AUTOMATIC MUSIC TRANSCRIPTION FOR POP MUSIC

Jonathan Yaffe
Tel Aviv University
jonathany@mail.tau.ac.il

Ben Maman
Tel Aviv University
benmaman@mail.tau.ac.il

Amit H. Bermano
Tel Aviv University
amberman@mail.tau.ac.il

ABSTRACT

Automatic Music Transcription (AMT) is the process of converting audio recordings of music into written musical notations with minimal human intervention, enabling the transformation of sound into tangible, editable, and analyzable forms. Current AMT approaches are restricted to piano and (some) guitar recordings, due to difficult data collection. In this project we attempt the task of AMT for pop music. This task is hard due to data collection difficulty and also the wide variety of pop instruments and sounds. The main idea is to take a model that was trained mainly on classic music and retrain it for pop music. Our project page with inference examples is available at `https://yoni-yaffe.github.io/pop_alignment`

***K*eywords** Automatic Music Transcription (AMT)

1 Introduction

Automatic music transcription is a technological marvel that bridges the worlds of sound and notation. It involves the intricate process of converting audio recordings of music into a tangible, symbolic representation, such as sheet music or a MIDI file, with minimal to no human intervention. This transformative technology holds the promise of revolutionizing the way we interact with music, enabling a wide range of applications, from aiding musicians in transcribing their compositions to facilitating the analysis of vast music databases. Automatic music transcription combines elements of signal processing, machine learning, and deep neural networks to decode the intricate layers of melody, harmony, and rhythm inherent in musical compositions. It not only enhances music accessibility but also empowers musicians, researchers, and the music industry to explore new creative horizons and unlock the hidden treasures of musical expression.

Transcribing multi-instrument music is often more challenging than transcribing single-instrument music due to the complexity of the task. Unlike single-instrument pieces, multi-instrument music requires the transcriber to identify and separate the different instrument sounds, decipher their individual rhythms and melodies, and then accurately notate them on the score. Additionally, there may be sections where different instruments are playing overlapping or harmonized parts, making it difficult to distinguish between them. This can require a high level of skill and expertise, as well as extensive knowledge of music theory and instrumentation. Nonetheless, transcribing multi-instrument music can also be more rewarding, as it allows for a more nuanced and detailed understanding of the composition, and can capture the intricacies of the musical performance.

The success of deep neural networks in image classification have attracted more and more attention from the AMT community. However DNN require massive amount of training data. Data collection is very difficult because manual annotation require a lot of work and it is often not accurate. The two instruments with the best gathered data are Piano and Guitar. For Guitar the annotation is done semi automatically with human verification. For piano data collection is simpler because of unique equipment (the Disklavier). Sensors record the movements of the keys, hammers, and pedals during a performance. Because piano has much more data most AMT literature focus on that. For pop music there is only manually annotated data by volunteers. The problem is that the data is low quality. People tend to change octave of certain instruments, skip certain parts of the song and have wrong timings. In addition people tend to give their own interpretation and change several notes. For that reason it is impossible to use this data to train as is.

We manually filtered the Lakh MIDI Dataset Clean, modified some of the songs(mainly fixed wrong octave in some of the songs) and created a new pop dataset with 300 songs with reasonable quality. In this work we adopt the method proposed in [1] - a method for simultaneously training a transcriber and aligning the scores to their corresponding performances, in a fully-automated process. We first take a model that was trained mainly on classic music and then use [1] method to align the midi to the audio performance and then retrain the model for pop music.

2 Related Work

We continue the work of [1]. We use the $Note_{EM}$ process. As detailed in the article the process consist of three parts: 1. Take an of-the-shelf architecture trained on synthetic data and bootstrap it's training on synthetic data. 2. The E-step, use the resulting network to predict the transcription of unlabeled recordings. The unaligned score is then warped based on the predictions as likelihood terms, and used as labeling 3. The M-step, the transcriber itself is trained on the new generated labels.

In this work we replace the model in stage 1 with a model that was trained on real recordings using the $Note_{EM}$ process. Then we start the process again on the pop dataset. The idea is that after training on classic music, the model will be good enough to detect pitch and then we can retrain for pop.

We use the architecture proposed by [2]. The approach is to predict pitch onsets events and then use the predictions to condition framewise prediction. During inference we do not allow for a new note to start unless the onset detector detect the start of the note. However only the onset stack will be trained. We freeze the velocity, offset and the frame stack.

3 Method

As detailed in [1] the key observation is that a weak transcriber can still produce accurate predictions if the global content of the outcome is known up to a warping function. In our case we have the unaligned labels for the pop music and we can take a pre-trained model and run the EM algorithm (Section1). We will also be using methods to deal with the low quality data as alignment and pseudo labels(Section 3.2). We will also use pitch shift augmentation(Section 3.5) to increase training data.

3.1 The EM Algorithm

Expectation–maximization (EM) algorithm is an iterative algorithm that aims to estimate parameters of a statistical model when some of the data is unobservable or latent. It alternates between two key steps: the "Expectation" step computes the expected values of the unobservable variables given the current estimates of the model parameters, and the "Maximization" step updates these parameters to maximize the likelihood of the observed data while incorporating the expected values obtained in the previous step. This iterative process continues until convergence is achieved, yielding maximum likelihood estimates for the model parameters. For our case we start by computing probabilities for each note using the current parameters. Then the unaligned labels are wrapped using DTW (the E step). Then the labels are used to retrain the model (M step).

3.2 Pseudo Labels and Alignment

In order to train on the pop data we modify the labels in the following ways: 1. We use a previously trained model on classic music to predict notes probabilities. Then we use the probabilities for running DTW and align the unaligned labels. That way we can overcome the inaccuracies in the labels. The inaccuracies can be different timings and also wrong tempo.

After we align the labels we also add pseudo labels. We use the probabilities from the previous model and we use two thresholds T_{pos} and T_{neg}. Predicted notes with probability higher than T_{pos} are added to the labels. That way we can transfer the learning of the previous model to the new one. In addition we do not back propagate through notes with probability lower than T_{neg}. That way we can get rid of wrong labeling or inaccurate labeling of the data. The main advantage of using pseudo labels is preserving knowledge of the model when training on pop music. Pop music labels can be very inaccurate so we want to improve the quality of the labels. This method turns out to be crucial for pop training as will be seen in (4.Experiments).

3.3 Instrument classes

Pop music has a wide variety of instruments so it is infeasible to detect each instrument with the current data. Thus we split the instruments into classes. The classes are Piano, Synth, Guitar, Electric Guitar, Bass, Human Voice. Human voice contain strings and wind instruments(such as Violin, Cello, Flute). The full mapping can be found on the GitHub page. Human voice class is large because in the pop dataset human voice is inconsistently labeled so we want to catch all cases. In addition the pop dataset is not very big so we want as least classes as possible.

3.4 Instrument Sensitive detection

Best results were obtained when training two models. One that detects only pitch(insensitive for instrument) and one that is sensitive for instruments (detect each class in 3.3). During inference we use the pitch model to predict the notes. Then we assign an instrument for each note using the probabilities of the instrument model(we simply take the instrument with the maximum probability). We also noticed that more instrument classes causes slower learning.

3.5 Pitch Shift Augmentation

As proposed in [3] we augment the data by stretching or shrinking our input audio with linear interpolation. We make pitch shifted copies by -5 to 5 semitones(overall 11 copies). In addition we apply continuous uniform distributed shift for each copy in the range of -0.1 to 0.1 semitones. Using the pitch shift help to deal with the problem of lack of data and also makes the model more robust to tuning in different recordings because usually in pop music the instruments are tuned perfectly.

3.6 Modulated Transcriber

We tried a new approach for detecting instruments. In the default configuration we have the audio as input and then we get predictions for notes and instruments each instrument separately. In this setting we also add the instrument as input and use it as "instrument embedding". Then the model predict only the pitch based on the embedding. That way we can have a one model that can predict all instruments with less parameters. This option can be more valuable when we need to predict a large amount of instrument. This method did not prove to be better at our experiments but we believe that the idea need to be explored with greater effort.

4 Experiments

4.1 Dataset

As mentioned before we formed a new dataset that is a subset of the Lakh MIDI Dataset. The new dataset consist of $\sim$ 300 songs. Each song was checked manually to confirm that the midi file is on the same octave and that the midi sound reasonable. Another problem that needed to be checked is songs length. Many midi files sometimes omit part of the song and that can create big differences between the audio and midi. Another problem is that the audio and the midi are not synchronized. To solve that issue we use the DTW algorithm to align the unaligned midi files from the dataset before we begin training. After the alignment we also add the pseudo labels in order to preserve the knowledge from the previous model. We need to do that because even after the alignment the labels are still very far from being accurate and many times background notes are not in the midi. We also notice that instrumnt assignment was not consistent for human voice. In some notations human voice was notated as a string instrument sometimes as guitar and sometimes as a trumpet. To overcome this issue the class of human voice consist of many instruments.

4.2 Training

We used the architecture from [2] but we only trained the onset stack. weights for velocity, frame and offset were frozen for training. We use convolutional filters of size 64/64/128, and linear layers of size 1024. We re-sampled all recordings to 16kHz sample rate, and used the log-mel spectrogram with 229 bins as the input representation. We used hop length 512. We used the mean BCE loss, with an Adam optimizer, with gradient clipped to norm 3, and batch size 32.

We trained two models. One that detects only pitch and one that also detect instruments(the classes in 3.3). We train for 100k iterations. For inference we use both models. The pitch model for detecting the notes and the instrument sensitive model to classify the instruments of each note. For the instrument model we add pseudo labels again after 100k steps. We trained using 4 Nvidia Tita XP. training took $\sim$30 hours.

4.3 Evaluation

For evaluation We use 2 different datasets. $Musicnet_{EM}$ based on the Musicnet after the modification from [1]. Guitar set, a dataset contains $\sim$ 3 hours of guitar recordings. Both are not "pop" dataset but we can use them to see how well the new train model compare to the previous one. we use $Note_{EM}$ as the baseline model from [1], it was trained on

Musicnet but no on Guitar Set. We only evaluated note metrics since the frame stack was frozen for training.

	$Musicnet_{EM}$			Guitar Set		
	P	**R**	**F1**	**P**	**R**	**F1**
$Note_{EM}$	92.8	88.4	90.3	93.9	72.4	81.2
pop model pitch	82.5	75.4	78.1	84.2	77.3	80.1
pop model inst	71.5	76.3	73.4	82.6	77.7	79.4
pop no pseudo labels	68.3	49.4	55.4	82.8	62.8	69.8

$Note_{EM}$ is the baseline model. it is instrument insensitive model. pop model pitch is instrument insensitive model with pseudo labels. pop model inst is instrument sensitive model with pseudo labels. pop no pseudo labels is instrument insensitive model without pseudo labels. All the models were trained with pitch shift augmentation.

4.4 Inference

Inference examples can be found on the project page on `https://yoni-yaffe.github.io/pop_alignment`. We tested on a wide variety of pop songs.

For inference we used two models one for pitch detection and one for instrument detection. We tried using both the pop model and the baseline model for the pitch detection. We found out that the pop model was better for most of the pop songs.

5 Conclusion

In this paper we attempted the task of AMT for pop. As expected pop music is complex and finding good data to train on can be really hard. From the experiment we can learn that the pop pitch model does not perform as well as the baseline on the Music-net and Guitar set datasets. However we see that the pseudo-labels help to preserve the knowledge of the baseline model. In the inference examples we can hear that on some performances the pop model perform better than the baseline. For the instrument sensitive model we noticed that guitar was the most classified instrument. Human voice class was much less frequently recognized because human voice has a wide variety of sounds and it also contains wind and string instrument so the data for this instrument was not of high enough quality. In addition as mentioned before instrument assignment is not consistent in the dataset, many instruments fall into the category of human voice. However our method produced a reasonable data to train on. We believe that more work can be done to improve training such as adding guitar data and also human voice data.

References

[1] Ben Maman and Amit H. Bermano. Unaligned supervision for automatic music transcription in the wild, 2022.

[2] Curtis Hawthorne, Erich Elsen, Jialin Song, Adam Roberts, Ian Simon, Colin Raffel, Jesse Engel, Sageev Oore, and Douglas Eck. Onsets and frames: Dual-objective piano transcription, 2018.

[3] John Thickstun, Zaid Harchaoui, Dean P. Foster, and Sham M. Kakade. Invariances and data augmentation for supervised music transcription. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2241–2245, 2018.