CS230

Milestone for Automatic Music Transcription

Project Category: Sound Recognition

Ruoyan Chen
Department of Electrical Engineering
Stanford University
ruoyan85@stanford.edu

Yiwen Liu
Department of Electrical Engineering
Stanford University
ywliu24@stanford.edu

1 Introduction Of Project

For a long time, the music sheet has been regarded as one the most effective media for musicians to communicate with each other. It is also an intuitive way for non-professionals to learn how to play a music instrument or sing a song. Nevertheless, music sheet might not be available for all compositions especially for those being protected by strict copyright regulations. Therefore, to better provide music beginners or amateurs with a chance to play these compositions, we came up a project idea of using deep learning techniques to perform automatic music transcription. The input to our algorithm would be raw audios. After training, our model will be able to translate music audios into the symbolic music representation output.

2 Details About The Dataset

For this project, we used MusicNet dataset. MusicNet is a collection of 330 freely-licensed classical music recordings, together with over 1 million annotated labels indicating the precise time of each note in every recording, the instrument that plays each note, and the note's position in the metrical structure of the composition. The labels are acquired from musical scores aligned to recordings by dynamic time warping, and are verified by trained musicians. Thus, officially they claim a labeling error rate of 4%.

The full dataset for this project is divided into two parts: a training set with 100,000 examples and a dev set with 50,000 examples. An example of the data and label is shown as Figure 1 and Figure 2.

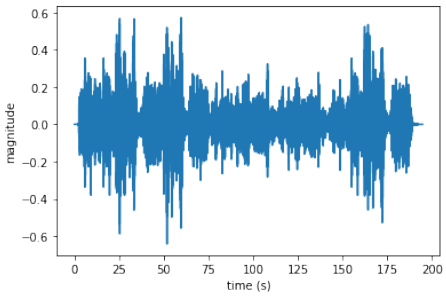

Figure 1: Audio data example

CS230: Deep Learning, Autumn 2020, Stanford University, CA.

	start_time	end_time	instrument	note	start_beat	end_beat	note_value
1	90078	124382	1	63	0.0	1.0	Quarter
2	90078	124382	1	75	0.0	1.0	Quarter
3	90078	110558	1	48	0.0	0.375	Dotted Sixteenth
4	114654	122334	1	55	0.5	0.375	Dotted Sixteenth
5	124382	139742	1	65	1.0	1.0	Quarter
6	124382	129501	1	50	1.0	0.375	Dotted Sixteenth
7	124382	139742	1	77	1.0	1.0	Quarter
8	133086	138206	1	55	1.5	0.375	Dotted Sixteenth
9	139742	146398	1	51	2.0	0.375	Dotted Sixteenth
10	139742	151006	1	67	2.0	0.75	Dotted Eighth
11	139742	151006	1	79	2.0	0.75	Dotted Eighth
12	147934	153054	1	55	2.5	0.375	Dotted Sixteenth
13	154078	159198	1	51	3.0	0.5	Eighth
14	154078	178654	1	79	3.0	2.0	Half
15	154078	178654	1	67	3.0	2.0	Half
16	159198	164318	1	55	3.5	0.375	Dotted Sixteenth
17	165854	173022	1	60	4.0	0.375	Dotted Sixteenth
18	174046	178142	1	55	4.5	0.375	Dotted Sixteenth
19	178654	186334	1	50	5.0	0.5	Eighth
20	178654	189918	1	68	5.0	0.75	Dotted Eighth

Figure 2: Label example

3 Approaches - Current Steps

To build a baseline model, we took advantage of a three-layer network with a fully connected layer interposed between the layer-one convolutions and the linear output layer. In this model, layer-one is a 1D-Conv layer that outputs the spectrogram of the input audio, thus the intermediate layer captures non-linear relationships between features of this spectrogram. By defining the loss function as the Mean Squared Error, we compared the results for two three-layer networks, one trained with a fixed log-spaced, cosine-windowed filterbank at layer-one, and the other trained end-to-end from the raw audio, and we see they both have test accuracy above 60% (Figure 3). We also made several observations during the implementation. For example, for the three-layer e2e model, because the end-to-end method requires a huge amount of training examples, it resulted in a high variance problem. Also, for the three-layer model, we tried Adam optimizer for both models but this unexpectedly resulted in a 10% decrease of test accuracy compared to the result of using Stochastic Gradient Descent optimizer suggested by its original author [1].

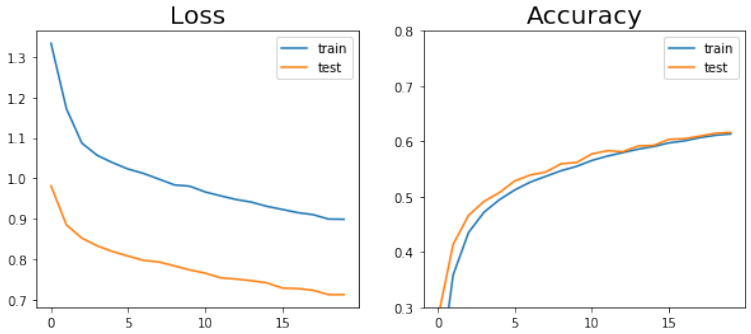

Figure 3: Loss and accuracy for three-layer network

4 Plans - Next Step

In previous days, due to some unspecified bugs, the runtime on Google Colab was inefficient and it took us almost a day to get results. Now, we switched to running locally and found a huge improvement in runtime, leaving a lot of room for hyperparameter tuning. Our first idea is to build much deeper models on top of either the raw audio or filterbank representation to see if the performance improves. Also, we are going to explore more complex network architectures based on the literature reviews.

In existing researches [2] [3] [4] [5] [6], several network architectures are implemented for music transcription problem, including DNNs, RNNs, and CNNs. In our future work, we will explore a

combination of networks to achieve better results, for example, the translation-invariant networks which introduces additional layers to capture specific invariances in the data.

References

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

[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. *arXiv preprint arXiv:1710.11153*, 2017.

[3] Yu-Lun Hsu, Chi-Po Lin, Bo-Chen Lin, Hsu-Chan Kuo, Wen-Huang Cheng, and Min-Chun Hu. Deepsheet: A sheet music generator based on deep learning. In *2017 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 285–290. IEEE, 2017.

[4] Jonggwon Park, Kyoyun Choi, Sungwook Jeon, Dokyun Kim, and Jonghun Park. A bi-directional transformer for musical chord recognition. *arXiv preprint arXiv:1907.02698*, 2019.

[5] Miguel A Román, Antonio Pertusa, and Jorge Calvo-Zaragoza. A holistic approach to polyphonic music transcription with neural networks. *arXiv preprint arXiv:1910.12086*, 2019.

[6] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic piano music transcription. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(5):927–939, 2016.