JOINTIST: JOINT LEARNING FOR MULTI-INSTRUMENT TRANSCRIPTION AND ITS APPLICATIONS

Kin Wai Cheuk[*1,3] **Keunwoo Choi**[2] **Qiuqiang Kong**[4] **Bochen Li**[4]
Minz Won[4] **Amy Hung**[4] **Ju-Chiang Wang**[4], **Dorien Herremans**[1]
[1] Singapore University of Technology and Design [2] Gaudio Lab
[3] Agency for Science, Technology and Researc, Singapore [4] ByteDance
kinwai_cheuk@mymail.sutd.edu.sg, k@gaudiolab.com

ABSTRACT

In this paper, we introduce Jointist, an instrument-aware multi-instrument framework that is capable of transcribing, recognizing, and separating multiple musical instruments from an audio clip. Jointist consists of the instrument recognition module that conditions the other modules: the transcription module that outputs instrument-specific piano rolls, and the source separation module that utilizes instrument information and transcription results. The instrument conditioning is designed for an explicit multi-instrument functionality while the connection between the transcription and source separation modules is for better transcription performance.

Our challenging problem formulation makes the model highly useful in the real world given that modern popular music typically consists of multiple instruments. However, its novelty necessitates a new perspective on how to evaluate such a model. During the experiment, we assess the model from various aspects, providing a new evaluation perspective for multi-instrument transcription. We also argue that transcription models can be utilized as a preprocessing module for other music analysis tasks. In the experiment on several downstream tasks, the symbolic representation provided by our transcription model turned out to be helpful to spectrograms in solving downbeat detection, chord recognition, and key estimation.

1. INTRODUCTION

Transcription, or automatic music transcription (AMT), is a music analysis task that aims to represent audio recordings as symbolic notations such as scores or MIDI (Musical Instrument Digital Interface) files [1–3]. AMT can play an important role in music information retrieval (MIR) systems since symbolic information – e.g., pitch, duration, and velocity of notes – determines a large part of our musical perception, distinguishing itself from other musical information such as timbre and lyrics. A successful AMT can ease the difficulty of many MIR tasks by providing a denoised version of music in a musically-meaningful, symbolic format. There are examples that include melody extraction [4], chord recognition [5], beat tracking [6], composer classification [7], [8], and emotion classification [9]. Finally, high-quality AMT systems can be used to build large-scale datasets as done in [10]. This would, in turn, accelerate the development of neural network-based music composition systems as these are often trained using symbolic data and hence require large-scale datasets [11–13]. However, music transcription is a tedious task for human and a challenging task even for experienced musicians. As a result, large-scale symbolic datasets of pop music are scarce, impeding the development of MIR systems that are trained using symbolic music representations.

In early research on AMTs, the problem is defined narrowly: transcription is done for a single target instrument, which is usually piano [14] or drums [15], and whereby the input audio only includes that instrument. The limitation of this strong and then-unavoidable assumption is clear: the model would not work for modern pop music, which occupies a majority of the music that people listen to. In other words, to handle realistic use-cases of AMT, it is necessary to develop a multi-instrument transcription system. Recent examples are Omnizart [16, 17] and MT3 [18] which we will discuss in Section 2.1.

In fact, the progress towards multi-instrument transcription has just begun, leaving several challenges related to the development and evaluation of such systems. In particular, the number of instruments in multiple-instrument audio recordings are not fixed. The number of instruments in a pop song may vary from a few to over ten. Therefore, it is limiting to have a model that transcribes a pre-defined fixed number of musical instruments in every music piece. Rather, a model that can adapt to varying number of target instrument(s) would be more robust and useful. This indicates that we may need to consider instrument recognition and instrument-specific behavior, during development as well as evaluation.

Motivated by the aforementioned recent trend and the existing issues, we propose Jointist – a framework that includes instrument recognition, source separation, and transcription. We adopt a joint training scheme to maximize the performance of transcription. First, we choose an ambitious scope of the task by including instrument recognition as a sub-task of the transcription and evaluating the performance of every existing instrument. This thorough evaluation would help researchers to deepen the un-

[*] The author conducted this work as an intern at ByteDance.

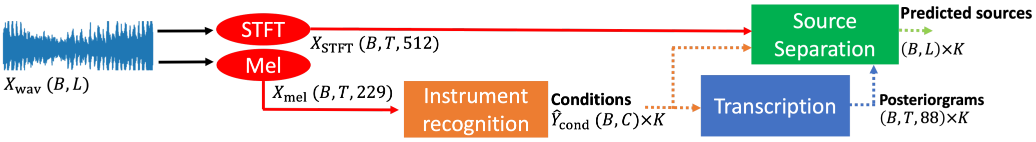

Figure 1. The proposed Jointist framework. Our actual framework can transcribe/separate up to 39 different instruments as defined in Table 7 of Appendix. B: batch size, L: audio length, C: instrument classes, T: number of time steps, K: number of predicted instruments. Dotted lines represent iterative operations for K times. Best viewed in color.

derstanding of the proposed framework – its properties, merits, and limitations. Second, we adopt a fully supervised training scheme and rely on an existing dataset. This means that the target instrument set is determined by the dataset we use. This is a practical choice, although we hope that few-shot and zero-shot learning can alleviate this issue in the future. Third, we provide experiment results that demonstrate the utility of transcription models as a pre-processing module of MIR systems. The result strengthens a perspective of transcription result as a symbolic representation, something distinguished from typical i) audio-based representation (spectrograms) or ii) high-level features [19, 20].

This paper is organized as follows. We first provide a brief overview of the background of automatic music transcription in Section 2. Then we introduce our framework, Jointist, in Section 3. After that, we describe the experimental details in Section 4 and discuss the experimental results in Section 5. We also provide the experiment results and discussion on the application of Jointist in Section 6. Finally, we conclude the paper in Section 7.

2. BACKGROUND

2.1 Multi-Instrument Automatic Music Transcription

While automatic music transcription (AMT) models for piano music are well developed and are able to achieve a high accuracy [2, 13, 21–25], multi-instrument automatic music transcription (MIAMT) is relatively unexplored. MusicNet [26, 27] and ReconVAT [28] are MIAMT systems that transcribe musical instruments other than piano, but their output is a *flat* piano roll that includes notes from all the instruments in a single channel. In other words, they are not instrument-aware. Omnizart [16, 29] is instrument-aware, but it does not scale up well when the number of musical instruments increases as discussed in Section 5.3. MT3 [18] is the current state-of-the-art MIAMT model. It formulates AMT as a sequence prediction task where the sequence consists of tokens of musical note representation. By adopting the structure of an NLP model called T5 [30], MT3 shows that a transformer architecture can perform successful transcription by learning from multiple datasets for various instruments.

2.2 AMT and Joint Learning

There have been attempts to jointly train a transcription model together with a source separation model. For example, [31] uses the F0 estimation to guide the singing voice separation. However, they only demonstrated their method with a monophonic singing track. In this paper, the Jointist framework extends this idea into polyphonic music. While [32] extends the joint transcription and source separation training into polyphonic music, their model is limited to up to five sources (Piano, Guitar, Bass, Drums, and Strings) which do not cover the diversity of real-world popular music; and ignores the instrument-aware adaptation that the Jointist performs. While [33] and [34] also use joint transcription and source separation training for a small number of instruments, they only use transcription as an auxiliary task, and there is no transcription during the inference phases. On the contrary, [35] applies a joint spectrogram and pitchgram clustering method to improve the multi-instrument transcription accuracy, their model is only capable of doing transcription. A zero-shot transcription and separation model was proposed in [36] but was only trained and evaluated on 13 classical instruments.

3. JOINTIST

We designed the proposed framework, Jointist, to jointly train music transcription and source separation modules so as to improve the performance for both tasks. By incorporating an instrument recognition module, our framework is instrument-aware and compatible with up to 39 different instruments.

As illustrated in Figure 1, Jointist consists of three submodules: the instrument recognition module f_{IR}, the transcription module f_{T}, and the music source separation module f_{MSS}. f_{IR} and f_{T} share the same mel spectrogram x_{mel} as the input, while f_{MSS} uses the STFT spectrogram x_{STFT} as the input.

Jointist is a flexible framework that can be trained end-to-end or individually. For ease of evaluation, we trained f_{IR} individually, and we trained f_{T} and f_{MSS} together. In one of our ablation studies (Table 4), we also studied the model performance when training f_{T} and f_{MSS} separately.

3.1 Instrument Recognition Module f_{IR}

Transformer networks [37] have been shown to work well for a wide range of MIR tasks [38–44]. In this paper, we adopt the music tagging transformer proposed in [44] as our musical instrument recognition module, f_{IR}. Similar to [44], our instrument recognition model consists of a convolutional neural network (CNN) front end and a transformer back end. The CNN front end has six convolutional blocks with output channels $[64, 128, 256, 512, 1024, 2048]$. Each convolutional block has two convolutional layers with kernel size (3,3), stride (1,1), and padding (1,1), followed by average pooling (2,2). To prevent overfitting, we perform dropout after each convolutional block with a rate of 0.2 [45]. We conduct an ablation study as shown in Table 1 to find the optimal number of transformer layers, which turns out to be 4. The instrument recognition loss is defined as $L_{\text{IR}} = \text{BCE}(\hat{Y}_{\text{cond}}, Y_{\text{cond}})$ where BCE is the binary cross-entropy, $\hat{Y}_{\text{cond}}$ is a tensor with the predicted instruments, and Y_{cond} is a tensor with the ground truth labels.

3.2 Transcription Module f_{T}

Our transcription module f_{T} is largely inspired by the model design of the onsets-and-frames model proposed in [24]. We modify this module to be conditioned by instrument vectors using FiLM [46], as proposed in [47] for source separation. By performing this modification, we have a model that is capable of transcribing different musical instruments based on a given one-hot vector I^i_{cond} of instrument i defined in Table 7. During inference, we combine the onset rolls and the frame rolls based on the idea proposed in [25] to produce the final transcription. In this procedure, the onset rolls are used to filter out noisy frame predictions, resulting in a cleaner transcription output.

f_{T} consists of a batch normalization layer [48], a frame model $f_{\text{T}}^{\text{frame}}$, and an onset model $f_{\text{T}}^{\text{onset}}$. The outputs from $f_{\text{T}}^{\text{frame}}$ and $f_{\text{T}}^{\text{onset}}$ are concatenated together and then passed to a biGRU layer followed by a fully connected layer (FC) to produce the final transcription $\hat{Y}_{\text{T}}$ [49]. Teacher-forced training is used, i.e. the ground truth instrumental one-hot vectors I^i_{cond} are used during training. During inference, f_{IR} is used to generate the instrumental condition $\hat{Y}_{\text{cond}}$. The transcription loss is defined as $L_{\text{T}} = \sum_{\text{j}} L_j$, where $L_i = \text{BCE}(\hat{Y}_j, Y_j)$ is the binary cross entropy and $j \in \{\text{onset}, \text{frame}\}$.

3.3 Source Separation Module f_{MSS}

We adopt a similar model architecture as in [47,50] for our source separation module f_{MSS}. In addition to the the musical instrument condition I^i_{cond} as the condition (either generated by f_{IR} or provided by human users), our f_{MSS} also uses the predicted piano rolls $\hat{Y}^i_T$ of the music as an extra condition. The source separation loss $L_{\text{MSS}} = \text{L2}(\hat{Y}^i_S, Y^i_S)$ is set as the L2 loss between the predicted source waveform $\hat{Y}^i_S$ and the ground truth source waveform Y^i_S. When combining the transcription output $\hat{Y}^i_{\text{T}}$ with X_{STFT}, we explored two different modes: summation $g(\hat{Y}^i_{\text{T}}) + X_{\text{STFT}}$ and concatenation $g(\hat{Y}^i_{\text{T}}) \oplus X_{\text{STFT}}$. Where g is a linear layer that maps the 88 midi pitches $\hat{Y}^i_{\text{T}}$ to the same dimensions as the X_{STFT}.

4. EXPERIMENTS

4.1 Dataset

Jointist is trained using the Slakh2100 dataset [51]. This dataset is synthesized from part of the Lakh dataset [52] by rendering MIDI files using a high-quality sample-based synthesizer with a sampling rate of 44.1 kHz. The training, validation, and test splits contain 1500, 225, and 375 pieces respectively. The total duration of the Slakh2100 dataset is 145 hours. The number of tracks per piece in Slakh2100 ranges from 4 to 48, with a median number of 9, making it suitable for multiple-instrument AMT. In the dataset, 167 different plugins are used to render the audio recordings.

Each plugin also has a MIDI number assigned to it, hence we can use this information to map 167 different plugins onto 39 different instruments as defined in Table 7 in the Appendix. [1] Our mapping is finer than the MIDI Instrument definition but slightly rougher than MIDI Programs. For example, MIDI numbers 0-3 are different piano types. While MIDI instrument treats all of the them as Piano, the MIDI Program considers them as different instruments. In this case, we believe that the MIDI Program is too fine for our task, hence we follow the MIDI Instrument mapping. However, in another example, MIDI numbers 6 and 7 are harpsichord and clarinet, which sound totally different from the piano. Yet, they are both considered as piano according to the MIDI Instrument. In this case, we follow the MIDI Program and consider them as different instruments from the piano. Finally, the original MIDI numbers only cover 0-127 channels. We add one extra channel (128) to represent drums so that our model can also transcribe this instrument.

4.2 Training

For all of the sub-modules of the Jointist framework, the audio recordings are resampled to 16 kHz, which is high enough to capture the fundamental frequencies as well as some harmonics of the highest pitch C_8 of the piano (4,186 Hz) [24, 25]. Following some conventions on input features [24,25,28], for the instrument recognition and transcription s, we use log mel spectrograms – with a window size of 2,048 samples, a hop size of 160 samples, and 229 mel filter banks. For the source separation, we use STFT with a window size of 1,028, and a hop size of 160. This configuration leads to spectrograms with 100 frames per second. Due to memory constraints, we randomly sample a clip of 10 seconds of audio from the full mix to train our models. The Adam optimizer [53] with a learning rate of 0.001 is used to train both f_{IR} and f_{T}. For f_{MSS}, the learning rate of 1×10^{-4} is chosen after preliminary experiments. All three sub-modules are trained on two GPUs with a batch size of six each. Pytorch and Torchaudio [54,55] are used to perform all the experiment and audio processing.

The overall training objective L is a sum of the losses of the three modules, i.e., $L = L_{\text{IR}} + L_{\text{T}} + L_{\text{MSS}}$.

[1] `https://jointist.github.io/Demo/appendix.pdf`

	mAP		F1	
#Layers (#Parameters)	Macro	Weighted	Macro	Weighted
1 (78.0M)	71.8	91.6	61.5	85.4
2 (78.8M)	72.1	92.1	65.0	86.2
3 (79.6M)	73.7	92.2	63.7	86.9
4 (80.3M)	**77.4**	**92.6**	**70.3**	**87.6**

Table 1. The accuracy of instrument recognition by the number of transformer layers.

	Flat F1		Piece. F1		Inst. F1	
Model	N.	N&O	N	N&O	N	N&O
[29]	26.6	13.4	11.5	6.30	4.30	1.90
[18]	76.0	57.0	N.A.	N.A.	N.A.	N.A.
T	57.9	25.1	59.7	27.2	48.7	23.2
iT	58.0	25.4	59.7	27.2	48.7	23.2
pTS(s)	**58.4**	26.2	61.2	**28.5**	50.6	24.7
ipTS(s)	**58.4**	**26.5**	N.A	N.A	N.A	N.A
TS(s)	47.9	18.6	45.7	18.4	34.9	15.8
pTS(c)	**58.4**	26.3	**61.3**	28.6	**50.8**	**24.8**
TS(c)	47.7	18.6	46.0	18.0	35.5	16.2

Table 2. Transcription accuracy by training methods. 'T' and 'S' specifies the trained modules, e.g., 'T' indicates that only the transcription module is trained (no source separation). The prefix 'p-' represents that the transcription module is pretrained. The prefix 'i-' represents that f_{IR} is used to obtain the instrument conditions. (s) and (c) indicate whether the piano rolls are summed or concatenated, respectively, to the spectrograms.

4.3 Evaluation

We report the F1 scores for f_{IR} with a classification threshold of 0.5 for all of the instruments. We chose this to ensure the simplicity of the experiment, even though the threshold can be tuned for each instrument to further optimize the metric [56]. To understand the model performance under different threshold values, we also report mean average precision (mAP) as shown in Table 1.

For f_{T}, we propose a new instrument-wise metric to better capture the model performance for multi-instrument transcription. Existing literature uses mostly flat metrics or piece-wise evaluation [18,24,25,28]. Although this can provide a general idea of how good the transcription is, it does not show which musical instrument the model is particularly good or bad at. Since frame-wise metrics do not reflect the perceptual transcription accuracy [57], we report only the note-wise (N.) and note-wise with offset (N&O) metrics in Table 2.

For f_{MSS}, we also report the instrument-wise metrics to better understand the model performance (Table 3).

5. DISCUSSION

5.1 Instrument Recognition

Table 1 shows the mAP and F1 scores of the instrument recognition module f_{IR} when different numbers of transformer encoder layers are used. Both the mAP and F1 scores improve as the number of layers increases. The best mAP and F1 scores are achieved when using four transformer encoder layers. Due to the instrument class imbalance in the Slakh2100 dataset, our f_{IR} performance is relatively low for instrument classes with insufficient training samples such as clarinet, violin, or harp. Therefore, the weighted F1/mAP is higher than the macro F1/mAP.[2]. Nevertheless, we believe that our f_{IR} is good enough to generate reliable instrument conditions for popular instruments such as drums, bass, and piano.

Model	Instrument	Piece	Source
S only	1.52	3.24	3.03
T *sum* S	**2.01**	3.72	3.50
T *cat* S	1.92	**3.75**	**3.52**
Upper bound T + S	4.06	4.96	4.81

Table 3. Source-to-Distortion Ratio (SDR) for different models. T represents transcriptor, S represents separator. *sum* and *cat* are different ways to merge the piano rolls with the spectrograms. The 4th row shows the upper bound when we use the ground truth piano rolls for source separation.

Feature merge	Model	Inst.	Piece	Source
sum	T+S	1.86	3.55	3.32
	pTS	2.01	3.72	3.50
	TS (STE)	1.45	3.31	3.10
	pTS (STE)	1.99	3.42	3.24
concat	TS	1.80	3.53	3.31
	pTS	1.92	**3.75**	**3.52**
	TS (STE)	2.01	3.58	3.37
	pTS (STE)	**2.13**	3.66	3.46

Table 4. Results for the ablation study with different feature merging and inferring methods. STE stands for straight-Through estimator; pT indicates that a pretrained weight is used to initialize the transcriber f_T for training.

5.2 Transcription

When evaluating the transcription module f_{T}, we assume that we have a perfect f_{IR} and use the ground truth instrument labels Y_{cond} as the conditions for f_{T}. This is a choice to isolate our evaluation to the transcription module; otherwise, incorrect predictions of f_{IR} will affect the evaluation of f_{T} part.

Table 2 shows the note-wise (N.) and note-wise with offset (N&O) transcription F1 scores for different models. In addition to training only the f_{T} (Model T), we also explore the possibility of training both f_{T} and f_{MSS} jointly (Model T+S) and study its effect on the transcription accuracy.

When training f_{T} standalone for 1,000 epochs, we can achieve an instrument-wise F1 score of 23.2. We presumed that joint training of f_{T} and f_{MSS} would result in better performance as the modules would help each other. Surprisingly, the joint training of f_{T} and f_{MSS} jointly from scratch results in a lower instrument-wise F1 score, 15.8. Only

[2] The F1 score for each instrument is available in the supplementary material, `https://jointist.github.io/Demo/`

when we use a pretrained f_{T} for 500 epochs and then continue training both f_{T} and f_{MSS} jointly, do we get a higher F1 score, 24.7. We hypothesize two reasons for this phenomenon. First, we believe that it is due to the noisy output $\hat{Y}^i_{\text{T}}$ generated from f_{T} in the early stage of joint training which confuses the f_{MSS}. It in turn causes a wrong gradient being backpropagated to the f_{T} model. The same pattern can be observed from the Source-to-Distortion Ratio (SDR) of the f_{MSS} which will be discussed in the next sub-section. Second, multi-task training often results in a lower performance than training a model for a single task, partly due to the difficulty in balancing multiple objectives. Our model might be an example of such a case.

When comparing the conditioning strategy, the difference between the summation and the concatenation modes is very subtle. The summation mode outperforms the concatenation mode by only 0.1 F1 score in terms of piece-wise F1 score; while the concatenation mode outperforms the summation mode by 0.001 in terms of flat F1 as well as the instrument-wise F1. We believe that the model has learned to utilize piano rolls to enhance the mel spectrograms. And therefore, summing both the piano rolls after the linear projection with the Mel spectrograms is enough to achieve this objective, and therefore no obvious improvement is observed when using the concatenation mode.

Compared to the existing methods, our model (24.8) outperforms Omnizart [29] (1.9) by a large margin in terms of Instrument-wise F1. This result proves that adding a control mechanism to alter the model behavior when predicting different musical instruments without changing the model architecture is more scalable than expanding the output channels of the models as the number of musical instruments increases.

Jointist (58.4), however, did not outperform the MT3 model (76.0) on the note-wise F1 score. This difference may come from two aspects. First, MT3 is based on the Transformer architecture, which outperforms earlier architectures such as ConvRNN, on which our model is based. Second, MT3 leverages multiple transcription datasets. These two aspects are independent from the main modification of Jointist (instrument-conditioning and joint training with source separation). Therefore, it would be possible in the future to train a system that has the merits of both of these frameworks. To test the robustness of proposed framework, we also evaluate the performance by using f_{IR} to generate I^i_{cond} (prefix 'I-' in Table 2). We obtain a similar flat F1 scores as our previous experiments, which implies that f_{IR} is good enough to pick up all of the necessary instruments for the transcription. Because false positive piano rolls and false negative piano rolls have undefined F1 scores, we are unable to report the piece-wise and instrument-wise F1 scores. The end-to-end transcription samples generated by Jointist are available in the supplementary material [2].

5.3 Source Separation

Table 3 shows the SDR for different source separation models. "S only" is standalone training of f_{MSS}. the models "T sum S" and "T cat S" have a jointly trained f_{T} and f_{MSS} with $\hat{Y}^i_{\text{T}}$ and X_{STFT} being summed and concatenated respectively. It can be seen that with the help of the transcription module f_{T}, Joinist is able to achieve a higher SDR. Similar to the discussion in Section 5.2, there is only a minor difference in instrument/piece/source SDR between the "sum" mode and "cat" (2.01/3.72/3.50 dB vs. 1.92/3.75/3.52 dB). The SDR for each instrument is available in the supplementary material [2]. Our experimental results in both Section 5.2 and Section 5.3 show that the joint training of f_T and f_{MSS} helps the other modules to escape local minimum and achieve a better performance compared to training them independently. The source separation samples produced by Jointist are available in the supplementary material [2].

We also explore the upper bound of source separation performance when the ground truth Y^i_{T} is used (last row of Table 3). When f_{MSS} has access to an accurate transcription result, its SDR can be greatly improved. This shows that AMT is an important MIR task that could potentially benefit other downstream tasks such as music source separation.

6. APPLICATIONS OF JOINTIST

6.1 Downbeat, Chord, and Key Estimations

It is intuitive to believe that symbolic information is helpful for beat and downbeat tracking, chord estimation, and key estimation. Literature has indicated that the timing of notes is highly related to beats [58]. Downbeats, on the other hand, correspond to bar boundaries and are often accompanied by harmonic changes [59]. The pitch information included in piano rolls offers explicit information about the musical key and chords [60, 61]. It also provides strong cues for modeling downbeats since they are correlated with harmonic changes. Given these insights, we attempt to apply our proposed hybrid representation to improve the performance of these tasks.

For audio processing, we use trained, 6-channel harmonic spectrograms (128 frequency bins) from [62]. We simplify our piano rolls into 2 channel (instrument index 0-37 as channel 0 and index 38 as channel 1). We use a 1-D convolution layer to project the piano rolls into the same frequency dimension as the spectrograms. The hybrid representation is a concatenation of the spectrograms and piano rolls, i.e., 8-channnel.

SpecTNT [39], a strong Transformer architecture, was chosen for modeling temporal musical events in audio recordings [63, 64]. The timestamp and label annotations are converted into temporal activation curves for the learning targets for SpecTNT as done in [39, 63]. Table 9 in Appendix summarizes our SpecTNT configurations for the three tasks.

We train beat and downbeat tracking tasks jointly following [63], but focus on downbeat evaluation, which is

Eval Data	Input	Downbeat (F1)	Chord (MajMin)	Key (Acc)
A	Audio Only	0.728	0.798	0.728
	Hybrid	**0.746**	**0.812**	**0.752**
B	Audio Only	0.620	0.766	-
	Hybrid	**0.663**	**0.785**	-

Table 5. Comparison of with and without pianoroll representation on each task (DB: downbeat). A and B represent the dataset Isophonics and RWC-POP, respectively

more challenging than beat tracking [59, 65]. We consider 24 classes of major and minor triad chords for chord estimation, and 24 classes of major and minor keys for key estimation, plus a "none" class for both tasks. The key and chord tasks are trained and evaluated separately.

For downbeat tracking, we use 7 datasets: Ballroom [66], Hainsworth [67], SMC [68], Simac [69], GTZAN [70], Isophonics [71], and RWC-POP [72]. We use either Isophonics or RWC-POP for evaluation while the remaining 6 datasets are used for training. For chord estimation, we use Billboard [73] [3] , Isophonics, and RWC-POP. We use either Isophonics or RWC-POP for evaluation while the remaining 2 datasets are used for training. For key estimation, we use Isophonics for evaluation and Billboard for training.

Table 5 presents the evaluation results for each task. We report the frame-wise Major/Minor score for chord, F-measure score for downbeat tracking, and the song-level accuracy. We observe that the model with the hybrid representation can consistently outperform that with spectrogram only across three tasks. This can be attributed to the advantage of piano rolls that provide explicit rhythmic and harmonic information to SpecTNT, which is frequency-aware (i.e., not shift invariant along the frequency axis).

6.2 Music Classification

We experimented if the piano roll is useful in music classification. MagnaTagATune dataset [74] is a widely used benchmark in automatic music tagging research. We used ≈21k tracks with top 50 tags following previous works introduced in [75].

Since music classification using symbolic data is a less explored area, we design a new architecture that imitates the music tagging transformer [44]. The size of piano roll input is (B, 39, 2913, 88), where B is the batch size, 39 is the number of instruments, 2913 is the number of time steps, and 88 is the number of MIDI note bins. The CNN front end has 3 convolutional blocks with residual connections, and the back end transformer is identical to the music tagging transformer [44]. The area under the receiver operating characteristic curve (ROC-AUC) and the area under the precision-recall curve (PR-AUC) are reported as evaluation metrics.

As shown in Table 6, the piano roll only model does not outperform the existing audio-based approaches. This is somewhat expected since we lose timbre-related information which is crucial in music tagging. However, after a careful analysis of tag-wise metrics, we conjecture strong dependency of the model on instrument information, which indicates that our hybrid approach in this section is under-optimized. This is based on the following observation.

Input	Model	ROC-AUC	PR-AUC
Audio Only	[76]	0.9106	0.4493
	[77]	0.9058	0.4422
	[75]	0.9129	0.4614
	[62]	0.9127	0.4611
Piano Roll Only	Transformer	0.8938	0.4063
Hybrid	Transformer	0.9090	0.4400

Table 6. Music tagging results on MagnaTagATune Dataset. The audio-only baseline systems are MusiCNN [76], Sample-level CNN [77], Short-chunk ResNet [75], and Harmonic CNN [62].

- The performance is comparable when an instrument tag is one of our 39 instruments (e.g., cello, violin, sitar), or a genre is highly correlated with our 39 instruments (e.g., rock, techno).
- The performance drops when a tag is not related to the 39 instruments (e.g., female vocal, male vocal), or the tag is related to acoustic characteristics that cannot be captured by piano rolls (e.g., quiet)

We further experimented with whether a hybrid model with a mid-level fusion (concatenating the audio embeddings and the piano roll embeddings before the transformer back end) improves the performance. As shown in Table 6, the hybrid model reported performance gain by taking advantage of acoustic features, but could not outperform audio-only models. However, this is a preliminary result and a more thorough experiment and optimization would be required to draw a conclusion.

7. CONCLUSION

In this paper, we introduced Jointist, a framework for transcription, instrument recognition, and source separation. Jointist was designed so that different but related tasks can help each other and improve the overall performance. In the experiments, we showed that jointly trained music transcription and music source separation models are beneficial to each other, leading to a highly practical music transcription model. We also showed that transcription results can be used together with spectrograms to improve the model performance of downbeat tracking, chord, and key estimation. While such a hybrid representation did not improve music tagging, the piano roll alone was enough to produce a decent music tagging result.

In the future, Jointist can be improved in many ways, e.g., replacing the ConvRNN with Transformers as done in [78]. We also hope more attempts to be made to use symbolic representations, complementary to audio representations, for progress towards a more complete music analysis system.

[3] Due to missing audio files for Billboard, we collected them manually from the Internet.

8. REFERENCES

[1] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, "Automatic music transcription: challenges and future directions," *Journal of Intelligent Information Systems*, vol. 41, no. 3, pp. 407–434, 2013.

[2] E. Benetos, S. Dixon, Z. Duan, and S. Ewert, "Automatic music transcription: An overview," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 20–30, 2018.

[3] M. Piszczalski and B. A. Galler, "Automatic music transcription," *Computer Music Journal*, pp. 24–31, 1977.

[4] G. Ozcan, C. Isikhan, and A. Alpkocak, "Melody extraction on midi music files," in *Seventh IEEE International Symposium on Multimedia (ISM'05)*. Ieee, 2005, pp. 8–pp.

[5] Y. Wu and W. Li, "Automatic audio chord recognition with midi-trained deep feature and blstm-crf sequence decoding model," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 27, no. 2, pp. 355–366, 2018.

[6] R. Vogl, M. Dorfer, G. Widmer, and P. Knees, "Drum transcription via joint beat and drum modeling using convolutional recurrent neural networks." in *18th International Society for Music Information Retrieval Conference, ISMIR 2017*, 2017, pp. 150–157.

[7] Q. Kong, K. Choi, and Y. Wang, "Large-scale midi-based composer classification," *arXiv preprint arXiv:2010.14805*, 2020.

[8] S. Kim, H. Lee, S. Park, J. Lee, and K. Choi, "Deep composer classification using symbolic representation," *ISMIR Late Breaking / Demo Session*, 2020.

[9] Y.-H. Chou, I. Chen, C.-J. Chang, J. Ching, Y.-H. Yang *et al.*, "Midibert-piano: Large-scale pre-training for symbolic music understanding," *arXiv preprint arXiv:2107.05223*, 2021.

[10] Q. Kong, B. Li, J. Chen, and Y. Wang, "Giantmidi-piano: A large-scale midi dataset for classical piano music," *arXiv preprint arXiv:2010.07061*, 2020.

[11] G. Brunner, A. Konrad, Y. Wang, and R. Wattenhofer, "Midi-vae: Modeling dynamics and instrumentation of music with applications to style transfer," *arXiv preprint arXiv:1809.07600*, 2018.

[12] X. Wu, C. Wang, and Q. Lei, "Transformer-xl based music generation with multiple sequences of time-valued notes," *arXiv preprint arXiv:2007.07244*, 2020.

[13] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C.-Z. A. Huang, S. Dieleman, E. Elsen, J. Engel, and D. Eck, "Enabling factorized piano music modeling and generation with the maestro dataset," *arXiv preprint arXiv:1810.12247*, 2018.

[14] A. Klapuri and A. Eronen, "Automatic transcription of music," in *Proceedings of the Stockholm Music Acoustics Conference*. Citeseer, 1998, pp. 6–9.

[15] J. Paulus and A. Klapuri, "Model-based event labeling in the transcription of percussive audio signals," in *Proc. Int. Conf. Digital Audio Effects (DAFX)*. Citeseer, 2003, pp. 73–77.

[16] Y.-T. Wu, Y.-J. Luo, T.-P. Chen, I. Wei, J.-Y. Hsu, Y.-C. Chuang, L. Su *et al.*, "Omnizart: A general toolbox for automatic music transcription," *arXiv preprint arXiv:2106.00497*, 2021.

[17] Y.-T. Wu, B. Chen, and L. Su, "Multi-instrument automatic music transcription with self-attention-based instance segmentation," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2796–2809, 2020.

[18] J. Gardner, I. Simon, E. Manilow, C. Hawthorne, and J. Engel, "Mt3: Multi-task multitrack music transcription," *arXiv preprint arXiv:2111.03017*, 2021.

[19] K. Choi, G. Fazekas, M. Sandler, and K. Cho, "Transfer learning for music classification and regression tasks," in *18th International Society for Music Information Retrieval Conference, ISMIR 2017*. International Society for Music Information Retrieval, 2017, pp. 141–149.

[20] R. Castellon, C. Donahue, and P. Liang, "Codified audio language modeling learns useful representations for music information retrieval," *arXiv preprint arXiv:2107.05677*, 2021.

[21] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, pp. 927–939, 2015.

[22] J. W. Kim and J. P. Bello, "Adversarial learning for improved onsets and frames music transcription," *International Society forMusic Information Retrieval Conference*, pp. 670–677, 2019.

[23] R. Kelz, S. Böck, and G. Widmer, "Deep polyphonic adsr piano note transcription," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 246–250.

[24] C. Hawthorne, E. Elsen, J. Song, A. Roberts, I. Simon, C. Raffel, J. Engel, S. Oore, and D. Eck, "Onsets and frames: Dual-objective piano transcription," *arXiv preprint arXiv:1710.11153*, 2017.

[25] Q. Kong, B. Li, X. Song, Y. Wan, and Y. Wang, "High-resolution piano transcription with pedals by regressing onset and offset times," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 29, pp. 3707–3717, 2021.

[26] J. Thickstun, Z. Harchaoui, and S. M. Kakade, "Learning features of music from scratch," in *ICLR*, vol. abs/1611.09827, 2016.

[27] J. Thickstun, Z. Harchaoui, D. P. Foster, and S. M. Kakade, "Invariances and data augmentation for supervised music transcription," *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2241–2245, 2017.

[28] K. W. Cheuk, D. Herremans, and L. Su, "Reconvat: A semi-supervised automatic music transcription framework for low-resource real-world data," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 3918–3926.

[29] Y.-T. Wu, B. Chen, and L. Su, "Polyphonic music transcription with semantic segmentation," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 166–170.

[30] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of Machine Learning Research*, vol. 21, pp. 1–67, 2020.

[31] A. Jansson, R. M. Bittner, S. Ewert, and T. Weyde, "Joint singing voice separation and f0 estimation with deep u-net architectures," in *2019 27th European Signal Processing Conference (EUSIPCO)*, 2019, pp. 1–5.

[32] E. Manilow, P. Seetharaman, and B. Pardo, "Simultaneous separation and transcription of mixtures with multiple polyphonic and percussive instruments," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 771–775.

[33] Y.-N. Hung, G. Wichern, and J. Le Roux, "Transcription is all you need: Learning to separate musical mixtures with score as supervision," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 46–50.

[34] K. Chen, X. Du, B. Zhu, Z. Ma, T. Berg-kirkpatrick, and S. Dubnov, "Zero-shot audio source separation through query-based learning from weakly-labeled data," *arXiv preprint arXiv:2112.07891*, 2021.

[35] K. Tanaka, T. Nakatsuka, R. Nishikimi, K. Yoshii, and S. Morishima, "Multi-instrument music transcription based on deep spherical clustering of spectrograms and pitchgrams," in *International Society for Music Information Retrieval Conference (ISMIR), Montreal, Canada*, 2020.

[36] L. Lin, Q. Kong, J. Jiang, and G. Xia, "A unified model for zero-shot music source separation, transcription and synthesis," in *International Society for Music Information Retrieval (ISMIR)*, 2021.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[38] T. Chen, S. Saxena, L. Li, D. J. Fleet, and G. Hinton, "Pix2seq: A language modeling framework for object detection," *arXiv preprint arXiv:2109.10852*, 2021.

[39] W.-T. Lu, J.-C. Wang, M. Won, K. Choi, and X. Song, "Spectnt: a time-frequency transformer for music audio," *arXiv preprint arXiv:2110.09127*, 2021.

[40] C.-Z. A. Huang, A. Vaswani, J. Uszkoreit, I. Simon, C. Hawthorne, N. Shazeer, A. M. Dai, M. D. Hoffman, M. Dinculescu, and D. Eck, "Music transformer: Generating music with long-term structure," in *International Conference on Learning Representations*, 2018.

[41] J. Park, K. Choi, S. Jeon, D. Kim, and J. Park, "A bi-directional transformer for musical chord recognition," *arXiv preprint arXiv:1907.02698*, 2019.

[42] M. Won, S. Chun, and X. Serra, "Toward interpretable music tagging with self-attention," *arXiv preprint arXiv:1906.04972*, 2019.

[43] R. Guo, I. Simpson, C. Kiefer, T. Magnusson, and D. Herremans, "Musiac: An extensible generative framework for music infilling applications with multi-level control," in *International Conference on Computational Intelligence in Music, Sound, Art and Design (Part of EvoStar)*. Springer, 2022, pp. 341–356.

[44] M. Won, K. Choi, and X. Serra, "Semi-supervised music tagging transformer," *Conference of the International Society for Music Information Retrieval Conference (ISMIR)*, 2021.

[45] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[46] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[47] G. Meseguer-Brocal and G. Peeters, "Conditioned-u-net: Introducing a control mechanism in the u-net for multiple source separations," *arXiv preprint arXiv:1907.01277*, 2019.

[48] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[49] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[50] A. Jansson, E. Humphrey, N. Montecchio, R. Bittner, A. Kumar, and T. Weyde, “Singing voice separation with deep u-net convolutional networks,” *18th International Society for Music Information Retrieval Conference, ISMIR 2017*, 2017.

[51] E. Manilow, G. Wichern, P. Seetharaman, and J. Le Roux, “Cutting music source separation some Slakh: A dataset to study the impact of training data quality and quantity,” in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019.

[52] C. Raffel, *Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching*. Columbia University, 2016.

[53] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2014.

[54] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, 2019.

[55] Y.-Y. Yang, M. Hira, Z. Ni, A. Astafurov, C. Chen, C. Puhrsch, D. Pollack, D. Genzel, D. Greenberg, E. Z. Yang *et al.*, “Torchaudio: Building blocks for audio and speech processing,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6982–6986.

[56] M. Won, J. Spijkervet, and K. Choi, *Music Classification: Beyond Supervised Learning, Towards Real-world Applications*. https://music-classification.github.io/tutorial, 2021. [Online]. Available: https://music-classification.github.io/tutorial

[57] K. W. Cheuk, Y.-J. Luo, E. Benetos, and D. Herremans, “Revisiting the onsets and frames model with additive attention,” in *Proceedings of the International Joint Conference on Neural Networks*. IEEE, 2021, p. In press.

[58] M. F. Matthew E. P. Davies, Sebastian B ock, *Tempo, Beat and Downbeat Estimation*. https://tempobeatdownbeat.github.io/tutorial/intro.html, 2021. [Online]. Available: https://tempobeatdownbeat.github.io/tutorial/intro.html

[59] S. Durand, J. P. Bello, B. David, and G. Richard, “Feature adapted convolutional neural networks for downbeat tracking,” in *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2016, pp. 296–300.

[60] S. Pauws, “Musical key extraction from audio.” in *5th International Society for Music Information Retrieval Conference*, 2004.

[61] E. J. Humphrey and J. P. Bello, “Four timely insights on automatic chord estimation.” in *Proc. ISMIR*, vol. 10, 2015, pp. 673–679.

[62] M. Won, S. Chun, O. Nieto, and X. Serrc, “Data-driven harmonic filters for audio representation learning,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 536–540.

[63] Y.-N. Hung, J.-C. Wang, X. Song, W.-T. Lu, and M. Won, “Modeling beats and downbeats with a time-frequency transformer,” in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 401–405.

[64] J.-C. Wang, Y.-N. Hung, and J. B. Smith, “To catch a chorus, verse, intro, or anything else: Analyzing a song with structural functions,” in *Proc. ICASSP*, 2022, pp. 416–420.

[65] S. Böck and M. E. Davies, “Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation.” in *21th International Society for Music Information Retrieval Conference, ISMIR 2020*, 2020.

[66] F. Krebs, S. Böck, and G. Widmer, “Rhythmic pattern modeling for beat and downbeat tracking in musical audio.” in *ISMIR*. Citeseer, 2013, pp. 227–232.

[67] S. W. Hainsworth and M. D. Macleod, “Particle filtering applied to musical tempo tracking,” *EURASIP Journal on Advances in Signal Processing*, vol. 2004, no. 15, pp. 1–11, 2004.

[68] F. Gouyon, *A computational approach to rhythm description-Audio features for the computation of rhythm periodicity functions and their use in tempo induction and music content processing*. Universitat Pompeu Fabra, 2006.

[69] A. Holzapfel, M. E. Davies, J. R. Zapata, J. L. Oliveira, and F. Gouyon, “Selective sampling for beat tracking evaluation,” *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 9, pp. 2539–2548, 2012.

[70] U. Marchand and G. Peeters, “Swing ratio estimation,” in *Digital Audio Effects 2015 (Dafx15)*, 2015.

[71] M. Mauch, C. Cannam, M. Davies, S. Dixon, C. Harte, S. Kolozali, D. Tidhar, and M. Sandler, “OMRAS2 metadata project 2009,” in *ISMIR Late Breaking and Demo*, 2009.

[72] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, “RWC Music Database: Popular, classical and jazz music databases.” in *Prc. ISMIR*, vol. 2, 2002, pp. 287–288.

[73] J. A. Burgoyne, J. Wild, and I. Fujinaga, “An expert ground truth set for audio chord recognition and music analysis.” in *ISMIR*, vol. 11, 2011, pp. 633–638.

[74] E. Law, K. West, M. I. Mandel, M. Bay, and J. S. Downie, "Evaluation of algorithms using games: The case of music tagging." in *Conference of the International Society for Music Information Retrieval Conference (ISMIR)*, 2009.

[75] M. Won, A. Ferraro, D. Bogdanov, and X. Serra, "Evaluation of cnn-based automatic music tagging models," *arXiv preprint arXiv:2006.00751*, 2020.

[76] J. Pons and X. Serra, "musicnn: Pre-trained convolutional neural networks for music audio tagging," *arXiv preprint arXiv:1909.06654*, 2019.

[77] J. Lee, J. Park, K. L. Kim, and J. Nam, "Sample-level deep convolutional neural networks for music auto-tagging using raw waveforms," *arXiv preprint arXiv:1703.01789*, 2017.

[78] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, "Sequence-to-sequence piano transcription with transformers," *arXiv preprint arXiv:2107.09142*, 2021.

Appendices

MIDI Index	MIDI Instrument	MIDI Program	Our Mapping	Our Index
0-3	Piano	Grand/Bright/Honky-tonk Piano	Piano	0
4-5	Piano	Electric Piano 1-2	Electric Piano	1
6	Piano	Harpsichord	Harpsichord	2
7	Piano	Clavinet	Clavinet	3
8-15	Chr. Percussion	Celesta, Glockenspiel, Music box, Vibraphone, Marimba, Xylophone, Tubular Bells, Dulcimer	Chr. Percussion	4
16-20	Organ	Drawbar, Percussive, Rock, Church, Reed Organ	Organ	5
21	Organ	Accordion	Accordion	6
22	Organ	Harmonica	Harmonica	7
23	Organ	Tango Accordion	Accordion	6
24-25	Guitar	Acoustic Guitar (nylon, steel)	Acoustic Guitar	8
26-31	Guitar	Electric Guitar (jazz, clean, muted, overdriven, distorted, harmonics)	Electric Guitar	9
32-39	Bass	Acoustic/Electric/Slap/Synth Bass	Bass	10
40	Strings	Violin	Violin	11
41	Strings	Viola	Viola	12
42	Strings	Cello	Cello	13
43	Strings	Contrabass	Contrabass	14
44	Strings	Tremolo Strings	Strings	15
45	Strings	Pizzicato Strings	Strings	15
46	Strings	Orchestral Harp	Harp	16
47	Strings	Timpani	Timpani	17
48-51	Ensemble	Acoustic/Synth String Ensemble 1-2	Strings	15
52-54	Ensemble	Aahs/Oohs/Synth Voice	Voice	18
55	Ensemble	Orchestra Hit	Strings	15
56	Brass	Trumpet	Trumpet	19
57	Brass	Trombone	Trombone	20
58	Brass	Tuba	Tuba	21
59	Brass	Muted Trumpet	Trumpet	19
60	Brass	French Horn	Horn	22
61-63	Brass	Acoustic/Synth Brass	Brass	23
64-67	Reed	Soprano, Alto, Tenor, Baritone Sax	Saxophone	24
68	Reed	Oboe	Oboe	25
69	Reed	English Horn	Horn	22
70	Reed	Bassoon	Bassoon	26
71	Reed	Clarinet	Clarinet	27
72	Pipe	Piccolo	Piccolo	28
73	Pipe	Flute	Flute	29
74	Pipe	Recorder	Recorder	30
75-79	Pipe	Pan Flute, Blown bottle, Shakuhachi, Whistle, Ocarina	Pipe	31
80-87	Synth Lead	Lead 1-8	Synth Lead	32
88-95	Synth Pad	Pad 1-8	Synth Pad	33
96-103	Synth Effects	FX 1-8	Synth Effects	34
104-111	Ethnic	Sitar, Banjo, Shamisen, Koto, Kalimba, Bagpipe, Fiddle, Shana	Ethnic	35
112-119	Percussive	Tinkle Bell, Agogo, Steel Drums, Woodblock, Taiko Drum, Melodic Tom, Synth Drum	Percussive	36
120-127	Sound Effects	Guitar Fret Noise, Breath Noise, Seashore, Bird Tweet, Telephone Ring, Helicopter, Applause, Gunshot	Sound Effects	37
128	Drums	Drums	Drums	38

Table 7. The instrument mapping used in our experiments. Our mapping is less detailed than the MIDI Program Number, but it is finer than the MIDI Instrument code, thus resulting in 39 different instruments.

		Full length SDR			**10s SDR**		
feature merge	Model	inst	piece	source	inst	piece	source
sum	T+S (binary)	1.35	3.22	2.97	3.14	5.32	4.31
	T+S (posterior)	1.86	3.55	3.32	3.47	5.60	4.54
	preT+S (binary)	0.20	2.75	2.44	1.86	5.34	4.11
	preT+S (posterior)	2.01	3.72	3.50	**3.69**	5.86	4.81
	T+S+STE (binary)	1.30	3.30	3.06	2.40	5.13	4.16
	T+S+STE (posterior)	1.45	3.31	3.10	2.41	4.96	4.07
	preT+S+STE (binary)	1.67	3.30	3.06	3.29	5.36	4.41
	preT+S+STE (posterior)	1.99	3.42	3.24	3.26	5.17	4.29
concat	T+S (binary)	1.28	3.20	2.95	2.72	5.06	4.20
	T+S (posterior)	1.80	3.53	3.31	3.21	5.50	4.51
	preT+S (binary)	-0.04	2.63	2.31	1.86	5.44	4.09
	preT+S (posterior)	1.92	**3.75**	**3.52**	3.50	**6.03**	**4.90**
	T+S+STE (binary)	1.89	3.60	3.37	3.63	5.70	4.66
	T+S+STE (posterior)	2.01	3.58	3.37	3.59	5.53	4.55
	preT+S+STE (binary)	1.72	3.47	3.22	3.18	5.78	4.61
	preT+S+STE (posterior)	**2.13**	3.66	3.46	1.86	5.44	4.09
spec patch	T+S (binary)	1.74	3.42	3.20	3.22	5.19	4.37
	T+S (posterior)	1.79	3.46	3.24	3.35	5.30	4.43
	preT+S (binary)	0.38	2.72	2.41	1.91	3.60	3.39
	preT+S (posterior)	1.91	3.61	3.40	3.58	5.52	4.65
	T+S+STE (binary)	1.74	3.46	3.24	3.50	5.28	4.42
	T+S+STE (posterior)	1.94	3.48	3.27	3.48	5.15	4.30
	preT+S+STE (binary)	1.71	3.47	3.23	3.28	5.40	4.53
	preT+S+STE (posterior)	2.01	3.58	3.38	3.57	5.30	4.48

Table 8. Full version of Table 4. 'spec patch' represents the model variation where the final learned mask is applied to the summation of the piano roll features and the spectrograms instead of the original spectrograms. We also experimented with two forms of piano rolls: posteriorgram and binary. Posteriorgram provides the probability of the existence of a note in the input audio to the model, which outperforms the model that uses binary version of the piano roll.

Task	Input length	(k, d)	(h_k, h_d)
Downbeat	6 seconds	(128, 96)	(4, 8)
Chord	12 seconds	(64, 256)	(8, 8)
Key	36 seconds	(128, 32)	(8, 4)

Table 9. SpecTNT parameters we used in each task, where k and d denote spectral and temporal feature dimensions; while h_k and h_d represent the number of heads for the spectral and temporal Transformer encoders, respectively.