Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 18 No. 8 (2024) | OPEN ACCESS

https://doi.org/10.3991/ijim.v18i08.45917

PAPER

A Proposed CNN Model for Audio Recognition on Embedded Device

Minh Pham Ngoc[1], Tan Ngo Duy[2](✉), Hoan Huynh Duc[3], Kiet Tran Anh[2]

[1]Institute of Information Technology, Vietnam Academy of Science and Technology, Hanoi, Vietnam

[2]Space Technology Institute, Vietnam Academy of Science and Technology, Hanoi, Vietnam

[3]Quy Nhon University, Binhdinh, Vietnam

ndtan@sti.vast.vn

ABSTRACT

The audio detection system enables autonomous cars to recognize their surroundings based on the noise produced by moving vehicles. This paper proposes the utilization of a machine learning model based on convolutional neural networks (CNN) integrated into an embedded system supported by a microphone. The system includes a specialized microphone and a main processor. The microphone enables the transmission of an accurate analog signal to the main processor, which then analyzes the recorded signal and provides a prediction in return. While designing an adequate hardware system is a crucial task that directly impacts the predictive capability of the system, it is equally imperative to train a CNN model with high accuracy. To achieve this goal, a dataset containing over 3000 up-to-5-second WAV files for four classes was obtained from open-source research. The dataset is then divided into training, validation, and testing sets. The training data is converted into images using the spectrogram technique before training the CNN. Finally, the generated model is tested on the testing segment, resulting in a model accuracy of 77.54%.

KEYWORDS

autonomous cars, convolutional neural networks (CNNs), vehicle classification, audio recognition

1 INTRODUCTION

In the last few decades, autonomous cars have achieved remarkable milestones, effectively demonstrating the feasibility of integrating self-driving technology into the real world [1] [2] [3]. One of the most important sub-technologies enabling this application is precise environmental perception, especially detecting the presence of other vehicles [1] [2]. While computer vision, a prevalent technique used in the autonomous industry, can provide accurate predictions regarding the types of surrounding vehicles, its effectiveness is often constrained by the limited field of view of the camera [4]. A multi-camera system, which raises the complexity of system design is required to address this challenge [5]. In this context, utilizing vehicle

Ngoc, M.P., Duy, T.N., Duc, H.H., Anh, K.T. (2024). A Proposed CNN Model for Audio Recognition on Embedded Device. *International Journal of Interactive Mobile Technologies (iJIM)*, 18(8), pp. 116–126. https://doi.org/10.3991/ijim.v18i08.45917

Article submitted 2023-10-16. Revision uploaded 2024-02-17. Final acceptance 2024-02-12.

© 2024 by the authors of this article. Published under CC-BY.

sound, or noise, as an identifier to complement the computer vision system emerges as a cost-effective approach due to its affordability and ability to provide adequate information despite potential camera occlusion.

To differentiate vehicles based on their sounds, one effective approach is to apply the convolutional neural network (CNN), a deep learning technique known for its proficiency in classification tasks. According to Li, J. et al.'s findings, the CNN shows a testing accuracy ranging from 73% to 82%, positioning it in the middle among the six architectures compared in their research, below the deep neural network (DNN) and its hybrid combination [6]. Consequently, CNN remains a commendable technique for sound recognition applications. Moreover, it has gained popularity in various sound recognition research domains, such as noise level assessment, environmental sound analysis, and speech recognition, showcasing positive results [7] [8] [9] [10] [11] [12].

A commonly used visualization technique with acoustic CNN models is spectrogram visualization [13] [14].

In this paper, we combined CNN to categorize vehicles based on their acoustic signatures. Moreover, before generating spectrogram images, each sound recording undergoes filtering using the fast Fourier transform (FFT) method to eliminate noise and retain significant frequencies. The resulting spectrogram images were partitioned into training, validation, and testing sets before being processed by the CNN model.

2 DATA COLLECTION

Due to limited data availability, we decided to use two open-source datasets: the IDMT-traffic and vehicle interior sound datasets [15] [16]. Our collected dataset includes cars, trucks, motorcycles, and buses. Each label contains approximately 700 five-second audio files. All audio files are recorded in '.wav' format and then inputted into Python programs for filtering, partitioning, and training. However, some modifications are necessary to ensure compatibility with the Keras CNN model. Firstly, each audio WAV file is rescaled to 16-bit integer type by dividing it by its maximum value and then multiplying it by 32767, which represents the limit for 16-bit data. Subsequently, a check for mono audio is performed to combine stereo audio recordings into a single (mono) channel. Following this, the processed audio files are organized into separate directories for training, validation, and testing.

3 DATA PREPROCESSING

3.1 Fast Fourier transform filtering

Currently, FFT is a common method in digital signal processing. The term FFT refers to an efficient algorithm for calculating the discrete Fourier transform (DFT), which is a close approximation of the continuous Fourier transform (CFT). Due to the high computational demands of CFT, which are often unaffordable for most projects, DFT has become the theory. Despite being a discrete representation, the DFT preserves the essential signal characteristics necessary for a variety of applications, such as digital signal classification [17].

The vehicle sound recognition system utilizes the FFT filter, employing the FFT algorithm from the Python SciPy library to compute the Fourier model for each signal. The equation governing the transformation is represented by the following:

$$y(n) = \sum_{k=0}^{N-1} e^{-j2\pi \frac{nk}{N}} \cdot x(k) \quad (1)$$

Where:
N: total number of samples.
$n = 0, 1, 2, ..., N\text{-}1$.
$y(n)$: DFT magnitude of nth frequency of the signal.
$x(k)$: input signal at kth discrete time.

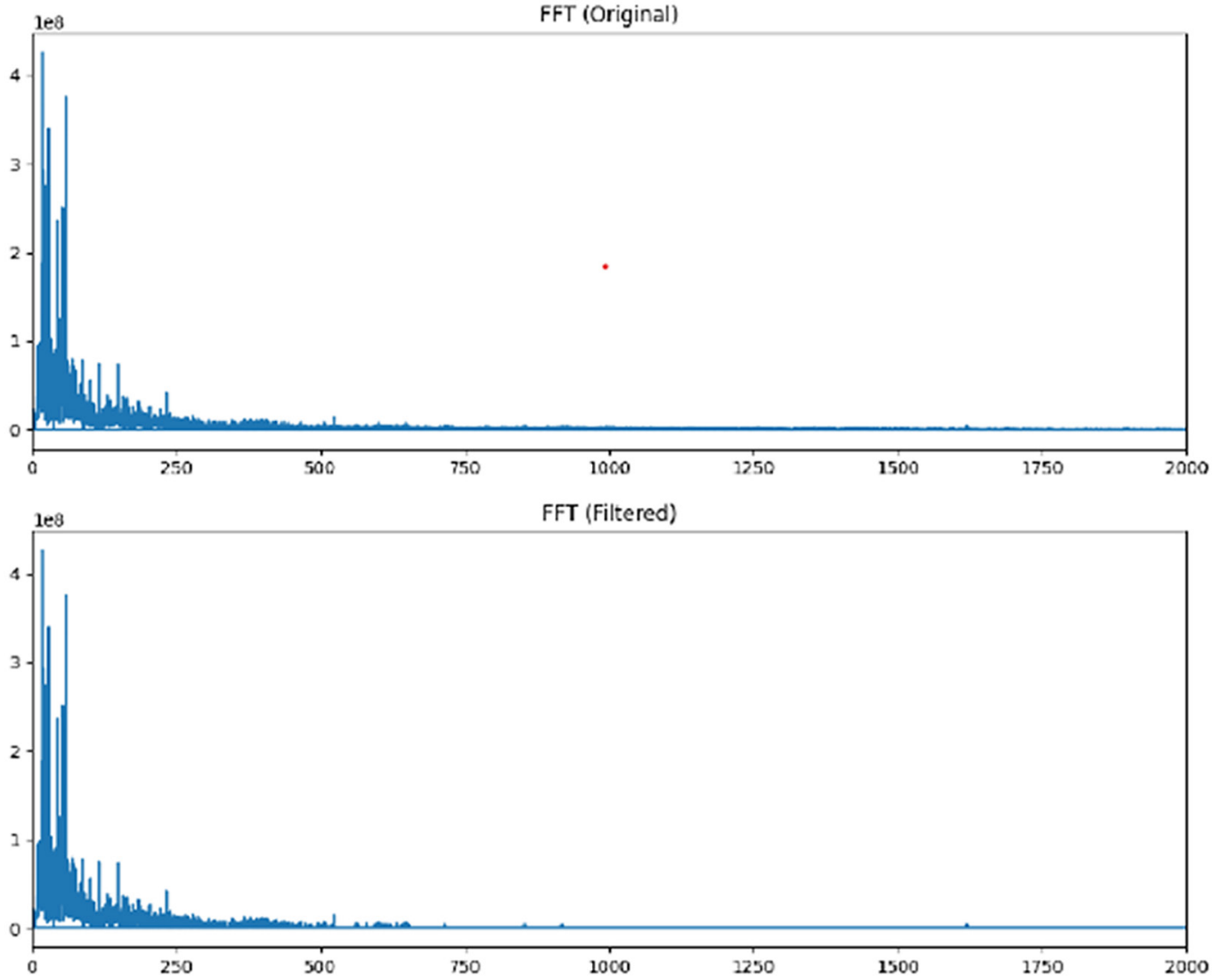


Fig. 1. FFT diagram of a bus record before (top) and after (bottom) filtering

Subsequently, the Fourier data is filtered using the 50th percentile to eliminate background noise and unimportant frequencies. However, it's crucial to exercise

caution while filtering the audio data to strike a balance between noise reduction and preserving data variation. Figure 1 shows the FFT results of the FFT filtering process, which removes almost all the noise at high frequencies. After the application of the filter, Figure 2 reveals that the overall shape of the sound remains, but the resulting record becomes cleaner.

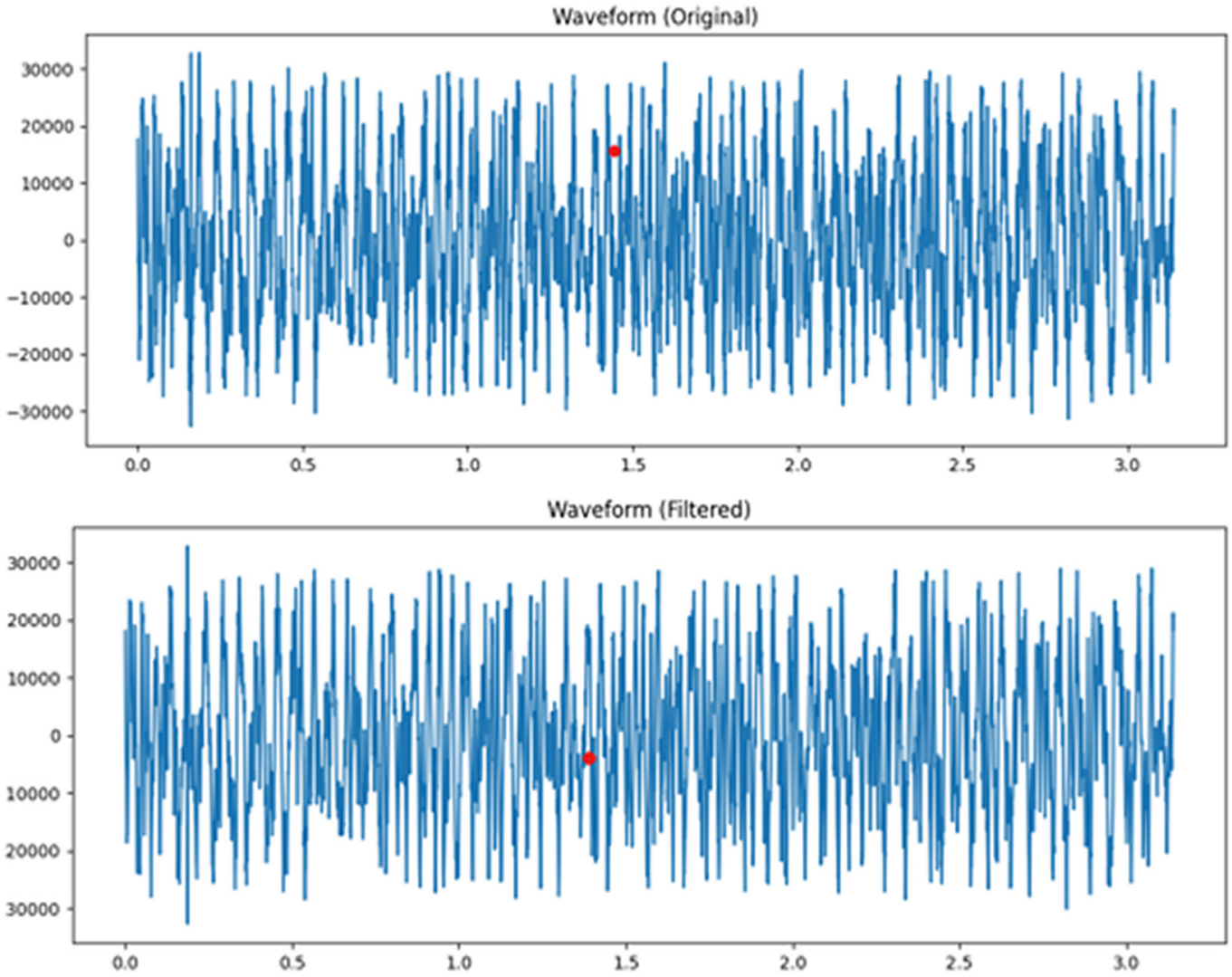
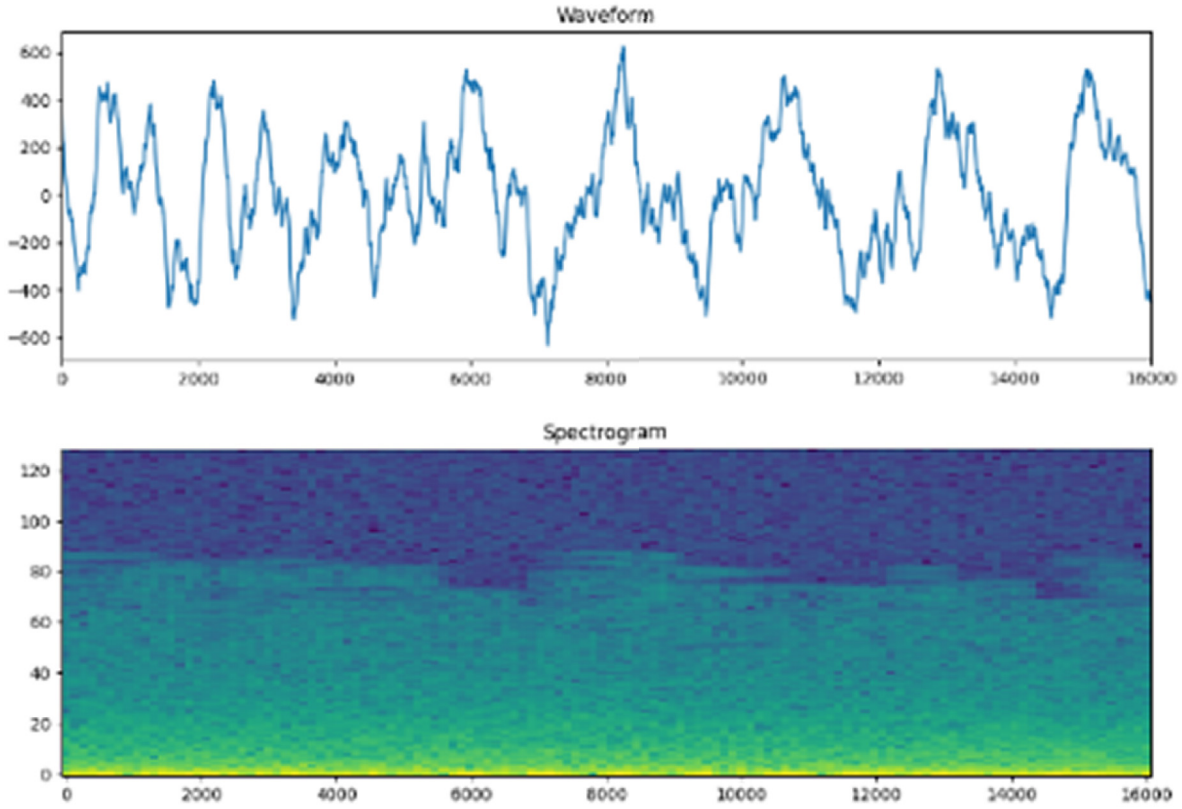


Fig. 2. Shows the waveform of the same bus record before (top) and after (bottom) filtering

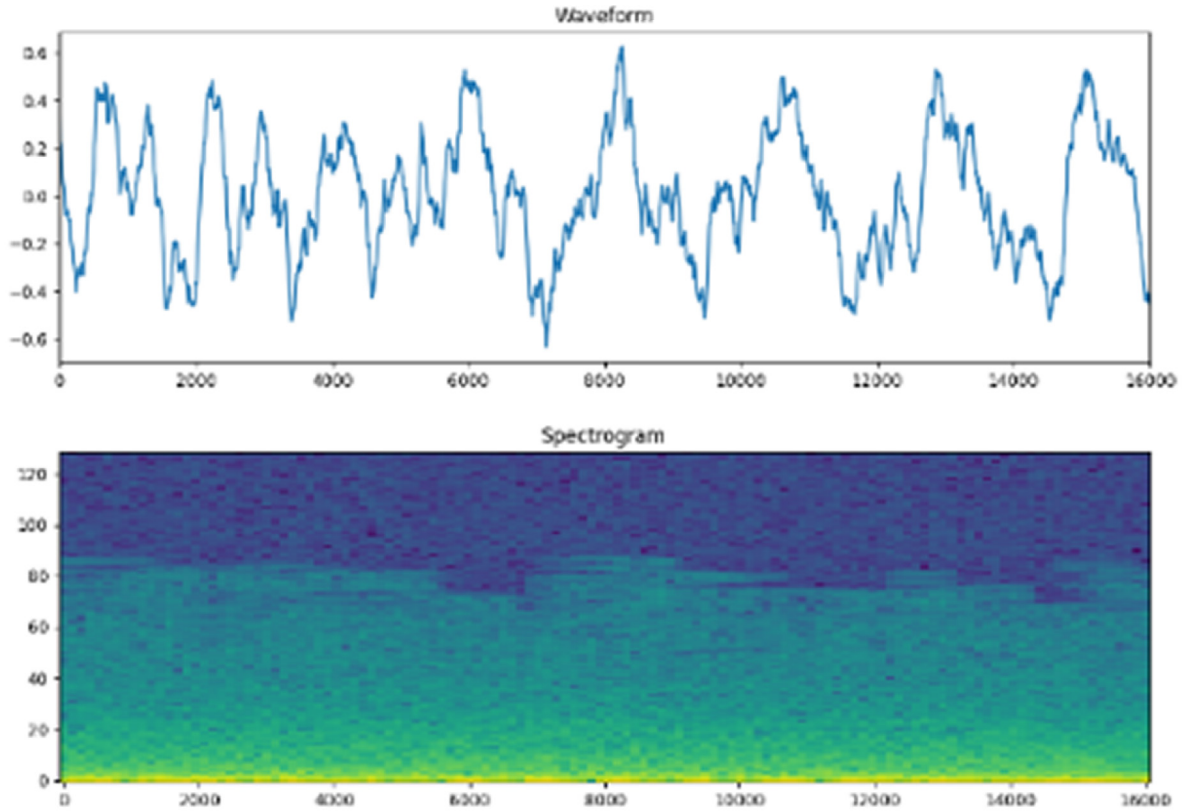
3.2 Spectrogram visualization

While running on a busy street, the presence of a large number of vehicles of the same type can amplify the noise level. This problem causes variations in the amplitude of the audio recording. For instance, recording the noise generated by a thousand motorbikes simultaneously will result in a higher amplitude compared to the noise produced by a hundred motorbikes. Moreover, the distance from the source to the recorder also significantly contributes to this variation. As a result, it is essential to apply a method that can handle sound amplification.

A spectrogram, a widely used visual representation method that depicts signal strength by comparing the amplitudes of various frequencies [11], effectively resolves the issue of sound amplification.



a) The original signal and its spectrogram



b) The amplified signal and its spectrogram

Fig. 3. The amplified signal (b) results in the same spectrogram as the original one (a)

To get spectrogram data, a short time Fourier transform (STFT) function available in TensorFlow, an open-source Python library, can be used. These data can finally be easily plotted by Matplotlib library's functions after being processed.

The spectrogram is a time-frequency analysis technique, however, converting a signal into spectrogram images can streamline the CNN training process compared to maintaining it in an array form.

4 MODEL DESCRIPTION AND TRAINING

4.1 Overview of the training process

At this phase of the research, to simplify the model training process and focus on finding an optimal algorithm to effectively preprocess the mixed-up sound while driving on the street, a basic CNN model from the Python TensorFlow library is selected. The model is a prospective choice because it can meet the general requirements of the system, which include multiclass classification and detecting the types of vehicles causing the sound recordings.

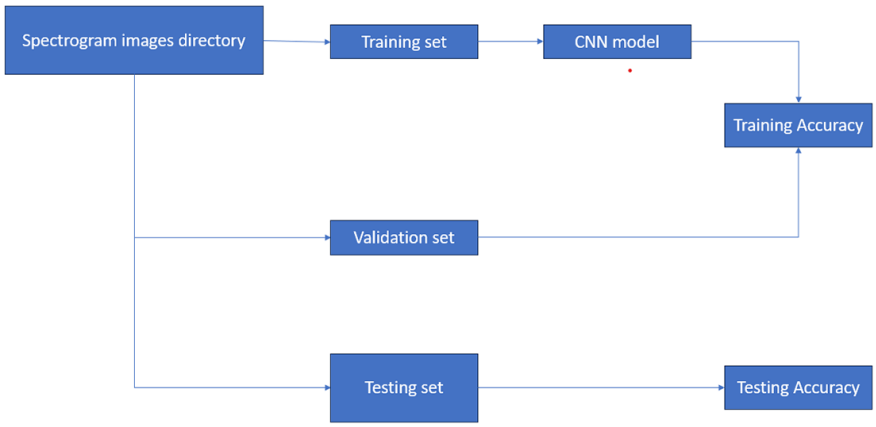


Fig. 4. Overview of model training process

During the model training step, all processed spectrogram images are sent to a directory to be separated into training, validation, and testing sets. The training set is used to calculate the coefficients of the CNN model, while the validation set is used to monitor model accuracy during training. Finally, the testing set assesses the generalization of the final model.

4.2 Convolutional neural network

Recently, CNN has become a popular option in various machine learning applications.

Every digital image is constructed by a matrix of colors, and each individual pixel can be defined by a 3D vector in the RGB scale. Therefore, in general, an image can be analyzed as matrices of numbers arranged in RGB scale [18].

The core algorithm of a CNN model involves using kernels, also known as convolutional filters, to traverse through the spectrogram image and produce feature maps. These feature maps contain crucial details that help determine the likelihood of the image belonging to a specific class [19] [20]. The kernel is a 3x3 matrix used to gather data within each nine-pixel square of a spectrogram. The feature map value

is determined by the sum of the element-wise product matrix of the kernel and the nine-pixel segment of the spectrogram.

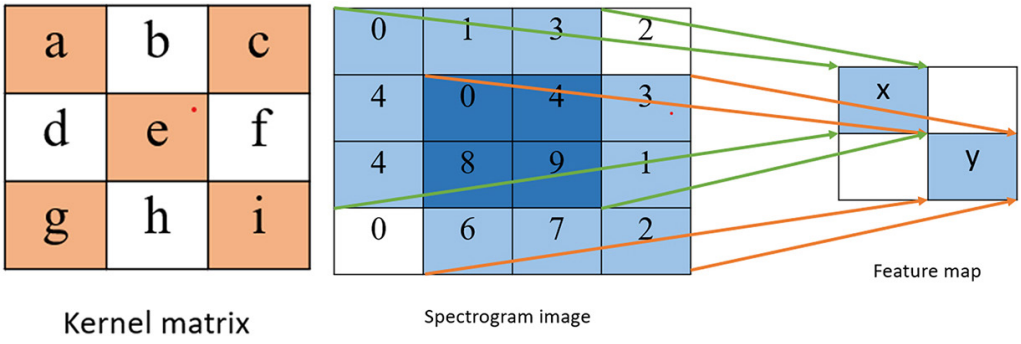


Fig. 5. Kernel matrix and convolutional process (The elements of the kernel matrix are calculated during the training process)

Next, the rectified linear unit (ReLU) activation function is introduced to transform linear data to a non-linear form, enabling non-linear form, approximation of the model. It follows the function f(x) = max (0, x) to replace the negative elements with zero in feature maps after convolution. This step enhances the computational power and the ability of the model to learn complex patterns.

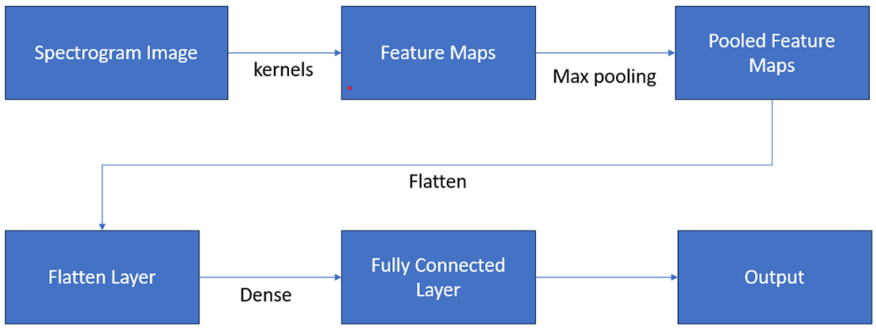


Fig. 6. Overview of convolutional neural networks

After being created, these feature maps are pooled to select the maximum value in each feature map instead of the average to minimize the impact of noise. The pooling layers are applied to extract the essential features from the maps. The number of featured maps and types of pooling layers depend on the application. The flatten layer is used to reorganize the output feature maps to match the inputs of a fully connected layer. The fully connected layer applies weights to each element of flattened feature maps to calculate the probability of the classifications.

Our model follows the traditional CNN architecture as provided by the TensorFlow library. The input data is formatted with a shape of (124, 129, 1). To streamline computation, a resizing function is applied to adjust the size to (32, 32). The last parameter, denoted as 1 in the input shape, signifies a single-channel spectrogram image, and it remains unchanged while the other dimensions undergo reduction. Subsequently, the spectrogram images pass through two convolutional layers with ReLU activation functions, resulting in the creation of 64 feature maps. These feature maps undergo a downsampling process through both max-pooling and a 25% dropout layer, resulting in 48 coefficients that act as input to a flattened layer. Following this, a dense layer with ReLU activation combines the 48 coefficients into 128 units. After establishing a full connection, 50% of the units are dropped out to reduce overfitting. Lastly, a dense layer with 64 final units is used to classify the data into four vehicle classes,

determining the output weights. In terms of optimization, the model is compiled using the Adam optimizer. The Adam optimizer, a widely adopted algorithm for training neural networks, combines adaptive learning rates with momentum-based optimization techniques. By dynamically adjusting learning rates for each parameter and maintaining moving averages, Adam efficiently converges during training and is a popular choice in various deep learning applications. A callback function is implemented to stop the training process if the validation accuracy decreases for 2 to 3 consecutive epochs.

After the training process, we calculate two important metrics: the loss value and the validation accuracy. The loss value measures the disparity between the predicted output and the correct labels. On the other hand, validation accuracy provides insights into the performance of the model on a validation dataset.

5 TESTING AND EVALUATION

With a dataset of approximately 3274 audio files divided into 4 classes, we first trained with a batch size of 64 and 10 epochs. In training, there are two models generated: with and without the FFT filter. The purpose of this step is to compare the test results obtained from each model and evaluate the effectiveness of integrating a FFT filtering step or locating the optimal percentiles for FFT filtering.

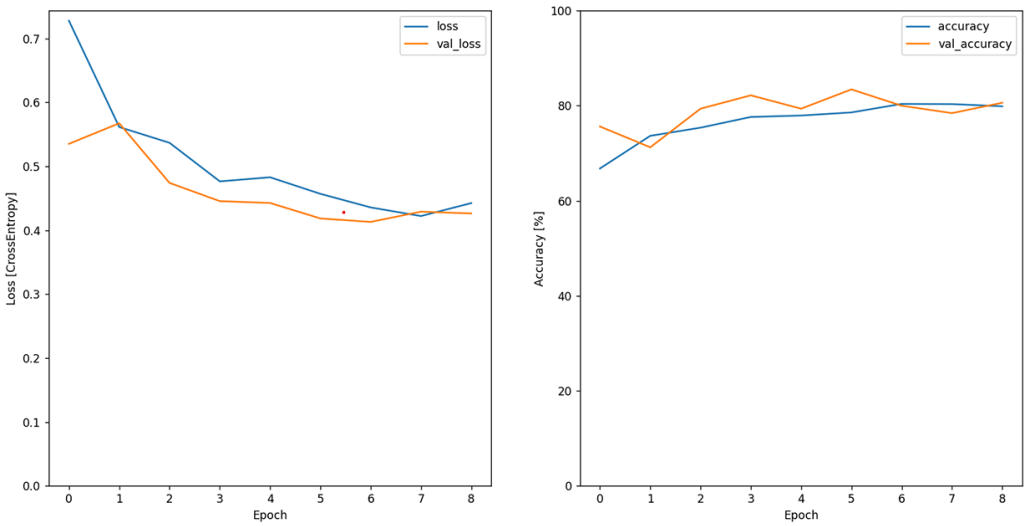


Fig. 7. Loss value (left) and validation accuracy (right) of the model without filter

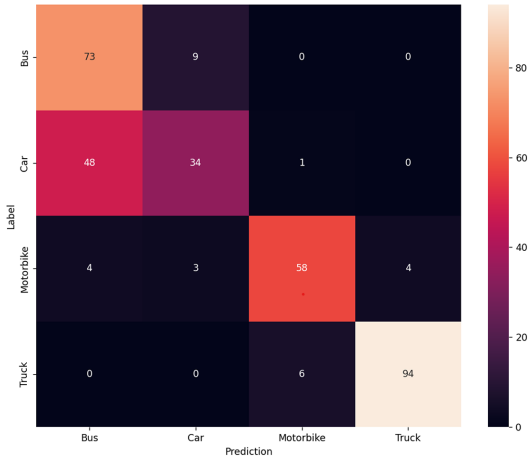


Fig. 8. Testing result of the model without filtering

In Figure 7, we observe a notable trend: the loss value decreases while the validation accuracy improves, which is an optimistic pattern. The highest accuracy achieved is 83.44%. Although the overall testing accuracy is 77.54%, Figure 8 highlights misclassifications between bus and car sounds. This confusion can be reasonably explained by the resemblance between the two in features and functions, which makes accurate prediction difficult.

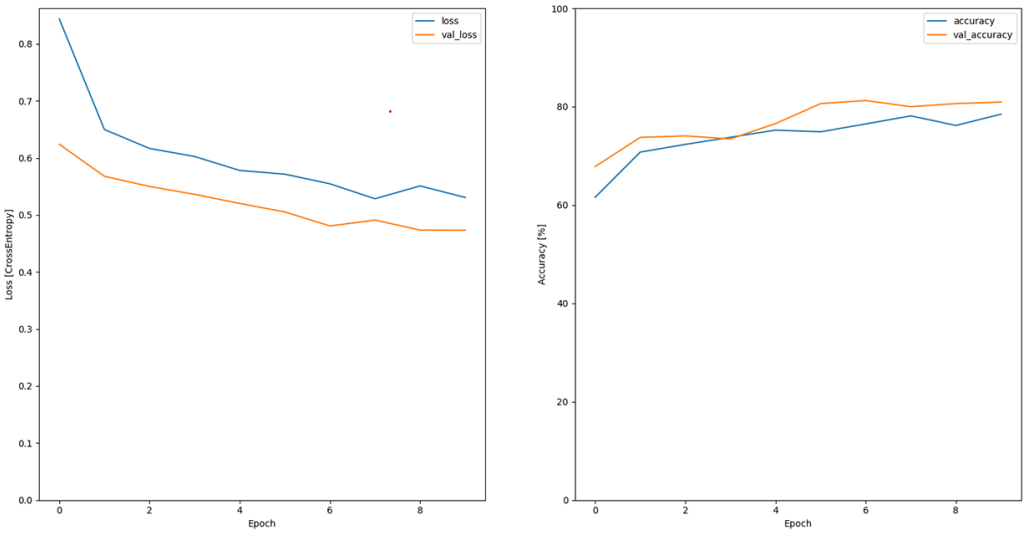


Fig. 9. Loss value (left) and validation accuracy (right) of the model with 50 percentile filter

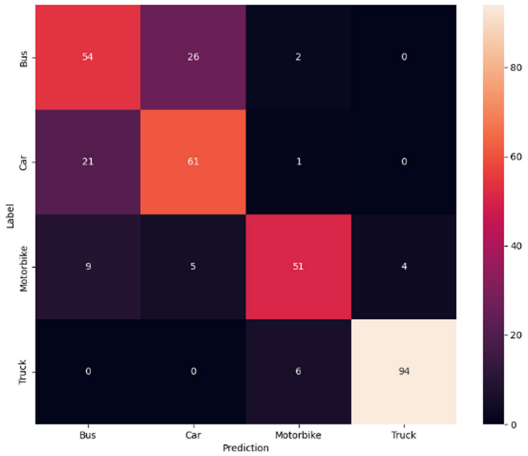


Fig. 10. Testing result with 50th percentile filter

On the other hand, with a 50th percentile filter, both the loss value and validation accuracy maintain consistent performance. Although the maximum validation accuracy experiences a slight decrease to 80.42% and the testing accuracy remains at 77.54%, it becomes more stable across each epoch. Additionally, the testing results in Figure 10 also indicate a noticeable improvement in the previously mentioned issue of confusion between bus and car sounds.

In general, the testing results show a positive sign since the model's accuracy is within the expected range of Juncheng Li's research [6]. For the future of our research, a DNN model or a DNN with the Smile6k feature can be applied to improve recognition accuracy [6].

6 ACKNOWLEDGEMENT

We extend our gratitude to the creators of the TensorFlow tutorial titled 'Simple Audio Recognition: Recognizing Keywords' for providing invaluable guidance on generating the initial source code used in this application. Simultaneously, this paper is funded by the project "Research and Proposal of an Audio Recognition Embedded Device Using IoT and Deep Learning Technology," CSCL02.03/22-23, by the Institute of Information Technology, Vietnam Academy of Science and Technology.

7 REFERENCES

[1] R. Hussain and S. Zeadally, "Autonomous cars: Research results, issues, and future challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1275–1313, 2019. https://doi.org/10.1109/COMST.2018.2869360

[2] J. Ondruš, E. Kolla, P. Vertaľ, and Ž. Šarić, "How do autonomous cars work?" *Transportation Research Procedia*, vol. 44, pp. 226–233, 2020. https://doi.org/10.1016/j.trpro.2020.02.049

[3] T. Manglani, R. Rani, R. Kaushik, and P. K. Singh, "Recent trends and challenges of diverless vehicles in real world application," in *International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, IEEE, 2022, pp. 803–806. https://doi.org/10.1109/ICSCDS53736.2022.9760886

[4] A. F. Abbas, U. U. Sheikh, F. T. AL-Dhief, and M. N. H. Mohd, "A comprehensive review of vehicle detection using computer vision," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 19, no. 3, p. 838, 2021. https://doi.org/10.12928/telkomnika.v19i3.12880

[5] Z. Zheng and K. Zhang, "A real-time vehicle indoor positioning algorithm based on multi-camera perception," in *2022 6th International Symposium on Computer Science and Intelligent Control (ISCSIC)*, IEEE, 2022, pp. 200–204. https://doi.org/10.1109/ISCSIC57216.2022.00050

[6] J. Li, W. Dai, F. Metze, S. Qu, and S. Das, "A comparison of deep learning methods for environmental sound detection," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2017, pp. 126–130. https://doi.org/10.1109/ICASSP.2017.7952131

[7] J. Cao, M. Cao, J. Wang, C. Yin, D. Wang, and P.-P. Vidal, "Urban noise recognition with convolutional neural network," *Multimed. Tools Appl.*, vol. 78, no. 20, pp. 29021–29041, 2019. https://doi.org/10.1007/s11042-018-6295-8

[8] J. Salamon and J. P. Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Process Lett.*, vol. 24, no. 3, pp. 279–283, 2017. https://doi.org/10.1109/LSP.2017.2657381

[9] F. Demir, M. Turkoglu, M. Aslan, and A. Sengur, "A new pyramidal concatenated CNN approach for environmental sound classification," *Applied Acoustics*, vol. 170, p. 107520, 2020. https://doi.org/10.1016/j.apacoust.2020.107520

[10] M. Massoudi, S. Verma, and R. Jain, "Urban sound classification using CNN," in *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, IEEE, 2021, pp. 583–589. https://doi.org/10.1109/ICICT50816.2021.9358621

[11] D. Palaz, M. Magimai-Doss, and R. Collobert, "Analysis of CNN-based speech recognition system using raw speech as input," in *Interspeech 2015*, pp. 11–15. https://doi.org/10.21437/Interspeech.2015-3

[12] G. Kovács, L. Tóth, D. Van Compernolle, and S. Ganapathy, "Increasing the robustness of CNN acoustic models using autoregressive moving average spectrogram features and channel dropout," *Pattern Recognit. Lett.*, vol. 100, pp. 44–50, 2017. https://doi.org/10.1016/j.patrec.2017.09.023

[13] A. M. Badshah, J. Ahmad, N. Rahim, and S. W. Baik, "Speech emotion recognition from spectrograms with deep convolutional neural network," in *2017 International Conference on Platform Technology and Service (PlatCon)*, IEEE, 2017, pp. 1–5. https://doi.org/10.1109/PlatCon.2017.7883728

[14] C. Wang *et al.*, "Real-time vehicle sound detection system based on depthwise separable convolution neural network and spectrogram augmentation," *Remote Sens. (Basel)*, vol. 14, no. 19, p. 4848, 2022. https://doi.org/10.3390/rs14194848

[15] E. Akbal, T. Tuncer, and S. Dogan, "Vehicle interior sound classification based on local quintet magnitude pattern and iterative neighborhood component analysis," *Applied Artificial Intelligence*, vol. 36, no. 1, 2022. https://doi.org/10.1080/08839514.2022.2137653

[16] J. Abeber, S. Gourishetti, A. Katai, T. Claub, P. Sharma, and J. Liebetrau, "IDMT-Traffic: An open benchmark dataset for acoustic traffic monitoring research," in *2021 29th European Signal Processing Conference (EUSIPCO)*, IEEE, 2021, pp. 551–555. https://doi.org/10.23919/EUSIPCO54536.2021.9616080

[17] E. O. Brigham, *The Fast Fourier Transform and Its Applications*. Prentice-Hall Signal Processing Series: Advanced Monographs. Prentice Hall, 1988. [Online]. Available: https://books.google.com.vn/books?id=XfJQAAAAMAAJ.

[18] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: An overview and application in radiology," *Insights Imaging*, vol. 9, no. 4, pp. 611–629, 2018. https://doi.org/10.1007/s13244-018-0639-9

[19] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Trans Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 6999–7019, 2022. https://doi.org/10.1109/TNNLS.2021.3084827

[20] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual understanding of convolutional neural network- a deep learning approach," *Procedia Comput. Sci.*, vol. 132, pp. 679–688, 2018. https://doi.org/10.1016/j.procs.2018.05.069

8 AUTHORS

Minh Pham Ngoc, Institute of Information Technology, Vietnam Academy of Science and Technology, Hanoi, Vietnam (ORCID: https://orcid.org/0009-0008-8300-1389).

Tan Ngo Duy, Space Technology Institute, Vietnam Academy of Science and Technology, Hanoi, Vietnam (E-mail: ndtan@sti.vast.vn; ORCID: https://orcid.org/0000-0002-5991-2755).

Hoan Huynh Duc, Quy Nhon University, Binhdinh, Vietnam (ORCID: https://orcid.org/0009-0006-4603-5622).

Kiet Tran Anh, Space Technology Institute, Vietnam Academy of Science and Technology, Hanoi, Vietnam (ORCID: https://orcid.org/0009-0004-7103-6163).