Solo Piano Detection: Comparing Supervised and Self-supervised approach

Huan Zhang

Apr 2022

1 Introduction

In this coursework, we aim to tackle a specific problem which is detecting the solo piano segment from a piece of recording, and isolating unwanted sound such as applause, speech or other instrument. Both supervised and self-supervised approaches are explored in this report, along with experiment details and results. The code and testing data will be released in `https://github.com/anusfoil/solo_piano`.

The motivation of this task is to provide robust post-processing for a dataset project that we are currently working on. With the final goal of collecting a virtuosic solo piano performance dataset with varying performers and standard piano repertoire, we were scraping and downloading a large body of recordings and albums from internet (over 10k tracks and 1k hours of music). Given that internet-sourced data may contain unwanted sounds such as applause in a live recording or speech in performer's interview, or even non-solo music such as violin sonata, a solo piano detector is necessary in maintaining the quality of such dataset. Without a doubt, such task is relatively niche and have no more applications beyond similar scenario. However, this task is still analogical to other domains such as machine anomaly detection, and our experiment may provides reusable insights.

The task is different from general sound event detection or audio tagging in the sense that it cares only about the positive samples but not negative ones. While the positive sample is always solo piano sound, the negative samples can be any sound in the wild (and we don't care about their exact label as well). Certainly, a binary classification can be performed, but grouping a number of different sound as "non-solo" seems biased. From a perceptual perspective, human would memorize what does a solo piano sounds like and then categorize anything else as non solo piano sound, without explicit training on negative samples. Thus, we also seek self-supervised approach in learning representations specific to piano sound.

2 Methodology

2.1 Supervised Baseline

We take the most intuitive way of approaching the problem as the baseline: Given a set of training data labeled as either solo piano and non solo piano sound, train a classifier that places audio segment into one of these two categories. In actual inference, a full piece of music is truncated into short segments (from 0.5 to 3 seconds), where predictions are obtained from the model. In this case, we obtain a series of timestamp labels that indicates the content of this segment, such that we can isolate unwanted sound.

Two backbone models are applied in this case:

1. **Baseline CNN** We provided a simple CNN baseline that's consisted of 3 convolutional layers, with batch normalization and max pooling in between.

2. **Musicnn** Proposed in [Pons et al., 2018], Musicnn is a musically motivated architecture for music tagging. The spectrogram front-end utilized filter shapes that effectively captures timbral features and temporal features.

2.2 Contrastive Learning

For the self-supervised approach, we employ two recent contrastive learning architectures: **SimCLR** and **Barlow Twins**

2.3 SimCLR

SimCLR was first proposed in [Chen et al., 2020], and it initiates the technique of contrastive learning. The model learns generic representations of data on an unlabeled dataset, and then it can be fine-tuned with a small amount of labeled data to achieve good performance for a given classification task. The generic representations are learned by simultaneously maximizing agreement between differently transformed views of the same image and minimizing agreement between transformed views of different images. Updating the parameters of a neural network using this contrastive objective causes representations of corresponding views to "attract" each other, while representations of non-corresponding views "repel" each other. In the image domain, SimCLR randomly draws examples from the original dataset, transforming each example twice using a combination of simple augmentations (random cropping, random color distortion, and Gaussian blur), creating two sets of corresponding views.

We follows the original SimCLR to incorporate the ResNet architecture to compute the audio representation. Afterwards, SimCLR computes a non-linear projection of the representation using a fully-connected network, which amplifies the invariant features and maximizes the ability of the network to identify different transformations of the same image. After pre-training on the unlabeled audio (which we used the same dataset as the supervised networks, just without labelling), we used the output of CNN to perform the downstream task of classifying solo piano sound.

2.4 Barlow Twins

Barlow Twins [Zbontar et al., 2021] is a self-supervised learning method that applies redundancy-reduction — a principle first proposed in neuroscience — to self supervised learning. The objective function measures the cross-correlation matrix between the embeddings of two identical networks fed with distorted versions of a batch of samples, and tries to make this matrix close to the identity. This causes the embedding vectors of distorted version of a sample to be similar, while minimizing the redundancy between the components of these vectors. Barlow Twins does not require large batches nor asymmetry between the network twins such as a predictor network, gradient stopping, or a moving average on the weight updates. Intriguingly it benefits from very high-dimensional output vectors.

3 Experiment details

3.1 Dataset

We utilize a self-crafted datast called **Solo Piano and Non Solo (SPNS)** dataset. It contains 400 audio clips, each runs for 10 seconds. Half of them were taken from classical piano recordings played by virtuoso, and half of them were taken from the AudioSet [Gemmeke et al., 2017] categories of applause, speech and various instrumental and environment sound. All audio are downsampled to 16k Hz and downmixed to mono. For the input representations, a melspectrogram was computed with 64 mels and 200 hop length, where each 1 second audio segment result in shape of (64, 81).

The testing data, also considered as the case study files, consists of 10 tracks that I took from my scraped album data that's representative as unclean piano recordings: 5 of them are solo piano live recordings with various length of applause at beginning or the end, the other 5 are performances in other instrumentation such as piano concertos.

3.2 Data Augmentations

For the contrastive learning approaches, data augmentations are required to create different views for self-supervision. We exploit the following audio transformations supported by the `torch-audiomentations` pacakge: `Polarity Inversion` - the amplitude is multiplied by -1. `Colored Noise` - adding colored noise to the original signal.

3.3 Evaluation metrics

During training, loss is the most important indication of model performance and is used for our checkpointing. Two types of experiments utilized different loss: For classification task we are using cross entropy loss, but for SimCLR we utilized InfoNCE loss, which compared the similarity of z_i and z_j (two augmented version of the same piece of data) to the similarity of z_i to any other representation in the batch by performing a softmax over the similarity values. For similarity computation we used cosine

similarity. For Barlow Twins, a novel loss was utilized control the invariance and redundancy:

$$L_{BT} = \sum_i (1 - C_{ii})^2 + \lambda \sum_i \sum_{j \neq i} C_{ij}^2$$

In the formula, C represent the correlation matrix, and the the two terms enforce the embeddings invariant to distortion of themselves, while decorrelates the different vector components of the embedding.

For both the direct classification model and downstream task of contrastive learning, we used validation accuracy for as the evaluation metrics.

3.4 Implementation and Training

The code was implemented in pytorch with pytorch-lightning framework. The specific set of parameters (learning rate, batch size, epoches) can be found in the code repository, as different experiments utilize different hyper-parameters.

4 Results and Discussion

trial	Experiment	Backbone	Final Loss	Validation Accuracy
1	Classification	Baseline CNN	0.3161	0.7834
2	Classification	Musicnn	0.2725	0.9231
3	Representation	SimCLR	0.297	0.8713
4	Representation	BarlowTwins	0.385	0.8479

The comparative results are shown in table 4. In terms of the final accuracy, the representation learning methods doesn't overperform the Musicnn classification model. But given that the self-supervision methods requires no labelling, the results are fairly promising. Another thing to notice is the training and inference speed of different sized models, where Musicnn is 3 times slower to train than baseline, and two contrastive learning models are roughly 10 times slower than the baseline on the same environment.

In figure 1, we showed the output taggram of one testing piece as well as its raw waveform. As pictured, the piece runs for 450 seconds with a long applause at the end. Notice that given silence samples exist in non solo category, the model tends to determine the pause within performance as short non solo clips.

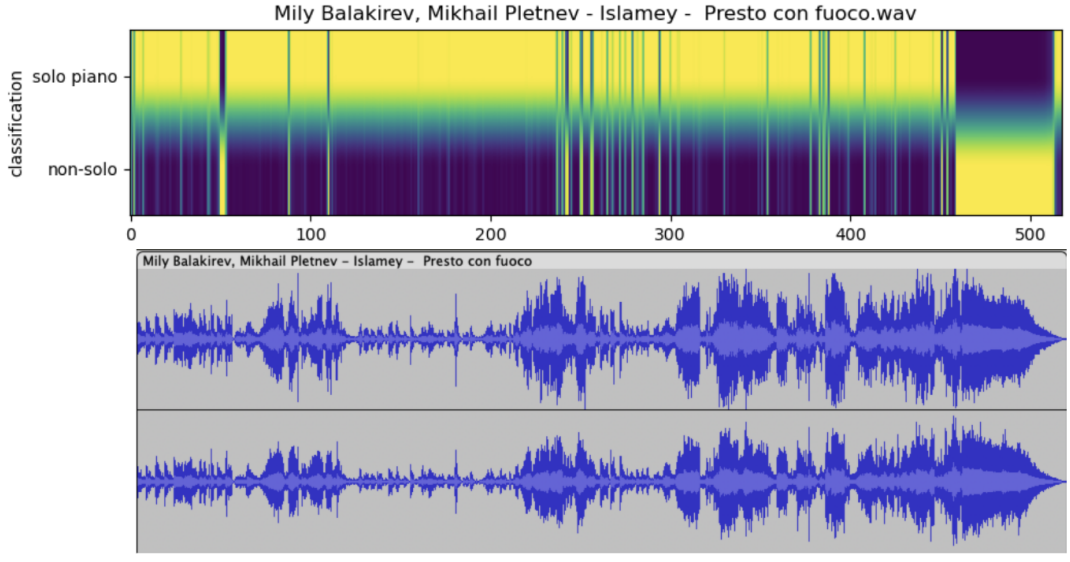

Figure 1: Classification Model: Taggram and waveform for one of our testing data. The applause at the end are effectively captured.

5 Application

As described in the beginning section, the solo piano detector was motivated by the need of dataset-cleansing. After a taggram was obtained, we applied the following post-processing rules to every track:

1. For the beginning 30% and concluding 30% of the recording durations, detect if there are continuous non-piano segment that runs for over 5 seconds.

2. If so, take the beginning or ending timeframe of non-solo segment and remove corresponding audio.

3. For the remaining audio piece, compute the percentage of non-solo clips from inference results. If the piece features other instrumentations, we would expect a large amount of non-solo clips throughout the audio. We set the threshold of 18% and output "non solo piano music" tag for the audio piece.

References

[Chen et al., 2020] Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. (2020). A Simple Framework for Contrastive Learning of Visual Representations. In *In proceedings of the International Conference on Machine Learning (ICML) 2020*.

[Gemmeke et al., 2017] Gemmeke, J. F., Ellis, D. P. W., Freedman, D., Jansen, A., Lawrence, W., Channing Moore, R., Plakal, M., and Ritter, M. (2017). Audio Set: An Ontology and Human-Labeled Dataset for Audio Events. In *In Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

[Pons et al., 2018] Pons, J., Nieto, O., Prockup, M., Schmidt, E., Ehmann, A., and Serra, X. (2018). End-to-end learning for music audio tagging at scale. *Proceedings of the 19th International Society for Music Information Retrieval Conference, ISMIR 2018*, pages 637–644.

[Zbontar et al., 2021] Zbontar, J., Jing, L., Misra, I., Lecun, Y., and Deny, S. (2021). Barlow Twins: Self-Supervised Learning via Redundancy Reduction. In *In proceedings of the International Conference on Machine Learning (ICML) 2021*.