Identification of South Asian Musical Instruments Using CNNs

Saad Qureshi
CS 2024

Faiz Haseeb
CS 2024

Aquib Ansari
CS 2024

***Abstract*–South Asia boasts a rich and diverse musical heritage with a wide array of instruments, each characterized by unique tonal qualities, shapes, and playing techniques. Instrument classification focuses on identifying the probability of a mixture of instruments being played in a piece of music. In this project, we are hoping to perform this task on a number of South Asian instruments, and provide the baseline for further work, such as development of software providing the real-time classification of instruments playing during a performance.**

I. Introduction

South Asian music holds immense cultural significance, reflecting the diverse heritage of the region. With roots deeply embedded in traditions that span across India, Pakistan, Bangladesh, Sri Lanka, and beyond, it serves as a vital aspect of cultural identity. The melodies, rhythms, and intricate compositions not only entertain but also convey stories, emotions, and spiritual depth. South Asian music, often associated with classical forms like Hindustani and Carnatic, has a unique ability to transcend language barriers, connecting listeners to the rich history and traditions of the subcontinent. Preserving and understanding this musical heritage is crucial for fostering cultural appreciation and global harmony.

Musical instrument classification is a field within audio signal processing and machine learning that involves the categorization of musical instruments based on various features such as sound characteristics, playing techniques, and tonal qualities. Musical instrument classification in general is a very important task for musical information retrieval systems, audio source separation, auto-matic music transcription, and genre classification.

In the context of our research, we are focused on the identification and classification of four famous South Asian instruments: Sitar, Bansuri, Harmonium, and Tabla. This process involves transforming audio samples into spectrograms, which are visual representations of the frequency content over time. By employing Convolutional Neural Networks (CNNs), a type of deep learning architecture well-suited for image and pattern recognition, we trained a model that accurately categorized spectrograms based on instruments present. The goal is to pave the way for a much more sophisticated system capable of providing real-time identification during musical performances. Our model and results are available at Code.

II. Research Question

“Taking an audio sample of any South Asian instrumental performance, classify the sample as a class of combinations of four instruments, Sitar, Bansuri, Harmonium, and Tabla.”

The input to our model is the spectrogram generated from an audio clip of a performance, essentially turning the entire task into image classification, rather than direct audio classification. We have a collection of over 8000 samples of performances from across the internet. The output to our model are the probabilities corresponding to all combinations of the instruments we have chosen shown below.

	Category
0	Sitar
1	Tabla
2	Harmonium
3	Bansuri
4	Sitar-Tabla
5	Sitar-Harmonium
6	Sitar-Bansuri
7	Tabla-Harmonium
8	Tabla-Bansuri
9	Harmonium-Bansuri
10	Sitar-Tabla-Harmonium
11	Sitar-Tabla-Bansuri
12	Sitar-Harmonium-Bansuri
13	Tabla-Harmonium-Bansuri
14	Sitar-Tabla-Harmonium-Bansuri

Table 1. Categories

Our key motivation is that in the realm of Raag performances, the intricate fusion of various South Asian musical instruments creates a rich tapestry of sound that can be very enchanting. However, for the untrained ear,

distinguishing between the diverse instruments employed during a performance can be a challenging endeavor. The motivation behind our initiative lies in addressing this very challenge.

Picture a scenario where the resonant notes of the sitar seamlessly intertwine with the soul-stirring tones of the tabla, and the ethereal sounds of the harmonium join in harmony. To the seasoned connoisseur, this melodic blend is a captivating symphony, but to those less familiar with South Asian instruments such as Western musicians, it can be a delightful yet perplexing auditory experience making it difficult for cross cultural performances.

As a start we have developed a model that will identify the instruments from a spectrogram of an audio clip and hope that this research will be carried forward and used to develop tools that can identify instruments in real time during performances and give instant feedback to the performers. It's a small step now, but we hope it'll lead to a big leap in making music experiences more dynamic and enjoyable for everyone and opening doors to more cross cultural performances with South Asian music.

III. Literature Review.

There may not be much research conducted on leveraging neural networks for identification of instruments in South Asian music. However, the field of Western instrument identification and classification has been thoroughly investigated, revealing a prevailing trend in utilizing image classification methodologies on segmented audio spectrograms which are visual representation of the audio signal's frequency content over time, serving as a pivotal input for the models. Furthermore, we have also looked at papers leveraging CNNs for image classification in general.

The ImageNet Classification with Deep Convolutional Neural Networks, a seminal paper by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton in 2012 [8], played a pivotal role in reshaping the landscape of deep learning with the introduction of the AlexNet architecture. This influential work showcased the remarkable capabilities of Convolutional Neural Networks (CNNs) in image classification tasks, significantly reducing the top-5 error rate on the ImageNet dataset. This meant that the model's predictions included the correct class within the top 5 predictions 83.6% of the time on a large-scale dataset with over 1.2 million images distributed across 1,000 object classes. While the ImageNet paper primarily focused on generic image classification, its architectural innovations and training strategies have since served as a foundational framework for various applications of CNNs, including the classification of spectrograms in audio data. In this study, we draw inspiration from the principles established in the ImageNet paper, adapting its groundbreaking concepts to the unique task of identifying classical South Asian musical instruments through spectrogram-based classification. This adaptation highlights the enduring impact of ImageNet on diverse domains beyond its original image classification context.

[3] In a study conducted by Haidar Ahmad, Convolutional Neural Networks (CNNs) were employed for the classification of audio streams from three distinct instruments - Piano, Drums, and Flute. The objective was to categorize these audio streams into three classes of the instruments above and a separate class labeled other instruments that were dominant. The dataset utilized in this research was Google's AudioSet, which encompasses human-labeled data derived from YouTube clips, presenting a diverse range of audio streams with various background noises and multiple instruments. Due to resource constraints, the study focused on a subset comprising 9,600 samples. To prepare the data for modeling, several preprocessing steps were employed, including downmixing to a single channel, normalization, standardization, and the generation of mel-spectrograms. Overfitting concerns were addressed through data augmentation and the addition of random white noise.

Fig 1. Pre-Processing Pipeline

Through the input spectrograms, the task of music identification becomes an image classification task where the spectral image is passed through the model and analyzed. The final CNN model consisted of seven layers with increasing filter counts. Parameter initialization was performed using He initialization, and the cross-entropy loss function was found to be effective for multi-class classification. During experimentation, the training phase involved 8,000 samples, utilizing a batch size of 128, 15 epochs, and a learning rate of 10e-3. The early stopping strategy was implemented to prevent overfitting. The CNN model demonstrated notable performance metrics, achieving

a precision of 70%, a recall of 65%, and an F1-score of 64 The largest source of errors for this model lies in the class "Other", where certain audio samples contain background sounds that do not correspond to any labeled instrument. Additionally, there are instances where certain samples are mislabeled or resemble the predicted class, further highlighting the complexity of collecting audio data.

A much more complex and more recent study was conducted by Kostek and Blaszke [1], which specifically employed sets of individual convolutional neural networks for each instrument. The dataset used in the experiment is the Slakh dataset, consisting of 2100 audio tracks with aligned MIDI files. Four instruments - bass, drums, guitar, and piano - were selected for the experiment. One noteworthy preprocessing step involves segmenting the mixed music into 4-second intervals which were then tagged according to the instruments present. Furthermore, If the level of instrument signal in the extracted part was lower than −60 dB, then this instrument was excluded from the example. This not only aids in computational efficiency but also provides a means to address the challenge of identifying complex audio signals. This entire process is known as Mel-frequency cepstral coefficients(MFCC). MFCCs are chosen for their effectiveness in identifying formants and timbre in sound. Distinguishing timbre is crucial in differentiating between instruments playing the same note with similar intensity.

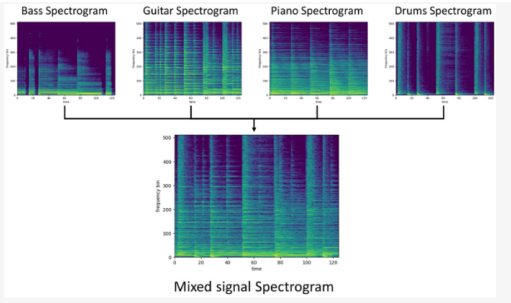


Fig 2. Mixed Signal Spectrogram

To ensure repeatability, the dataset was divided into three parts: training set (116,413 examples), validation set (5970 examples), and evaluation set (6983 examples). The class weighting vector was applied to balance the results between instruments.The proposed model architecture is distinctive in that it employs sets of individual submodels—one for each instrument. This enables flexibility in using models with varying architectural complexities for different instruments which stands out as a departure from the more conventional single-model paradigms than the one conducted by Ahmad.

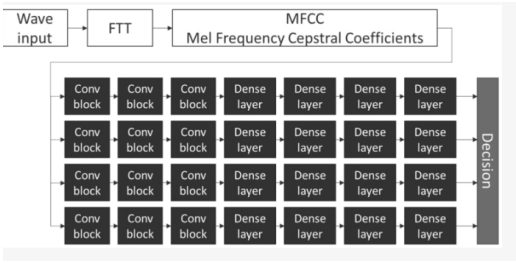


Fig 3. Model Architecture

The model initially generates MFCC from the raw audio signal, which is passed into a series of convolution layers, max pooling, and batch normalization operations. The model is designed to make instrument predictions based on the extracted features where each submodel's convolution block is tuned according to the instrument it predicts. Furthermore, all submodels are using ReLU activation since instrument identification is a highly non-linear problem due to the spectral contrast, Other blocks such as batch normalization and dense layers remain consistent across the models. Regarding scalability, this architecture allows for the addition of new instruments to a previously trained neural network. This is achieved by extending the set of submodels to accommodate the new instrument.

A different strategy was adopted by Mahanta, Khilji, and Pakray [2] which employed an artificial neural network (ANN) model trained on the London philharmonic orchestra dataset, focusing solely on mel-frequency cepstral coefficients (MFCCs) of the audio data. The dataset consists of twenty different classes of musical instruments categorized into four families: woodwinds, brass, percussion, and strings.The dataset is imbalanced with violin having 1502 examples and banjo only having 74. However, this model still achieves results without the need for data augmentation by stratified splitting with a ratio of 8:2 between training and testing sets. The proposed model architecture consists of multiple layers, including dense hidden layers with varying numbers of neurons, ReLU activation functions, and dropout layers for regularization. The output layer employs a Softmax activation function, which provides the confidence scores for each class. The Adam optimizer is used for training the model. It is an adaptive learning rate optimization algorithm that adjusts the learning rate during training. During the training process, the model achieves a peak training accuracy of 99.13% and a

validation accuracy of 97.26%. The high validation accuracy suggests that the model has learned to generalize well to unseen data. However, the author mentions it could not be concluded as the best metrics since the dataset is imbalanced.

The oldest paper we came across was, "Instrument classification using Hidden Markov Models" by Eichner, Wolff, and Hoffmann [9] which delved into musical instrument classification using traditional Hidden Markov Models (HMMs). The authors built a database comprising 600 samples, encompassing classical guitar, violin, trumpet, and clarinet recordings, with variations in instruments, rooms, solo pieces, players, and repetitive playings. The audio signals were processed using a 31-channel mel-scale filter bank, and statistical principal component analysis was applied to reduce the feature space to 25 dimensions. The study conducted two pivotal experiments: the first focused on recognizing individual notes across instruments, while the second extended this to identify specific instruments. Despite the relatively smaller dataset, this paper reported an accuracy rate of 76%, showcasing the greater efficiency of CNNs as compared to older AI techniques in image classification.

Majority of the papers we looked into used data sets of audio files containing multiple instruments playing in unison, with the instruments present in the clip being used to label the clip. Those audio clips were then being converted into spectrograms for feature extraction. We opted for a similar approach to our data sets. Below is a table indicating the dataset size, instruments and accuracy of all the papers we looked into starting from the most recent. Based on the accuracy and datasets, we came to the conclusion that CNNs were most suited to handle large image datasets for computer vision tasks and began working on our own model with an accuracy target of 90%.

Year	Instruments	Model	DataSet	Accuracy
[1] 2022	Bass, Guitar, Piano, Drums	Set of Individual CNNs	Slakh DataSet(2100 Audio Tracks,120,000 samples)	93%
[2] 2021	Basoon, Bass Clarinet, Banjo, Flute, Flute, French Horn, Cello, Clarinet, Contrabassoon, Etc	ANN	London Philharmonic Orchestradatase(13679)	97%
[3] 2020	Banjo, Cello, Violin, Tuba Etc	KNN	Undefined	98%
[4] 2018	Piano, Drums, Flute, Other	CNN	Google AudioSet(9600)	70%
[5] 2018	Cello, Piano, Trumpet, Flute, Violin	ANN	Electronic Music Studio Dataset(Undefined)	90%-93%
[6] 2015	Pianos, Woodwinds, Brass and String Instruments	CNNs	UIOWA MIS database(Undefined)	93%-99%
[7] 2015	Harmonium, Flute, Sitar	KNN	Undefined	99%
[8] 2012	None	CNN	ImageNet(1.2 Million Images)	83.60%
[9] 2012	Cello, Flute, Oboe, Trumpet, Viola	CNN & KNN	600 Samples	90.00%
[10] 2006	Guitar, Trumpet, Clarinet, Trumpet	Mixture of HMMs	600 Mixed Recordings	76%-78%

Table 2. Literature Review Summary

IV. Dataset.

1) Data Collection: In this section we will detail our methods of data gathering. Our only source of data was videos of instrumental performances from Youtube. Using online tools, we downloaded the videos and converted them to .Wav sound files, separating them into classes based on what instruments were present in the video. This required some manual effort, as finding videos of unaccompanied and accompanied performances without unnecessary background noise proved to be difficult. Moreover, downloading the video and then converting it to the .Wav file format proved to be tedious. We then partitioned the sound files in each class into 4-second clips. To accommodate some combinations of instruments, we merged audio files of single instruments. The partitioning and merging of sound files was accomplished using a program that we wrote ourselves that utilized the AudioSegment module from the Pydub Python library. Multi-instrument classes were not all populated using combinations of single instrument classes, but with a mixture of combinations and videos of accompanied (multi-instrumental) performances. We have accumulated approximately 6000 4-second audio files across the 15 different classes. With the class by class breakdown given below:

Category	Samples
Bansuri	1044
Harmonium	432
Harmonium-Bansuri	34
Sitar	721
Sitar-Bansuri	809
Sitar-Harmonium	270
Sitar-Harmonium-Bansuri	329
Sitar-Tabla	739
Sitar-Tabla-Bansuri	224
Sitar-Tabla-Harmonium	118
Sitar-Tabla-Harmonium-Bansuri	64
Tabla	899
Tabla-Bansuri	202
Tabla-Harmonium	111
Tabla-Harmonium-Bansuri	37

Table 3. Per Category Samples

Our data can be found in the following google drive.

2) Data Preprocessing: Before the model was ready to train, there were some preprocessing steps involved in order to change the input. Torchaudio, which is a library for audio and signal processing with PyTorch, was used for preprocessing. Firstly, the audio files were resampled, in which the sampling rate of the sound file was changed, which is the number of audio samples carried in a second, measured in Hertz (Hz). The audio files were upsampled, in order to get a higher quality for more preprocessing. After that, the channel of the audio file was checked to see if they were stereo or mono, and converted to stereo. Stereo is when the audio file has sound separated to the right and to the left, as it is in headphones, whereas mono is when the sound is coming from one direction. Thirdly, the rechanneled and resampled audio was padded to a standard length of 6 seconds. After these steps, the files were ready to be converted into their mel-spectrograms. This was done using the transforms module of the Torchaudio library, which essentially computes the Short-time fourier transform (STFT) of the audio signal to convert it into the time-frequency domain, and then passing the STFT through mel filter banks, which are filters that approximate how humans perceive pitch. The resulting mel-spectrogram is then converted to decibels and then visualized on a time-frequency graph, with color intensity representing the intensity of the sound in decibels.

3) Spectrogram Analysis: Below is the comparison of the mel-spectrograms of the four instruments in our dataset.

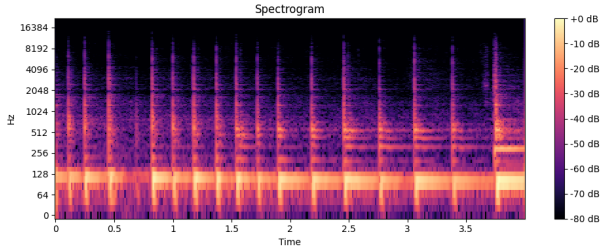


Fig 4. Tabla Mel-Spectrogram

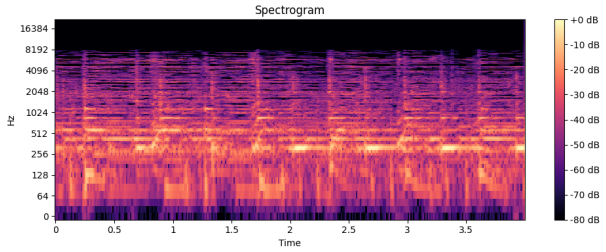


Fig 5. Sitar Mel-Spectrogram

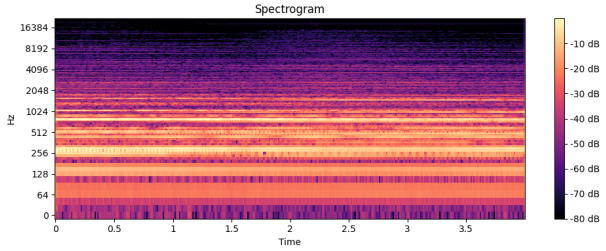


Fig 6. Harmonium Mel-Spectrogram

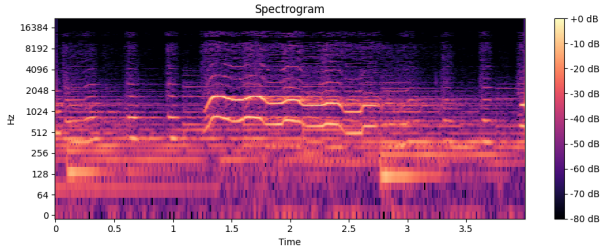


Fig 7. Bansuri Mel-Spectrogram

As we can see from the mel-spectrograms of the instruments, each has their own characteristics which are indicative of the features of the instrument. The tabla's spectrogram, for one, shows the low end frequency range of the drum, while the sharp lines of high intensity are indicative of the way the tabla sounds and is played, short sharp beats of the drum.

The sitar's spectrogram is much different, with it occupying a broad frequency range due to its larger range of notes over multiple octaves, with the main strings occupying the middle of the range frequencies, coupled with the sympathetic strings, which are in the higher frequencies. The intensity pattern of the main strings shows how it is played, the main string are plucked and can be sustained, bent or vibrated, before moving on to the next note, while the sympathetic strings are strummed alongside the plucking, which is most likely why the sharp lines in the higher frequencies are present.

The harmonium's spectrogram is interesting, showing sustained intensities in the mid to low range of frequencies, displaying the quality of the sound produced by the instrument. long unwavering notes that cannot be manipulated in any way, such as how notes can be bended with the sitar, as well as combinations of notes to make chords can all be identified from the spectrogram.

Finally, the bansuri's spectrogram is also indicative of the quality of the sound of the instrument. Depending on the size of the bansuri, with the one represented by the spectrogram being a larger bansuri, it can occupy anywhere between the high to mid range of frequencies. The intensity patterns also show what kind of sound the instrument produces. Quieter, quivering single notes are shown on the spectrogram.

V. CNN Model

Due to limited time, we acquired a CNN from Kaggle that was used for classification of string instruments and trained it on our dataset. The CNN model has been implemented using Keras API provided by Tensorflow framework. It has a 3x3 convolutional layer followed by three 2x2 convolutional layers all with 2x2 max pooling. The number of filters were 32, 64, 128, and 256 with ReLu activation. The model uses sigmoid as the decision activation function in order to assign probabilities to the 15 categories. The total trainable parameters are 8,565,807.

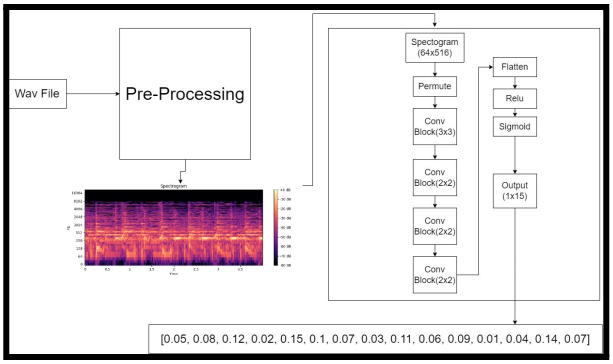


Fig 8. Model Pipeline

The code for this model implemented using keras is shown below.

```
model1=Sequential()
model1.add(layers.InputLayer(input_shape=(2, 64, 516)))
model1.add(layers.Permute((2, 3, 1)))
model1.add(Conv2D(32,(3,3), activation="relu",padding='same'))
model1.add(MaxPooling2D(pool_size =(2,2)))
model1.add(Conv2D(64,(2,2),activation="relu",padding='same'))
model1.add(MaxPooling2D(pool_size =(2,2)))
model1.add(Conv2D(128,(2,2),activation="relu",padding='same'))
model1.add(MaxPooling2D(pool_size =(2,2)))
model1.add(Conv2D(256,(2,2),activation="relu",padding='same'))
model1.add(MaxPooling2D(pool_size =(2,2)))
model1.add(Flatten())
model1.add(Dense(256,activation="relu"))
# Change Dense Layer Input to number of classes.
model1.add(Dense(15,activation='softmax'))

model1.summary()

model1.compile(optimizer = keras.optimizers.Adam(learning_rate = 0.001),
               loss = "binary_crossentropy", metrics =["accuracy"])
```

Fig 9. Model Code

VI. Experiments

The training of the model was performed on Google Colab using a T4 GPU. After shuffling and splitting our 6000 samples into 80% train and 20% test our model was trained for 10 epochs with a batch size of 32 and a learning rate of 0.001 with binary cross entropy as the loss measure. We arrived at a learning rate of 0.001 as greater learning rates had lesser accuracies and smaller learning rates would have taken greater time.

Parameters	Value
Epochs	10
Learning Rate	0.001
Batch Size	32

Table 4. Model Parameters

Learning Rate	Precision
0.001	0.94
0.01	0.91
0.1	0.03

Table 5. Learning Rates and Precisions

Our primary metric for analysis and comparison with previous works is the accuracy of the model. Each episode took an average of 5 seconds to run with the first episode being an outlier of 20 seconds. In the first episode we started with an accuracy of 0.4 and reached an accuracy of 0.98 in the final episode. Evaluating the model on the test data we achieved a precision of 0.939 with a recall of 0.94 and F1 score of 0.939. Based on the curve below for our model we can see there was little fluctuation in training on the dataset.

Category	Precision	Recall	f1-score	support
Sitar	0.91	0.97	0.94	123
Tabla	1	1	1	164
Harmonium	0.98	0.97	0.97	95
Bansuri	0.96	0.95	0.96	217
Sitar-Tabla	0.93	0.98	0.95	161
Sitar-Harmonium	0.67	0.69	0.68	42
Sitar-Bansuri	0.93	0.92	0.92	165
Tabla-Harmonium	1	0.96	0.98	23
Tabla-Bansuri	1	0.87	0.93	38
Harmonium-Bansuri	1	1	1	12
Sitar-Tabla-Harmonium	0.7	0.53	0.6	30
Sitar-Tabla-Bansuri	0.93	0.98	0.95	52
Sitar-Harmonium-Bansuri	0.98	1	0.99	61
Tabla-Harmonium-Bansuri	1	0.88	0.93	8
Sitar-Tabla-Harmonium-Bansuri	0.93	0.87	0.9	15
accuracy			0.94	1207
macro avg	0.93	0.9	0.91	1207
weighted avg	0.94	0.94	0.94	1207

Table 6. Classification Report

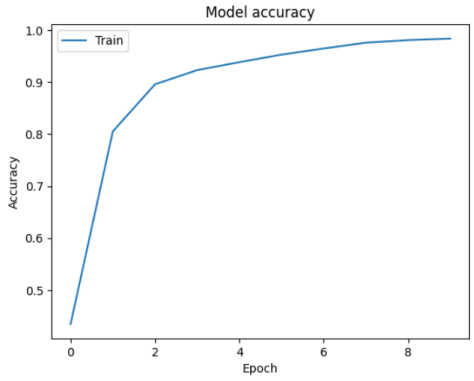


Fig 10. Accuracy Graph

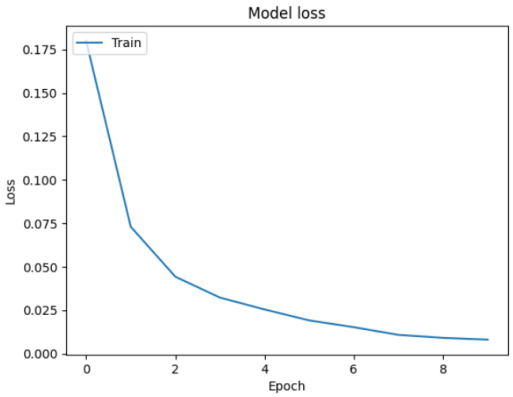


Fig 11. Loss Graph

In order to analyze which categories our model struggled the most with classifying we generated a confusion matrix. Out of all the labels category 5 and category 10 had the highest proportion of mislabels with the category test size. For category 5 the test samples were 43 out of which only 29 were labeled correctly. For category 10 the test samples were 23 out of which 16 were labeled correctly.

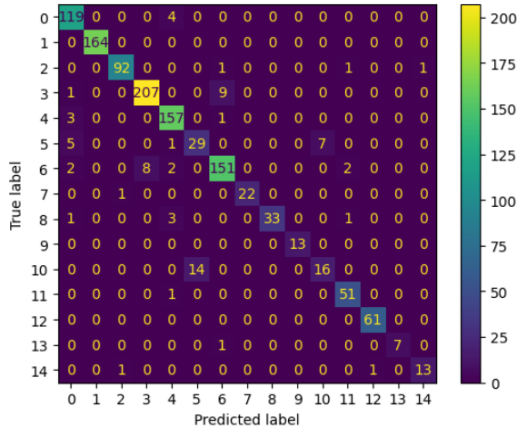
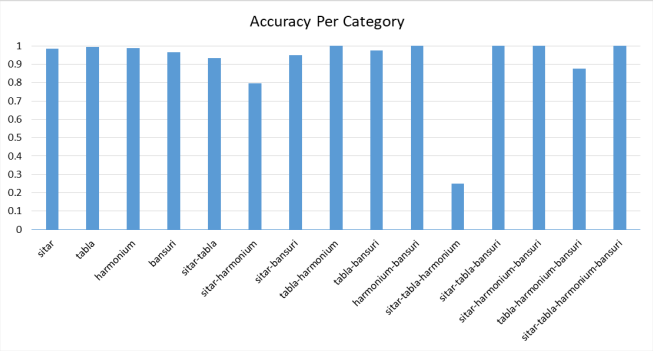


Fig 12. Confusion Matrix

Based on the findings from the confusion matrix, we measured the average accuracy for each category from 5 evaluations in order to see the improvements needed in our dataset. We came to see that out of all combinations, sitar-tabla-harmonium(10) and sitar-harmonium(5) had the least accuracy of 0.25 and 0.79 respectively. When we analyzed the outputs for the categories we saw that in most cases they were being mistaken for each other by the model hinting at an absence of spectrograms with tabla frequencies in sitar-tabla-harmonium causing our neural network to think one as the other leaving room for further data collection in these categories.



Histogram 1. Per Category Accuracy

```
Category: sitar-tabla-harmonium
Data Shape - Train: (94, 2, 64, 516), Test: (24, 2, 64, 516)
1/1 [==============================] - 0s 23ms/step
[ 5 10 10  5 10  5  5  5  5  5  5  5 10 10  5  5  5  5 10  5  5  5  5]
Category: sitar-tabla-harmonium, Accuracy: 0.25
```

Fig 13. Sitar-Tabla-Harmonium Test Output Classes

```
Category: sitar-harmonium
Data Shape - Train: (216, 2, 64, 516), Test: (54, 2, 64, 516)
2/2 [==============================] - 0s 12ms/step
[ 5  5  5  5  5  5  5  5  5  5  5 10  0  5  5  5 10  5 10  5  5  5 10  0
  5  5  5  5  5  5  5  5 10  0  5  5  5  5  5 10  5  5  5  5  5  0  6
  5  5  5  5  5  5]
Category: sitar-harmonium, Accuracy: 0.7962962962962963
```

Fig 14. Sitar-Harmonium Test Output Classes

VII. Results and Discussion

After training on CNN, we were able to acquire a model for one of the papers by Mohandas and Dhivya [9] in our literature review that trained a K-nearest neighbor classifier on 600 samples with 6 classes reaching an accuracy of over 90%. We trained over 6000 samples on this classifier however, our accuracy was far lesser coming in at 45%. This could be as a result of differences in pre-processing and or the K-neighbors model not being suited for 15 categories with over 6000 samples.

	Precision	Recall	f1-score	support
Sitar	0.64	0.64	0.64	260
Tabla	0.1	0.08	0.09	108
Harmonium	0.19	0.23	0.21	116
Bansuri	0.3	0.34	0.32	180
Sitar-Tabla	0.52	0.51	0.52	202
Sitar-Hamonium	0.16	0.15	0.16	176
Sitar-Bansuri	0.96	1	0.98	82
Tabla-Harmonium	0.98	0.99	0.99	184
Tabla-Bansuri	0.98	1	0.99	56
Harmonium-Bansuri	0.07	0.07	0.07	138
Sitar-Tabla-Harmonium	0.22	0.19	0.2	124
Sitar-Tabla-Bansuri	0.46	0.43	0.44	225
Sitar-Harmonium-Bansuri	1	0.96	0.98	50
Tabla-Harmonium-Bansuri	0.18	0.19	0.19	135
Sitar-Tabla-Harmonium-Bansuri	0.08	0.09	0.08	117
Accuracy			0.43	2153
Macro Average	0.46	0.46	0.46	2153
Weighted Average	0.43	0.43	0.43	2153

Table 7. Mohandas and Dhivya KNN Evaluation Classification Report

To confirm our suspicions we used the second CNN model from the same paper and replaced it with our and measured the accuracy. In the interest of time, we did not apply the literature review's pre-processing steps and directly modified the input for its CNN. Training was done for 10 episodes where each episode took an average of 10 seconds. We started with an accuracy of 0.23 and approached 0.9 in the final episode. Further evaluating the model on the test set we received a favorable precision of 0.89, f1 score of 0.89 and recall of 0.9.

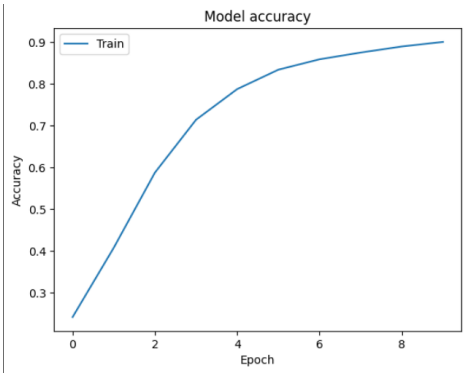


Fig 15. Mohandas and Dhivya CNN Training Accuracy Graph

Category	Precision	Recall	f1-score	support
Sitar	0.9	0.95	0.92	123
Tabla	1	0.99	1	164
Harmonium	0.94	0.89	0.92	95
Bansuri	0.83	1	0.91	217
Sitar-Tabla	0.93	0.98	0.96	161
Sitar-Harmonium	0.55	0.86	0.67	42
Sitar-Bansuri	0.97	0.7	0.81	165
Tabla-Harmonium	0.95	0.91	0.93	23
Tabla-Bansuri	0.97	0.97	0.97	38
Harmonium-Bansuri	0.9	0.69	0.78	12
Sitar-Tabla-Harmonium	0	0	0	30
Sitar-Tabla-Bansuri	1	0.98	0.99	52
Sitar-Harmonium-Bansuri	1	1	1	61
Tabla-Harmonium-Bansuri	1	1	1	8
Sitar-Tabla-Harmonium-Bansuri	0.65	0.87	0.74	15
accuracy			0.9	1207
macro avg	0.84	0.85	0.84	1207
weighted avg	0.89	0.9	0.89	1207

Table 8. Mohandas and Dhivya CNN Evaluation Classification Report

Based on the report from our literature review's CNN model and much more favorable results from our first model, it is clear that CNNs are better suited for the task of musical instruments classification due to their effectiveness for tasks involving image and pattern recognition. KNNs, on the other hand, rely on explicit feature vectors provided by the user. They don't inherently learn features from the data, and their performance can be sensitive to the choice of features.

One of our major motivators for using a CNN model was the research conducted by Blaskze and Kostek who leveraged not one but four CNNs per category on a dataset of over 120,000 samples. They had an accuracy of 0.93 with the lowest category accuracy being 0.83 leaving more room for improvement in our research with better data collection since our lowest category accuracy is 0.25. However, they are employing individual CNNs for mixed spectrograms compared to our technique of using a single CNN on separate mixed instruments categories. Our paper also has the more distinct instrument classes, not previously collected on such a scale.

Table 2. Results per instrument.

Metric	Bass	Drums	Guitar	Piano
Precision	0.94	0.99	0.82	0.87
Recall	0.94	0.99	0.82	0.91
F1 score	0.95	0.99	0.82	0.89
True positive	5139	6126	2683	3811
True negative	1072	578	2921	2039
False positive	288	38	597	589
False negative	301	58	599	361

Table 9. Blaskze and Kostek Classification Report

VIII. Conclusion

Our adaptable framework for instrument identification stands out for its capacity to seamlessly handle large datasets, offering ample opportunities for expanding sample collections with a broader range of diverse instrument categories including sarangi, shenai, and veena. This not only enhances the precision of our model but also positions it for integration into software capable of delivering real-time instrument classification—an integral aspect of our project's primary goals. With the advent of more advanced GPUs, our model can be dynamically trained, showcasing the potential for continuous improvement. Additionally, this paper marks a pioneering effort by being among the first to curate thousands of samples from various online sources, showcasing the rich musical tapestry of our beloved subcontinent.

References

[1] Blaszke, M., & Kostek, B. (2022, April 15). "Musical Instrument Identification Using Deep Learning Approach." MDPI. https://www.mdpi.com/1424-8220/22/8/3033

[2] Mahanta, Pakray, Khilji, et al. (2021, October). "Deep Neural Network for Musical Instrument Recognition Using MFCCs." ResearchGate. https://www.researchgate.net/publication/351347578_Deep_Neural_Network_for_Musical_Instrument_Recognition_Using_MFCCs

[3] Prabavathy, S., Rathikarani, V., & Dhanalakshmi, P. (2020). Classification of Musical Instruments using SVM and KNN. International Journal of Innovative Technology and Exploring Engineering (IJITEE) https://www.ijitee.org/wp-content/uploads/papers/v9i7/G5836059720.pdf

[4] Haidar-Ahmad, L. (2018). "Music and Instrument Classification Using Deep Learning." CS230 Deep Learning. Stanford University. https://cs230.stanford.edu/projects_fall_2019/reports/26225883.pdf

[5] Chakraborty, S., & Parekh, R. (2018). "Improved Musical Instrument Classification using Cepstral Coefficients and Neural Networks." ResearchGate. https://www.researchgate.net/publication/327806078_Improved_Musical_Instrument_Classification_Using_Cepstral_Coefficients_and_Neural_Networks

[6] Park, T, and Lee, T. (2015). "Musical Instrument Sound Classification with Deep Convolutional Neural Network Using Feature Fusion Approach." https://arxiv.org/ftp/arxiv/papers/1512/1512.07370.pdf

[7] Joshi, S., & Chitre, A. (2015). Identification of Indian Musical Instruments by Feature Analysis with Different Classifiers https://dl.acm.org/doi/10.1145/2818567.2818588

[8] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). "ImageNet classification with deep convolutional neural networks." In Advances in Neural Information Processing Systems (Vol. 25). https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[9] Dhivya, S., & Mohandas, P. (2012). Comparison of Convolutional Neural Networks and K-Nearest Neighbors for Music Instrument Recognition https://link.springer.com/chapter/10.1007/978-3-031-18444-4_9

[10] Eichner, M., Wolff, M., & Hoffmann, R. (2006). "Instrument classification using hidden Markov models." ISMIR. https://archives.ismir.net/ismir2006/paper/000112.pdf