$\boldsymbol{a} \mapsto \boldsymbol{X}$ maps raw audio to a feature representation more conducive to learning. A melody of length N is a sequence of notes $\boldsymbol{y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_N] \in \mathbb{Y}^N$ consisting of onset-pitch pairs $\boldsymbol{y}_i = (t_i, n_i) \in \mathbb{Y} = \mathbb{R}^+ \times \mathbb{V}$ where $t_i < t_j$ if $i < j$. Given a featurization $\boldsymbol{X}$, the melody transcription task is to construct a transcription algorithm $\texttt{Transcribe} : \mathbb{R}^{Tf_k \times d} \to \mathbb{Y}^N$ such that $\boldsymbol{X} \mapsto \boldsymbol{y}$.

3.1 Evaluation

To evaluate a melody transcription method `Transcribe`, we adopt a standard metric commonly used for evaluation in polyphonic music transcription tasks, namely, "onset-only note-wise F-measure" [36]. This metric scores an estimated transcript $\texttt{Transcribe}(\boldsymbol{X})$ by first matching its note onsets to those in the reference $\boldsymbol{y}$ with 50ms of tolerance (default in [37]), and then computes a standard `F1` score where an estimated note is treated as correct if it is the same pitch as its matched reference note. This "note-wise" metric represents a departure from the "frame-based" metrics typically used to evaluate melody extraction algorithms—Ycart et al. demonstrate in [36] that this particular note-wise metric correlates more strongly with human perception of transcription quality than any other common metric, including frame-based ones.

We make a slight modification to this note-wise metric specific to the melody transcription setting: an estimate $\texttt{Transcribe}(\boldsymbol{X})$ may receive full credit if it is off by a fixed octave shift but otherwise identical to the reference. In downstream settings, melody transcriptions are likely to be used in an octave-invariant fashion, e.g., they may be shifted to read more comfortably in treble clef, or performed by singers with different vocal ranges. Hence, we modify the evaluation criteria by simply taking the highest score over octave shifted versions of the estimate:

$$\max_{\sigma \in \mathbb{Z}} \texttt{F1}(\texttt{OctaveShift}(\texttt{Transcribe}(\boldsymbol{X}), \sigma), \mathbf{y}).$$

Henceforth, we refer to this octave-invariant metric as `F1`.

4. DATASET OVERVIEW

A major obstacle to progress on melody transcription is the lack of a large volume of data for training. To the best of our knowledge, there are only two datasets available with annotations suitable for melody transcription: the RWC Music Database [38–40] (RWC-MDB), and a dataset labeled by Laaksonen [26]. The former is larger but the annotations are inconsistent—Ryynänen and Klapuri note that only 8.7 hours (130 songs) are usable for melody transcription [24], while the latter only contains 1.5 hours.

We derive a suitably large dataset for melody transcription using crowdsourced annotations from HookTheory.[5] HookTheory is a platform where users can easily create and share musical analyses of particular recordings hosted on YouTube, with Wikipedia-style editing. The dataset contains annotations for 22k segments of 13k unique recordings totaling 50 hours of labeled audio. The audio content covers a wide range of genres—there is a skew towards pop and rock but many other genres are represented including EDM, jazz, and even classical. We create an artist-stratified 8:1:1 split of the dataset for training, validation, and testing. The dataset also includes chord annotations which may facilitate chord recognition research.

While HookTheory data has been used previously for MIR tasks like harmonization [41, 42], chord recognition [43], and representation learning [44], making use of this platform for MIR is currently cumbersome. One obstacle is that the annotations are created via a "functional" interface, i.e., one which uses scale degrees and roman numerals relative to a key signature instead of absolute notes and chord names. In contrast, most MIR research favors absolute labels. Hence, we convert annotations from this functional format to a simple (JSON-based) absolute format. One caveat is that the HookTheory annotation interface uses a relative octave system, so there is no way to reliably map annotations to a ground truth octave. Thus, melodies in our dataset also contain only relative octave information, consistent with the octave-invariant evaluation proposed in Section 3.1.

5. METHODS

Similar to state-of-the-art methodology used for polyphonic transcription [45], our approach to melody transcription involves training Transformer models [14] to predict notes from audio features. However, to address the unique challenges of melody transcription, our approach differs in two distinct ways. First, because melody transcription involves operating on broad audio, we leverage representations from pre-trained models as drop-in replacements for the handcrafted spectrogram features used as inputs to other transcription systems. Secondly, because alignments in our dataset are approximate, we propose a new strategy for training transcription models under such conditions.

5.1 Pre-trained representations

We explore representations from two different pre-trained models for use as input features to transcription models. In [17], Castellon et al. demonstrate that representations from Jukebox [1]—a generative model of music audio pre-trained on 1M songs—constitute effective features for many MIR tasks, though notably they do not experiment on transcription. We adopt their approach to extract features from Jukebox ($f_k \approx 345$ Hz, $d = 4800$), though we use a deeper layer (53) than their default (36) which improved transcription performance in our initial experiments.

We also explore features from MT3 [35], an encoder-decoder transcription model pre-trained on a multitude of different transcription tasks (though not melody transcription). For this model, we use the encoder's outputs as features ($f_k = 125$ Hz, $d = 512$). The two models have different trade-offs with respect to our setting: Jukebox was pre-trained on audio similar to that found in our dataset but in a generative fashion, whereas MT3 is pre-trained on transcription but for different audio domains.

[5] HookTheory annotations are published under a CC BY-NC-SA 3.0 license, which our dataset inherits.

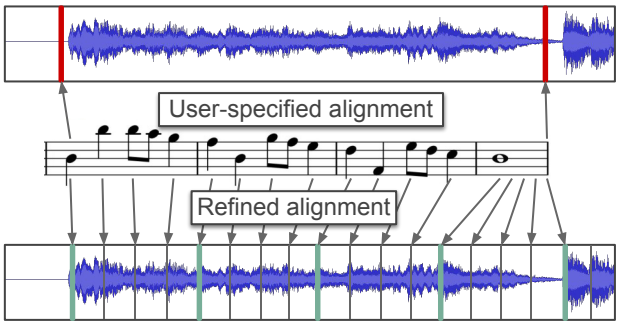

Figure 2. We refine the crude user-specified alignments from HookTheory by using beat and downbeat tracking. The first segment beat is mapped to the detected downbeat nearest to the user-specified starting timestamp, and remaining beats are mapped to subsequent detected beats.

5.2 Refined Alignments

The alignments between audio and HookTheory annotations are crude—users provide only an approximate starting and ending timestamp of their annotated segment within the audio. Because transcription methodology generally depends on precise alignments, we make an effort to refine the user-specified ones. To this end, we make use of the beat and downbeat detection algorithm from `madmom` [12, 13]. Specifically, our approach aligns the first beat of the segment to the detected downbeat which is nearest to the user-specified starting timestamp. Then, we align the remaining beats to the subsequent detected beats (see Figure 2 for an example). This provides a beat-level alignment for the entire segment, which we linearly interpolate to fractional subdivisions of the beat. Formally, we construct an alignment function $\texttt{Align} : [0, B) \to [0, T)$ that assigns each of B beats in the metrical structure to a time $t \in [0, T)$ in the audio. In an informal listening test, this produced an improved alignment for 95 of 100 segments, where the primary failure mode in the remaining 5 segments occurred when `madmom` detected the wrong beat as the downbeat. We use these refined alignments for training and evaluation and release them alongside the dataset.

5.3 Beat-wise resampling

Here we outline our approach for training transcription models in the presence of imprecise alignments. Existing transcription methods were largely designed for domains where perfect alignments are readily available, e.g., piano transcription data captured by a Disklavier. Despite our best efforts, the refined HookTheory alignments are still imprecise when compared to alignments in the datasets used to develop existing methods. Consequently, in initial experiments, we found that naively adopting existing methods (specifically, [16, 45]) resulted in poor performance on our dataset and task. Additionally, initial experiments on training models with an alignment-free approach [46] also resulted in poor performance.

Accordingly, to sidestep small alignment deviations, we perform a beat-wise resampling of audio features $\boldsymbol{X} \in \mathbb{R}^{Tf_k \times d}$ to yield features that are uniformly spaced in subdivisions of the beat (using `Align`—see Section 5.2) rather than in time. For an audio recording with B beats, we sample features $\tilde{\boldsymbol{X}} \in \mathbb{R}^{4B \times d}$ at sixteenth-note intervals. The value $\tilde{\boldsymbol{X}}_i$ is constructed by averaging all feature vectors in $\boldsymbol{X}$ that are nearest to the i'th sixteenth note into a single vector which acts as a proxy feature. For example, if a recording has a tempo of 120 BPM, a sixteenth note represents 125 ms of time, which would entail averaging across 43 feature vectors from Jukebox ($f_k \approx 345$ Hz). The intuition is that, while our alignments may not be precise enough to identify which of those 43 frames contains an onset, we can be reasonably confident that it occurs *somewhere* within them, and thus the relevant frame will be incorporated into the proxy. A similar approach was previously explored for song structure analysis in [47].

5.4 Modeling

Together with the beat-wise resampling $\tilde{\boldsymbol{X}} \in \mathbb{R}^{4B \times d}$, we convert the sparse task labels $\boldsymbol{y} \in (\mathbb{R}^+ \times \mathbb{V})^N$ into a dense sequence $\tilde{\boldsymbol{y}} \in \{\{\varnothing\} \cup \mathbb{V}\}^{4B}$, which indicates whether or not an onset occurs at each sixteenth note.[6] Formally,

$$\tilde{\boldsymbol{y}}_i = \begin{cases} n_j & \text{if } \texttt{Align}(\frac{i}{4}) = t_j \text{ for some note } \boldsymbol{y}_j, \\ \varnothing & \text{otherwise.} \end{cases}$$

We formulate melody transcription as an aligned sequence-to-sequence modeling problem and attempt to predict the sequence $\tilde{\boldsymbol{y}}$ given $\tilde{\boldsymbol{X}}$. Specifically, we train models of the form $f_\theta : \mathbb{R}^{4B \times d} \to \mathbb{R}^{4B \times (|\mathbb{V}|+1)}$, which parameterize probability distributions $p_\theta(\tilde{\boldsymbol{y}}_i | \tilde{\boldsymbol{X}}) = \texttt{SoftMax}(f_\theta(\tilde{\boldsymbol{X}})_i)$ over elements of the sequence $\tilde{\boldsymbol{y}}$. One unique aspect of our dataset is that absolute octave information is absent (see Section 4). Hence, we construct an octave-tolerant cross-entropy loss by identifying the octave shift amount that minimizes the standard cross-entropy loss (denoted `CE`) when applied to the labels:

$$\min_{\sigma \in \mathbb{Z}} \sum_{i=0}^{4B-1} \texttt{CE}(p_\theta(\tilde{\boldsymbol{y}}_i | \tilde{\boldsymbol{X}}), \texttt{OctaveShift}(\tilde{\boldsymbol{y}}_i, \sigma)).$$

We require a thresholding scheme to convert the dense sequence of soft probability estimates $p_\theta(\tilde{\boldsymbol{y}}_i | \tilde{\boldsymbol{X}})$ into a sparse sequence of notes required by our task (see Section 3). Given a threshold $\tau \in \mathbb{R}$ (in practice, tuned on validation data), we define a sorted *onset list*

$$\mathcal{I} = \texttt{Sort}(\{i \in \{0, \ldots, 4B-1\} : p_\theta(\tilde{\boldsymbol{y}}_i = \varnothing | \tilde{\boldsymbol{X}}) < \tau\}).$$

This should be interpreted as a list of N metrical positions where an onset likely occurs. The timings of these onsets are given by the alignment, and we will predict the note-value with the highest probability. The sparse melody transcription is thus defined for $j = 1, \ldots, N$ by

$$\texttt{Transcribe}(\tilde{\boldsymbol{X}})_j = (t_j, n_j), \text{ where}$$
$$t_j = \texttt{Align}\left(\frac{\mathcal{I}_j}{4}\right),$$
$$n_j = \arg\max_{v \in \mathbb{V}} p_\theta(\tilde{\boldsymbol{y}}_{\mathcal{I}_j} = v | \tilde{\boldsymbol{X}}).$$

[6] This requires quantizing labels to the nearest sixteenth note. In practice, less than 1% of notes in our dataset are affected by this quantization.

Features	d	F1
Mel	229	0.514
MT3	512	0.550
Jukebox	4800	**0.615**
Mel, MT3	741	0.548
Mel, Jukebox	5029	0.617
MT3, Jukebox	5312	0.622
Mel, MT3, Jukebox	5541	**0.623**

Table 1. HookTheory test set performance for Transformers trained with different features (top) and combinations (bottom). Features are complementary—combining all three yields highest performance—but marginally so compared to Jukebox alone.

Approach	F1 (All)	F1 (Vocal)
MT3 Zero-shot [35]	0.133	0.085
Melodia [48] + Segmentation	0.201	0.268
Spleeter [31] + Tony [28]	0.341	**0.462**
DSP + HMM [24]	**0.420**	0.381
Mel + Transformer	0.631	0.621
MT3 + Transformer	0.701	0.659
Jukebox + Transformer	**0.744**	**0.786**

Table 2. Performance of different approaches on a subset of RWC-MDB [38–40]. The bottom three approaches were trained on the HookTheory dataset. For fair comparison to vocal transcription baselines, we also separately report performance on the vocal portions of this dataset.

6. EXPERIMENTS

Here we describe our experimental protocol for training melody transcription models on the HookTheory dataset. The purpose of these experiments is two-fold. First, we compare representations from different pre-trained models to handcrafted spectrogram features to determine if pre-training is helpful for the task of melody transcription (Section 6.1). Second, we compare our trained models holistically to other melody transcription baselines (Section 6.2).

All transcription models are encoder-only Transformers with the default hyperparameters from [14], except that we reduce the number of layers from 6 to 4 to allow models to be trained on GPUs with 12GB of memory. During training, we select random slices from the annotated segments of up to 96 beats or 24 seconds in length (whichever is shorter). We train using our proposed loss function from Section 5.4 and perform early stopping based on max F1 score across thresholds τ on the validation set, using the best validation τ for testing. All models converge within 15k steps or about a day on a single K40 GPU.

6.1 Comparing input features

We compare representations from Jukebox [1] and MT3 [35] (see Section 5.1) to handcrafted spectrogram features, which are commonly used by existing transcription methods. Specifically, we compare to log-amplitude Mel spectrograms using the formulation from [16] ($f_k \approx 31$, $d = 229$). Because features may contain complementary information, we also experiment with all combinations of these three features. Note that our beat-wise resampling strategy allows for trivial combination of these features (by concatenation) despite their differing rates. In Table 1, we report F1 (as described in Section 3.1) on the HookTheory test set for all input features.

Overall, using representations from Jukebox as input features results in stronger melody transcription performance than using either representations from MT3 or conventional handcrafted features. Representations from both MT3 and Jukebox outperform conventional handcrafted features, implying that both pre-training strategies are helpful for melody transcription. Note that these two pre-training approaches are compared holistically—these models differ on several axes (number of parameters, pre-training data semantics, pre-training task), and thus it is impossible to disentangle the individual contributions of these different factors without retraining the models.

Qualitatively speaking, there is a noticeable difference in performance across the three different input features which correlates with quantitative performance (see footnote 1 for sound examples). Using representations from Jukebox tends to result in fewer wrong notes than the other features, and substantially reduces the number of egregiously wrong notes (e.g., notes outside of the key signature). Representations from Jukebox also appear to aid in the detection of more nuanced rhythmic patterns. Moreover, using handcrafted features will often result in several repeated onsets during a longer sustained melody note—in contrast, using representations from Jukebox appears to mitigate this failure mode.

Different features also appear to complement one another to a degree. The strongest performance overall is obtained by combining all three features, though the improvement over Jukebox alone is marginal. The practical downsides of combining all features outweigh the marginal benefits—running both pre-trained models effectively doubles the overall runtime, and the models have incompatible software dependencies. Hence, in the remainder of this paper we focus on models trained on individual features.

6.2 Comparison to melody transcription baselines

We compare overall performance of our proposed melody transcription approach to several baselines. We evaluate all methods on a small subset of 10 songs from RWC-MDB [38–40], another dataset which includes melody transcription labels. We chose this specific subset in an effort to compare to early DSP-based work on melody transcription—none of the early approaches [22–25] shared code, however [24] shared melody transcriptions for this 10-song subset.

In addition to [24], we also compare to a baseline which applies a note segmentation heuristic [19] to a melody extraction algorithm [48]. We additionally compare to MT3 in a zero-shot fashion—this model was not trained on melody transcription but was trained on some tasks which incorporate vocal transcription. Finally, because the vo-

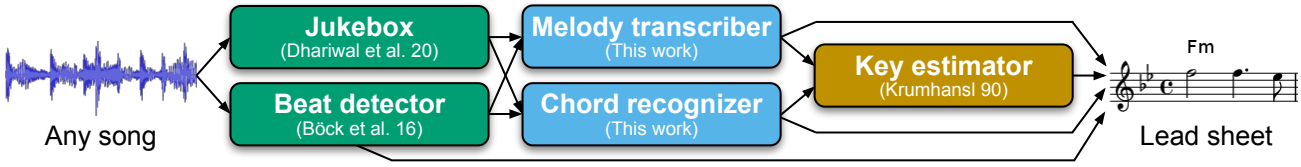

Figure 3. Inference procedure for Sheet Sage, our proposed system which transcribes any Western music audio into lead sheets (scores which depict melody as notes and harmony as chord names). The green, blue, and yellow boxes respectively take audio, features, and symbolic music data as input. Green boxes are modules that we built as part of this work—both are Transformers [14] trained on their respective tasks using audio features from Jukebox [1] and data from HookTheory [49].

cals often carry the melody in popular music, we compare to a baseline of running the Tony [28] monophonic transcription software on source-separated vocals isolated with Spleeter [31]. Because this approach will only work for vocals, we also separately report performance on a subset of our evaluation set where the vocals represent the melody. Scores for all methods and baselines appear in Table 2.

Overall, our approach to training Transformers with features from Jukebox significantly outperforms the strongest baseline in both the vocals-only and unrestricted settings ($p < 0.01$ using a two-sided t-test for paired samples). Qualitatively speaking, the stronger baselines produce transcriptions where a reasonable proportion of the notes are the correct pitches, but they have poor rhythmic consistency with respect to the ground truth. In contrast, our best model produces the correct pitches more often and with a higher degree of rhythmic consistency.

7. SHEET SAGE

As a bonus demo, here we describe Sheet Sage, a system we built to automatically convert music audio into lead sheets (see footnote on first page for examples), powered by our Jukebox-based melody transcription model. In Western music, a piece can often be characterized by its melody and harmony. When engraved as a lead sheet—a musical score containing the melody as notes on a staff and the harmony as chord names—melody and harmony can be readily interpreted by musicians, enabling recognizable performances of existing pieces. Hence, for some music, a lead sheet represents the essence of its underlying composition. Existing services like Chordify [50] can already detect a subset of the information needed to produce lead sheets (specifically, chords, beats, and keys) for broad music audio. However, despite past research efforts [24, 25], no user-facing service yet exists which can convert broad music audio into lead sheets, presumably due to the poor performance of existing melody transcription systems.

To build Sheet Sage, we also train a Jukebox-based chord recognition model on the HookTheory data, using the same methodology that we propose for melody transcription (we simply replace the target vocabulary of onset pitches with one containing chord labels). Passing audio through our Jukebox-based melody transcription and chord recognition models results in a score like format containing raw note names and chord labels per sixteenth note. Engraving this information as a lead sheet requires additional information: the key signature and the time signature. We estimate the former using the Krumhansl-Schmuckler algorithm [51, 52], which takes the symbolic melody and chord information as input. For the latter, we use `madmom` [12, 13]. Finally, we engrave a lead sheet using Lilypond [53]. See Figure 3 for a full schematic.

Subjectively speaking, Sheet Sage often produces high-quality lead sheets, especially for the chorus and verse segments of pop music which have more prominent melodies. Performance is fairly robust across styles and instruments, even those which are less represented in the training data—one user reported particularly strong success on Bollywood music. However, the system occasionally struggles, especially with quieter vocals, layered harmonies, unusual time signatures, or poor intonation. Sheet Sage is also limited to fixed time and key signatures due to limitations of its downbeat detection and key estimation modules.

8. CONCLUSION

We present a new method and dataset which together improve melody transcription on broad music audio. Our method benefits from the rich representations learned by generative models pre-trained on broad audio. This suggests that further improvement in melody transcription may be possible without additional data, i.e., by scaling up or otherwise improving the pre-training procedure. By open sourcing our models and dataset, we hope to spark renewed interest for melody transcription in the MIR community, which may in turn reduce the gap between human perception and machine recognition of a fundamental aspect of music.

9. ETHICAL CONSIDERATIONS

Our definition of melody transcription incorporates equal temperament, a Western-centric tuning system. This could lead to disparate treatment of non equal-tempered music, e.g., if a streaming service were to use melody transcriptions for recommendation. We therefore advocate for the deployment of transcription only in contexts where users are self-selecting music to listen or play along to. Transcription may also be used to create training data for generation—as with any work on generation, there are risks of plagiarism and labor displacement. We recommend that any work on generation involve careful auditing and mitigation of plagiarism. Due to the incomplete nature of a melody, we argue that melody generation tools are more likely to be *incorporated* into co-creation workflows (see [54]) rather than used to displace musicians.

10. ACKNOWLEDGEMENTS

Thanks to Annie Hui-Hsin Hsieh, John Hewitt, Maggie Henderson, Megha Srivastava, Nelson Liu, Pang Wei Koh, Rodrigo Castellon, Sam Ainsworth, and Zachary C. Lipton for helpful discussions, support, and advice. We thank all reviewers for their helpful comments.

11. REFERENCES

[1] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, and I. Sutskever, "Jukebox: A generative model for music," *arXiv:2005.00341*, 2020.

[2] M. Ryynänen, T. Virtanen, J. Paulus, and A. Klapuri, "Accompaniment separation and karaoke application based on automatic melody transcription," in *IEEE International Conference on Multimedia and Expo*, 2008.

[3] K. Droe, "Music preference and music education: A review of literature," *Applications of Research in Music Education*, 2006.

[4] D. Bainbridge, C. G. Nevill-Manning, I. H. Witten, L. A. Smith, and R. J. McNab, "Towards a digital library of popular music," in *ACM Conference on Digital Libraries*, 1999.

[5] A. Ghias, J. Logan, D. Chamberlin, and B. C. Smith, "Query by humming: Musical information retrieval in an audio database," in *ACM International Conference on Multimedia*, 1995.

[6] S. Ewert, B. Pardo, M. Muller, and M. D. Plumbley, "Score-informed source separation for musical audio recordings: An overview," *IEEE Signal Processing Magazine*, 2014.

[7] 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," in *ICLR*, 2019.

[8] M. Goto and S. Hayamizu, "A real-time music scene description system: Detecting melody and bass lines in audio signals," in *International Joint Conference on Artificial Intelligence Workshop on Computational Auditory Scene Analysis*, 1999.

[9] M. Goto, "A real-time music-scene-description system: Predominant-F0 estimation for detecting melody and bass lines in real-world audio signals," *Speech Communication*, 2004.

[10] J. Salamon, E. Gómez, D. P. Ellis, and G. Richard, "Melody extraction from polyphonic music signals: Approaches, applications, and challenges," *IEEE Signal Processing Magazine*, 2014.

[11] K. S. Rao, P. P. Das *et al.*, "Melody extraction from polyphonic music by deep learning approaches: A review," *arXiv:2202.01078*, 2022.

[12] S. Böck, F. Krebs, and G. Widmer, "Joint beat and downbeat tracking with recurrent neural networks," in *ISMIR*, 2016.

[13] S. Böck, F. Korzeniowski, J. Schlüter, F. Krebs, and G. Widmer, "`madmom`: A new python audio and music signal processing library," in *International Conference on Multimedia*, 2016.

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[15] S. Sigtia, E. Benetos, and S. Dixon, "An end-to-end neural network for polyphonic piano music transcription," *IEEE Transactions on Audio, Speech, and Language Processing*, 2016.

[16] 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," in *ISMIR*, 2018.

[17] R. Castellon, C. Donahue, and P. Liang, "Codified audio language modeling learns useful representations for music information retrieval," in *ISMIR*, 2021.

[18] J. S. Downie, X. Hu, J. H. Lee, K. Choi, S. J. Cunningham, and Y. Hao, "Ten years of MIREX: reflections, challenges and opportunities," in *ISMIR*, 2014.

[19] J. Salamon, "`audio_to_midi_melodia`," https://github.com/justinsalamon/audio_to_midi_melodia, 2015.

[20] R. Nishikimi, E. Nakamura, K. Itoyama, and K. Yoshii, "Musical note estimation for F0 trajectories of singing voices based on a Bayesian semi-beat-synchronous HMM," in *ISMIR*, 2016.

[21] R. Nishikimi, E. Nakamura, M. Goto, K. Itoyama, and K. Yoshii, "Scale-and rhythm-aware musical note estimation for vocal F0 trajectories based on a semi-tatum-synchronous hierarchical hidden semi-markov model," in *ISMIR*, 2017.

[22] R. P. Paiva, T. Mendes, and A. Cardoso, "An auditory model based approach for melody detection in polyphonic musical recordings," in *International Symposium on Computer Music Modeling and Retrieval*, 2004.

[23] R. P. Paiva, T. Mendes, and A. Cardoso, "On the detection of melody notes in polyphonic audio," in *ISMIR*, 2005.

[24] M. P. Ryynänen and A. P. Klapuri, "Automatic transcription of melody, bass line, and chords in polyphonic music," *Computer Music Journal*, 2008.

[25] J. Weil, T. Sikora, J.-L. Durrieu, and G. Richard, "Automatic generation of lead sheets from polyphonic music signals," in *ISMIR*, 2009.

[26] A. Laaksonen, "Automatic melody transcription based on chord transcription," in *ISMIR*, 2014.

[27] E. Molina, L. J. Tardón, A. M. Barbancho, and I. Barbancho, "SiPTH: Singing transcription based on hysteresis defined on the pitch-time curve," *IEEE Transactions on Audio, Speech, and Language Processing*, 2014.

[28] M. Mauch, C. Cannam, R. Bittner, G. Fazekas, J. Salamon, J. Dai, J. Bello, and S. Dixon, "Computer-aided melody note transcription using the Tony software: Accuracy and efficiency," in *International Conference on Technologies for Music Notation and Representation*, 2015.

[29] R. Nishikimi, E. Nakamura, M. Goto, K. Itoyama, and K. Yoshii, "Bayesian singing transcription based on a hierarchical generative model of keys, musical notes, and F0 trajectories," *IEEE Transactions on Audio, Speech, and Language Processing*, 2020.

[30] R. Nishikimi, E. Nakamura, M. Goto, and K. Yoshii, "Audio-to-score singing transcription based on a CRNN-HSMM hybrid model," *APSIPA Transactions on Signal and Information Processing*, 2021.

[31] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *Journal of Open Source Software*, 2020.

[32] E. Benetos, S. Dixon, D. Giannoulis, H. Kirchhoff, and A. Klapuri, "Automatic music transcription: challenges and future directions," *Journal of Intelligent Information Systems*, 2013.

[33] J. Thickstun, Z. Harchaoui, and S. Kakade, "Learning features of music from scratch," in *ICLR*, 2017.

[34] 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 *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2019.

[35] J. Gardner, I. Simon, E. Manilow, C. Hawthorne, and J. Engel, "MT3: Multi-task multitrack music transcription," in *ICLR*, 2022.

[36] A. Ycart, L. Liu, E. Benetos, and M. Pearce, "Investigating the perceptual validity of evaluation metrics for automatic piano music transcription," *TISMIR*, 2020.

[37] C. Raffel, B. McFee, E. Humphrey, J. Salamon, O. Nieto, D. Liang, and D. Ellis, "`mir_eval`: A transparent implementation of common MIR metrics," in *ISMIR*, 2014.

[38] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Popular, classical and jazz music databases," in *ISMIR*, 2002.

[39] M. Goto, H. Hashiguchi, T. Nishimura, and R. Oka, "RWC Music Database: Music genre database and musical instrument sound database," in *ISMIR*, 2003.

[40] M. Goto, "Development of the RWC Music Database," in *International Congress on Acoustics*, 2004.

[41] Y.-W. Chen, H.-S. Lee, Y.-H. Chen, and H.-M. Wang, "SurpriseNet: Melody harmonization conditioning on user-controlled surprise contours," in *ISMIR*, 2021.

[42] Y.-C. Yeh, W.-Y. Hsiao, S. Fukayama, T. Kitahara, B. Genchel, H.-M. Liu, H.-W. Dong, Y. Chen, T. Leong, and Y.-H. Yang, "Automatic melody harmonization with triad chords: A comparative study," *Journal of New Music Research*, 2021.

[43] J. Jiang, G. G. Xia, and D. B. Carlton, "MIREX 2019 submission: Crowd annotation for audio key estimation," *MIREX*, 2019.

[44] J. Jiang, G. G. Xia, D. B. Carlton, C. N. Anderson, and R. H. Miyakawa, "TransformerVAE: A hierarchical model for structure-aware and interpretable music representation learning," in *ICASSP*, 2020.

[45] C. Hawthorne, I. Simon, R. Swavely, E. Manilow, and J. Engel, "Sequence-to-sequence piano transcription with transformers," in *ISMIR*, 2021.

[46] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks," in *ICML*, 2006.

[47] B. McFee and D. Ellis, "Analyzing song structure with spectral clustering," in *ISMIR*, 2014.

[48] J. Salamon and E. Gómez, "Melody extraction from polyphonic music signals using pitch contour characteristics," *IEEE Transactions on Audio, Speech, and Language Processing*, 2012.

[49] C. Anderson, D. Carlton, R. Miyakawa, and D. Schwachhofer, "Hooktheory TheoryTab DB," https://www.hooktheory.com/theorytab.

[50] B. de Haas, J. P. Magalhaes, D. Ten Heggeler, G. Bekenkamp, and T. Ruizendaal, "Chordify: Chord transcription for the masses," in *ISMIR Late-Breaking Demos*, 2014.

[51] C. L. Krumhansl, *Cognitive foundations of musical pitch*. Oxford University Press, 1990.

[52] D. Temperley, "What's key for key? The Krumhansl-Schmuckler key-finding algorithm reconsidered," *Music Perception*, 1999.

[53] H.-W. Nienhuys and J. Nieuwenhuizen, "LilyPond, a system for automated music engraving," in *Colloquium on Musical Informatics*, 2003.

[54] C.-Z. A. Huang, H. V. Koops, E. Newton-Rex, M. Dinculescu, and C. J. Cai, "AI Song Contest: Human-AI co-creation in songwriting," in *ISMIR*, 2020.

SOURCE SEPARATION OF PIANO CONCERTOS WITH TEST-TIME ADAPTATION

Yigitcan Özer **Meinard Müller**
International Audio Laboratories Erlangen, Germany
{yigitcan.oezer, meinard.mueller}@audiolabs-erlangen.de

ABSTRACT

Music source separation (MSS) aims at decomposing a music recording into its constituent sources, such as a lead instrument and the accompaniment. Despite the difficulties in MSS due to the high correlation of musical sources in time and frequency, deep neural networks (DNNs) have led to substantial improvements to accomplish this task. For training supervised machine learning models such as DNNs, isolated sources are required. In the case of popular music, one can exploit open-source datasets which involve multitrack recordings of vocals, bass, and drums. For western classical music, however, isolated sources are generally not available. In this article, we consider the case of piano concertos, which is a genre composed for a pianist typically accompanied by an orchestra. The lack of multitrack recordings makes training supervised machine learning models for the separation of piano and orchestra challenging. To overcome this problem, we generate artificial training material by randomly mixing sections of the solo piano repertoire (e.g., piano sonatas) and orchestral pieces without piano (e.g., symphonies) to train state-of-the-art DNN models for MSS. As our main contribution, we propose a test-time adaptation (TTA) procedure, which exploits random mixtures of the piano-only and orchestra-only parts in the test data to further improve the separation quality.

1. INTRODUCTION

Music source separation (MSS) is a central task in music information retrieval, which seeks to recover individual musical sources in audio recordings. In general, a musical source can refer to singing, an instrument, or an entire group of instruments providing an accompaniment. Since musical signals often exhibit non-stationary spectro–temporal properties and may be highly correlated in time and frequency, MSS proves to be a challenging task in music processing [1]. In the last years, deep neural networks (DNNs) have led to substantial improvements in separating musical sources [2–11]. Despite their effectiveness, one disadvantage of data-driven deep models is their need for a large training dataset, which in the case of MSS consists of multitrack recordings with (isolated) individual sources or stems. Most of the open-source datasets with isolated stems are limited to popular music, e.g., MUSDB18 [12]. However, for western classical music, professionally produced multitrack recordings are quite rare [13, 14].

In this article, we consider the separation of piano concertos, which are a genre of central importance in western classical music. From the Baroque era onward, numerous composers have written piano concertos, which are compositions highlighting the virtuosity of pianists. As an example, Figure 1 shows an excerpt from the first movement of Piano Concerto in D minor (KV 466) by Wolfgang Amadeus Mozart. In addition to the large number of compositions, there also are many prominent historical recordings of piano concertos in classical music archives. In this context, separation of piano and orchestra can enable applications such as editing and upmixing historical and modern piano concerto recordings.

As the piano is the lead instrument and the orchestra takes over the accompaniment, separation of piano concertos can be regarded as a lead instrument and accompaniment separation task [15–17]. The piano has distinct timbral characteristics, e.g., clear onsets, which intuitively may help a separation model in distinguishing it from orchestral instruments such as strings, woodwinds, and brass. Nevertheless, the high spectro–temporal correlations between the piano and orchestral parts in concertos constitute a challenging problem.

In this paper, we adapt the spectral-based Spleeter model [5] to address the separation of piano and orchestra in piano concertos. Spleeter has achieved impressive results for the separation of four stems (vocals, drums, base, and others) in the SiSEC challenge [18]. Building upon its standard architecture, which is an encoder–decoder convolutional neural network (CNN), we train our baseline model using a proprietary dataset.

When training deep MSS models, generating random mixes of solo instrument recordings may improve the separation quality [4, 19]. Random mixing for data generation and augmentation has opened up new paths for separating instrument mixtures, for which multitrack recordings are not available. For example, Chiu et al. [20] created their own synthetic dataset comprising classical violin and pop piano solo recordings, which serve as training material of an MSS model for the separation of piano and violin duos. Inspired by the recent advances in deep learning and data

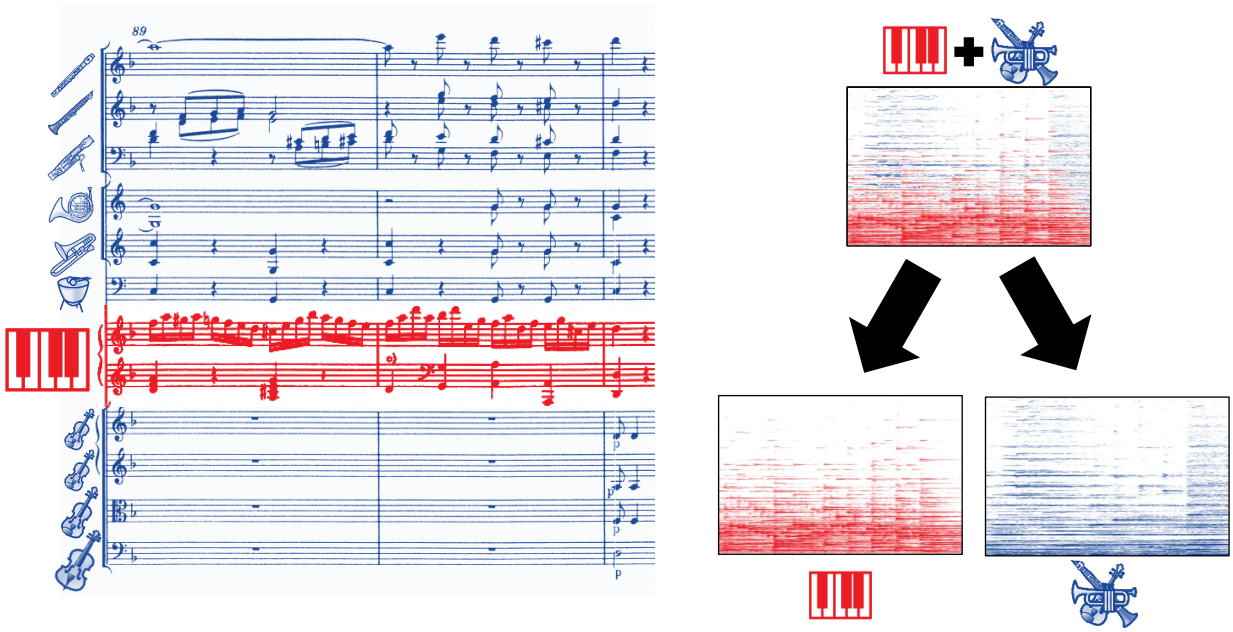

Figure 1: An excerpt from the first movement of the Piano Concerto in D minor (KV466) by Wolfgang Amadeus Mozart. Our goal is to estimate the magnitude spectrograms of the piano part (red) and orchestral part (blue).

augmentation, we generate in our setting an artificial training dataset through randomly mixing samples from the solo piano repertoire (e.g., piano sonatas, mazurkas, etc.) and orchestral pieces without piano (e.g., symphonies) to simulate piano concertos.

Whereas one can improve the performance of data-based models using artificially generated data, a supervised machine learning model necessitates a representative training set to ensure its robustness during the testing phase. In the case of MSS, many recordings have specific acoustic properties (e.g., historical recordings) that are not reflected in training datasets, thus leading to a poor separation performance. To overcome this issue, one can exploit the occurence of repeating patterns in the same recording [21], use bootstrapping to improve separation results [22, 23], or adapt a pre-trained model to one specific target mixture [24, 25]. In this work, our approach is based on the latter strategy. To this end, we first train on our artificial dataset and then finetune the model at the testing stage. As our main contribution, we propose a test-time adaptation (TTA) method similar to [26], where we exploit that a piano concerto typically has relatively long piano-only and orchestra-only passages. Generating random mixes of these sections, we adapt the separation model at the test time individually for each piano concerto in our test dataset. Our systematic experiments highlight the benefits of TTA trained with the spectrogram-domain MSS model Spleeter [5]. To evaluate the performance of our models, we use the widely-used Signal to Distortion Ratio (SDR) [27], and the *2f-model* [28], which is an objective quality measure. Furthermore, we conduct Multiple Stimulus with Hidden Reference and Anchors (MUSHRA) listening tests [29] to assess the perceptual separation quality.

The remainder of the paper is organized as follows. In Section 2, we describe our MSS approach, explore the recent state-of-the-art spectrogram-domain DNN model Spleeter to address the MSS task and describe our experimental setting. In Section 3, we introduce the TTA procedure to improve the separation quality of piano concertos and present our dataset. In Section 4, we report on the quantitative empirical results, including a subjective evaluation. Finally, we conclude in Section 5 with prospects on future work.

2. MUSIC SOURCE SEPARATION APPROACH

Recent DNN approaches for MSS can be divided into two categories: waveform-domain architectures [6, 7] and spectrogram-domain approaches [2–5]. In this paper, our focus is the latter type of models, which learn to approximate the magnitude spectrogram of a target source. To reconstruct the separated audio signals, spectrogram-based models typically use binary masking, soft masking or multichannel Wiener filtering [30].

In particular, we adapt the Spleeter model [5] for separating piano concertos. Its default setting is based on a *U-Net* [31], which has recently been a widely-used architecture in the MIR community to address the MSS task [2, 6, 25, 32, 33]. Following this trend of research, we adapt a U-Net model to predict the magnitude spectrograms of the constituent piano and orchestral parts in a piano concerto. In the following, we revisit the U-Net architecture in Section 2.1 and present our experimental setting in Section 2.2.