InstruNET: Convolutional-based Model For Instrument Identification in Raw Audio Signals

Daphne Baron
Division of Engineering Science
University of Toronto
1008956934
daphne.baron@mail.utoronto.ca

Amsal Gilani
Division of Engineering Science
University of Toronto
1008983557
amsal.gilani@mail.utoronto.ca

Hanhee Lee
Division of Engineering Science
University of Toronto
1009486441
hanheeeng.lee@mail.utoronto.ca

GitHub Repository

Abstract

In this paper, we present InstruNET, a convolutional neural network (CNN) designed for multi-label instrument classification. Originally, our objective was to develop an end-to-end pipeline that could take a raw, mixed audio track and generate sheet music for each individual instrument. To address this separation task, we experimented with Non-Negative Matrix Factorization (NMF) and a custom Spectrogram U-Net to reconstruct individual spectrograms from the mixed .wav file. However, due to limitations such as poor signal reconstruction, high model complexity, and constrained computational resources, we redefined our project scope to focus on the more foundational problem of detecting instrument presence within mixed audio files. Under this new scope, we curated a high-quality dataset from the BabySlakh dataset by extracting metadata, isolating stems, and organizing them into folders grouped by instrument class. InstruNET was then trained on Mel spectrograms of this refined dataset to perform multi-label classification across 14 classes. Our model significantly outperformed different approaches, including YAMNet (72.1%) and a shallow feed-forward network (58.9%), achieving an accuracy of 82.1%. These results show the effectiveness of convolutional architectures for instrument identification in mixed-instrument music files and lay the groundwork for future work in source separation and transcription.

1 Introduction

1.1 Motivation

The challenge of obtaining high-quality sheet music for different instruments of songs directly from unprocessed and raw audio is inspired by the increasing demand for AI-assisted tools in music production, education, and performance analysis. While there exist numerous models that can take a file of a specific instrument and turn it into sheet music, using note analysis and Fourier transforms, there is still a lack of a direct pipeline from raw mixed-instrument audio files to sheet music for each individual instrument. To create a pipeline like this, three main tasks need to be completed. Firstly, we must accurately identify the instruments within the raw audio file to recognize instruments in various parts of the song. Secondly, we need to divide the raw audio file into individual instrument files. Thirdly, we use pre-existing tools that can be used to properly convert the files into sheet music.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

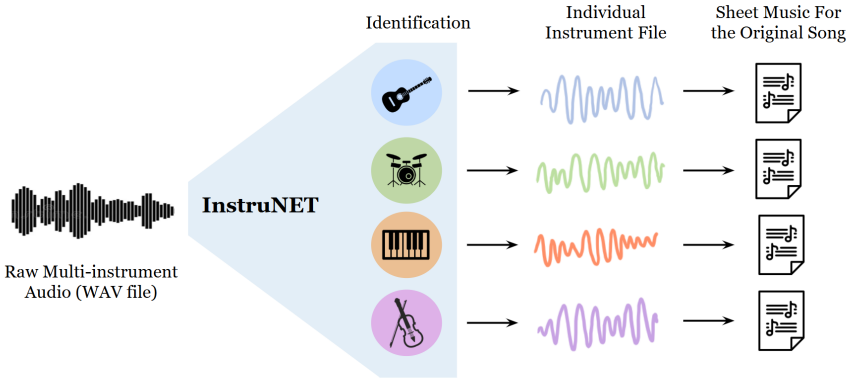

Figure 1: Full pipeline of raw audio to sheet music including the InstruNET task of instrument identification.

1.2 Initial Scope

Initially, we wanted to explore the use of neural networks to both effectively identify and separate individual instruments in a song, generating distinct audio files for each isolated instrument. Once these audio files for each instrument were generated, the goal was to use a pre-existing tool to then create sheet music for educational purposes, shown in Figure 1. However, the scope was narrowed down to focus exclusively on the first task: identifying the instruments from the mixed song file, deferring the separation and transcription for next steps. The re-scoping process is further expanded upon in Section 3 and 4.

2 Data Exploration

2.1 Dataset

BabySlakH contains the first 20 tracks of the open source data set called SlakH. It includes the .wav and MIDI files for each track along with all associated isolated instrument files [1]. For each track, there is also metadata, which includes basic information regarding the instrument classes present, the volume of the tracks and additional information. Moreover, each track in the dataset contains at minimum a piano, bass, drums, and guitar part. From the metadata, we found the average number of instruments per track to be 11. In addition, there is a large range of instrument distributions as shown in Figure 2. We also found that the tracks have an average duration of 243 seconds.

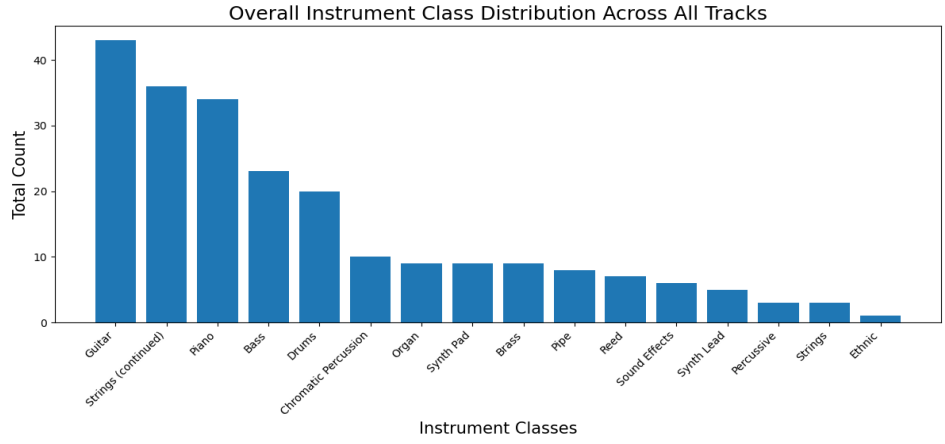

Figure 2: Unique instrument counts in the 20 tracks of BabySlakh.

2.2 UMAP

For further data exploration, we utilized UMAP to visualize the embeddings of individual instruments across the 20 tracks to understand how similar instruments are in a 2-dimensional space. Specifically, we down-sampled the audio, reducing the number of data points per second while preserving useful information. Additionally, we fixed the audio length for each of the instrument stems to standardize the data. From our initial attempts, we noticed equally scattered data points rather than any clustering of similar instruments. This indicated that a simple reduction of dimensionality may not be sufficient.

To combat this, we explored creating a UMAP using Mel-frequency Cepstral Coefficients (MFCC), a commonly used feature representation for audio signals [2]. We believed that this change would help capture the instrument-specific characteristics more robustly. Furthermore, within the MFCC technique of UMAP, we adjusted the parameters such as the number of MFCC coefficients, delta features, frame aggregation, and normalization [3]. However, even with all these modifications, UMAP was not very successful in clustering the data, leading us to utilize other data exploration techniques, such as spectrogram and waveform visualizations to observe and learn from the data.

3 Preliminary Results and Analysis

To target the intial scope of our project our baseline model was NMF. This model was run on our personal laptops with up to 16GB of RAM.

3.1 NMF

For a preliminary trial of splitting a mixed .wav file, an NMF was utilized to take a first shot at isolating instruments and learn more about how a machine learning model can work towards finding a solution to our challenge [4]. The model takes in a spectrogram of a track and uses a Euclidean cost function to learn two matrices that correspond to feature importance and activations.

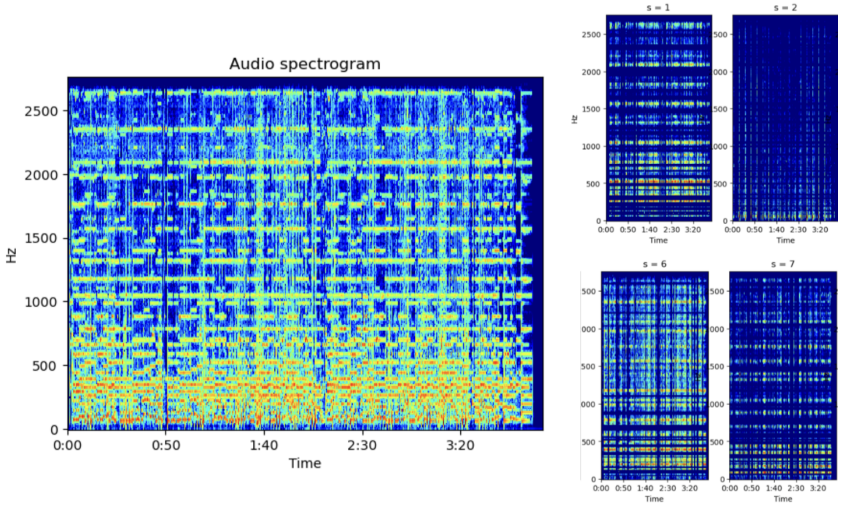

Figure 3: One spectrogram is split into 4 different spectrograms that can be used to obtain individual audio files for each split.

While the NMF was successful in learning features and splitting the spectrogram to individual stems, it was unable to fully capture the effects of a specific instrument sound, as shown on the right side of Figure 3. The resulting stems could be heard to include sounds completely unrelated to the one the stem was capturing. Signal to Distortion Ratios (SDRs) and correlation coefficients were calculated for each stem using the best fitting actual stem [5]. Specifically, since the NMF outputs splits in no specific order, we used the Hungarian algorithm to find the ordering of the outputs that best fits the stems we have. We got negative SDR metrics for each split which show poor performance by the model. This agrees with the findings we had by listening to the reconstructed signals. Additionally,

correlation was near zero, pointing to no linear relationship between the reconstructed stem and the real instrument. A NMF model has limitations due to its additive method of capturing features. In complex cases, when instruments overlap, it is difficult for an NMF to properly recognize the individual features.

3.2 Spectrogram U-Net and WaveNet

We attempted our initial scope of decomposing the spectrogram of a mixed audio track (mix.wav) into its individual instrument stems, using a U-Net based model. Our custom U-Net architecture used three convolutional blocks for both encoding and decoding, with skip connections to retain spatial features across layers. The input of the model was a single-channel normalized spectrogram of the mixed track and has an output of multiple spectrograms, each corresponding to the individual instrument stems. We also used spectrogram normalization using dB scaling. However, the model did not learn as meaningfully as expected based on the spectrograms produced, shown in Figure 4. This was quantitatively evidenced by a high mean absolute error (MAE) of 1.48, which indicates poor predictions. Additionally, attempts to increase the model's complexity by adding parameters led to GPU memory issues, limiting further experimentation. WaveNet is another convolution based model which we attempted for our initial scope but was unsuccessful, due to time constraints [6].

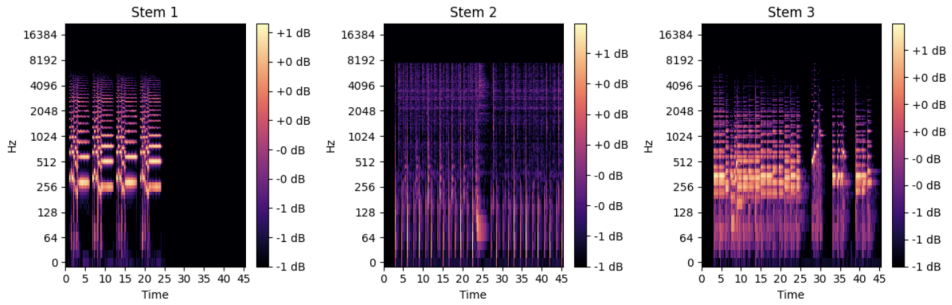

Figure 4: One spectrogram is split into 10 different spectrograms that can be used to obtain individual audio files for each split.

4 Re-scoping

These barriers across methods presented in Section 3 led us to pivot toward a more feasible multi-label classification approach, focusing on detecting instrument presence rather than attempting an end to end signal reconstruction.

5 Comparison Models

5.1 Baseline Model: Shallow Feed Forward Neural Network

Our baseline model performs multi-label instrument classification using a shallow feed forward architecture. It consists of a single fully connected linear layer which is applied to a flattened log-Mel spectrogram input which treats a spectrogram as a 2D vector. This vector is then mapped directly to a vector of logits corresponding to the number of instrument classes. The model uses a binary cross-entropy loss and the Adam optimizer. Other hyperparameters include a learning rate of 0.001 and a batch size of 16. Additionally, the dataset has an 80/20 training and test split. After training for 10 epochs, the model achieved a test accuracy of 58.9%. These results demonstrate that the model is able to capture some structural patterns in the input feature. However, the model's lack of spatial awareness (due to the flattening) and inductive bias limits its ability for generalization. As a result, a convolutional architecture or transformer-based model may improve performance significantly.

5.2 Pre-trained Model: YAMNet

YAMNet is a pre-trained deep CNN developed by Google. It was trained on the AudioSet dataset, which contains over 2 million 10-second audio clips sourced from YouTube [7],[8]. The model

supports classification over 521 audio classes. Although not explicitly designed for instrument identification, YAMNet's embeddings are useful for classification tasks due to the diversity and size of its training data.

For our implementation, instrument stems across BabySlakH's 20 tracks were preprocessed by converting them to mono, 16 kHz .wav format, then normalizing and segmenting them into 2-second clips. Each segment was then used as input to YAMNet, from which embeddings were extracted for every overlapping window. These embeddings formed the basis of a classification pipeline designed to map audio segments to specific instrument labels [9]. The extracted features were used to train a logistic regression model [10]. The results of the model are recorded in the Section 7. While YAMNet embeddings enable classification, there are some limitations to the model. For example, since YAMNet is not fine-tuned for the specific dataset used in this project, there are some suboptimal recognitions of less-represented or similar sounding, more specific, instruments [11].

6 InstruNET

The InstruNET is a CNN used for multilabel instrument identification in songs. It takes in Mel spectrograms as input and outputs labels across 14 instrument classes (13 instruments and an additional 'no_music' class).

6.1 Pre-processing

To prepare the dataset for training our InstruNET model, we began by extracting instrument information from the metadata files associated with each track in the BabySlakh dataset. Originally, the audio data was organized by track, with each track containing multiple isolated instrument stems. Using the metadata, we systematically parsed each file to identify which instruments were present and then reorganized the raw audio data into new folders, grouping the stems by instrument class. After performing an initial data exploration, we identified the five most frequently occurring instruments in the dataset—guitar, strings, piano, bass, and drums—and created an additional folder called combined, which contained various multi-instrument permutations of these five categories. This was crucial for our model to perform well in the multilabel classification task. Furthermore, to avoid sparsity issues during training, we excluded less prominent instruments such as sound effects, percussive, and ethnic, which only appeared in brief or isolated segments of the tracks. This helped create a more balanced dataset which was used for training our InstruNET model.

6.2 Architecture

InstruNET consists of three convolutional layers with increasing channel sizes (32, 64, 128) as can be seen in Figure 5. The increasing channel sizes allow for the network to learn more complex features through the deeper layers. The smaller layers are used to detect simpler patterns, like edges or highlighted regions in the spectrograms, whereas the deeper layers allow for more channels to model more complicated features like instrument timbre, the unique sound quality of an instrument, or harmonic structure. Each of the layers is followed by batch normalization which helps speed up the training time of the model by normalizing layer outputs, therefore reducing covariate shift. Batch normalization also makes the network less sensitive to weight initialization leading to a more stable network. They are also followed by ReLU activation to introduce some nonlinearity into the model's pattern recognition. Max pooling is then used for downsampling, further lowering computation time, and to add translation invariance by preserving the most prominent features of each region.

After the final convolutional block, we use an adaptive average pooling layer, which is flexible for different input dimensions, to reduce the spatial dimensions to 1 by 1. This reduction helps the model focus on a global summary of the input. This is then passed through two fully connected layers with dropout between to prevent overfitting. Lastly, it outputs the probabilities across the 14 instrument classes described above which are used to calculate binary cross entropy logit loss with the one-hot encoding of the true labels of the inputs. Finally, a sigmoid function can be applied to obtain the final instrument labels predicted by the model.

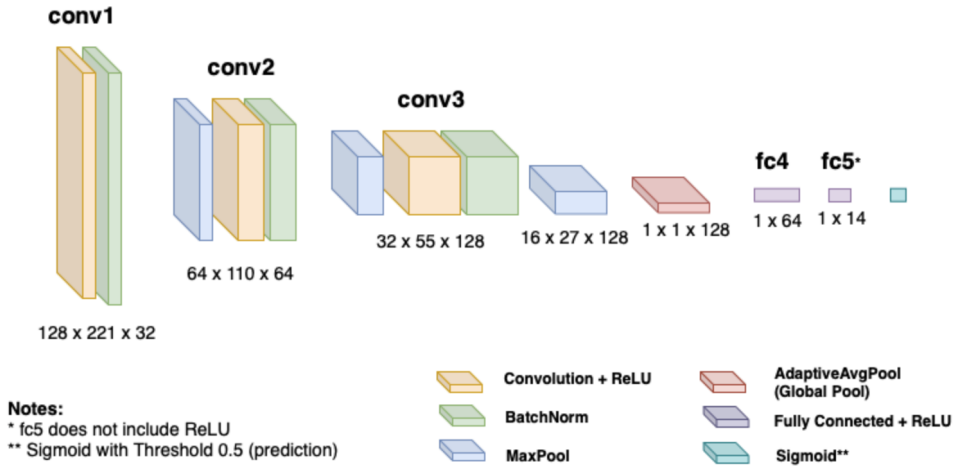

Figure 5: Visual overview of the InstruNET architecture. The final layer outputs probabilities for 14 instrument classes using a sigmoid activation.

6.3 Hyperparameter Optimization

To determine the optimal hyperparameters for InstruNET, we began by fixing the model's architecture as mentioned above. The choice of hidden dimensions, number of layers, and activation function was motivated by balancing model capacity with overfitting risk. Through early experiments, we noticed that increasing the depth beyond three convolutional blocks offered minimal gains while significantly increasing needed compute power.

To further fine-tune the model, we used a structured grid search approach over two hyperparameters: dropout rate and kernel size, as shown in Table 1. We trained each iteration on 5% of the data and 10 epochs to enhance efficiency. We found that a dropout of 30% and a kernel size of 3 consistently offered the best test accuracy. We hypothesize that these hyperparameter choices strike the best balance between regularization and receptive field size. This manual grid search approach was interpretable and effective in our case, since the parameter space was small and well-understood, unlike higher-dimensional spaces that might require Bayesian optimization.

Table 1: Validation accuracy (%) for different combinations of dropout rates and kernel sizes during grid search.

Dropout \ Kernel Size	1	3	5
0.2	59.36	59.21	65.20
0.3	56.29	**77.19**	67.11
0.4	69.59	76.46	65.79

6.4 Early Stopping and Learning Rate Scheduler

The InstruNET training process uses both early stopping and adaptive learning rate scheduling to improve model generalization and training convergence. Specifically, early stopping checks the training and validation accuracy and stops the process if no improvement is seen over 5 consecutive epochs. In addition, the adaptive learning rate scheduling with PyTorch's ReduceLROnPlateau, decreasing the learning rate by a factor of 0.1 after two epochs without validation accuracy improvement. Training begins with an initial learning rate of 0.001, which is gradually lowered to support convergence. Ultimately, these strategies work together to prevent overfitting, improve convergence, and ensure model generalization for the test set.

6.5 Interpretability

To interpret the model's predictions, we generated saliency maps which highlight regions of the spectrogram that are most relevant to the final labels. These visualizations help in interpreting where the model focused to make a classification decision. In Figure 6, we show an example of a saliency map for the piano class. In tracks with multiple instruments, the saliency maps reveal how the model learns to capture important features. For piano, as seen in the figure, the model paid attention to both the low and mid-frequency ranges most, capturing the broad frequency spectrum of piano tones. This indicates that the model was accurately able to recognize not just isolated, prominent features, but also more subtle, context-dependent patterns. Furthermore, since our model is multi-labeling, the model must consider both the presence and absence of different instruments. Even with similar instruments, it was able to accurately identify the unique spectral characteristics of different instruments.

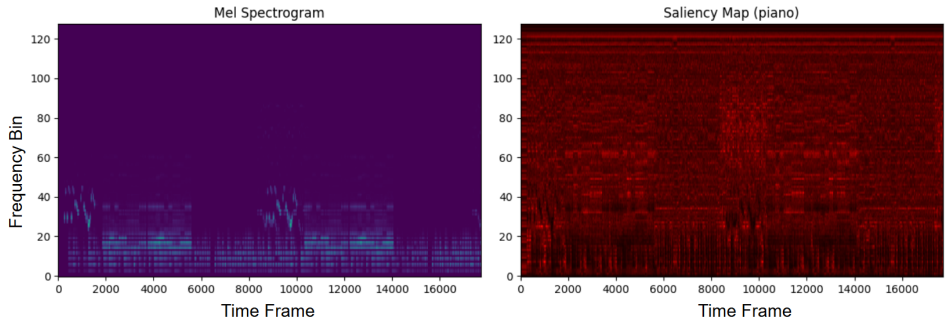

Figure 6: Mel spectrogram of a piano stem (left) and its corresponding saliency map (right).

7 Results

7.1 InstruNET Results

Overall, the model performs strongly across most classes, as indicated by the predominantly green bars representing true positives and true negatives in Figure 7. However, bass, strings, guitar, and piano have higher false positive or false negative counts compared to others. This may be due to overlapping frequency ranges, which make them harder to distinguish in a multi-label setting. In future work, integrating temporal attention to better capture context over time may help improve precision and recall.

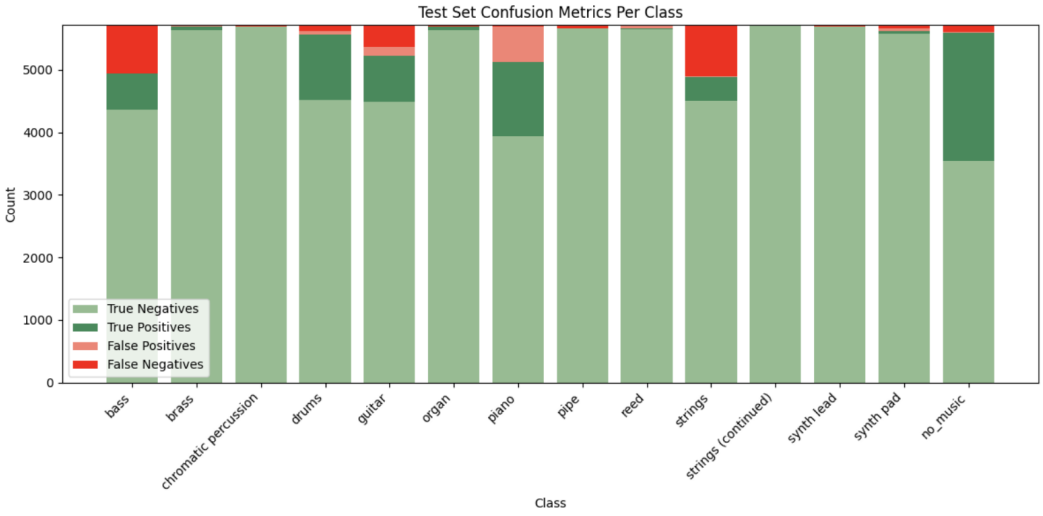

Figure 7: Per-class confusion metrics on the test set for InstruNET.

7.2 Comparison Results

The performance comparison in Table 2 highlights the effectiveness of InstruNET over both the baseline and YAMNet models across all evaluation metrics. InstruNET achieved the highest accuracy at 82.1%, significantly outperforming YAMNet (72.1%) and the baseline (58.9%). It also led in precision (0.848) and recall (0.839), indicating strong performance in both correctly identifying and retrieving relevant instrument labels. These results demonstrate that InstruNET's convolutional architecture is highly effective for multi-label instrument classification.

Table 2: Comparison of weighted metrics across Shallow Feedforward Neural Network (baseline), YAMNet (pre-trained) and InstruNET.

Metric	**Baseline**	**YAMNet**	**InstruNET**
Accuracy	58.9%	72.1%	**82.1%**
F1-score	0.772	0.776	**0.832**
Precision	0.779	0.738	**0.848**
Recall	0.773	0.731	**0.839**

8 Next Steps

Ultimately, the project resulted in a model capable of identifying the instruments in a mixed audio track. As a result, the next step involves doing source separation to isolate each identified instrument into distinct audio files. One way to do so, would be to use a similar technique to Demucs, a state-of-the-art model developed by Facebook AI Research for music source separation tasks [12]. This architecture would involve adding an encoder-decoder structure with bidirectional LSTM layers, enabling it to reconstruct time-domain waveforms with high resolution. Demucs' architecture would be useful for developing a unique model that can handle more of the instruments seen in the Slakh dataset as Demucs can only handle bass, vocals, guitar and others. After obtaining the separated instrument stems using an updated model, each audio file could be converted into MIDI format using the open-source transcription tool Basic Pitch from Spotify [13]. Basic Pitch uses neural networks to transcribe music into MIDI representations. The output from Basic Pitch for each instrument stem can be exported as standard MIDI files. From this, a conventional music notation software such as MuseScore can be used to generate legible sheet music [14]. This three-stage pipeline: instrument identification, source separation, and transcription using Basic Pitch and MuseScore presents a framework that will transform raw mixed instrument audio into sheet music for each instrument. This tool could help greatly in improving the accessibility of music education and allow for musicians to quickly and accurately turn recordings into playable written music scores.

References

[1] E. Manilow, G. Wichern, P. Seetharamanand J. Le Roux, “BabySlakh”. Zenodo, Oct. 20, 2019. Available: 10.5281/zenodo.4603870

[2] M. Rocamora and A. Pardo, "Separation and classification of harmonic sounds for singing voice detection," in *CIARP 2012, LNCS, vol. 7441*, L. Alvarez et al. (Eds.), Springer-Verlag Berlin Heidelberg, 2012, pp. 707–714. Available: https://scispace.com/pdf/separation-and-classification-of-harmonic-sounds-for-singing-52nzzu25x3.pdf

[3] "librosa.feature.mfcc – Mel-frequency cepstral coefficients," Librosa Documentation, 2024. [Online]. Available: https://librosa.org/doc/main/generated/librosa.feature.mfcc.html. [Accessed: Mar. 11, 2025].

[4] ZahraBenslimane, “SoundSourceSeparationusingNMF/Audio-Source-Separation-using-NMF.ipynb at main · ZahraBenslimane/SoundSourceSeparationusingNMF,” GitHub, 2025. https://github.com/ZahraBenslimane/SoundSourceSeparationusingNMF/blob/main/Audio-Source-Separation-using-NMF.ipynb (accessed Mar. 11, 2025).

[5] J. L. Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, “SDR - half-baked or well done?,” arXiv.org, 2018. https://arxiv.org/abs/1811.02508 (accessed Mar. 11, 2025).

[6] A. Van Den Oord et al., “WAVENET: A GENERATIVE MODEL FOR RAW AUDIO.” Available: https://arxiv.org/pdf/1609.03499

[7] Tensorflow, “Models/research/audioset/yamnet at master · Tensorflow/models,” GitHub, https://github.com/tensorflow/models/tree/master/research/audioset/yamnet (accessed Mar. 11, 2025).

[8] “AudioSet,” Google, https://research.google.com/audioset/ (accessed Mar. 11, 2025).

[9] “Sound classification with YAMNet: tensorflow hub,” TensorFlow, https://www.tensorflow.org/hub/tutorials/yamnet (accessed Mar. 11, 2025).

[10] Marl, “Marl/openl3: OpenL3: Open-source deep audio and image embeddings,” GitHub, https://github.com/marl/openl3 (accessed Mar. 11, 2025).

[11] Tensorflow, “Models/research/audioset/vggish/readme.md at master · Tensorflow/models,” GitHub, https://github.com/tensorflow/models/blob/master/research/audioset/vggish/README.md# (accessed Mar. 11, 2025).

[12] S. Rouard, F. Massa, and A. Défossez, “Hybrid transformers for music source separation,” ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5, Jun. 2023. doi:10.1109/icassp49357.2023.10096956

[13] R. M. Bittner, J. J. Bosch, D. Rubinstein, G. Meseguer-Brocal, and S. Ewert, “A lightweight instrument-agnostic model for polyphonic note transcription and multipitch estimation,” ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 781–785, May 2022. doi:10.1109/icassp43922.2022.9746549

[14] Musescore, “MuseScore ,” GitHub, https://github.com/musescore/MuseScore

Note: Usage of AI in the above document was for sentence construction and editing purposes.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Through looking at the abstract, claims were made regarding how existing methods for music separation; however, there is not an accurate pipeline that can take a raw audio and fully split into different instruments and create sheet music from it. This is mentioned in the motivation and scope part of our introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [Yes]

Justification: In the UMAP, there are numerous techniques that could be used as next steps to improve the reduction split. The NMF section also states some limitations regarding the difficulty in splitting sound that has multiple instruments playing at once.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We do not have any theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

Justification: We discussed in the paper the way we adjusted the original models that were found online - included in citation.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Github is on Quercus

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The baseline models that were used did not require training and test details as they were sourced online or pre-trained models.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: There aren't any statistical analyses that are reported

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

Justification: The compute information for the baseline models are included in the baseline models - section 3.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: BabaySlakh is an openly available dataset. We linked it and referenced it on this report. The work we did regarding baseline models were also taken through open-source platforms (ex. Github) and are referenced as well.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The societal impacts are included in the abstract and introduction (motivation section).

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: Yes, when using the YAMNet model, it is trained on AudioSet, which is a dataset from YouTube videos. Even though there may be biases and privacy issues from these video clips, the model produces a classification of 512 predefined, non-harmful labels. As a result, even if there is a malicious user who inputs harmful audio waveforms, the model will still output only instruments, ensuring that it won't cause harmful outputs.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The pre-trained models are cited appropriately in the references section.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: No new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.