

LEARNING MUSIC SEQUENCE REPRESENTATION FROM TEXT SUPERVISION

Tianyu Chen^{1 3} [†], Yuan Xie⁴ [†], Shuai Zhang^{1 3}, Shaohan Huang², Haoyi Zhou^{1 3}, Jianxin Li^{1 3}

¹BDBC, Beihang University, China ²Microsoft Research Asia, China

³SKLSDE, Beihang University ⁴The Institute of Acoustics of the Chinese Academy of Sciences, China

ABSTRACT

Music representation learning is notoriously difficult for its complex human-related concepts contained in the sequence of numerical signals. To excavate better **M**usic **S**Equence **R**epresentation from labeled audio, we propose a novel text-supervision pre-training method, namely **MUSER**. MUSER adopts an audio-spectrum-text tri-modal contrastive learning framework, where the text input could be any form of meta-data with the help of text templates while the spectrum is derived from an audio sequence. Our experiments reveal that MUSER could be more flexibly adapted to downstream tasks compared with the current data-hungry pre-training method, and it only requires 0.056% of pre-training data to achieve the state-of-the-art performance.

Index Terms— Contrastive learning, deep learning, music representation, cross-modal learning

1. INTRODUCTION

Music is an inseparable part of human culture, containing complex emotional or narrative concepts in audio sequences. It evokes a crucial question for computer scientists - “*Can our computer understand the meaning of music?*” Over the recent years, various music understanding benchmarks have been built, including music tagging [1, 2, 3] and genre classification [4].

Inspired by the success of deep learning on computer vision and natural language processing, researchers introduce large-scale pre-training techniques into music sequence representation learning [5, 6]. The learned representation can be further transferred to different downstream tasks, achieving better performance than former end-to-end methods. However, pre-training is data-hungry. Further improvements on downstream tasks rely heavily on more hand-crafted tags [7] or other metadata [8, 9], where the music labels require professional knowledge, becoming luxury.

A question is naturally brought, “*Are existing labels enough to learn better music representation?*” Music data is relatively quantity-small but with sufficient supervision information in their text-form metadata (e.g., lyrics, album

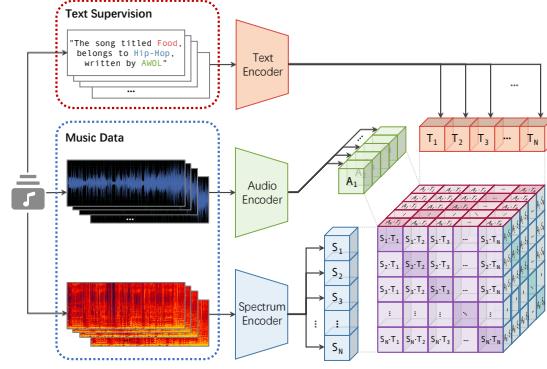


Fig. 1. The framework of MUSER.

descriptions, lyricist, composer, singer, comments), which are still under-explored. In this paper, we propose a novel text-supervision method to learn directly from text-form metadata, called MUSER. First, we convert different forms of metadata into a unified plain text format (e.g., one typical text format contains song name and genre class - “The song titled Food, belongs to Hip-Hop”). Then, the plain text sequence and the audio sequence of music will be encoded respectively into a shared embedding space. Also, the music spectrum is encoded to learn text-form concepts from multi-view. By using the shared embedding space, we can apply CLIP-style [10][11] contrastive learning, which aims to distinguish music sequence by their corresponding text. After contrastive pre-training stage on limited training data, we fine-tune the encoders on genre classification and auto-tagging benchmarks, achieving comparable or even better results than state-of-the-art pre-training methods. Our contributions are: (1) We are the first to introduce text supervision for exploring the fine-grained feature of distributed songs;(2) Additional spectrogram encoder that greatly improves data efficiency of the CLIP-style framework; (3) A novel tri-modal contrastive pre-training framework - MUSER and new state-of-the-art on music-related benchmarks.

2. METHOD

2.1. The Framework of MUSER

As illustrated in Fig.1, three individual encoders are involved in our MUSER framework. To fully excavate the text super-

[†] indicates equal contribution. The corresponding author is Jianxin Li. This work is supported by the NSFC through grant No.61872022.

vision signals, we initialize our text encoder and spectrum encoder from OpenAI pre-trained CLIP [10], where the music spectrum encoder is considered as a special “image encoder”. To get more information, we use templates to stitch together various music metadata. Templates splice descriptive phrases into a complete sentence as text input. And a befitting template could help define the decision boundaries for classification tasks.

Text Encoder: The text encoder is a base-size Transformer [12] network. Before operating on text tokens, the encoder employs a byte pair encoding (BPE) tokenizer to convert plain text into a sequence of discrete tokens. The text sequence will be bracketed with special tokens, [SOS] and [EOS]. The activations of the [EOS] at the highest layer will be treated as the feature representation of the text and then linearly projected into the shared embedding space.

Music Spectrum Encoder: We use a ResNet-50 image encoder as music spectrum encoder [13]. Unlike the raw version, we change the global average pooling layer into an attention pooling mechanism, where a “transformer-style” QKV attention[14] is implemented. The image representation output from the ResNet encoder will serve as a query.

Music Audio Encoder: ESResNeXt [15] serves as our audio encoder, which is used in AudioClip [16]. Based on ResNet-50, ESResNext includes a trainable time-frequency transformation with complex frequency B-spline wavelets [17].

2.2. Tri-modal Contrastive Learning

In this section, we will introduce our training algorithm in detail. Given a batch of labeled input music sequences, we denote the i -th music audio sequence as A_i , the corresponding text label as T_i . Then a music spectrum S_i could be calculated by Short-Time Fourier Transform (STFT).

Then we can get the embeddings of the three modals by forwarding the input into their encoders:

$$E_{A_i}, E_{T_i}, E_{S_i} = F_{\text{aud}}(A_i), F_{\text{txt}}(T_i), F_{\text{spec}}(S_i). \quad (1)$$

The core idea of contrastive learning is to pull the similar embeddings closer while pushing the other embeddings away. In our scenario, we pull the E_{T_i} closer to E_{A_i} and E_{S_i} while push the E_{T_i} far away from E_{A_j} and E_{S_j} , where j -th example is another music sequence in the same batch. Now we get the optimization goal for modal Q and K :

$$L(i, j; \theta_Q, \theta_K) = -\log \frac{\exp(D(E_{Q_i}, E_{K_i})/\tau)}{\sum_{j=1, j \neq i}^N \exp(D(E_{Q_i}, E_{K_j})/\tau)}, \quad (2)$$

where N is the batch size and τ is a learnable parameter. θ_Q is the parameter of the encoder for modal Q and θ_K is the parameter of the encoder for another modal K . D is a distance function to evaluate the similarity between E_{Q_i} and E_{K_i} . As

Algorithm 1: Contrastive Learning of MUSER.

Data: all labeled training music sequence \mathcal{A} , text \mathcal{T} , spectrum \mathcal{S} pairs, and label \mathcal{Y} .
Input: encoders $F_{\text{aud}}, F_{\text{txt}}, F_{\text{spec}}$; weights W_a, W_t, W_s ; temperature parameter τ ; batch size n .

```

1 while not done do
2   Sample batches  $(A_i, T_i, S_i, Y_i) \sim (\mathcal{A}, \mathcal{T}, \mathcal{S}, \mathcal{Y})$ .
3   for all  $(A_i, T_i, S_i, Y_i)$  do
4      $E_{A_i}, E_{T_i}, E_{S_i} \leftarrow F_{\text{aud}}(A_i), F_{\text{txt}}(T_i), F_{\text{spec}}(S_i)$ .
5      $E_{A_i}, E_{T_i}, E_{S_i} \leftarrow W_a E_{A_i}, W_t E_{T_i}, W_s E_{S_i}$ .
6     Compute logits  $\hat{Y}_{AT}, \hat{Y}_{TA}, \hat{Y}_{ST}, \hat{Y}_{TS}$  as: e.g.,
7        $\hat{Y}_{AT_i} = E_{A_i} E_{T_i} e^\tau$ . Compute losses
8        $\ell_{AT_i}, \ell_{TA_i}, \ell_{ST_i}, \ell_{TS_i}$  as: e.g.,
9        $\ell_{AT_i} = \text{CrossEntropy}(\hat{Y}_{AT_i}, Y_i, axis = 0)$ .
10      Compute overall loss
11       $\ell_i = (\ell_{AT_i} + \ell_{TA_i} + \ell_{ST_i} + \ell_{TS_i})/4$ .
12    end
13    Update encoders and weights with loss  $\mathcal{L} = \sum_i \ell_i$ .
14 end

```

depicted in Fig.1 , for all three modals, we optimize them together with asymmetric contrastive loss:

$$\mathcal{L}(i, j; \theta_A, \theta_T, \theta_S) = \mathcal{L}(i, j; \theta_A, \theta_T) + \mathcal{L}(i, j; \theta_S, \theta_T), \quad (3)$$

where $\theta_A, \theta_T, \theta_S$ represent the parameters of audio encoder, text encoder and music spectrum encoder, respectively. A more detailed training algorithm is illustrated in Alg. 1.

3. EXPERIMENT SETUP

We experiment with different music benchmarks with an extra small music dataset for pre-training. For fair comparisons, we include state-of-the-art methods as baselines and report results under the same evaluation settings.

3.1. Benchmark Tasks

Two downstream music-related tasks could benchmark the performance of music sequence representation learning: (1) genre classification (2) automatic tagging.

3.1.1. Genre Classification

Genre classification involves assigning the most appropriate genre from a given song. We choose GTZAN as the dataset for this task, which contains 1,000 tracks of 30-second length. In consideration of several annotation errors on the dataset [18], we adopt the “fault-filtered” split [19] to minimize the impact of error labels. The filtered training set (443 tracks) is regarded as a part of the fusion training set. We use accuracy as the evaluation metrics.

Table 1. Datasets for Music Sequence Pre-training.

Method	Source	Audio / Text
VGGish[21]	YouTube-8M	350000h / 8M
CLMR[22]	Not mentioned	2200h / 260k
CALM[23]	Jukebox	240000h / 1.2M
MUSER	FMA (small subset)	66.7h / 8k
	MTT (train)	127h / 15k
	GTZAN (train)	3.7h / 0.4k
Total		195.8h / 23.5k

3.1.2. Automatic Tagging

Automatic tagging aims to find the most appropriate tags for target songs, also viewed as a multi-label classification problem. We choose MagnaTagATune (MTT) [20] as the benchmark dataset. To ensure enough training data for each tag, we limit the vocabulary to the top 50 most popular tags. We use the standard 12:1:3 train/val/test split for MTT [7]. Two macro-averaged over tags metrics are reported: area under the receiver operating characteristic curve (ROC-AUC), and average precision (AP).

3.2. Pre-training Datasets

Free Music Archive (FMA): FMA is a large-scale dataset for evaluating several tasks in Music Information Retrieval. We use the small subset of FMA, a balanced subset containing 8,000 clips. We rely on templates to concatenate the genre, parent genre, and top-level tag of each audio together.

Traditional models can only be trained on a single dataset, which tends to result in poor generalization performance. In contrast, the powerful transformer-based text encoder enables MUSER to fuse all kinds of datasets with different annotations together. We fuse two benchmark datasets with an extra FMA dataset for pre-training.

In total, we use around 23,500 clips as pre-training data, only 0.056% of the data for the state-of-the-art pre-training method. The details are depicted in Table 1.

3.3. Baseline Methods

A handful of recent music sequence representation learning methods will be included in our comparison:

VGGish [21]: Pre-trained on a large-scale video dataset (AudioSet [24]) with a classification task, VGGish is the first to explore the best performance of fully connected DNNs using CNN Architectures.

CLMR [22]: CLMR also noticed the problem of expensive music labels. It is the first work to introduce the contrastive pre-training techniques, which enable unsupervised music sequence representation learning.

CALM [23]: CALM is first proposed for unconditional speech generation. It codifies a high-rate continuous audio

Table 2. Performance on music understanding benchmarks.

Method	Tags (AUC)	Tags (AP)	Genre (ACC)
VGGish	89.4	42.2	75.2
CLMR	89.4	36.1	68.6
CALM	91.5	41.4	79.7
AE only (MT., PT)	88.7	38.4	59.7
AE only (MT., PT+FT)	88.9	38.9	76.9
State-of-the-art	91.5	42.2	82.1
MUSER (AE only)	87.5	36.3	66.6
MUSER (w/o spec)	88.1	39.6	75.2
MUSER (PT)	88.7	41.6	72.6
MUSER (PT+FT)	89.5	43.0	82.5

¹ MT. refers to Multi-task.

² AE only refers to only with Audio-Encoder.

sequence into low-rate discrete codes. Then a language model is trained on resulting codified audio and optional meta-data to produce high-quality contextual representations.

3.4. Text Template Design

The core idea of our approach is to learn musical perception from text supervision formed by manual templates. Traditional classification methods map their metadata into a numeric *id* of label (e.g, 0 for “jazz” and 1 for “hip-hop”). This leads to inconsistency between different music datasets where similar tags can be represented by different numerical *ids*. Instead of label mapping, text input takes advantage of semantic similarity, guiding a fair distribution in embedding space.

Utilizing manually designed templates to combine music metadata into unified text input, MUSER could bridge the distribution gap. The template helps to restrict the decision boundary, making the embedding space more targeted. Finding an appropriate template is an art - requiring both domain expertise and an understanding of the text encoder’s inner workings. We find the text template “a song of {genre}, belongs to {tag}”. to be a good default that helps clarify the concepts in music metadata.

During testing, the similarity of music sequence with each class-filled text template will be calculated to find the most similar class.

4. RESULTS AND DISCUSSION

We first show the results of MUSER, compared with baseline pre-training methods. Then we investigate the text semantics learned by MUSER and provide suggestions for text template engineering. Finally, we study the high data efficiency of MUSER under few-shot settings.

4.1. Role of Text Encoder

Three settings are proposed to prove the text-supervision effect: (1) **MUSER (AE only):** Without text-supervision, we

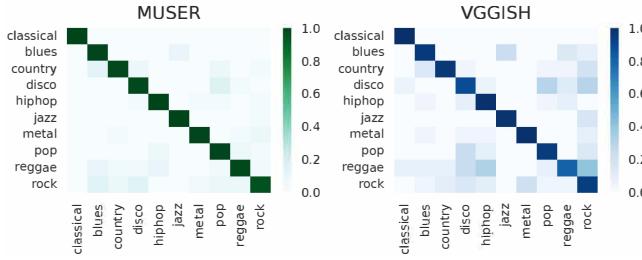


Fig. 2. Confusion matrices on GTZAN genre classification.

Table 3. Performance of different text templates on MTAT.

Template	Tags (AUC)	Tags (AP)
No template	88.5	41.9
“tags for the {genre} music is {tag}”	88.3	41.4
“the {genre} music is characterized by {tag}”	88.5	42.0
“a song of {genre}, belongs to {tag}, whose style is {style}”	89.5	43.0

convert the metadata into numerical labels and utilize the audio encoder for training and test. (2) **MUSER (PT)**: Pre-train MUSER encoders on the fused dataset, then directly test them on benchmark tasks. (3) **MUSER (PT+FT)**: After text-supervision pre-training, we continual fine-tune all encoders on benchmark dataset train set.

In Table 2, we observe that our MUSER (PT+FT) outperforms the state-of-the-art⁰ on both tagging and genre classification tasks. It is not surprising that without text supervision, only fine-tuning the audio encoder with numerical labels leading to poor performance. And a small portion of music domain data with MUSER pre-training can provide a good initialization for further transfer. The fine-tuning strategy also works in our contrastive setting and benefits a better result.

From Fig.2, directly learning from plain text, our MUSER can more easily distinguish similar genres like “rock” and “reggae”, where VGGish gets confused.

In Table 3, we analyze the influence of different text templates on MTAT. The selection of text templates is a key factor to our MUSER. Of several text templates we tried, “a song of {genre}, belongs to {tag}, whose style is {style}” performs best. We assume the optimal template could be found by reinforcement learning, left for a further study.

4.2. Role of Spectrum Encoder

We observe that the music spectrum encoder benefits contrastive pre-training. Generated by analyzing the music signal, the spectrum does not introduce additional information

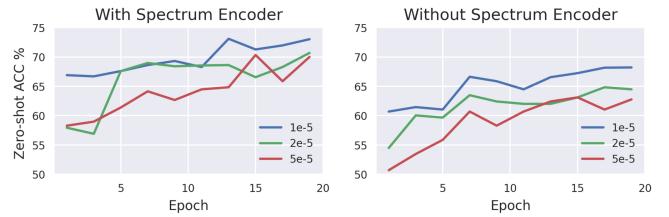


Fig. 3. Ablation experiments w/o spectrum encoder. Evaluate with zero-shot accuracy on GTZAN for the first 20 epochs.

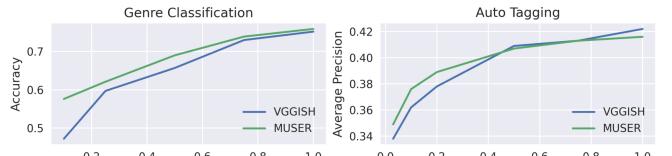


Fig. 4. Comparisons of fine-tuning efficiency on % ratio of training samples for different downstream tasks, with only FMA-small dataset for pre-training.

but enables the MUSER encoders to learn music from different views. From Table 2, MUSER(without spec) settings remove spectrum encoder during pre-training and fine-tuning, we find the spectrum brings a 7.3% improvement on genre accuracy and 3.4% improvement on tagging average precision.

4.3. Data Efficiency

In Fig.4, we illustrate the performance of our method and VGGish with limited training examples for downstream tasks. The quantity of pre-training data used is less than 0.1% of other pre-trained models. By less than 40% of training data, our pre-training method can exhibit a better few-shot ability than VGGish. We hypothesize traditional pre-trained methods need a new task-related network layer on top of them, which is hard to well-initialize with limited data.

5. CONCLUSIONS

In this paper, we explore the possibility of learning music sequence representation from text supervision with contrastive pre-training. Compared with SOTA pre-training methods, our MUSER can perform competitively on downstream tasks with far less labeled data. We also reveal that continual fine-tuning with task-related data can further improve performance. For future work, we are interested in exploring the performance boundary of our MUSER method with a large-scale music dataset. Furthermore, prompt-based fine-tuning is promising where the encoders remain fixed. A proper text template can contribute to SOTA performance. This technique may break the boundary between music sequence and natural language description, which helps flexibly generalize pre-trained models to data-limited downstream tasks.

⁰We only compared with pre-training models that use 30-second clips as training data for fair.

6. REFERENCES

- [1] Keunwoo Choi, György Fazekas, and M. Sandler, “Automatic tagging using deep convolutional neural networks,” in *ISMIR*, 2016.
- [2] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik M. Schmidt, A. F. Ehmann, and X. Serra, “End-to-end learning for music audio tagging at scale,” in *ISMIR*, 2018.
- [3] Minz Won, Sanghyuk Chun, and X. Serra, “Toward interpretable music tagging with self-attention,” *ArXiv*, vol. abs/1906.04972, 2019.
- [4] Hareesh Bahuleyan, “Music genre classification using machine learning techniques,” *ArXiv*, vol. abs/1804.01149, 2018.
- [5] J. Cramer, Ho-Hsiang Wu, J. Salamon, and J. Bello, “Look, listen, and learn more: Design choices for deep audio embeddings,” in *ICASSP*, 2019.
- [6] Jordi Pons and X. Serra, “musicnn: Pre-trained convolutional neural networks for music audio tagging,” in *ISMIR*, 2019.
- [7] Aäron van den Oord, S. Dieleman, and B. Schrauwen, “Transfer learning by supervised pre-training for audio-based music classification,” in *ISMIR*, 2014.
- [8] Jongpil Lee, Jiyoung Park, and Juhan Nam, “Representation learning of music using artist, album, and track information,” in *ICML*, 2019.
- [9] Qingqing Huang, A. Jansen, Li Zhang, D. Ellis, R. Saurous, and John R. Anderson, “Large-scale weakly-supervised content embeddings for music recommendation and tagging,” in *ICASSP*, 2020.
- [10] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever, “Learning transferable visual models from natural language supervision,” in *ICML*, 2021.
- [11] Teli Ma, Shijie Geng, Mengmeng Wang, Jing Shao, Jiasen Lu, Hongsheng Li, Peng Gao, and Yu Qiao, “A simple long-tailed recognition baseline via vision-language model,” *ArXiv*, vol. abs/2111.14745, 2021.
- [12] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, “Attention is all you need,” in *NIPS*, 2017.
- [13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in *CVPR*, 2016.
- [14] X. Wang, Ross B. Girshick, Abhinav Kumar Gupta, and Kaiming He, “Non-local neural networks,” in *CVPR*, 2018.
- [15] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel, “Esresne (x) t-fbsp: Learning robust time-frequency transformation of audio,” in *IJCNN*, 2021.
- [16] Andrey Guzhov, Federico Raue, Jörn Hees, and Andreas Dengel, “Audioclip: Extending clip to image, text and audio,” *arXiv preprint arXiv:2106.13043*, 2021.
- [17] Anthonio Teolis, “Computational signal processing with wavelets,” in *Applied and numerical harmonic analysis*, 1998.
- [18] Bob L. Sturm, “The gtzan dataset: Its contents, its faults, their effects on evaluation, and its future use,” *ArXiv*, vol. abs/1306.1461, 2013.
- [19] Corey Kereliuk, Bob L. Sturm, and J. Larsen, “Deep learning and music adversaries,” *IEEE Trans Multimedia*, 2015.
- [20] Edith Law, Kris West, Michael I. Mandel, Mert Bay, and J. S. Downie, “Evaluation of algorithms using games: The case of music tagging,” in *ISMIR*, 2009.
- [21] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al., “Cnn architectures for large-scale audio classification,” in *ICASSP*, 2017.
- [22] Janne Spijkervet and J. Burgoyne, “Contrastive learning of musical representations,” in *ISMIR*, 2021.
- [23] Rodrigo Castellon, Chris Donahue, and Percy Liang, “Codified audio language modeling learns useful representations for music information retrieval,” in *ISMIR*, 2021.
- [24] Jort F Gemmeke, Daniel PW Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R Channing Moore, Manoj Plakal, and Marvin Ritter, “Audio set: An ontology and human-labeled dataset for audio events,” in *ICASSP*, 2017.