Musical Instrument Recognition in Polyphonic Audio through Convolutional Neural Network and Spectrograms

Rujia Chen, Akbar Ghobakhlou, Ajit Narayanan

***Abstract*—**This study investigates the task of identifying musical instruments in polyphonic compositions using Convolutional Neural Networks (CNNs) from spectrogram inputs, focusing on binary classification. The model showed promising results, with an accuracy of 97% on solo instrument recognition. When applied to polyphonic combinations of 1 to 10 instruments, the overall accuracy was 64%, reflecting the increasing challenge with larger ensembles. These findings contribute to the field of Music Information Retrieval (MIR) by highlighting the potential and limitations of current approaches in handling complex musical arrangements. Future work aims to include a broader range of musical sounds, including electronic and synthetic sounds, to improve the model's robustness and applicability in real-time MIR systems.

***Keywords*—** Binary Classifier, CNN, Spectrogram, Instrument

I. Introduction

The task of recognizing multiple musical instruments in polyphonic compositions introduces a significant challenge within the realm of Music Information Retrieval (MIR). Conventional multi-class classification approaches, where one classifier attempts to distinguish many instruments in the same piece of music, become untenable due to the potentially unlimited combinations of instruments playing simultaneously. In the worst case, the classifier may need to be trained to recognize all probable instruments, or a new instrument may require the entire classifier to be retrained. This research uses a different solution, employing a One-vs-All (OvA) binary classification approach involving Convolutional Neural Networks (CNNs) and spectrogram input. OvA requires one classifier trained to recognize a specific instrument in polyphonic music (i.e., where multiple instruments occur). Instrument recognition is performed through an ensemble of classifiers identifying whether their instrument is playing or not in a section of music. This methodological innovation allows for the practical and scalable identification of individual instruments within complex audio signals. By synthesizing polyphonic audio samples and adapting CNNs for OvA classification, this study addresses the inherent challenge of distinguishing between instruments playing together. Moreover, the OvA approach adopted here uses only spectrograms of musical signals, marking a significant advancement in MIR over other approaches that adopt feature extraction or feature selection techniques prior to training and classification.

II. Literature Review

Musical instrument recognition in humans and machines has been examined for decades. For instance, Patil et al. delve into the nuances of timbre perception, investigating how the auditory system discerns the distinctive signatures of instrument sounds, emphasizing the intricacies involved in auditory processing [1]. Chi et al. contribute to this understanding by proposing a multiresolution spectrotemporal analysis that broadens the comprehension of sound attributes, enhancing sound structure analysis [2]. Agus et al. focus on the rapid categorization of musical sounds through their timbre, exploring the cognitive speed in identifying the timbre of musical sounds [3]. Yang et al. direct their research towards the auditory representations of acoustic signals, studying the auditory system's ability to encode and interpret a wide array of acoustic information [4]. Kostek's work stands out in the classification of musical instruments and duet analysis, applying music information retrieval techniques to automate sound recognition and categorization [5]. Further research in the field examines the classification of musical instruments using spectral features, demonstrating the efficacy of machine learning methods like support vector machines and quadratic discriminant analysis based on a set of features such as inharmonicity and spectral centroid [6]. Krishna and Sreenivas take recognition a step further by extending it from isolated notes to solo phrases, employing pattern recognition to enhance instrument identification across complex musical fragments [7].

On the other hand, some recent studies focus more narrowly on deep learning, assessing, for instance, how well CNNs work for instrument classification using spectrogram-based analysis [8], [9], [10], [11], [12], [13] [14]. These foundational efforts primarily utilize spectrogram analysis for converting signals into images for CNN architectures. Many of these approaches focus on identifying one or two instruments. Identifying the full

Rujia Chen is with the Auckland University of Technology, Auckland, New Zealand (corresponding author, e-mail: rujia.chen@autuni.ac.nz).

Akbar Ghobakhlou is with the Auckland University of Technology, Auckland, New Zealand (e-mail: akbar.ghobakhlou@aut.ac.nz).

Ajit Narayanan is with the Auckland University of Technology, Auckland, New Zealand (e-mail: ajit.narayanan@aut.ac.nz).

combination of instruments, rather than just the predominant one, remains an area less explored.

The study [15] introduced a novel approach, using a dedicated classifier for each instrument. This method, however, mainly assessed the accuracy of individual instruments' presence, rather than detecting every single instrument within a piece.

Our methodology enhances this [15] model by utilizing a One-vs-All (OvA) strategy [16], [17], a technique for tackling multi-class classification [18], [19], [20] [21], to address the challenges of multilabel classification (Figure 1). This approach allows our model to effectively differentiate between multiple overlapping instruments in complex audio settings, filling a gap in the identification of all instruments within polyphonic compositions.

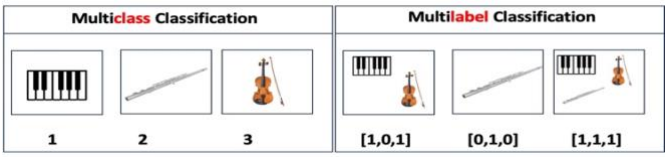

Fig. 1. Example of difference Between Multi-Class and Multilabel Classification, with output class values below each instrument.

According to Figure 1, our adoption of multilabel classification stems from the desire to identify all instruments in an audio sample, rather than just one. While the paper [8] may have honed in on recognizing individual instruments (single-label classification as indicated in the left part of Figure 1), our methodology aims to detect not just one output label, but also duos, trios, and full ensembles combinations (right part of Figure 1). This approach culminates in a model capable of recognizing, for instance, 16 distinct classes that cover all potential combinations of four instruments. Such a multilabel strategy, inspired by the One-vs-All technique and fundamentals outlined in [9] through [14], offers a more comprehensive analysis of polyphonic compositions.

III. Data Representation

A. Complexity of Polyphony

In real-world music, instruments often play simultaneously, creating polyphonic audio that traditional classification approaches struggle to accurately decode. That is why previous research usually only classifies the predominant instrument in a piece of polyphonic music.

To summarize the challenge of defining target classification classes for musical instrument recognition formulaically, let us consider how the number of potential classes grows with the number of instruments, n. This approach addresses the complexity of accounting for every possible combination of instruments playing together, from solo performances to full ensembles. Given n instruments, the number of possible combinations (including the scenario where no instrument is playing, represented as the "zero" case) can be calculated using the formula for the sum of combinations:

$$Total\ Combinations = 1 + \sum_{k=1}^{n} \binom{n}{k} \quad (1)$$

where $\binom{n}{k}$ is the binomial coefficient, representing the number of ways to choose k instruments out of n without regard to order. The "+1" accounts for the case where no instrument is playing. This formula can be simplified using the property of binomial coefficients, which states that the sum of combinations from $k = 0$ to n is 2^n:

$$Total\ Combinations = 2^n \quad (2)$$

Here, each "combination" represents a possible target classification class for the model, including:

- No instrument playing (2^0 = 1 combination),
- Solo performances for each instrument (2^1 = 2 combinations),
- Duo performances for each instrument (2^2 = 4 combinations). For example, the combinations are no instrument, the first instrument solo, the second instrument solo, first and section instrument duo performance.
- Decet performances for each instrument (2^{10} = 1024 combinations)
- And so on...

This exponential growth highlights the impracticality of designing a single multi-class classifier for polyphonic music recognition as n increases, underscoring the necessity and efficiency of employing a multiple binary classification strategy for each instrument individually where a classifier output '1' if its instrument is playing, '0' otherwise.

B. Binary Array Representation Methodology

The encoding approach taken to classify musical instruments from the NSynth dataset [22] (excluding the 'Synth Lead ' instrument due to the imbalance in its data distribution) involves converting the possible combinations of instruments into a binary array representation. Table I shows how the binary classification system can be summarized for 10 instruments using a 10-bit array.

TABLE I. Binary Encoding Schema for Instrument Presence

Type	Label
No Instrument	[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Bass Solo	[1, 0, 0, 0, 0, 0, 0, 0, 0, 0]
Brass Solo	[0, 1, 0, 0, 0, 0, 0, 0, 0, 0]
Flute	[0, 0, 1, 0, 0, 0, 0, 0, 0, 0]
Guitar	[0, 0, 0, 1, 0, 0, 0, 0, 0, 0]
Keyboard solo	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0]
Mallet Solo	[0, 0, 0, 0, 0, 1, 0, 0, 0, 0]
Organ Solo	[0, 0, 0, 0, 0, 0, 1, 0, 0, 0]
Reed Solo	[0, 0, 0, 0, 0, 0, 0, 1, 0, 0]
String Solo	[0, 0, 0, 0, 0, 0, 0, 0, 1, 0]
Vocal Solo	[0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
Bass & Brass	[1, 1, 0, 0, 0, 0, 0, 0, 0, 0]
Bass & Flute	[1, 0, 1, 0, 0, 0, 0, 0, 0, 0]
... (other duo combinations)	
Bass & Vocal	[1, 0, 0, 0, 0, 0, 0, 0, 0, 1]
... (other trio quartet quintet sextet septet octet nonet decet combinations)	
All instruments	[1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
Total Combinations:	1024

Each row signifies a unique instrumental combination in a 10-bit array, where '1' indicates presence and '0' indicates absence.

Each instrument is represented by a binary digit in a 10-bit array, where a '1' indicates the presence and a '0' indicates the absence of that instrument in each sample. The 'No Instrument' class is represented by all zeros, while solo performances are indicated by a single '1' in the position corresponding to the instrument played. For combinations of instruments, the array contains a '1' for each instrument present in the combination. The array for 'All instruments' would be all ones, indicating that every instrument is playing simultaneously. The total number of combinations that can be represented in this binary classification system $2^{10} = 1024$, which includes all solo, duo, trio, quartet, quintet, sextet, septet, octet, nonet, and decet performances, as well as the case where no instrument is playing.

This binary representation is an effective method to handle multilabel classification problems where each label combination is treated independently. It is a compact and scalable way to encode the presence or absence of each instrument in the dataset, facilitating the binary classification process for each possible instrument inclusion within an audio sample. This encoding system is especially suitable for computational models such as neural networks, where multilabel classification can be inherently complex. Using a binary array simplifies the output layer of the model to a set of independent binary decisions, making it computationally efficient and easier to interpret.

IV. Methodology

Our methodology encompasses a comprehensive data collection process, incorporating a diverse array of audio samples to ensure a well-rounded and effective training dataset to simulate real-world scenarios accurately. Given that no prior feature extraction or feature selection is performed, the learning architecture must extract and select the most important features for itself from the spectrograms using supervised learning.

TABLE II. Procedure for Multilabel Classification Using Binary Classifiers

	Description of Experiment	
	Step	***Detail***
1	Data Collection	Select and preprocess a diverse set of instruments from the NSynth dataset. Generate spectrograms by Librosa *.STFT*() function [23] from audio samples to use as input features.
2	Training & Validation	Train separate CNN-based binary classifiers for each instrument by utilizing convolutional layers for feature extraction and dense layers for classification.
3	Testing on NSynth Solo Samples	Test each binary classifier on a testing set to measure individual accuracy. Evaluate the models' ability to identify the presence of multiple instruments within a sample through the overlap of binary classification outputs.
4	Validation on Multiple labeled Data	Generate synthetic audio by overlapping 0 to 10 instrument sounds to create 11 classes, including a 'no instrument' class. Validate classifiers on synthesized data to assess performance across varying levels of polyphony using accuracy and Exact Match Ratio (EMR) metrics.

Table II outlines the sequential steps taken in the experiment, from data collection to validation, detailing the methods and processes used at each stage.

The architecture of our OvA model uses CNN's capabilities to interpret spectrograms as images, enhancing the network's proficiency in extracting features for accurate instrument identification within polyphonic pieces. This is due, in part, to their ability to identify critical spatial relationships through their convolution and pooling layers, thus optimizing the CNN's processing power for complex audio recognition [24], [25]. The experiment design is presented in Table II.

A. Data Collection and Preparation

The NSynth dataset[22], known for its rich collection of musical instrument sounds synthesized using deep neural networks, serves as the foundation for our training data. To create a comprehensive training set, we select a diverse range of instruments from the NSynth dataset, ensuring a broad representation of musical families (bass, brass, flute, guitar, keyboard, mallet, organ, reed, string, vocal). The audio samples are standardized with regard to length, normalized with regard to volume levels, and converted into a suitable format for spectrogram generation. This step ensures uniformity and consistency across the dataset, crucial for effective learning including training, validation, and testing datasets.

All NSynth samples involved acoustic (not electronic) instruments and comprised 108900 acoustic samples to train (1000382 samples) and test (8518) the OvA instrument classifier. Feature Extraction with Spectrograms

TABLE III. Dataset Distribution for Model Training and Evaluation

	Family	**Training (validation split=0.2)**	**Testing**	**Total**
1	Bass	94	28	122
2	Brass	12487	1273	13,760
3	Flute	6166	406	6,572
4	Guitar	12423	920	13,343
5	Keyboard	7899	609	8,508
6	Mallet	25867	1855	27,722
7	Organ	151	25	176
8	Reed	13035	1227	14,262
9	String	18724	1786	20,510
10	Vocal	3536	389	3,925
	Total	100382	8518	108900

Table III lists the number of samples for each instrument family used during training (with a 20% validation split) and testing, alongside their cumulative totals.

Figure 2 presents the spectrogram of a flute audio sample, visualizing its frequency content over time. The spectrogram was generated using a Short-Time Fourier Transform (STFT) with specific parameters aimed at capturing the intricate details of the sound. The process involved converting the time-domain audio signal into a time-frequency representation, where the vertical axis represents frequency (in Hertz), and the horizontal axis denotes time (in seconds). The colour intensity at each point on the spectrogram correlates with the amplitude (in decibels) of a particular frequency at a given time, thus providing a powerful visual insight into the characteristics of the sound. The brighter the colour, the higher the amplitude of that frequency component at that time.

The STFT parameters included a window size of 2048 samples and a hop length of 512 samples, selected to balance

the resolution in the time and frequency domains effectively. By applying these parameters, the nuances in the audio signal's frequency spectrum are adequately captured, highlighting the harmonics and overtone structure unique to the flute's sound. This visual representation aids in understanding the audio sample's texture and is instrumental in training the convolutional neural network models, with a portion of the data being allocated for validation to assess the model's learning efficacy.

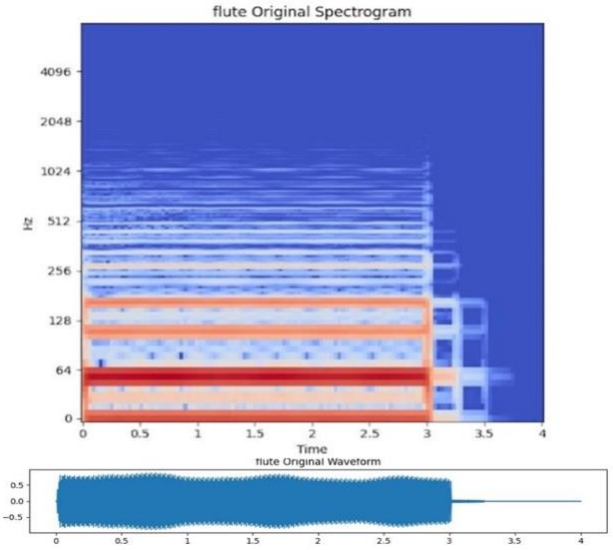

Fig. 2. Spectrogram of a Flute Audio Sample Generated using STFT

B. Binary Classifier Design and CNN Configuration

Binary Classifiers: For each instrument, a separate binary classifier was trained capable of distinguishing whether a given training sample (100,382 samples, representing ten distinct musical instruments, as detailed in Table III) contained that specific instrument (positive class: target instrument) or not (negative class: not target music instrument).

A One-vs-All (OvA) model architecture using Convolutional Neural Networks (CNNs) was used for each instrument in the NSynth dataset. The CNN Architecture is tailored to extract hierarchical features from spectrograms, including convolutional layers for feature detection and pooling layers for dimensionality reduction. (Table IV).

Table IV outlines the detailed layer-by-layer architecture of the Convolutional Neural Network (CNN) used for classifying instruments in audio samples. Each Conv2D layer, designed to extract features from the input spectrogram, is followed by a batch normalization layer to standardize the activations. Subsequent MaxPooling2D layers reduce dimensionality, retaining the most significant features.

The network concludes with fully connected Dense layers, including a Dropout layer to prevent overfitting, and culminates in a sigmoid activation function for binary classification.

TABLE IV. CNN ARCHITECTURAL CONFIGURATION FOR INSTRUMENT CLASSIFICATION

Layer Type	Configuration
Conv2D	32 filters, 3x3 kernel, activation='relu', padding='same'
BatchNormalization	-
MaxPooling2D	2x2 pool size
Conv2D	64 filters, 3x3 kernel, activation='relu', padding='same'
BatchNormalization	-
MaxPooling2D	2x2 pool size
Conv2D	128 filters, 3x3 kernel, activation='relu', padding='same'
BatchNormalization	-
MaxPooling2D	2x2 pool size
Flatten	-
Dense	128 nodes, activation='relu'
Dropout	rate=0.5
Dense	1 node, activation='sigmoid'

C. Synthetic Data Generation for Polyphony

To evaluate the models' performance in polyphonic settings, we synthesized new audio samples by overlapping sounds from multiple instruments. We create synthetic audio samples by overlapping 2 to 10 different instrument sounds from the NSynth dataset, generating a range of polyphonic complexities. For each synthetic sample, multiple labels corresponding to the instruments are assigned, thereby preparing the data for multilabel classification evaluation.

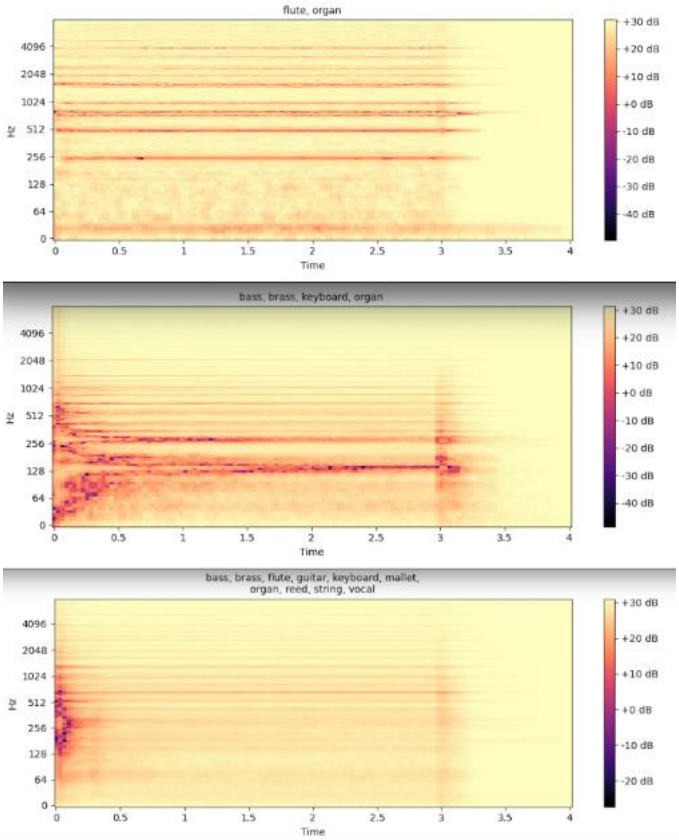

Fig. 3. Examples of Spectrogram Complexity of Duo, Quartet and Decet Overlaps (To maintain conciseness, we present only three spectrograms).

Figure 3 presents visualizations of spectrograms polyphonic overlaps. The top visualizes a duo overlap of organ and flute, introducing bi-instrument complexity. The middle shows a quartet blend of bass, brass, keyboard, and organ, showcasing increased polyphonic density. Lastly, the bottom represents a synthesized overlap of all ten instruments, presenting the highest level of complexity tested. These selected examples are indicative of the model's capacity to handle varying degrees of polyphony, chosen for illustration due to space constraints in the paper.

Thus, mirroring the spectrogram examples in Figure 3, Table V presents our methodology for constructing a dataset to evaluate the model's capability to recognize multiple instruments simultaneously. We synthesized audio samples with an increasing number of instrument sounds to create classes that represent various levels of polyphony, from solo pieces to decets. Each class contains 100 samples, resulting in a total of 1100 samples. This approach guarantees a balanced distribution of samples, reflecting a wide array of polyphonic complexities for the multilabel classification evaluation.

From no instrument class [0,0,0,0,0,0,0,0,0,0] to all instrument class [1,1,1,1,1,1,1,1,1,1] , we randomly synthesized 100 samples for each class. The detail of the spectrogram and the label representation are shown in table V.

TABLE V. SYNTHESIZED POLYPHONIC COMPLEXITY SAMPLES

Spectrogram Samples	Class Label of Synthesised Spectrogram	Number of Samples
No Instrument	10-bit array with all zeros [0, 0, 0, 0, 0, 0, 0, 0, 0, 0], indicating the spectrogram of no instrument.	100
Random Solo Pieces (such as the spectrogram on figure 3 top-left)	10-bit array with one '1' and the rest zeros	100
Random Duo (such as figure 3 top-right)	10-bit array with two '1' and the rest zeros	100
Random Trio	10-bit array with three '1' and the rest zeros	100
Random Quartet	10-bit array with four '1' and the rest zeros	100
Random Quintet (such as the spectrogram on figure 3 bottom-left)	10-bit array with five '1' and the rest zeros	100
Random Sextet	10-bit array with six '1' and the rest zeros	100
Random Septet	10-bit array with seven '1' and the rest zeros	100
Random Octet	10-bit array with eight '1' and the rest zeros	100
Random Nonet	10-bit array with nine '1' and the rest zeros	100
Random Decet (such as the spectrogram on figure 3 bottom-right)	10-bit array with all '1's [1, 1, 1, 1, 1, 1, 1, 1, 1, 1], indicating ten instruments Spectrogram	100
Total		**1100**

Each polyphonic class in the dataset is constructed by randomly overlaying the indicated number of distinct instrument sounds from the NSynth dataset, ensuring a balanced representation of polyphonic diversity for evaluation purposes.

D.Evaluation Metrics

In the context of multilabel classification, the overall accuracy of a model is computed by averaging the accuracy of each instance. This is mathematically represented in the following equation:

$$Accuracy = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i \cap z_i|}{|y_i \cup z_i|} \qquad (3)$$

In (3), n represents the total number of instances in the dataset. y_i denotes the actual labels for the i-th instance, and represents the predicted labels for the n-th instance. The intersection $|y_i \cup z_i|$ calculates the number of correctly predicted labels, while the union $|y_i \cup z_i|$ represents the total unique labels (both actual and predicted) for that instance. The accuracy for each instance is the ratio of the intersection to the union, and overall accuracy is the average of these ratios across all instances. This accuracy allows the partially correct predictions. Multilabel Accuracy sometimes can be called as Intersection Over Union.

On the other hand, the Exact Match Ratio (EMR) is a stricter metric that quantifies the percentage of samples for which the model has made completely correct identifications. The EMR is defined by the equation:

$$EMR = \frac{Number\ of\ exact\ match}{Total\ Number\ of\ samples} \qquad (4)$$

With EMR, only those instances where the predicted labels exactly match the true labels are considered correct, providing no tolerance for partially correct predictions.

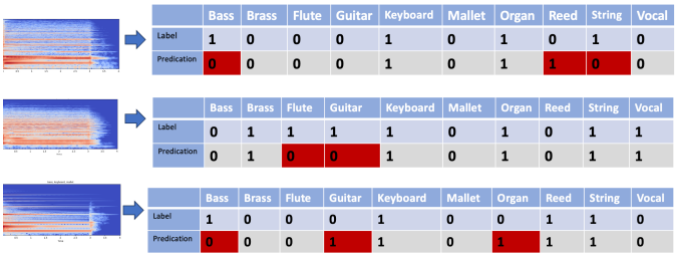

Fig. 4. Example of Multilabel Classification Result: The left side of the figure presents spectrograms of audio samples with various instrument mixtures, while the right side shows the corresponding predictions of the binary classifiers for each instrument type. The red highlights indicate discrepancies between the ground truth labels and the predictions made by the model. This juxtaposition illustrates the process of classification and the model's precision in identifying multiple instruments within a single sample.

For example, Figure 4 shows a fictitious instance of a multilabel classification result. To calculate the precision of our multilabel classification approach, we applied formulas (3) and (4) as detailed in the methodology section. In the first example, the top spectrogram reveals a polyphonic mixture of four instruments, yet the model encounters three misclassifications, highlighted in red. The middle spectrogram exhibits two errors in prediction, while the third displays three inaccuracies. The collective accuracy calculates to 73%, derived from the mean of individual accuracies (70%, 80%, and 70%). The Exact Match Ratio (EMR) stands at 0%, indicating no perfect matches across the examples. This dual metric evaluation offers a nuanced view of model performance, with EMR providing a stringent assessment of complete prediction accuracy.

V. Results

A. Outcomes of Binary Classifier Training

Figure 5 presents the training outcomes for the bass and guitar classifiers. Similar results were observed across the other eight classifiers. The training was capped at 1000 epochs with a termination criterion set to a precision threshold of 98%, ensuring that all binary classifiers achieved an accuracy exceeding 98%. According to Figure 5, the top left graph represents the accuracy of the bass classifier over a number of training epochs. As the number of epochs increases (the x-axis), the accuracy (the y-axis) improves, reaching over 90% after 200 epochs. The top right graph shows the loss value for the bass during training, which indicates how much the prediction deviates from the actual result; as training progresses, this loss decreases, reflecting improved performance of the classifier. The bottom graphs correspond to the guitar classifier, exhibiting a similar trend. Both accuracy and loss for the guitar show improvement over epochs, with accuracy increasing and loss decreasing, analogous to the bass classifier's performance.

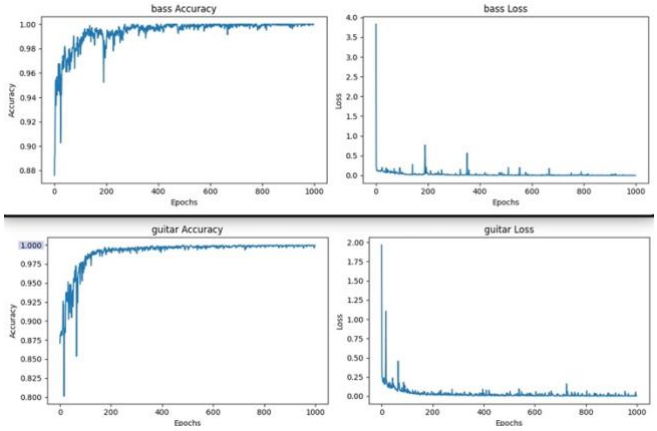

Fig. 5. The left charts represent the accuracy evolution of the bass (top) and guitar (bottom) classifiers with increasing epochs, where the x-axis denotes the epoch count and the y-axis represents the accuracy percentage. The right charts depict the corresponding loss metrics for each classifier, with the x-axis again tracking the epoch count and the y-axis illustrating the loss magnitude. Both classifiers exhibit a convergence in accuracy and a decrease in loss as the number of epochs increases.

To maintain brevity in the paper, only these two graphs are shown, although the same performance trend is observed in the classifiers for the other eight instruments.

B. Binary Classifier Results for Single Instrument Recognition

Table VI illustrates the testing results for the binary classifiers of various musical instruments when presented individually. The overall accuracy across all classifiers is 97%. In table VI, recall, and F1-scores for each classified instrument on the testing dataset are presented separately. These metrics reflect the classifiers' ability to accurately identify the presence of specific instruments in the audio samples. The 'Sample' column indicates the number of instances each instrument was tested within the dataset. An aggregate accuracy, along with macro and weighted averages, encapsulates the overall performance across all instrument classifiers. Excluding the bass, which has a notably lower precision likely due to an imbalanced sample size (28 in total) from NSynth testing dataset, all instrument classifiers exhibit over 90% accuracy on the testing set. The reed, brass, guitar, keyboard, and string classifiers all got 90% plus precision and recall rates.

TABLE VI. Performance Metrics of Classifiers on Individual Instruments

Class	Precision	Recall	F1-Score	Sample
bass	42%	100%	60%	28
brass	98%	99%	98%	1273
flute	90%	96%	93%	406
guitar	98%	96%	97%	920
keyboard	99%	98%	99%	609
mallet	99%	98%	98%	1855
organ	93%	100%	96%	25
reed	98%	95%	96%	1227
string	98%	98%	98%	1786
vocal	100%	100%	100%	389
Accuracy	97%	97%	97%	**8518**

To appraise the performance of our 10 binary classifiers, we employed a two-phased comparative approach. First, we undertook a single instrument-to-instrument comparison, comparing our classifiers' ability to identify bass, brass, flute, etc., against three other convolutional models which used different instruments, datasets and ML architectures. This comparison provides a comparative, single-instrument perspective where appropriate on classifier efficacy, as detailed in Table VII.

TABLE VII. Comparative Performance Metrics of Our Model and Previous Single Instrument Results

Class	Our Model Precision (acoustic subset)	NSnyth Dataset (full dataset)[26]	IRMAS Dataset [11]	Slakh dataset [15]
bass	42%	70%	-	94%
brass	98%	79%	-	-
flute	90%	81%	55%	-
guitar	98%	71%	70%	82%
keyboard	99%	70%	-	-
mallet	99%	73%	-	-
organ	93%	97%	-	-
reed	98%	88%	-	-
string	98%	93%	-	-
vocal	100%	45%	72%	-

Table VII illustrates the precision scores of our model on an acoustic subset alongside performance metrics from the NSynth, IRMAS, and Slakh datasets. It highlights the precision for each instrument class, showcasing the effectiveness of our model in comparison to the results provided by other techniques using different datasets. Our model demonstrates better precision across most classes, particularly when evaluated on acoustic samples. The dashes indicate where comparable data were not available for a specific dataset.

Table VII includes the precision of identical instrument labels for a fair comparison. For example, the term 'piano' in other studies may be considered under the broader category of

'keyboard' in our study but is not included here since they are not identically labelled. 'Keyboard' covers other instruments apart from the piano.

The second phase as described in Table VIII compares music sample model performance on the same NSynth dataset. Here, while models cited in [27], and [28] are evaluated across acoustic, electronic, and synthetic audio sample types, our evaluation remains confined to acoustic samples.

TABLE VIII. EVALUATION OF MODEL PERFORMANCE AGAINST PREVIOUS RESULT

Sample		**Model Performance**		
Type	**Sample Size**	**Efficient-LEAF** [27]	**LEAF** [28]	**Our Model**
Acoustic	108,978	☑	☑	☑
Electronic	110,224	☑	☑	
Synthetic	86,777	☑	☑	
Accuracy		72.1%	69.2%	97%

Table VIII compares the performance of our classification model with the Efficient-LEAF and LEAF results across acoustic, electronic, and synthetic sample types. Our model achieved an overall accuracy of 97%, indicating a substantial improvement over previous result. The 'R' notation signifies the availability of results, with a blank space indicating that the corresponding dataset was not evaluated. This direct comparison underscores our model's effectiveness, especially in acoustic sample recognition.

C. Binary Classifier on multiple-instrument samples

As seen in Figure 6, our model achieves a 100% accuracy rate for no instrument (silence) spectrograms and an EMR of 1, indicating flawless recognition of silence. For solo instrument spectrograms, our model demonstrates a 97% accuracy and a 79% EMR, showing high effectiveness in identifying single instruments. In duo scenarios, the accuracy slightly dips to 77%, with a 4% EMR, suggesting more difficulty in precise recognition of two-instrument combinations. As we progress to trios, the model's accuracy is 71% and the EMR drops to 2%, continuing the trend of decreased recognition precision with increased polyphony. The quartet scenarios further challenge the model, with accuracy at 62% and EMR at 0%. The diminishing trend continues through quintet to decet categories, with respective accuracies of 58%, 53%, 48%, 45%, 43%, and 37% and EMR remaining consistently low or non-existent.

The overall accuracy, calculated by equation (3), which is the accuracies for no instrument through decets, is 64%, and the overall EMR, calculated by equation (4), determined by the mean of EMRs across all classes, is 17%.

According to research outlined in paper [18], [19], accuracy in multilabel classification includes partially correct predictions, whereas EMR strictly accounts for completely accurate predictions. In our 10-instrument classification, with 10 binary classifications, a random guess, according to equation (4), would result in an accuracy equivalent to the probability of each class being correct by chance, while a random EMR would be the inverse of the number of all possible label combinations, which is $1/2^{10}$, or approximately 0.00098. Therefore, a EMR of 17%, suggests that the model is performing well above chance levels (0.098%).

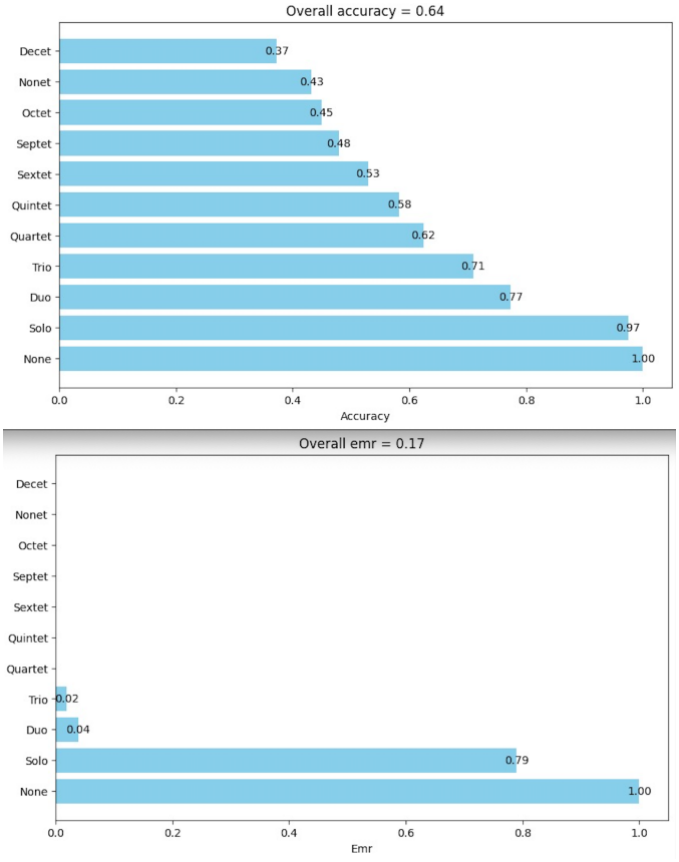

Fig. 6. Performance Metrics for Polyphonic Instrument Classification: The top chart details the classification accuracy for solo to decet classes, while the bottom chart presents the Exact Match Ratio (EMR).

The graph shown in Figure 7 charts the individual accuracy trends for each instrument classifier as it processes an increasing number of ensemble sizes. Each line represents a distinct instrument, tracing the model's ability to correctly identify the presence of that instrument within a varying mix of other sounds. The trend lines suggest that while some instruments maintain a relatively stable accuracy rate, others exhibit a more pronounced decline as the ensemble complexity grows.

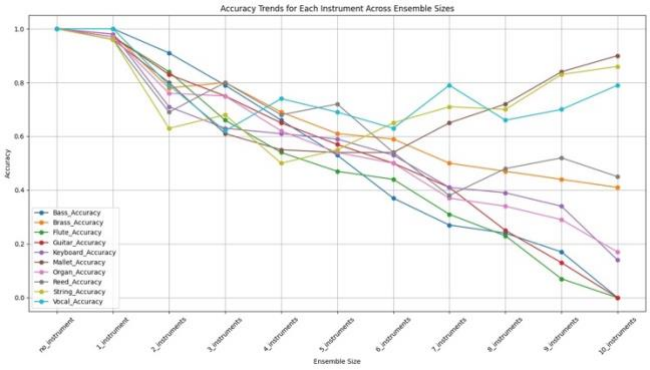

Fig. 7. Individual Instrument Accuracy Trends Over Ensemble Sizes: This graph depicts the trajectory of classification accuracy for each instrument as the number of instruments in the ensemble increases.

In Figure 8, we see a comprehensive view of how each instrument's recognition true positive rate is affected by the number of ensemble layers. The converging and diverging lines indicate the model's varying levels of performance, offering a direct comparison between different instruments. The graph provides a detailed perspective on the specific challenges that polyphonic recognition presents, highlighting the nuanced capabilities of our classification system in distinguishing among a range of instrumental combinations.

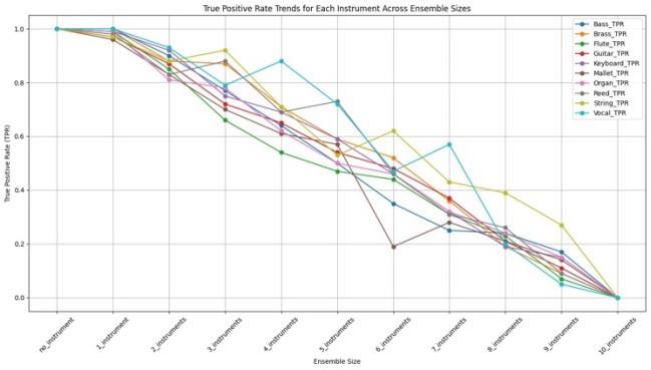

Fig. 8. Comparative Instrument Recognition Trends: The plot illustrates the variance in model accuracy for individual instruments against the expanding complexity of ensemble compositions.

VI. Discussion

According to Figure 6, the overall accuracy, calculated by equation (3), which is the accuracies for no instrument through decets, is 64%. In detailed disscusion, the no instrument senario gets perfect scores (100%) for cases with no instruments, suggesting good silence detection. Solo instruments achieve a high accuracy (97%) yet lower EMR (79%) for solo pieces indicate potential over-prediction but good instrument identification. Regarding 2 instruments, despite decent accuracy (77%), a low EMR (4%) for duos shows challenges in exact instrument pairing recognition. The result of 3 instruments to 6 instruments is declining accuracy and EMR across these classes reveals the model's struggles with identifying multiple overlapping instruments. Regarding the larger ensembles, for groups larger than 6 instruments, accuracy below 50% implies the model's performance is insufficient for complex ensembles.

Figure 7 and Figure 8, show the accuracy and True Positive Rate (TPR) of 10 instrument in 10 different overlayed scenarios. In the scenario of silence (no instruments playing), the model's accuracy and TPR stand at 100%, showcasing its capacity to accurately detect the absence of instrumental sound. However, as we introduce solo instruments into the mix, we observe that while most instruments maintain an accuracy and TPR of 100%, a few starts to deviate slightly, with brass at 98% accuracy and mallet at 96%. For duos, the accuracy for bass drops to 91%, and brass to 78%, with corresponding TPRs of 90% and 88%, indicating the model's beginning to struggle with multi-instrument recognition. By the time we reach trios, instruments like flute and keyboard drop to accuracy rates of 66% and 63%, and TPRs of 66% and 75% respectively, highlighting a more pronounced difficulty in recognizing simultaneous instruments. In quartets, the accuracy for flute further declines to 54%, and guitar to 65%, with their TPRs also falling to 54% and 65%. When evaluating quintets, accuracy for bass is at 53% and for guitar at 57%, while TPR for both dips below 55%, suggesting that the complexity of audio patterns is beginning to notably affect performance. As we introduce sextets, the model's performance declines steeply with bass accuracy at 37% and flute at 44%, and their TPRs at 35% and 44% respectively. In septet arrangements, the trend continues with even lower accuracy rates; bass falls to 27%, and flute to 31%. When it comes to octets and nonets, the accuracy for bass plummets to 24% and 17%, indicating the model's increasing struggle in dense polyphonic environments. The TPR for brass in nonets is just 9%, revealing significant issues with identifying specific instruments in complex audio samples.

Additionally, it is also necessary to discuss the 10 scenarios by confusion matrix.

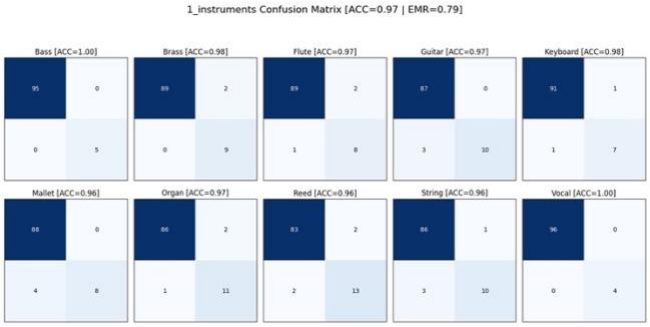

Fig. 9. Confusion Matrix of each instrument in Solo.

Figure 9 illustrates the solo instrument scenario. The accuracy of bass and vocal achieve 100% accuracy with the rest of the instruments like Brass, Flute, Guitar, Keyboard, and Mallet achieving above 96%. The TPR for all instruments remained at or above 96%, showcasing the model's precision in identifying individual instruments.

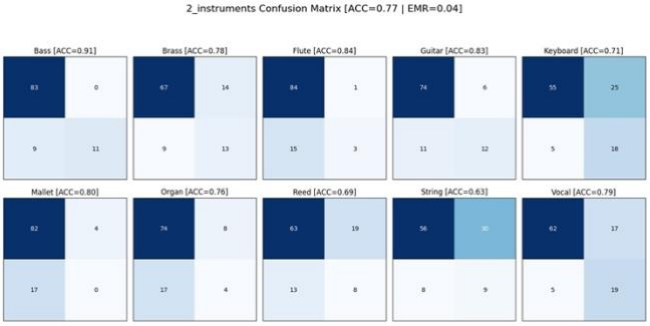

Fig. 10. Confusion Matrix of each instrument in Duo.

In duo combinations, with two instruments playing together, as shown in Figure 10, there is a slight dip in accuracy, with bass showing the highest decrease to 91% and TPR declines to 90%. The worst performed classifier is string classifier with an accuracy of 63% and TPR of 88%.

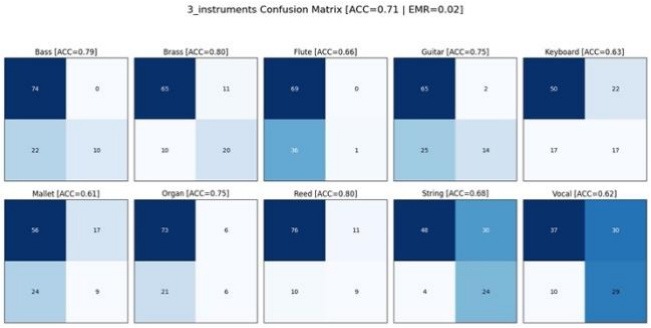

Fig. 11. Confusion Matrix of each instrument in Trio.

Figure 11 is the confusion matrix of trio instrument scenario. Introducing a third instrument brought more complexity, reflected in lower accuracy and TPR values. Bass, for example, dropped to 79% in accuracy and 77% in TPR. This suggests that as polyphony increases, the model starts to face difficulties in maintaining its high recognition performance.

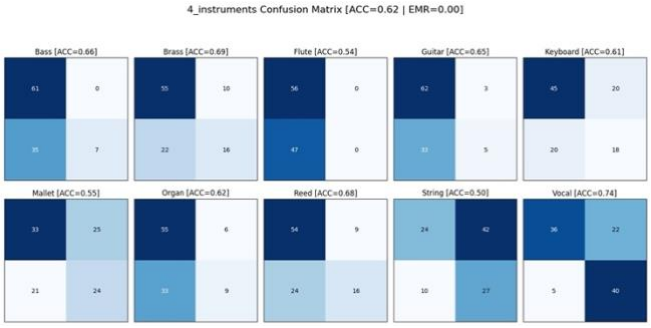

Fig. 12. Confusion Matrix of each instrument in the Quartet.

Also, according to Figure 12, with four instruments, the model's performance showed further challenges. The accuracy for most instruments fell below 70%, with Flute experiencing the most significant reduction to 54%. This marks the beginning of substantial difficulties in polyphonic detection.

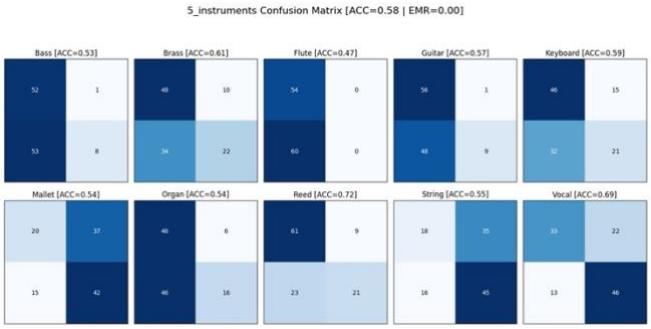

Fig. 13. Confusion Matrix of each instrument in Quintet.

Quintet Instrument Scenario (figure 13): With five instruments, the model's accuracy generally hovered around the 50-60% range. The TPR also reflected similar trends, indicating the model's struggle to discern individual instruments amidst the complex sound mixtures.

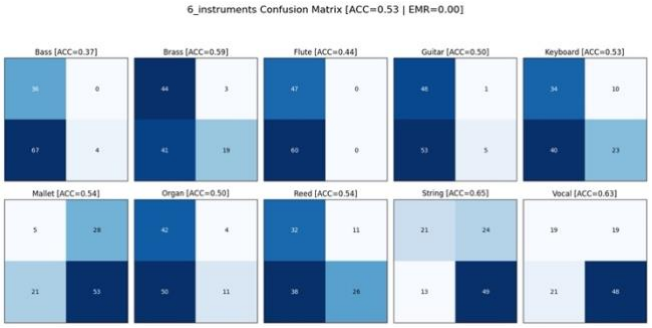

Fig. 14. Confusion Matrix of each instrument in Sextet.

Sextet Instrument Scenario: Accuracy and TPR continued to decrease as the ensemble size grew to six instruments. The model's capability to correctly identify the instruments was notably hampered, with Bass accuracy dropping to 37%.

To maintain brevity and due to the declining performance observed in scenarios with seven or more instruments, detailed confusion matrices for Septet to Decet are omitted. From the Septet scenario onwards, the overall accuracy consistently fell below 50%, indicating the model's limited capability in handling high levels of polyphony. This trend underscores the model's performance boundaries in complex audio settings, which become increasingly pronounced as the number of simultaneous instruments rises.

Finally, these results should be put in the context of whether humans can accurately recognize all instruments in a short sample of music. Single instruments can be expected to be easily recognized by trained musicians, as well as combinations of two or three instruments. However, when multiple instruments are playing, even expert musicians may need longer to accurately identify all the instruments.

VII. Conclusion and Future Work

In conclusion, by leveraging the robust capabilities of Convolutional Neural Networks (CNNs) and binary classification strategies, our study makes a useful contribution to the domain of musical instrument recognition in polyphonic compositions. By implementing a One-vs-All (OvA) approach and focusing on spectrogram-based feature extraction, we have identified the potential to enhance accuracy and adaptability in recognizing complex musical arrangements. Our methodology advances the precision of detecting individual instruments in polyphony, evidencing an appreciable performance in ensembles of two to four instruments. Beyond this complexity, however, we observe a decline in model efficacy, particularly when confronted with arrangements involving more than four instruments, underscoring the intricate challenges of polyphonic recognition. Recognizing more than three instruments in short may also be challenging for humans given only 7 seconds [29] of music.

Our next steps involve expanding our dataset to include a broader spectrum of music genres, incorporating both electronic and synthetic instruments. This expansion will enable our system to become more adaptive and comprehensive. Ultimately, our goal is to advance the capabilities of Music Information Retrieval (MIR) technology,

paving the way for advancements in digital music analysis and the automation of music transcription processes. This development has the potential to enhance the precision of multiple musical instrument recognition system in real time (e.g. during streaming) and contribute significantly to the field of real-time MIR.

VIII. Reproducibility

To ensure the reproducibility of our research, all code and related materials have been made available in a public GitHub repository[1]. This includes detailed instructions for setting up the environment, processing the data, and running the experiments.

References

[1] K. Patil, D. Pressnitzer, S. Shamma, and M. Elhilali, 'Music in our ears: the biological bases of musical timbre perception', PLoS computational biology, vol. 8, no. 11, p. e1002759, 2012.

[2] T. Chi, P. Ru, and S. A. Shamma, 'Multiresolution spectrotemporal analysis of complex sounds', The Journal of the Acoustical Society of America, vol. 118, no. 2, pp. 887–906, 2005.

[3] T. R. Agus, C. Suied, S. J. Thorpe, and D. Pressnitzer, 'Fast recognition of musical sounds based on timbre', The Journal of the Acoustical Society of America, vol. 131, no. 5, pp. 4124–4133, 2012.

[4] X. Yang, K. Wang, and S. A. Shamma, 'Auditory representations of acoustic signals', IEEE transactions on information theory, vol. 38, no. 2, pp. 824–839, 1992.

[5] B. Kostek, 'Musical instrument classification and duet analysis employing music information retrieval techniques', Proceedings of the IEEE, vol. 92, no. 4, pp. 712–729, 2004.

[6] G. Agostini, M. Longari, and E. Pollastri, 'Musical instrument timbres classification with spectral features', EURASIP Journal on Advances in Signal Processing, vol. 2003, pp. 1–10, 2003.

[7] A. Krishna and T. V. Sreenivas, 'Music instrument recognition: from isolated notes to solo phrases', in 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing, IEEE, 2004, pp. iv–iv.

[8] D. Kim, T. T. Sung, S. Y. Cho, G. Lee, and C. B. Sohn, 'A single predominant instrument recognition of polyphonic music using CNN-based timbre analysis', International Journal of Engineering & Technology, vol. 7, no. 3.34, p. 590, 2018.

[9] Y. Han, J. Kim, and K. Lee, 'Deep convolutional neural networks for predominant instrument recognition in polyphonic music', IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no. 1, pp. 208–221, 2016.

[10] J. S. Gómez, J. Abeßer, and E. Cano, 'Jazz Solo Instrument Classification with Convolutional Neural Networks, Source Separation, and Transfer Learning.', in ISMIR, 2018, pp. 577–584.

[11] A. Solanki and S. Pandey, 'Music instrument recognition using deep convolutional neural networks', International Journal of Information Technology, vol. 14, no. 3, pp. 1659–1668, 2022.

[12] D. Szeliga, P. Tarasiuk, B. Stasiak, and P. S. Szczepaniak, 'Musical instrument recognition with a convolutional neural network and staged training', Procedia Computer Science, vol. 207, pp. 2493–2502, 2022.

[13] P. Shreevathsa, M. Harshith, A. Rao, and others, 'Music instrument recognition using machine learning algorithms', in 2020 International Conference on Computation, Automation and Knowledge Management (ICCAKM), IEEE, 2020, pp. 161–166.

[14] L. Zhong, D. Saito, and N. Minematsu, 'Predominant instrument recognition in polyphonic music based on transfer learning with vanilla ResNet-50', IEICE Technical Report; IEICE Tech. Rep., vol. 122, no. 387, pp. 232–237, 2023.

[15] M. Blaszke and B. Kostek, 'Musical instrument identification using deep learning approach', Sensors, vol. 22, no. 8, p. 3033, 2022.

[16] M. Galar, A. Fernández, E. Barrenechea, H. Bustince, and F. Herrera, 'An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes', Pattern Recognition, vol. 44, no. 8, pp. 1761–1776, 2011.

[17] R. Rifkin and A. Klautau, 'In defense of one-vs-all classification', The Journal of Machine Learning Research, vol. 5, pp. 101–141, 2004.

[18] M.-L. Zhang and Z.-H. Zhou, 'A review on multi-label learning algorithms', IEEE transactions on knowledge and data engineering, vol. 26, no. 8, pp. 1819–1837, 2013.

[19] G. Tsoumakas and I. Katakis, 'Multi-label classification: An overview international journal of data warehousing and mining', The label powerset algorithm is called PT3, vol. 3, no. 3, 2006.

[20] M.-L. Zhang and Z.-H. Zhou, 'Multi-label learning by instance differentiation', in AAAI, 2007, pp. 669–674.

[21] M. R. Boutell, J. Luo, X. Shen, and C. M. Brown, 'Learning multi-label scene classification', Pattern recognition, vol. 37, no. 9, pp. 1757–1771, 2004.

[22] J. Engel et al., 'Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders'. 2017.

[23] B. McFee et al., 'librosa: Audio and music signal analysis in python', in Proceedings of the 14th python in science conference, 2015.

[24] N. Sharma, V. Jain, and A. Mishra, 'An analysis of convolutional neural networks for image classification', Procedia computer science, vol. 132, pp. 377–384, 2018.

[25] L. Chen, S. Li, Q. Bai, J. Yang, S. Jiang, and Y. Miao, 'Review of image classification algorithms based on convolutional neural networks', Remote Sensing, vol. 13, no. 22, p. 4712, 2021.

[26] T. Nylén and V. Stenmark, 'Comparison of CNN and LSTM for classifying short musical samples'. 2023.

[27] J. Schlüter and G. Gutenbrunner, 'Efficientleaf: A faster learnable audio frontend of questionable use', in 2022 30th European Signal Processing Conference (EUSIPCO), IEEE, 2022, pp. 205–208.

[28] N. Zeghidour, O. Teboul, F. de C. Quitry, and M. Tagliasacchi, 'LEAF: A learnable frontend for audio classification', arXiv preprint arXiv:2101.08596, 2021.

[29] F.-R. Stöter, M. Schoeffler, B. Edler, and J. Herre, 'Human ability of counting the number of instruments in polyphonic music', in Proceedings of Meetings on Acoustics, AIP Publishing, 2013.

[1] https://github.com/fireHedgehog/music-intrument-OvA-model/tree/main