Instrument Recognition in Full-Length Polyphonic Songs: An Approach Using Foundational Models

Humberto Navarro de Carvalho[1],
Yuri de Almeida Malheiros Barbosa[1],
Thaís Gaudêncio do Rêgo[1]

[1]Centro de Informática – Universidade Federal da Paraíba (UFPB)

humbertonavarroc@gmail.com, yuri@ci.ufpb.br, gaudenciothais@gmail.com

Resumo. *A identificação de instrumentos musicais em músicas polifônicas é uma tarefa crucial no campo da Recuperação de Informação Musical (MIR), com impactos na catalogação, indexação e análise de música. Tradicionalmente, a pesquisa sobre identificação de instrumentos é tipicamente limitada à classificação de pequenos trechos de sons individuais ou mixes. No entanto, muitas aplicações práticas requerem a detecção de instrumentos em músicas polifônicas completas. Neste artigo, propomos um método para identificar instrumentos musicais em tais músicas, utilizando modelos fundacionais. Primeiro, extraímos as características das instâncias do conjunto de dados OpenMIC usando modelos de áudio fundacionais. Em seguida, treinamos um perceptron multicamadas para identificação de instrumentos em trechos de áudio. Por último, aplicamos diferentes métodos de agregação para obter predições de instrumentos em músicas inteiras. Para testar o modelo, realizamos testes tanto no nível de trechos de áudio, utilizando a divisão de teste do conjunto de dados OpenMIC, quanto no nível de músicas, usando uma versão modificada do conjunto de dados MoisesDB. Como as classes eram diferentes em ambos os conjuntos de dados, foi criado um mapeamento para converter previsões do modelo para nove classes de instrumentos. O modelo alcançou uma pontuação F1 média macro de 0,806 para as 20 classes nos trechos e 0,772 nas músicas de longa duração para 9 classes. Em cinco instrumentos chave (voz, guitarra, baixo, bateria e piano), o modelo teve uma pontuação F1 de 0,91 e 0,914 nos níveis de trecho e de música, respectivamente. Ao integrar uma variedade de técnicas de agregação, nosso método oferece um avanço sobre os modelos tradicionais que estão principalmente limitados à análise de segmentos curtos de áudio, demonstrando a aplicabilidade prática do modelo em cenários do mundo real.*

Palavras-chave: *Identificação de Instrumentos, Recuperação de Informação Musical, Modelos Fundacionais, Classificação Musical, Aprendizado de Máquina.*

Catalogação na publicação
Seção de Catalogação e Classificação

C331i Carvalho, Humberto Navarro de.
Instrument recognition in full-Length polyphonic songs: an approach using foundational models / Humberto Navarro de Carvalho. - João Pessoa, 2024.
15 f. : il.

Orientação: Yuri Barbosa.
Coorientação: Thaís Rêgo.
TCC (Graduação) - UFPB/CI.

1. Identificação de instrumentos. 2. Recuperação de informação musical. 3. Modelos fundacionais. 4. Classificação musical. 5. Aprendizado de máquina. I. Barbosa, Yuri. II. Rêgo, Thaís. III. Título.

UFPB/CI CDU 004.8

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

Abstract. *Identifying musical instruments in polyphonic music is a crucial task in the field of Music Information Retrieval (MIR), with impacts on music cataloging, indexing, and analysis. Traditionally, research on instrument identification is typically limited to the classification of short snippets of individual sounds or mixes. However, many practical applications require the detection of instruments in full-length, polyphonic songs. In this paper, we propose a method for identifying musical instruments on entire polyphonic songs, by using foundational models. First, we extract the features of the instances in the OpenMIC dataset using foundational audio models. Then, we train a multilayer perceptron for instrument identification on snippets. Last, we apply different aggregation methods to obtain song-level instrument predictions. To test the model, we conducted both snippet-level tests, using the test split from the OpenMIC dataset, and song-level tests, using a modified version of the MoisesDB dataset. Since the classes were different on both datasets, a mapping was created to convert predictions from the model to nine instrument classes. The model achieved a macro-averaged F1 Score of 0.806 for the 20 classes on the snippets and 0.772 on full-length songs for 9 classes. On the five key instruments (voice, guitar, bass, drums and piano), the model had an F1 Score of 0.91 and 0.914 on the snippet and song level, respectively. By integrating a range of aggregation techniques, our method offers an advancement over traditional models that are primarily limited to analyzing short audio segments, demonstrating the practical applicability of the model in real-world scenarios.*

Keywords: *Instrument Recognition, Music Information Retrieval, Foundational Models, Music Classification, Machine Learning.*

1. Introduction

Instrument recognition in music is a critical area within Music Information Retrieval (MIR) with applications in music cataloging, indexing, automated music analysis, and dataset tagging. Accurately identifying instruments in full-length polyphonic songs can significantly improve these systems, aiding tasks such as music recommendation, playlist curation, and interactive learning tools for musicians and producers. However, existing research predominantly focuses on the classification of small polyphonic snippets of songs or on monophonic snippets [1–15]. This narrow focus on short snippets often fails to account for the temporal continuity and the complex interactions within complete songs. In the real world, the full-length, polyphonic nature presents a more nuanced challenge for accurate instrument recognition, which most current research fails to consider. Therefore, this paper has two main contributions:

(i) Training a multilayer perceptron (MLP) to predict instruments in 10-second snippets using embeddings from foundational audio models. (ii) Extending these techniques to handle full-length songs by applying various aggregation methods to consolidate snippet-level predictions into comprehensive song-level classifications.

In addition to our main objective, we explore secondary research questions concerning the effectiveness of specific aggregation methods for different instruments.

The MLP was trained and tested on the OpenMIC-2018 dataset [16], and the

model's capability of translating these results to full-length songs was evaluated on a modified version of the MoisesDB dataset [17], comprising entire songs with annotated instrument presence.

Our evaluation on the MoisesDB dataset included nine different instrument classes, providing a broad perspective on the model's performance. For a more detailed analysis, we also evaluate the model's performance on just five key instruments: voice, guitar, bass, drums, and piano. These instruments were chosen for their prominence in contemporary popular music, and due to their essential roles in shaping a song's structure and feel. The human voice carries melody, emotion, and lyrics; the guitar and piano establish the harmonic foundation and define chord progressions; the bass adds depth and rhythm, complementing the drums, which drive tempo and dynamics.

Our research focuses on addressing the gap in instrument recognition for complete polyphonic songs, extending the applicability of snippet-based models to real-world scenarios.

2. Related Works

As mentioned in the Introduction, current research in music instrument recognition primarily emphasizes three distinct areas: monophonic recognition on short snippets, identifying the predominant instrument in snippets, and polyphonic recognition on short audio fragments. For instance, studies on monophonic recognition focus on distinguishing isolated instrument sounds, which simplifies the audio analysis but limits applicability to real-world music scenarios where multiple instruments interact [1–3]. In contrast, research on predominant instrument recognition seeks to identify the main instrument in polyphonic audio mixes [10–12], yet often misses secondary instruments that contribute to the music's texture. Polyphonic recognition, while advancing the state of multi-instrument analysis, typically confines itself to brief audio clips [4–9, 13–15], thereby lacking the broader context of entire compositions. These approaches do not align with our focus on full-length polyphonic songs.

Some studies [9, 13–15, 18] use the OpenMIC dataset and similar metrics to ours on snippet-based recognition, allowing for performance comparison for the first half of our work.

Closer to our approach, some works evaluate instrument prediction on full-length songs [19–21]. However, these studies differ from ours due to them utilizing a single model for recognition, and smaller time resolutions. [19, 20] also use a fixed threshold of 0.5 for binary predictions and a single aggregation method for predictions in full-length songs. In contrast, we use a foundational model with a classification head to obtain predictions, five different aggregation methods and calculate unique thresholds for each instrument.

3. Methodology

This section outlines the datasets used, the methodology for training the MLP, the process for aggregating model predictions from 10-second clips to full-song predictions, the method for obtaining thresholds to convert probabilities into binary predictions, and finally, the complete pipeline for predicting instrument presence in full-length songs.

3.1. OpenMIC Dataset

The OpenMIC dataset comprises 20,000 10-second audio clips annotated with the presence of 20 different instrument classes. This dataset is ideal for our model training because of its large number of instruments and due to its snippet size ensuring the presence of instruments throughout the entire clip – a feature not guaranteed in full songs. We used the OpenMIC's official train/test partition and dedicated 10% of the train data for validation, resulting in 13,424 clips for training, 1,491 for validation, and 5,085 for testing.

Annotations in the OpenMIC dataset are sourced from crowd-sourcing platforms, where annotators assign the presence or absence of an instrument in a snippet, and the dataset indicates the fraction of annotators who confirmed the presence of an instrument for that snippet. For binary classification purposes, we applied a threshold of 0.5, converting relevance scores to binary labels.

3.2. MoisesDB

For evaluating the methodology of identifying instruments on full-length songs, we utilized a modified version of the MoisesDB dataset. The original dataset contains 240 full-length polyphonic songs, annotated for 75 different instruments, making it highly suitable for our experiments.

The MoisesDB dataset does not have the same instrument annotations as the OpenMIC dataset used to train our MLP model. Therefore, we created a mapping to convert the predictions from the 20 OpenMIC classes and the 75 MoisesDB classes to 9 unified instrument classes. This mapping was based on the parent class of each instrument in MoisesDB, and the 9 unified classes were chosen to align with our desired use cases. Table 1 shows the result of this mapping.

By consolidating diverse instruments into broader categories such as "Wind" and "Others," we risk losing important nuances that differentiate the individual instruments. Distinctive timbral characteristics may be overlooked, affecting the model's ability to accurately classify instruments within these broad categories.

Table 1. Mapping for instrument conversion between datasets.

Proposed Classes	OpenMIC Classes
Bass	Bass
Drums	Cymbals, Drums
Guitar	Guitar
Voice	Voice
Piano	Piano
Keys	Organ, Synthesizer
Wind	Trombone, Trumpet, Saxophone, Clarinet, Flute
Strings	Violin, Cello
Others	Banjo, Ukulele, Mandolin, Accordion, Mallet Percussion

Our goal was to accurately validate how well each instrument was detected. Using the original MoisesDB dataset, where over 94% of the songs contain bass, guitar, drums, and vocals, would not provide sufficient insight into the model's performance in the absence of these instruments, since identifying false positives is crucial for determining the model's real-world applicability.

To address this, we utilized MoisesDB's isolated stems to remove instruments from specific songs, ensuring a sufficient number of false cases for each class. This method enabled us to accurately evaluate the model's ability to detect each instrument. Table 2 shows the number of occurrences for each class in the modified dataset.

Table 2. Number of occurrences of each class in the 240 songs of the modified dataset.

Classes	Drums	Bass	Guitar	Voice	Piano	Keys	Others	Strings	Winds
# of occurr.	161	120	120	120	110	103	101	43	22

3.3. Embedding Models and Temporal Support

To generate audio embeddings, in order to train the MLP, we employed three open-source foundational models and compared the effectiveness of using them to train our neural network:

- MusicFM [22], a foundational model that uses random tokenization and self-supervised learning to create robust audio embeddings, tailored to various downstream MIR tasks, by capturing both token-level details and sequence-level patterns;
- PaSST [23], a transformer model that applies the patchout technique to selectively omit portions of audio spectrogram inputs during training, thereby reducing computational demands and improving the model's ability to generate robust embeddings for downstream tasks;
- BEATs [24], an audio pre-training model that employs an iterative framework with acoustic tokenizers to extract semantic-rich embeddings, improving performance across diverse audio classification tasks by prioritizing high-level semantic abstraction over low-level feature reconstruction.

Previous research [18] has shown that subdividing audio into smaller segments (called the *Temporal Support - TS*) before embedding can improve the performance of downstream tasks, such as instrument recognition. Inspired by these results, we experimented $TS \in \{1, 2, 3, 5, 10\}$ seconds, TS being the duration of audio input considered to extract an embedding. Each 10-second audio snippet is then divided into T fragments, where $T = 10/TS$, i.e., for $TS = 5$ we have two fragments. Each fragment is subsequently embedded using the foundational models, resulting in the two embedded fragments $\{\mathbf{x}_1, \mathbf{x}_2\}$. In another case, if $TS = 2$, for example, we would have five embedded fragments as a result $\{\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4, \mathbf{x}_5\}$. For $TS = 3$, the last second is discarded, using only the first 9 seconds of the snippet.

3.4. Multilayer Perceptron Training

Given that our input consists of polyphonic songs with multiple instrument classes, we framed the problem as multi-label classification. To achieve our goal of predicting instruments from audio embeddings, we designed our MLP as follows: an input layer with E neurons, where E is the embedding size, different for every foundational model, with ReLU activation function. A hidden layer with 256 neurons with ReLU activation. Last, an output layer with C neurons, where C is the number of classes, with sigmoid activation function. These configurations were chosen based on empirical experiments. A simple

neural network was chosen as a classifier because the foundational models (more complex architectures) already extract the audio representations that a simpler model can classify.

We train the MLP using snippets from the OpenMIC dataset. For each snippet, only a sub-set of all instrument classes were annotated. To deal with the issue of these partial annotations, we applied two different training methods for the MLP, to determine which approach would yield better results.

The first method, which we called Partial Loss training, consisted in adapting the loss formulation proposed in section 3.1 of [25]. This approach aims to only take it into account known labels while calculating the loss, with the goal of excluding unknown labels in each instance from the training process.

In terms of notation, we denote by C the number of instruments and N the number of training audio snippets. We denote the training data by $\mathcal{D} = \{(\mathcal{I}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathcal{I}^{(N)}, \mathbf{y}^{(N)})\}$, where $\mathcal{I}^{(i)}$ is the i-th audio snippet, $\mathbf{y}^{(i)} = [y_1^{(i)}, \ldots, y_C^{(i)}] \in \mathcal{Y} \subseteq \{-1, 0, 1\}^C$ is the label vector, and $\mathbf{y} = [\mathbf{y}^{(1)}, \ldots, y^{(N)}] \in \{-1, 0, 1\}^{N \times C}$ is the label matrix for the training set. For the i-th audio snippet and c-th instrument, $y_C^{(i)} = 1$ means the instrument is present with certainty, and $y_C^{(i)} = -1$ and $y_C^{(i)} = 0$ mean the instrument is absent with certainty or its presence is unknown, respectively. Finally, f denotes our MLP model and $\bar{\mathbf{s}}^{(i)} = [\bar{s}_1^{(i)}, \ldots, \bar{s}_C^{(i)}] \in R^C$ is the average of the logits of the model across all T fragments of audio snippet $\mathcal{I}^{(i)}$, i.e., $\bar{s}_c^{(i)} = \frac{1}{T} \sum_{t=1}^{T} f_c(\mathbf{x}_t)$, where $f_c(\mathbf{x}_t)$ is the c-th logit of model f for the t-th audio fragment. The partial-BCE loss is then given by Equation (1).

$$\mathcal{L}(\bar{\mathbf{s}}, \mathbf{y}) = \frac{g(p_{\mathbf{y}})}{C} \sum_{c=1}^{C} \left[1_{[y_c=1]} \log \left(\frac{1}{1 + \exp(-\bar{s}_c)} \right) + 1_{[y_c=-1]} \log \left(\frac{\exp(-\bar{s}_c)}{1 + \exp(-\bar{s}_c)} \right) \right], \tag{1}$$

where $p_{\mathbf{y}} \in [0, 1]$ is the proportion of known labels and $g(p_{\mathbf{y}}) = \alpha p_{\mathbf{y}}^{\gamma} + \beta$ is a normalization function with hyperparameters α, β and γ. For our experiments we used the same hyperparameters values as in [25], i.e., $\alpha = -4.45$, $\beta = 5.45$, and $\gamma = -1$.

We also employed a different approach for training for the model, which we called Yes and No Training. Each instrument was given two classes, Yes-Instrument and No-Instrument, and the truth for each class was the definitive presence (in the Yes-Instrument class) or the definitive absence (in the No-Instrument class), and the false values were when we did not have information for that instrument. After making a ground truth value according to this, the training was conducted using Binary Cross Entropy (BCE) as the loss function, with twice as many classes as the previous method. Since the goal of the project is to correctly identify instruments, only the Yes-Instrument classes were taken into consideration for testing, and only for the instruments which we had full information in the ground truth.

After conducting empirical tests with batch sizes of 8, 16, 32, 64, and 128, as well as learning rates of 10^{-2}, 5×10^{-3}, 10^{-3}, 5×10^{-4} and 10^{-4}, we selected the Adam

optimizer with a learning rate of 10^{-3} and a batch size of 128. This configuration yielded the best results, and we utilized early stopping with a patience of 3 epochs since the validation loss did not decrease after this point.

In total, we trained 30 models (three embedding models across five temporal supports, with two different training methods each) and evaluated their performance. Each configuration was trained five times, and their average performance on the test dataset was recorded. We also experimented with eight other Multilayer Perceptron configurations (adding one or two layers, removing a layer, decreasing the number of neurons, increasing number of neurons, using dropout, using batch normalization and changing the activation function), but the difference in results was not significant, so the original network was chosen for complete testing and for the aggregation methods.

On Figure 1, we observe that the training process was efficient, with the optimal validation loss achieved after just 5 epochs. Notably, the validation loss stabilized close to its best value after the first epoch, indicating a rapid convergence in the MLP's training process. This rapid convergence allowed us to conduct numerous experiments efficiently.

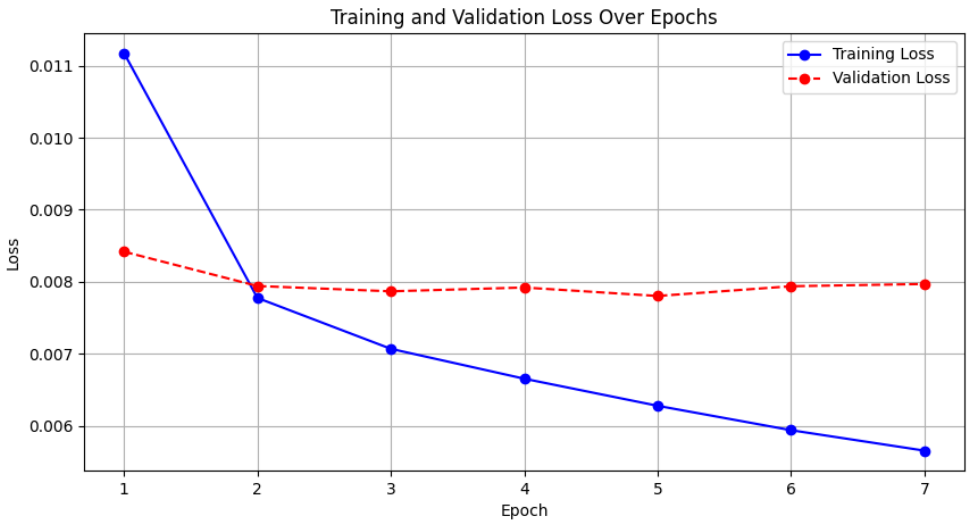

Figure 1. Graph of training and validation loss over epochs.

3.5. Aggregation

To extend our model's instrument detection capabilities from 10-second snippets to full-length songs, we employed different aggregation strategies. Each song was divided into continuous, non-overlapping 10-second snippets, and predictions were made for each snippet. The following aggregation methods were tested for each instrument class:

1. **Average**: Calculates the mean probability of detecting an instrument across all snippets. This approach allows for evaluating the instrument's presence throughout the entire song.
2. **Max**: Focuses on the highest probability detected among the snippets. This method is particularly effective for highlighting instruments that may only be played briefly within the song.

3. **Harmonic Mean**: Computes the harmonic mean of probabilities from all snippets. This method also averages probabilities but prioritizes consistency, giving more weight to uniformly strong probabilities throughout the song.
4. **Top 3 Average**: Considers the average of the three highest probabilities, ensuring multiple snippet evaluations while emphasizing the most significant parts of the song where the instrument is present.
5. **Third Highest**: Selects the third-highest probability value. It serves a similar purpose to the Max method but reduces the potential distortion from outliers, providing a more balanced view.

These strategies aimed to find the most effective method for each instrument, enhancing the overall prediction accuracy.

3.6. Thresholding

To convert probability outputs into binary predictions of instrument presence, we determined optimal, unique threshold values for each instrument. Various strategies were explored, including fixed thresholds set at values like 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, and 0.5, as well as dynamic thresholds calculated using Precision-Recall and ROC curves.

Ultimately, the method using Precision-Recall Curves was selected as it produced the highest F1 Score, effectively balancing precision and recall in binary predictions. These thresholds were subsequently applied to the aggregated probabilities to produce final binary predictions for each song.

3.7. Full-Length Song Instrument Prediction

The complete process for obtaining instrument predictions from full-length songs is illustrated in Figure 2.

In summary, the song is divided into continuous, non-overlapping 10-second snippets, which are passed into the BEATs models to be converted into embeddings. Then, the MLP model generates predictions for each snippet based on the 20 instrument classes. Predictions for each snippet are then aggregated using multiple methods. To convert predictions from the 20 MLP classes to the 9 new classes, we selected the highest probability value within each group as the value for the corresponding new class, for every prediction made. Finally, the probabilities for the 9 classes are converted into binary predictions using the previously calculated thresholds.

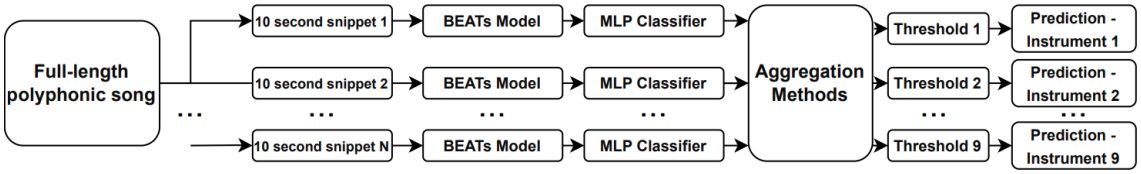

Figure 2. Diagram of pipeline from full-length songs to instrument predictions.

4. Results and Discussion

This section evaluates the model's performance on both 10-second snippets and full-length songs, employing different metrics for specific purposes. We use Mean Average

Precision (mAP) to assess the effectiveness of the model's probability outputs for instrument detection, providing insight into its performance before thresholding. For binary classification assessment after thresholding, we utilize F1 Score, Precision, and Recall. Additionally, we compare our results to those of previous studies.

4.1. Training Methods Results

Upon evaluating the Partial Loss Training and Yes and No Training methods across various Temporal Support (TS) values, we observed that Partial Loss consistently outperformed Yes and No Training for the PaSST and BEATs models (see Figure 3). Notably, while MusicFM showed better results with Yes and No Training in certain cases, it still underperformed compared to the other two models overall. Based on these findings, we selected Partial Loss Training for subsequent experiments.

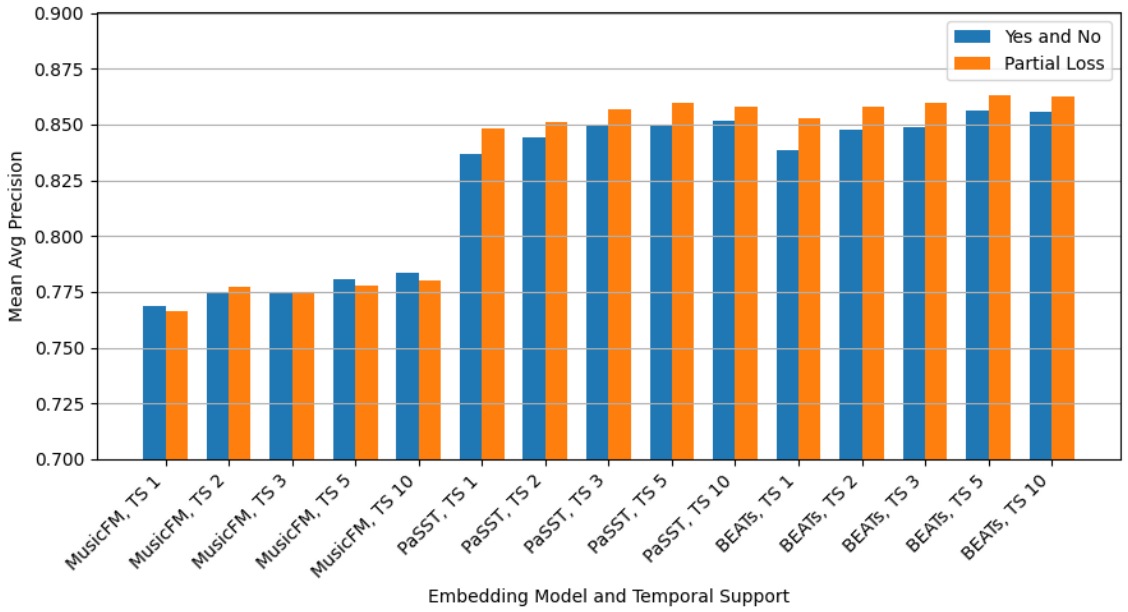

Figure 3. Comparison of Mean Average Precision values for the two Loss Methods, using different Embedding Models and Temporal Supports.

4.2. Results on 10 second snippets

To validate the MLP's capability to detect instruments on 10-second snippets with various TS, we compared our macro-averaged mean average precision (mAP) values for all TS and embedding models with the ones reported on [18], since they also utilize TS and embeddings from foundational models to train their model for classification on OpenMIC. Table 3 demonstrates that our work was able to reproduce their results, confirming the validity of our training pipeline. After analyzing these results, we selected the BEATs model with a TS value of 10 seconds for subsequent steps.

Using this model, we evaluated its performance on the OpenMIC test split by computing thresholds for each instrument from the Precision-Recall curve on the validation set and applying these thresholds to the test set predictions. We evaluated the model's performance both on the 20 classes from the dataset, and on the 5 key instruments we selected for our analysis (voice, guitar, bass, drums and piano), due to their overwhelming

Table 3. Comparison of mAP results between our work and [18].

TS	EM	This Paper	[18]
1s	MusicFM	0.767	
	PaSST	0.847	0.851
	BEATs	0.852	0.852
3s	MusicFM	0.779	
	PaSST	0.86	0.866
	BEATs	0.863	0.862
5s	MusicFM	0.782	
	PaSST	0.864	0.866
	BEATs	0.866	0.869
10s	MusicFM	0.779	
	PaSST	0.861	0.861
	BEATs	0.866	0.866

presence on popular music, and their importance in the structure of a song. The results are shown in Table 4.

Table 4. Macro-averaged results for the model on the 20 instruments of the test set, and on the 5 key instruments.

Instruments	F1 Score	Precision	Recall
All 20	0.806	0.773	0.858
5 Key Instruments	0.91	0.892	0.932

On the OpenMIC test set, [9] achieved an mAP of 0.853, while [13], [14], and [15] reported macro-averaged F1 Scores of 0.81, 0.837, and 0.811, respectively. Although our snippet-level predictions did not surpass the state-of-the-art, our results are closely competitive, suggesting minimal impact on the model's predictions for full-length songs.

4.3. Aggregation Results

With the model's capabilities of detecting instruments on polyphonic songs in 10 seconds snippets verified, we now evaluate how different aggregation methods translate these results to full-length songs. This test was conducted on 80% of the modified MoisesDB dataset, comprising 192 polyphonic songs with 9 classes. The remaining 20% of the dataset is reserved for calculating thresholds specific to each class.

We present the mAP results for each of the five aggregation methods. Additionally, for each instrument class, we identified the best-performing aggregation method, which is the one that achieved the highest mAP on the 20% of the dataset reserved for validation (see Table 5). The mAP metric was chosen to determine the best aggregation method because it does not require binary classification thresholds, which are not available for the validation set used to calculate these thresholds.

Using threshold values derived from the Precision-Recall Curve of 20% of the modified MoisesDB dataset, we further evaluated the model's binary prediction performance for the presence of instruments in full-length songs, our primary goal. We compared how different aggregation methods translated these into binary predictions. These

Table 5. mAP results for the different Aggregation Methods (BMfeC stands for Best Method for each Class)

Aggregation Method	mAP Results
Average	0.791
Max	**0.829**
Harmonic	0.705
Top 3 Average	0.821
Third Highest	0.81
BMfeC	**0.829**

tests were also conducted on 80% of the modified MoisesDB dataset. The results are shown in Table 6.

Table 6. Macro-averaged results for the model on the 9 instrument classes, and on the Top 5 instruments

Classes	Method	F1 Score	Precision	Recall
All 9	Average	0.77	0.764	**0.796**
	Max	0.758	**0.859**	0.73
	Harmonic	0.65	0.672	0.743
	Top 3 Avg	**0.772**	0.812	0.783
	3rd Highest	0.755	0.767	**0.796**
	BMfeC	0.757	0.845	0.735
5 Key Instruments	Average	0.888	0.902	0.877
	Max	**0.914**	**0.924**	0.905
	Harmonic	0.792	0.721	0.887
	Top 3 Avg	**0.914**	0.916	**0.916**
	3rd Highest	0.889	0.871	0.915
	BMfeC	0.913	**0.924**	0.904

On Figure 4, we present the F1 Scores for the top three aggregation methods across individual instruments. The analysis of aggregation methods revealed that Top 3 Averages and Max were the most effective, highlighting the importance of focusing on high-confidence snippets for accurate instrument detection.

We can also observe that the five key instruments — voice, guitar, bass, drums, and piano — exhibited high F1 Scores, indicating that the model is particularly adept at recognizing these prominently featured instruments in popular music. This success can be attributed to their frequent presence in training data and distinct auditory signatures. However, as previously discussed, by consolidating diverse instruments into broader categories such as ”Wind” and ”Others,” the model risks missing important nuances that differentiate individual instruments. This generalization is reflected in the less satisfactory results observed for these categories.

We are unable to directly compare our results with those outlined in [19–21], as their research was evaluated on different datasets, with differing instrument classes from ours.

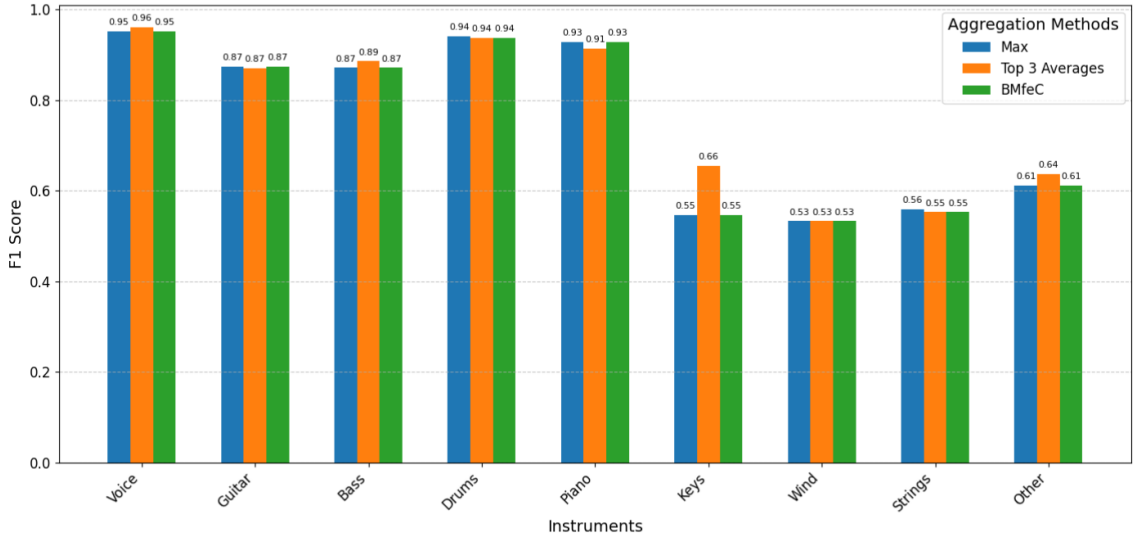

Figure 4. Comparison of Aggregation Methods by Instrument, by F1 Score.

5. Conclusion

In this paper, we developed and evaluated a method for detecting instruments in full-length polyphonic songs. We trained a MLP model to effectively predict instruments in 10-second snippets. We then translated these predictions to full-length songs by utilizing several aggregation methods and calculated thresholds. The model achieved a macro-averaged F1 Score of 0.772 for the nine proposed classes and 0.914 for the top five key instruments using the Top 3 Averages method.

The performance on full-length songs closely matched the results observed for 10-second snippets, with a small increase in performance for the 5 Key Instruments (from a F1 Score of 0.91 on the snippet-level to 0.914 on the song level), demonstrating that our approach effectively scales from snippets to full songs, highlighting its applicability in real-world scenarios.

Different aggregation methods produced varying results across instruments, with distinct methods proving most effective for different instruments. Notably, the Top 3 Averages and Max strategies emerged as the most effective overall. This highlights the benefits of using multiple aggregation methods together.

The developed methods offer substantial opportunities to improve music analysis systems. They can improve recommendation engines and automatic music categorization by providing detailed instrumentation data, allowing for personalized song suggestions and more precise classification. Additionally, by automating dataset tagging with instrumental information, these methods can support various MIR tasks such as genre classification, mood detection, and music similarity analysis, thereby increasing the value and utility of datasets for research and model development.

Our analysis showed that instruments with greater dataset representation yielded better results than those that were less represented. To improve overall performance, future work should focus on increasing the training data for underrepresented instruments, achievable through data augmentation techniques or by sourcing new songs to create a

more balanced dataset.

Additionally, using a wider selection of instruments for training and testing could address problems linked to broad classes like ”Wind” and ”Others.” Implementing a hierarchical classification method may offer improvements in managing the complexity of these categories. Finally, conducting experiments that incorporate music genre information can assess the model’s performance across different styles, highlighting specific areas for improvement.

References

[1] Andrew Wise, Anthony S. Maida, and Ashok Kumar. Attention augmented cnns for musical instrument identification. In *2021 29th European Signal Processing Conference (EUSIPCO)*, pages 376–380, 2021.

[2] Arindam Dutta, Dibakar Sil, Aniruddha Chandra, and Sarbani Palit. Cnn based musical instrument identification using time-frequency localized features. *Internet Technology Letters*, 5(1):e191, 2022.

[3] Debdutta Chatterjee, Arindam Dutta, Dibakar Sil, and Aniruddha Chandra. Deep single shot musical instrument identification using scalograms. In *2023 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 386–389, 2023.

[4] Dhananjay Mukhedkar. Polyphonic music instrument detection on weakly labelled data using sequence learning models. Master's thesis, KTH, School of Electrical Engineering and Computer Science (EECS), 2020.

[5] Maciej Blaszke and Bozena Kostek. Musical instrument identification using deep learning approach. *Sensors*, 22:3033, 04 2022.

[6] Maciej Blaszke, Gražina Korvel, and Bożena Kostek. Exploring neural networks for musical instrument identification in polyphonic audio. *IEEE Intelligent Systems*, pages 1–11, 2024.

[7] Lekshmi Chandrika Reghunath and Rajeev Rajan. Transformer-based ensemble method for multiple predominant instruments recognition in polyphonic music. *EURASIP Journal on Audio, Speech, and Music Processing*, 2022(1), 2022.

[8] Sally M Elghamrawy and Shehab Edin Ibrahim. Audio signal processing and musical instrument detection using deep learning techniques. In *2021 9th International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-ECC)*, pages 146–149. IEEE, 2021.

[9] Dading Chong, Helin Wang, Peilin Zhou, and Qingcheng Zeng. Masked spectrogram prediction for self-supervised audio pre-training. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[10] Arun Solanki and Sachin Pandey. Music instrument recognition using deep convolutional neural networks. *International Journal of Information Technology*, 14(3):1659–1668, 2022.

[11] C. R. Lekshmi and Rajeev Rajan. Predominant instrument recognition in polyphonic music using convolutional recurrent neural networks. In Mitsuko Aramaki, Keiji Hirata, Tetsuro Kitahara, Richard Kronland-Martinet, and Sølvi Ystad, editors, *Music in the AI Era*, pages 214–227, Cham, 2023. Springer International Publishing.

[12] Dong Yu, Huiping Duan, Jun Fang, and Bing Zeng. Predominant instrument recognition based on deep neural network with auxiliary classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 28:852–861, 2020.

[13] Siddharth Gururani, Mohit Sharma, and Alexander Lerch. An attention mechanism for musical instrument recognition. *arXiv preprint arXiv:1907.04294*, 2019.

[14] Zhi Zhong, Masato Hirano, Kazuki Shimada, Kazuya Tateishi, Shusuke Takahashi, and Yuki Mitsufuji. An attention-based approach to hierarchical multi-label music instrument classification. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5, 2023.

[15] Yuxian Mo, Jian Hu, Chaonan Bao, and Dawei Xiong. A novel deep learning-based multi-instrument recognition method for polyphonic music. In *2023 3rd International Symposium on Computer Technology and Information Science (ISCTIS)*, pages 1069–1073, 2023.

[16] Eric Humphrey, Simon Durand, and Brian McFee. Openmic-2018: An open data-set for multiple instrument recognition. In *ISMIR*, pages 438–444, 2018.

[17] Igor Pereira, Felipe Araújo, Filip Korzeniowski, and Richard Vogl. Moisesdb: A dataset for source separation beyond 4-stems. *arXiv preprint arXiv:2307.15913*, 2023.

[18] Aurian Quelennec, Michel Olvera, Geoffroy Peeters, and Slim Essid. On the choice of the optimal temporal support for audio classification with pre-trained embeddings. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 976–980. IEEE, 2024.

[19] Siddharth Gururani, Cameron Summers, and Alexander Lerch. Instrument activity detection in polyphonic music using deep neural networks. In *ISMIR*, pages 569–576, 2018.

[20] Peter Li, Jiyuan Qian, and Tian Wang. Automatic instrument recognition in polyphonic music using convolutional neural networks. *arXiv preprint arXiv:1511.05520*, 2015.

[21] Hannes Bradl, Markus Huber, and Franz Pernkopf. Transfer learning using musical/non-musical mixtures for multi-instrument recognition. In *Speech Communication; 15th ITG Conference*, pages 51–55. VDE, 2023.

[22] Minz Won, Yun-Ning Hung, and Duc Le. A foundation model for music informatics. In *ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1226–1230. IEEE, 2024.

[23] Khaled Koutini, Jan Schlüter, Hamid Eghbal-Zadeh, and Gerhard Widmer. Efficient training of audio transformers with patchout. *arXiv preprint arXiv:2110.05069*, 2021.

[24] Sanyuan Chen, Yu Wu, Chengyi Wang, Shujie Liu, Daniel Tompkins, Zhuo Chen, and Furu Wei. Beats: Audio pre-training with acoustic tokenizers. *arXiv preprint arXiv:2212.09058*, 2022.

[25] Thibaut Durand, Nazanin Mehrasa, and Greg Mori. Learning a deep convnet for multi-label classification with partial labels. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 647–657, 2019.