.jp/naid/10009375717/en/ (accessed on 12 January 2022).
61. Umesh, S.; Cohen, L.; Nelson, D. Fitting the mel scale. In Proceedings of the 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No. 99CH36258), Phoenix, AZ, USA, 15–19 March 1999; Volume 1, pp. 217–220.
62. Douglas, O.; Shaughnessy, O. *Speech Communications: Human and Machine*; IEEE Press: New York, NY, USA, 2000; pp. 367–433.
63. Rabiner, L.; Schafer, R. *Theory and Applications of Digital Speech Processing*; Prentice Hall Press: Hoboken, NJ, USA, 2010.

64. Reshi, A.A.; Rustam, F.; Mehmood, A.; Alhossan, A.; Alrabiah, Z.; Ahmad, A.; Alsuwailem, H.; Choi, G.S. An efficient CNN model for COVID-19 disease detection based on X-ray image classification. *Complexity* **2021**, *2021*, 6621607. [CrossRef]
65. Rustam, F.; Siddique, M.A.; Siddiqui, H.U.R.; Ullah, S.; Mehmood, A.; Ashraf, I.; Choi, G.S. Wireless capsule endoscopy bleeding images classification using CNN based model. *IEEE Access* **2021**, *9*, 33675–33688. [CrossRef]
66. Ari, S.; Hembram, K.; Saha, G. Detection of cardiac abnormality from PCG signal using LMS based least square SVM classifier. *Expert Syst. Appl.* **2010**, *37*, 8019–8026. [CrossRef]
67. Yu, M.; Huang, Q.; Qin, H.; Scheele, C.; Yang, C. Deep learning for real-time social media text classification for situation awareness–using Hurricanes Sandy, Harvey, and Irma as case studies. *Int. J. Digit. Earth* **2019**, *12*, 1230–1247. [CrossRef]
68. Rustam, F.; Ashraf, I.; Mehmood, A.; Ullah, S.; Choi, G.S. Tweets classification on the base of sentiments for US airline companies. *Entropy* **2019**, *21*, 1078. [CrossRef]
69. Omar, B.; Rustam, F.; Mehmood, A.; Choi, G.S. Minimizing the overlapping degree to improve class-imbalanced learning under sparse feature selection: application to fraud detection. *IEEE Access* **2021**, *9*, 28101–28110.
70. Powrie, W. On track: The future for rail infrastructure systems. *ICE Proc. Civ. Eng.* **2014**, *167*, 177–185. [CrossRef]
71. Oslakovic, I.S.; ter Maat, H.; Hartmann, A.; Dewulf, G. Climate change and infrastructure performance: should we worry about? *Procedia-Soc. Behav. Sci.* **2012**, *48*, 1775–1784. [CrossRef]
72. Jaroszweski, D.; Quinn, A.; Baker, C.; Hooper, E.; Kochsiek, J.; Schultz, S.; Silla, A. Guidebook for enhancing resilience of European railway transport in extreme weather events. In *MOVE-IT—Management Weather Events in Transport System—Railways*; 7th Framework Program; European Commission: Brussels, Belgium, 2014.
73. Falamarzi, A.; Moridpour, S.; Nazem, M. A review on existing sensors and devices for inspecting railway infrastructure. *J. Kejuruter.* **2019**, *31*, 1–10.