An Approach to Instrumental Song Classification Utilizing Spectrogram and Convolutional Neural Networks

Anh Tuan Le, Hien Thanh Thi Nguyen, Hoa Huu Nguyen, Hai Thanh Nguyen ✉

Abstract Searching for a song is a necessity, where the copyright of the song is a significant concern. This study proposes a method to classify and identify songs based on specific features that the model learns from music data. Python and CNN programming languages are used to build the model. In the first process, support libraries are used to extract audio data from the computer in WAV format. The dataset ***A*** includes 100 songs without lyrics, while the dataset ***B*** includes 100 audio files with the same song name but played in different types of musical instruments. We randomly cut the original audio files into clips less than 10 seconds long because users often use a specific code to find the entire track. The original audio files are split into clips of different lengths in the training set, including 1s, 3s, 5s, 10s, 20s, 30s, 60s, and 90s. Next, the Short-Time Fourier Transform was used to convert the audio data to the frequency domain. Finally, a shallow Convolutional Neural Network (CNN) and a Fully Connected layer (FC) were used to perform song classification tasks. We found that data augmentation by dividing the entire song into small pieces based on length significantly improved classification performance compared to those not using this technique. This research positively contributes to the advancement of e-commerce music systems, where listeners can enjoy music conveniently and memorably.

1 Introduction

Musical instruments have become an integral part of the daily lives of countless individuals worldwide. The meaning of musical instruments reflects people's diverse and rich musical preferences. According to data from Spotify platform [1], there

Anh Tuan Le, Hien Thanh Thi Nguyen, Hoa Huu Nguyen, Hai Thanh Nguyen ✉
Can Tho University, Can Tho, Vietnam,
e-mail: anhb1906362@student.ctu.edu.vn, ntthien@ctu.edu.vn, nhhoa@ctu.edu.vn, nthai.cit@ctu.edu.vn

are more than 10 million users across 7 European countries; they access a variety of music genres such as relaxing, Ambient, classical, electronic, jazz, etc. Through a mobile app or streaming, individuals can enjoy the instrument, which helps to have a tranquil environment, reduces stress, and fosters mental serenity. This experience improves mood and concentration during various activities, including work and study. Furthermore, musical instruments have universal appeal, overcoming language barriers and allowing people from different cultures to connect with music on an emotional level. Therefore, there are many demands to find a song from a short piece of music that people accidentally listen to from somewhere. Besides, the music industry's problem of copying and copyright infringement is enormous and familiar today.

With the development of technology and the internet, copying, distributing, or illegally using musical works has become more accessible and more popular than ever. The challenge is defining a song from its short audio excerpt. This study will research solutions to this challenge. The CNN [2] are used to analyze spectral images obtained from audio data, allowing users to explore diverse musical genres based on their acoustic characteristics. This approach lets users quickly and easily discover their favorite tracks, enhancing their music discovery journey. The critical contribution of this study consists of two important phases. Firstly, the proposed method randomly cuts original audio files into audio segments of less than 10 seconds because users often use a specific small segment to find the entire track. Secondly, during the training period, the original audio files were divided into audio segments of different lengths, including 1s, 3s, 5s, 10s, 20s, 30s, 60s and 90 seconds. This method determines which segments will provide the most accurate detection performance when applied to the recognition model. Besides, this study performs the classification and identifying process using machine learning models, mainly CNN, on spectrograms [3]. This approach highlights the study's uniqueness and demonstrates the specificity of using CNN to classify and search for specific acoustic properties of instrumental music. The following are our main contributions:

- We proposed an approach of using a shallow CNN network that can classify the song based on the spectrogram of the segment of the song. We also conducted to compare the experimental results related to data fragmentation and the effect of training on audio segments and the whole music; we found that when we break down a piece of music into smaller segments and train each part, classification results on segments are better than on the whole music.
- We proposed that data augmentation by dividing the songs into multiple segments could improve the classification accuracy and that CNN outperforms FC on both datasets. We found that performing data augmentation by dividing the dataset into segments improved the classification performance more than without applying this technique. This approach has significantly increased the performance of the classification model. The data enhancement consists of dataset A, and additional data is added as fragments of different lengths, helping the model focus on learning and classification based on the unique characteristics of each part of the data in each song.

- In addition, we did a study to determine the importance of choosing a test song with a small number of seconds. The results show that choosing a shorter test segment improved the accuracy of the classification process. This can be explained by the fact that shorter segments often contain only a tiny portion of the music, focusing on the unique characteristics of specific instruments.
- Both CNN and FC models are applied to this study. By comparison, a vital exploration found that the CNN has a higher potential aggregation and performs better in the test phase.

This article is structured in the following order. In Section 2, we will first get acquainted with the research related to the topic. Then, in Section 3, we will investigate how to unravel the problem. Next, Section 4 Introduction to data used in research, data characteristics, and division, environmental setups. Finally, in Section 5, we will present our conclusions based on the experimental results.

2 Related work

At present, there have been many remarkable studies with increasingly improved algorithms, bringing high Efficiency in finding instrumental music. These studies have achieved high accuracy and attracted the interest of music lovers worldwide, especially in countries with a long history of music with various instruments. Diverse traditions are imbued with ethnic cultures such as China, Thailand, India, and Mexico. The work in [4] was essential to the Search Engine Case Study. This study used approximately nearest neighbors to preprocess instrumental songs and extracted the characteristics of the tracks in the archive using Mel frequency Cepstral Factor (MFCC) extraction. The study resulted in 100 tracks of different lengths, with a sampling frequency of 16000, and each MFCC having a length of 13, yielding the best results. The accuracy in the Top 1 was 36%, the Top 5 was 4%, and the Top 10 was 44%. Additionally, the study applied the ANN algorithm and MFCC feature extraction to perform song search by harmony, helping to select the most suitable parameters for the most accurate results and the fastest search time and built a website that integrated algorithms to help users find songs more easily. The work in [5] was an essential work in the field of environmental sound recognition using deep neural network models, especially CNN. This article evaluated the potential of convolutional neural networks in classifying short audio clips of environmental sounds. A plunging model consisted of 2 convolutional layers with maximum pooling and two fully connected layers trained on a low-level representation of audio data (segmented spectrogram) with delta. The network's accuracy was evaluated on three publicly available datasets of urban and environmental records. This model outperformed baseline implementations based on mel frequency cepstral coefficients and achieved results comparable to other state-of-the-art approaches.

The work in [6] was essential to musical information extraction and automatic note recognition. The study focused on converting an audio signal into a frequency-time (spectral) representation and using a machine-learning model to predict musical notes and corresponding musical instruments to develop a system. Systems automatically extracted information from audio signals, such as note names and specific instruments in polyphonic music. They used pitch and fundamental frequency features to describe audio characteristics in expressions with spectrum maps. The study employed machine learning models such as Support Vector Machines (SVM) to classify musical notes and instruments. Models were trained on musical datasets annotated with musical notes. The authors tested the music dataset and compared the predicted results with the label information. The results showed the feasibility and potential of the method in realizing musical notes and instruments from audio signals. This led to the development of AMT using machine learning models, which opened the way to automating the processing of traditional music recordings.

The study [7] proposed a scheme to distinguish between vocal (song) and non-vocal (instrumental) music signals. They used similarity features based on spectral images to classify audio signals. Research observed that spectral images of musical instrument signals showed more stable peaks over time, and this was not the case for a song. It promoted research into finding features based on spectral images. Contextual features were calculated based on the occurrence pattern of the most critical frequencies over the time scale and the overall texture pattern revealed by the time-frequency distribution of the signal intensity. For classification, the authors relied on the Random Sample Consensus (RANSAC) technique that could handle many data types in one class. RANSAC created models for each class and performed classification model production based on it. The author prepared a music database consisting of 300 instrument files and 300 song files. Each file contained audio of approximately 40-45 seconds in duration. Files were obtained from burning CDs, recording live shows, and downloading from various sites on the Internet. The sampling of the given frequency data was 22050 Hz. Samples were 16-bit and monochrome. Data on flute, piano, guitar, and drums were available and saved. The songs also came in many genres like classical, jazz, rock, and Bhangra (also a North Indian genre). Audio files were divided into frames to calculate the characteristics. Corresponding to each frame was usually obtained. Each frame comprised 256 samples, including 128 cross-samples between two consecutive frames. Achieved results: Neural Network: instrument 85%, Song (80.33%), Overall (82.67%), RANSAC: instrument 97%, Song (93%), Overall (95%). In this study, the performance of RANSAC on the computed feature set was examined, and Neural Network performance on the same feature set showed that the performance of RANSAC was much better than that of Neural Network.

One of the notable studies was conducted by the authors of the study [8]. Unlike traditional methods, the research explored a classification scheme for musical instruments using features learned from a CNN. However, another method for feeding the phase information into a neural network was proposed in this research. Because a time series is a sequence of one-dimensional data, the filter (kernel) used for the time series in CNN should also be one-dimensional. This restriction gave rise to a limitation when applying a filter convolution because information in a specific temporal location could only be convoluted once. In contrast, a two-dimensional spectrogram image could be analyzed using a two-dimensional filter multiple times per single temporal location, providing more dimensions for analysis. The research initiative aimed at enhancing the performance of a classifier through data augmentation techniques. Specifically, the researchers shifted a time block by 13 samples, creating 13 temporally shifted Multiresolution Recurrence Plots (MRP) datasets. These datasets were then utilized to construct seven layers of RP images, each with a quadrupled length compared to the preceding block in the time series. The temporal sizes of these layers were set at 25, 27, 29, 211, 213, 215, and 217. By combining the proposed MRPs and spectrogram images with a multi-column network, they obtained an improved instrument classification performance using the UIOWA MIS database. The research also tested the performance of the combined network for a piano classification task. The results show that the classification of musical instrument timbre could be improved using MRP data with a spectrogram image and by feeding the data and image into multi-column CNN.

This study [9] had two primary contributions: first, it proposed a deep convolutional neural network architecture for environmental sound classification - model SB-CNN, PiczakCNN, SKM. Second, it proposed the use of audio data augmentation to overcome the problem of data scarcity and explored the influence of different augmentations on the performance of the proposed CNN architecture. The deep CNN architecture proposed in this study comprised three convolutional layers interleaved: 24, 48, and 48 filters with two pooling operations, followed by two fully connected (dense) layers. The research experimented with four different audio data augmentations (deformations): Time Stretching (TS), Pitch Shifting (PS1), Pitch Shifting (PS2), Dynamic Range Compression (DRC), and Background Noise (BG). Each deformation was applied directly to the audio signal before converting it into the input representation to train the network (log-mel-spectrogram). Combined with data augmentation, the proposed model achieved state-of-the-art results for environmental sound classification. The study improved performance by combining a deep, high-capacity model and a boosted training dataset. This combination outperformed the proposed CNN without boosting and the model ”shallow” dictionary geometry with enhancement. Finally, the study examined the influence of each boost on the model’s classification accuracy for each class and observed that each boost affected each class’s accuracy differently, suggesting that the model performance could be further improved by applying conditional data augmentation to each layer.

Although many studies attempted to propose methods based on deep CNNs to perform song detection and data augmentation by shifting the time block, the efficiency of increasing the samples by separating the complete original songs into smaller pieces trained by a shallow CNN has not yet been clarified on instrumental songs. In addition, it also starts from the fact that we request the name of the song, but we usually only have a small piece of the song. Therefore, our study aims to evaluate the potential benefit of such an approach in improving the instrumental song classification tasks and support seeking a song with only a tiny piece of the song.

3 Methods

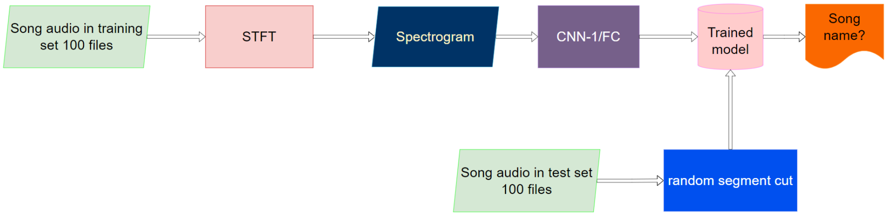

Song audio in training set 100 files
STFT
Spectrogram
CNN-1/FC
Trained model
Song name?
Song audio in test set 100 files
random segment cut

Fig. 1: Overall workflow for song classification with Spectrogram and machine learning algorithms.

In the study, the strategy proposes a method to classify songs based on specific characteristics that the model learns from the music data. Python and CNN programming languages are used to build the model. In the first process, supporting libraries extract audio data from the computer in WAV format. The dataset ***A*** includes 100 songs without lyrics, while the dataset ***B*** includes 100 audio files with the same song name but played with different types of musical instruments. Then, the pydub library [10] randomly cuts the original audio files into clips less than 10 seconds long because users often use a specific snippet to find the entire track. The original audio files are split into clips of different lengths in the training set, including 1s, 3s, 5s, 10s, 20s, 30s, 60s, and 90s. Next, Short-Time Fourier Transform (STFT) is used to extract the musical features from the audio signal, converting them into spectral images. Next, the CNN model is built after having the optical image feature data spectrum. This neural network architecture is specifically designed to work with 2D data. In addition, using the CNN-1/FC models helps us to learn features at local locations on the image, thereby generating higher-level features for image classification. The workflow is shown in Figure 1.

3.1 Data collection and division of songs

A technical method to split an audio file into small segments of different duration, including 1 second, 3 seconds, 5 seconds, 10 seconds, 20 seconds, 30 seconds, 60 seconds, and 90 seconds. We used the Python pydub library. The process starts with choosing the length of the sub-audio tracks, which can be 1 second, 3 seconds, 5 seconds, 10 seconds, 20 seconds, 30 seconds, 60 seconds, or 90 seconds. Then, we go through each part of the original audio file. For each part, we calculate the start and end times. Check if the end time exceeds the length of the audio file, and if so, restrict it from exceeding it. Then, we create a name for the new sub-audio file by combining the original filename with the part number.

We use the calculated start and end times to perform the audio clipping from the original file and convert the child audio track into a new WAV file. The code then continues by setting the output directory path and looping through the audio files in the input directory. For each audio file, the code defines the input audio file path and the target directory for the sub. If the destination directory does not already exist, it will be created. Finally, split the audio file into several small parts, with each part length, and save it in a folder. A technical method used to split up audio files randomly to custom lengths is to use the pydub library in Python. This process begins with the total time of the original audio file. Then, randomly select the start time to cut to generate random audio parts. Once the audio sections have been created, we create an output directory based on the name of the original audio file to store the split audio tracks. These audio tracks are exported in WAV format and are automatically saved to the output folder. The result is a directory of custom-length random audio tracks from the original audio file.

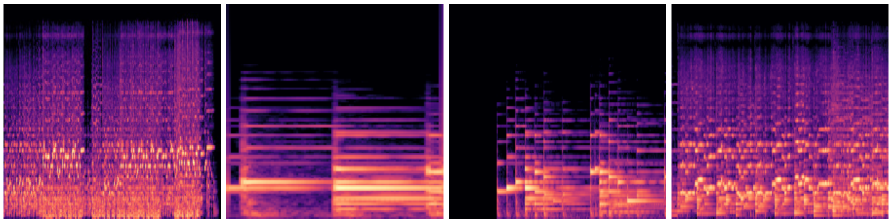
In Table 1, dataset ***A*** includes 100 songs without lyrics, while dataset ***B*** includes 100 audio files with the exact song title but played with different instruments. This brings variety and richness to our data sets to study and analyze the influence of instruments and duration distribution on music. AXstest and BXstest include samples where each sample is a piece with a length of X seconds randomly extracted from a song in datasets A and B, respectively. At the same time, AYstrain consists of sub-sequences of audio files with the length of Y seconds extracted from A. As shown in Table I, for example, A03strain contains audio segments cut from dataset A, having the same length of three seconds. Similarly, A01strain, A05strain, A10strain, A30strain, A60strain, and A90strain include samples with lengths of 1, 5, 10, 30, 60, and 90 seconds, respectively.

Table 1: Information on datasets used in the experiments. AXstest, BXstest includes samples where each sample is a piece with a length of X seconds randomly extracted from a song in dataset A, B, respectively, while AYstrain consists of sub-sequences of audio files with the length of Y seconds extracted from A.

Datasets	**The number of samples**
A	100
B	100
A10stest	100
B10stest	100
B05stest	100
A01strain	26122
A03strain	8738
A05strain	5265
A10strain	2660
A20strain	1353
A30strain	919
A60strain	481
A90strain	346

3.2 Transforming audio signal to image with Spectrogram

In our research on spectrogram finding and classifying instrumental music based on the method of the STFT, an essential technique in signal and audio processing. The audio signal from the files is divided into small frames with a fixed length. This division helps determine the frequency variation over time in each moment. Apply the Discrete Fourier Transform (DFT) to the audio signal in each frame. DFT decomposes the signal into individual frequency components and displays the magnitude of each frequency. The result of the DFT in each frame is a sequence of frequency-intensity values. The result is a 2D histogram of the different frequency bands in each timeframe. Spectroscopy allows us to observe changes in the frequency of sounds over time and detect musical characteristics such as tone, instrumentation, and noise changes. In the image, the time axis represents the track's duration, and the frequency axis represents the range of audio frequencies. Colors and variations in the spectral image represent the level and distribution of audio frequencies at a particular time. This is a unique way of expressing music information with images, helping us capture and analyze the characteristics and sound elements of the song. Fig 2 is an illustration of the spectrum of the original song "Ai chung tinh duoc mai" and the individual audio cuts (a complete song) from the dataset into one-second, 10-second (and 30-second.



(a) The original. (b) one seconds (c) 10 seconds (d) 30 seconds

Fig. 2: Some illustrations of spectral images with various lengths cut from a song.

3.3 The networks for song recognition

We selected the architecture illustrated in Figure 3; we called the network CNN-1 (with one convolutional layer) for short. A shallow convolutional network architecture can work well on Synthetic Image Representations as revealed in [11] with the input size of 32×32. The images generated by Spectrogram in this study also have the size of 32×32, so we use a similar architecture with [11] to perform the song classification tasks. Figure 3 exhibits that the network receives three-channel color images (audio file visualized by Spectrogram) as the input. This image is passed through a series of convolutional layers with 64 kernels of size 3×3 (with the stride 1), then passed through a maximum 2×2 composite layer (stretch 2). The ReLU activation function is applied after each convolution layer to produce the output. Next, a maximum composite layer (MaxPooling2D) is added to the model. This layer performs up to 2×2 summation on the output of the previous convolutional layer. It helps reduce the output size and retain the essential features. The next layer is the flat layer (Flatten), which transforms the output from the previous layers into a 1D vector. Finally, a fully connected (Dense) class is added to the model. This class has units=NUM-CLASSES, where NUM-CLASSES is 100 songs. The activation function used is 'softmax,' an activation function commonly used in classification problems to calculate the probability of each class given the input data. See the architecture of CNN-1 illustrated for the 32×32 image in Figure 3 for more details. Besides CNN-1, we have also implemented a Fully Connected neural network (FC) that received the images and performed the linear regression to provide 100 outputs corresponding to 100 songs.

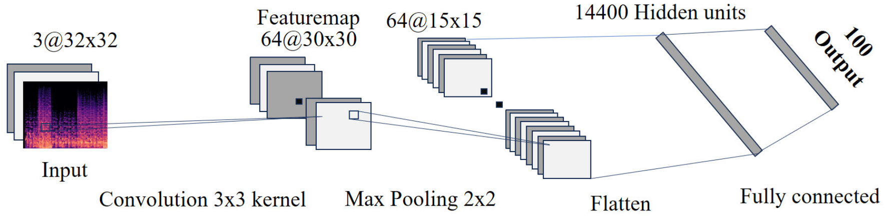


Fig. 3: The CNN-1 architecture includes a stack of one convolutional layer with 64 filters of 3x3 and a max pooling of 2x2, followed by one fully connected layer.

4 Experimental results

4.1 Experimental Setup

All networks have used Adam for the optimizer, a batch size of 64, a learning rate of 0.0001, running to 500 epochs. CNN-1 has 1,441,892 Parameters, while FC owns 307,300. Several libraries are also implemented, such as annoy (version 1.17.0), librosa (version 0.9.2), python-speech-features (version 0.6), numpy (version 1.21.5)), jar (version 1.1.2), tensorflow (version 2.8)), and matplotlib (version 3.5.1).

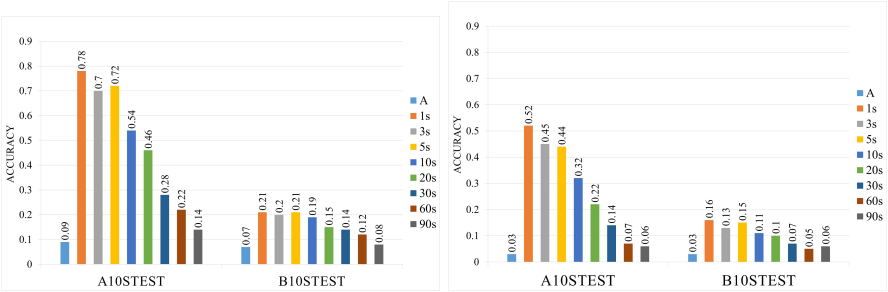
4.2 The length of pieces extracted from the song can affect the song detection performance

AXstest, BXstest includes samples where each sample is a piece with a length of X seconds randomly extracted from a song in dataset A, B, respectively. Our idea was to test how many seconds of audio would best fit into the recognition model and be able to detect the song most accurately. In addition, we applied two methods, CNN-1 and FC models, to this dataset to compare the performance between the two network architectures in the task of instrumental music classification based on audio data. Furthermore, we highlighted the impressive results on the datasets to facilitate precise observation and comparison of the song classification performance between the CNN-1 and FC models based on Table 2.

Table 2: Efficiency of separation into small pieces in song classification tasks.

	Models	**CNN-1**	**FC**	**Models**	**CNN-1**	**FC**	**Models**	**CNN-1**	**FC**
Roles	**Datasets**	**Accuracy**	**Accuracy**	**Datasets**	**Accuracy**	**Accuracy**	**Datasets**	**Accuracy**	**Accuracy**
Training set	A	1	1	A	1	0.9900	A	1	0.9400
Test set	A10stest	0.0900	0.0300	B10stest	0.0700	0.0300	B05stest	0.0500	0.0300
Training set	A01strain	**0.8318**	**0.6670**	A01strain	**0.9934**	**0.7598**	A01strain	**0.6236**	0.6805
Test set	A10stest	**0.7800**	**0.5200**	B10stest	**0.2100**	**0.1600**	B05stest	**0.2000**	0.1700
Training set	A03strain	0.9489	0.7906	A03strain	0.9205	0.7906	A03strain	0.9205	0.7906
Test set	A10stest	0.7000	0.4500	B10stest	0.2000	0.1300	B05stest	0.1800	0.1500
Training set	A05strain	0.9823	0.7504	A05strain	0.9853	0.7504	A05strain	0.9781	**0.7813**
Test set	A10stest	0.7200	0.4400	B10stest	0.2100	0.1500	B05stest	0.2000	**0.2000**
Training set	A10strain	0.9987	0.8680	A10strain	0.9857	0.701	A10strain	0.9751	0.7650
Test set	A10stest	0.5400	0.3200	B10stest	0.1900	0.1100	B05stest	0.1600	0.1000
Training set	A20strain	0.8847	0.8861	A20strain	0.9970	0.8580	A20strain	0.9741	0.7450
Test set	A10stest	0.4600	0.2200	B10stest	0.1500	0.1000	B05stest	0.1500	0.0900
Training set	A30strain	0.9028	0.9270	A30strain	0.9488	0.9140	A30strain	0.9630	0.8650
Test set	A10stest	0.2800	0.1400	B10stest	0.1400	0.0700	B05stest	0.1100	0.0700
Training set	A60strain	0.9563	0.5571	A60strain	0.9708	0.4636	A60strain	1	0.5051
Test set	A10stest	0.2200	0.0700	B10stest	0.1200	0.0500	B05stest	0.1200	0.0600
Training set	A90strain	0.9826	0.6184	A90strain	1	0.5028	A90strain	0.9971	0.9219
Test set	A10stest	0.1400	0.0600	B10stest	0.0800	0.0600	B05stest	0.0800	0.0600

Figure 4 presents a performance comparison between the two network structures CNN-1 and FC in classifying instrumental music in Tables 2 and Table 2. Audio data is presented as cluster cells. The variety of test suites has yielded outstanding results for comparison and analysis. Both CNN-1 and FC models show high performance on data set with 1s music,*A10stest* FC reaches 52%, CNN-1 model gives better results at 78%,*B10stest* FC achieves 16%, CNN-1 model gives better results at 21%. This impression is essential in realistic sound classification simulations since, in practice, listeners can start listening to any part of the track at random. Besides, the CNN-1 model still shows superiority over FC; CNN-1's ability to generalize and process complex data makes it a robust instrumental music classifier.



(a) CNN-1

(b) FC

Fig. 4: Comparison of FC and CNN-1 model performance on A10stest and B10stest datasets.

4.3 Data augmentation on songs

Table 3: Song classification performance using and without data augmentation.

				CNN-1		**FC**	
Roles	**Datasets**	**Increased samples**	**Total**	**Accuracy**	**Time(seconds)**	**Accuracy**	**Time(seconds)**
Without data augmentation							
Training set	A		100	1	14.1267	1	10.8605
Test set	A10stest		100	0.0900		0.0300	
Using data augmentation							
Training set	A + A01strain	26122	26222	**0.8312**	10091.3252	**0.6822**	360.4965
Test set	A10stest		100	**0.8000**		**0.5200**	
Training set	A + A03strain	8738	8838	0.9729	1679.7152	0.8176	213.0935
Test set	A10stest		100	0.6900		0.4800	
Training set	A + A05strain	5265	5365	0.9852	1371.6816	0.7940	109.1950
Test set	A10stest		100	0.7300		0.5000	
Training set	A + A10strain	2660	2760	0.9880	830.8003	0.8000	42.6509
Test set	A10stest		100	0.5500		0.3000	
Training set	A + A20strain	1353	1453	0.9587	477.6910	0.8066	28.0938
Test set	A10stest		100	0.4700		0.2200	
Training set	A + A30strain	919	1019	0.9754	177.2855	0.7595	26.2689
Test set	A10stest		100	0.3300		0.1300	
Training set	A + A60strain	481	581	0.9552	125.3534	0.7796	14.2146
Test set	A10stest		100	0.2600		0.0800	
Training set	A + A90strain	346	446	0.9910	97.6455	0.9192	20.5288
Test set	A10stest		100	0.1700		0.0900	

Table 3 evaluates the performance of dataset A in both cases: one is when no data augmentation is applied, and the other is when we did not add more data for the training set. AXstest where each sample is a piece with a length of X seconds randomly extracted from a song in dataset A, respectively. During training, we built the training dataset by augmenting the ”A” set data with audio clips of different lengths: 1s, 3s, 5s, 10s, 20s seconds, 30 seconds, 60 seconds, and 90 seconds. We also performed modeling of both CNN-1 and FC networks on the advanced dataset to compare the performance between the two network architectures in the task of instrumental music classification based on audio data. For example, as in Table 3, *A03strain+A* contains audio-based substrings of the same length of 3 seconds, separate from the added *A* dataset, original dataset A. Similarly, *A01strain+A*, *A05strain+A*, *A10strain+A*, *A20strain+A*, *A30strain+A*, textitA60strain +A, *A90strain+A* consists of a data-enhanced original dataset A, samples of length 1, 5, 10,20, 30, 60, 90, sec, respectively. Also, in the Advanced Sample column, we calculate the percentage of sample gain and how much % in the tested dataset; we highlighted the impressive results on the datasets to facilitate precise observation and comparison of the song classification performance between the CNN-1 and FC models.

4.4 Classification algorithms comparison

In this study, we conduct a detailed analysis of the performance of two models, FC and CNN-1, on audio datasets for instrument classification based on acoustic data. More specifically, in Table 2, the results of song detection performance from an excerpt randomly cut from 100 songs in the original data set ”A” of a 10-second length FC model have an accuracy of only 52%. In contrast, the CNN-1 model has an accuracy of 78%. In addition, the song detection performance results from an excerpt randomly cut from 100 songs in the original data set ”B” with a length of 10 seconds. The CNN-1 model has an accuracy of 21%. , compared to the FC model, only 16%. In addition, the song detection performance results from an excerpt randomly cut from 100 songs in the original dataset ”B” with a length of 5 seconds. CNN-1 model achieved 20% accuracy when the FC model only reached 17%. In addition to the above test results, we performed performance evaluation on both cases: no data enhancement and data enhancement are applied as presented in Table 3. In addition, we model both CNN-1 and FC networks on the advanced dataset to compare the performance between the two network architectures in the task of Classification of Instruments based on acoustic data. The exciting result is that the FC model only gives 3% accuracy without data enhancement, while the CNN-1 model achieves 9%. However, when applying data enhancement, the performance of the FC model improves significantly to 52%, but the CNN-1 model still outperforms with an impressive accuracy of 80%. The research results clearly show the superiority of the CNN-1 model over the FC model in the classification task based on audio data. This is proven through various tests on different data sets.

5 Conclusion

We introduced an approach involving data fragmentation based on the power of convolutional neural networks on images to perform song recognition. The results showed that data augmentation by dividing the dataset into small chunks based on length significantly improved classification performance compared to not using this technique. This contributed significantly to the tool classification process and produced a significant increase in the performance of the classification model. In addition to the above studies, we have presented test results on both CNN-1 and FC models, providing a detailed view of the comparison between the two network architectures in the task of instrument classification based on audio data. We found that FC had a lower synthesis potential and resulted in worse performance than CNN-1 on the test set, which is an important finding. The scalability of the proposed approach is a significant concern. With the potential for the number of classes to change as new songs are added, the paper should address how the model can adapt to this scenario. Future work could explore solutions to this scalability challenge.

References

1. Goldmann, M., Kreitz, G.: Measurements on the spotify peer-assisted music-on-demand streaming system. In: 2011 IEEE International Conference on Peer-to-Peer Computing. pp. 206–211 (2011)
2. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)
3. Melchert, O., Roth, B., Morgner, U., Demircan, A.: Optfrog—analytic signal spectrograms with optimized time–frequency resolution. SoftwareX 10, 100275 (2019)
4. Nguyen, H.T., Vo, L.D., Tran, T.T.: Approximate nearest neighbour-based index tree: A case study for instrumental music search. Applied Computer Systems 28(1), 156–162 (Jun 2023), https://doi.org/10.2478/acss-2023-0015
5. Piczak, K.J.: Environmental sound classification with convolutional neural networks. In: 2015 IEEE 25th international workshop on machine learning for signal processing (MLSP). pp. 1–6. IEEE (2015)
6. Benetos, E., Dixon, S., Giannoulis, D., Kirchhoff, H., Klapuri, A.: Automatic music transcription: Breaking the glass ceiling (ISMIR 2012)
7. Ghosal, A., Chakraborty, R., Dhara, B.C., Saha, S.K.: Song/instrumental classification using spectrogram based contextual features. In: Proceedings of the CUBE International Information Technology Conference. pp. 21–25 (2012)
8. Park, T., Lee, T.: Musical instrument sound classification with deep convolutional neural network using feature fusion approach. arXiv preprint arXiv:1512.07370 (2015)
9. Salamon, J., Bello, J.P.: Deep convolutional neural networks and data augmentation for environmental sound classification. IEEE Signal processing letters 24(3), 279–283 (2017)
10. Boteju, W., Herath, H., Peiris, M., Wathsala, A., Samarasinghe, P., Weerasinghe, L.: Deep learning based dog behavioural monitoring system. In: 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS). pp. 82–87. IEEE (2020)
11. Nguyen, T.H., Prifti, E., Sokolovska, N., Zucker, J.D.: Disease prediction using synthetic image representations of metagenomic data and convolutional neural networks. In: 2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF). IEEE (Mar 2019), https://doi.org/10.1109/rivf.2019.8713670