Position Papers of the of the 17th Conference on Computer Science and Intelligence Systems pp. 13–18
DOI: 10.15439/2022F279
ISSN 2300-5963 ACSIS, Vol. 31

Searching For Loops And Sound Samples With Feature Learning

Jan Jakubik
Wroclaw University of Science and Technology
Faculty of Information and Communication Technology
Department of Artificial Intelligence
jan.jakubik@pwr.edu.pl

***Abstract*—In this paper, we evaluate feature learning in the problem of retrieving subjectively interesting sounds from electronic music tracks. We describe an active learning system designed to find sounds categorized as samples or loops. These retrieval tasks originate from a broader R&D project, which concerns the use of machine learning for streamlining the creation of videogame content synchronized with soundtracks. The method is expected to function in the context of limited data availability, and as such cannot rely on supervised learning of what constitutes an "interesting sound". We apply an active learning procedure that allows us to find sound samples without predefined classes through user interaction, and evaluate the use of neural network feature extraction in the problem.**

***Index Terms*—music information retrieval, machine learning, signal processing**

I. Introduction

THE USE of machine learning methods in Music Information Retrieval (MIR) has developed significantly in the past decade thanks to the improvements in machine learning areas such as deep neural networks [1] and increased availability of big data. We now have large datasets available for problems such as genre recognition and auto-tagging [2], emotion recognition [3], and more specialized problems such as pitch tracking have seen massive improvements too [4]. The most limiting factor for many narrow MIR problems remains the lack of massive training datasets required to train well-performing deep models.

This issue becomes more problematic when we consider the limitations of existing datasets in practical applications. Many methods are continuously developed on existing well-defined problems, but when new practical demands arise, it is often hard to find appropriate data. The MIR problems described in this work were defined in cooperation with a business partner interested in streamlining the creation of music-synchronized videogame content. Such content relies on the ability to trigger in-game events, such as playing particle effects or animations, in sync with the audio. One of the functionalities the developer was interested in was a retrieval system in which the creator points to a single example of a particular sound in the context of an existing music track, and all other occurrences of that sound can be automatically marked.

The desired scenario creates an ambiguity impossible to resolve with just a single sample. As such, we have opted for an active learning solution in which the annotations are obtained through the interaction of the developer with the retrieval results. Our goal was to limit the effort of the user in finding other occurrences of the sound (ideally working perfectly with just a single occurrence). The system was also expected to perform in an open-set recognition scenario, where sound types cannot be pre-defined.

In this work, we focus on empirical evaluation of deep feature learning to the described retrieval scenario. We have previously published early results relying on the use of feature extraction techniques considered standard for music audio in this task [5]. We build upon the earlier work by extending the method through the use of deep feature learning and evaluate the result on an improved version of the dataset. The improved dataset contains more annotations and a clear distinction between two categories of repeating sounds typically found in electronic music - *loops* and *samples*.

II. Related Work

Retrieval of sound effects has been an active topic in MIR, and we can relate our problem to multiple existing ones. Sound event detection and classification [7] were considered supervised tasks in multiple contexts, including non-musical ones. These supervised tasks are usually defined as retrieval of specific, predefined sound classes from large-scale collections, which makes the supervised methods ill-fitting for our problem. However, zero-shot learning approaches which relate to the way our task is defined found success in sound effect classification [8] and could potentially be applied to ours. These rely on a pre-trained deep neural representation and transfer learning. Source separation [9] and onset detection [10] are related to our task in that we need to separate the sample and detect its times of appearance. There are onset detection datasets [11] for a limited range of sounds, usually drums, that could potentially serve as a benchmark for that type of sound only. However, our base assumption about the desired functionality of the system is that there are no limitations on the types of sounds users can mark as interesting.

Loop discovery has been considered in several papers, although it is not a very active research topic. In the area of music structure analysis, there is a concern with finding repeated patterns [12]. However, more relevant to our application, there exists a loop retrieval approach based on

tensor decomposition introduced in [14]. The idea originates from [13] in which modeling of audio using predefined loops was attempted. The tensor decomposition approach improves upon that work by being able to decompose audio without loops being known in advance. It was later developed to create Unmixer [15], a publicly available system for fully unsupervised loop decomposition.

Active learning has been applied to MIR tasks on multiple fronts: genre recognition [16], mood recognition [17] and narrow tasks such as singing voice detection [18]. These works assume a scenario in which the goal is to maximize performance given a limited annotation budget. As such, they use a variety of metrics to select samples that are most beneficial to annotate. Metrics for sample selection can be broadly classified into two categories: based on uncertainty and based on correlation [19]. In the first case, the most uncertain samples or samples that would result in the largest model change are chosen. In the second case, samples are chosen to represent a significant subset of the data, e.g., each sample is representative of a particular cluster obtained in dataset clustering.

Unsupervised feature learning focuses on using large amounts of unannotated data to train general-purpose neural networks that can find use in downstream tasks with a small amount of training data. Early approaches to this problem usually utilize an encoder-decoder architecture [20] and the information bottleneck principle. A network that first encodes and then reconstructs the data implicitly creates a compact and robust lower-dimensional representation that leverages patterns within the unlabeled dataset. More recently, a lot of attention has been given to contrastive learning which leverages the power of data augmentation. A self-supervised network is trained by comparing data created from a single sample through different randomized augmentations, with the goal of creating a representation that is invariant to the augmentation. It has been shown that using the principle of contrastive learning alone is sufficient to train robust representations without labels for any supervised tasks [21].

III. Materials and Methods

Below we describe all data and methods used in the study. The dataset has been previously used in [5], but here it is developed further with the separation of two distinct types of sounds. The method of active retrieval and the feature representation it uses are descrived in subsections B-D.

A. Dataset Description

The dataset consists of 300 songs from the Creative Commons repository sampleswap.org. Audio files within the dataset are complete songs, ranging from 2 to 7.5 minutes in length. The songs have been selected from 4 musical genres (House, Dubstep, Drum&Bass and Downtempo) and annotated by three workers based on their subjective perceptions of interesting *sound samples* and *loops*. In the creation of electronic music, a *sound sample* is a pre-recorded sound that can be used for its interesting sonic qualities, while *loop* is a pattern that can seamlessly repeat, usually with both melodic and percussive components. Note that these aren't mutually exclusive, as any sound sample can be used within a loop, and any loop can be sampled. Our game developer partner was interested in retrieving both reused samples and actual loops, which lead to our attempt to develop a general method for any "standout" repeatable sounds, while still maintaining the distinction between both categories in the dataset.

When defining these concepts to the annotators, we asked them to consider *samples* to be audio effects and characteristic sounds that stand out against the musical background and can be heard at least twice in the same track, whereas *loops* were described as seamlessly repeating musical and rhythmic patterns. The annotations were created with 0.1-second precision. Within the *sample* category, annotators preferred short sounds: the majority of the sounds chosen were less than 3 seconds long. However, some persistent background sounds as long as 24 seconds were perceived and marked as a single sample of interest. *Loops* were longer on average, however, some loops as short as 2 seconds also occur in the dataset. Overall, there is a decent variety of what a potential user could understand as *samples* and *loops* represented within the dataset.

B. Active Learning Retrieval Approach

The desired system works as follows: given an audio file representation X, time of occurrence t_0 and duration d, the goal is to find a set of times $\{t_1, ..., t_n\}$ marking all other occurrences of this sound within the audio file. This search is performed according to Algorithm 1, which repeatedly polls the user with new retrieval results and then adds the responses to the growing set of positive samples P or negative samples N. The function $UserResponse(newsample)$ corresponds to the user giving a yes/no answer whether $newsample$ is a correct result. The algorithm is limited by $patience$, a parameter that represents the number of negative samples that can be returned before the user gives up on searching.

Algorithm 1 Active Retrieval Procedure

```
function RETRIEVE(X, t_0, d, patience)
    P ← {t_0}
    N ← ∅
    while |N| < patience do
        newsample ← GetBestSamples(X, P, N, d)
        if UserResponse(newsample) = true then
            P ← P ∪ {newsample}
        else
            N ← N ∪ {newsample}
        end if
    end while
    return P
end function
```

The key issue in defining a method for solving this problem is the implementation of the function $GetBestSamples$ which uses some representation of the sound file X, the set of samples identified as positive so far P and the set of samples

identified as negative so far N. This function should retrieve the most fitting candidate for a new positive sample, and return its time of occurrence. A natural choice for this function is nearest neighboor search in a feature space that represents the percieved similarity between sound excerpts well. In that case, the key element becomes the choice of the feature space.

C. Feature Representations of Audio

The baseline methods we present can be applied to multiple representations of audio, including a vector sequence obtained from a pre-trained deep learning model. In evaluation, we focus on the following vector sequence representations derived from the audio spectrogram:

1) Mel spectrogram: Mel spectrogram is an example of a spectral representation that takes the psychoacoustic properties of sound into account. Mel spectrogram transforms short frames of the signal into the frequency domain, using a logarithmically spaced Mel frequency spectrum. This corresponds to human perception of frequency better than the linearly spaced Short Time Fourier Transform. Mel spectrogram is not used directly as a feature representation (in preliminary tests, it achieved worse results than MFCC), but instead, serves as an input to a feature learning network described in subsection D.

2) MFCC: Mel-frequency Cepstral Coefficients are features commonly used in speech recognition that have found success in multiple MIR tasks. MFCC vectors capture timbral properties of sound well but lose precise frequency information.

D. Unsupervised Feature Learning

Our unsupervised feature learning setup combines autoencoding and contrastive learning losses. We have found that using only one of these was not sufficient, which will be elaborated on in Section IV. For the contrastive loss, we chose to base the network on the BYOL (Bootstrap Your Own Latent) approach [22], an evolution of the earlier SimCLR method [23]. For the autoencoding objective, we use a standard MSE reconstruction loss with no additional modifications.

1) Bootstrap Your Own Latent: The BYOL approach is an evolution of earlier contrastive learning methods, resulting from the observation that the previous methods took some unnecessary precautions from creating a loss with trivial, bad global optima. The loss $\mathcal{L}_{BYOL}$ for a pair of samples (x, x') created through data augmentation is written in a simplified form in Eq. 1:

$$\mathcal{L}_{BYOL}(x, x') = \|N(Pr(P(E(x)))) - N(P_f(E_f(x')))\|^2 \tag{1}$$

Two samples resulting from data augmentation: x and x', pass through four consecutive components: E denotes an encoder network, P denotes a projection network, Pr denotes a predictor network and N denotes vector normalization. The f subscript denotes the "frozen" version of components (i.e., not updated through gradient descent steps).

While network E is the feature extractor we are trying to obtain as the end goal of feature learning, other components exist to improve the training procedure. The projection layer P is optional but has been shown by the authors of the original SimCLR paper to improve the quality of trained representations in downstream tasks. The predictor network Pr helps prevent the collapse of training by making the architecture asymmetric. Unlike earlier contrastive learning methods, BYOL does not explicitly prevent a collapse to a bad global optimum in its loss (for example, if the encoder E outputs the same vector for any input, the MSE could be easily reduced to 0). However, the creators of the method have shown empirically that in a practical setting, with random initialization and gradient descent training of the N, P, and Pr components, the training procedure is not expected to collapse.

E, P and Pr networks are trained with gradient descent. E_f and P_f components are instead updated as an exponential running mean of respectively E and P. I.e., the parameters θ_f of a frozen network are updated based on the parameters θ of a respective unfrozen network, using Eq. 2 with a hyperparameter $\alpha \in (0, 1)$:

$$\theta_f = \alpha\theta_f + (1 - \alpha)\theta \tag{2}$$

2) Autoencoder Network: For autoencoder training objective $\mathcal{L}_{AE}$, we use the simplest possible formulation, as shown in Eq. 3:

$$\mathcal{L}_{AE}(x) = \|D(E(x)) - x\|^2 \tag{3}$$

The loss is calculated on an example x using encoder network E and decoder network D.

IV. Results

The results presented below are obtained through Algorithm 1 with $patience$ set to 5 and nearest neighboor implementation of the function $GetBestSamples$. Between different experiments, we only change the representation of sound supplied to the algorithm.

Implementation we use to obtain our results utilizes librosa [24] for the extraction of the audio features: Mel-spectrogram and MFCC. All features were extracted at a 22kHz sampling rate, with default parameters for the size of spectrogram frames (window sizes of 1024 and hop lengths of 512, Hamming window). The neural network was implemented and trained in Pytorch [25]. Matrix and vector computations are performed in NumPy [26], and for more computationally expensive matrix operations (distance calculations for determining the nearest neighbor) we also use Pytorch. On a system with an NVIDIA 2080Ti GPU, the use of GPU for distance calculations results in a significant speedup of approximately 2x when processing the entire dataset for evaluation.

For the encoder neural network in both BYOL and autoencoder approaches, we use a 5-layer convolutional neural network with kernel size 3 and 128 channels in each layer. For the decoder module in autoencoder, and the projection

and predictor modules of BYOL, we found 2 layer convolutional networks sufficient, and adding layers to those modules resulted in no improvements. The network is trained for 1000 iterations with Adam optimizer and 256 batch sizes.

For BYOL augmentations we have tested the following: addition of Gaussian noise (mean 0, standard devaitaion 0.3),randomly removing 20% of the 128 Mel-spectrogram frequency bins, transposition by a fixed number of mel frequency bins and an augmentation based on Harmonic-Percussive Sound Separation (HPSS). The last augmentation applies HPSS to separate the harmonic components from one of the samples in pair (x, x') in the loss function of BYOL (Eq. 1). Final results are obtained with a combination of all agumentations.

As our end goal is to find all occurrences of the sound, the key figure of merit is recall, and as the task is practically oriented, our evaluation is based on the recall achievable within given $patience$. Precision of the system is less relevant, as $patience$ directly limits how many false-positive answers can occur.

To contextualize the following results, we measured the results of a naive nearest neighboor approach without the use of active learning. The naive baseline achieves a recall of 0.3 on the Samples subset of the data, and 0.51 on the Loops subset of the data.

A. Results In The Samples Category

Results in the audio samples category are shown in Fig. 1, including recall over specific genres. A learned feature extractor outperforms the standard MFCC feature extractor. The improvements are seen mainly in Drum&Bass and Downtempo songs. Within the House genre, retrieval achieves equally good results for both approaches, which can be largely explained by the low structural complexity of songs in this part of the dataset (several tracks have a single repeating loop as a baseline for the entire track, which makes retrieval significantly easier). We can see that the main difficulty appears in the Dubstep genre, where feature learning achieves no improvement.

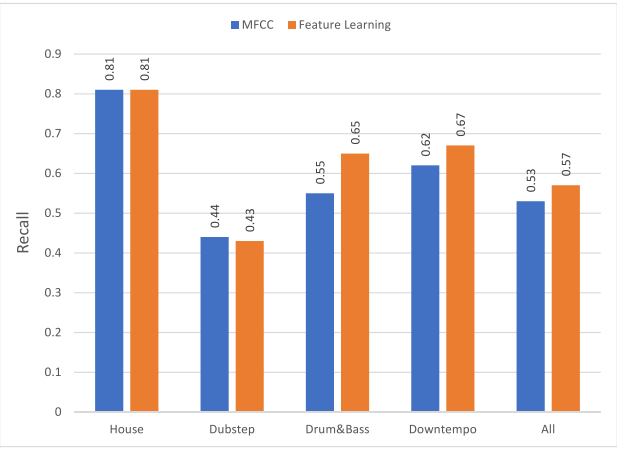

Fig. 1. Results in the *sound samples* category

B. Results In The Loops Category

Results in the audio loops category are shown in Fig. 2, including recall over specific genres. We can see that the retrieval of loops can be significantly easier than finding audio samples, likely stemming from the fact that by our definition loops share melodic and rhythmic qualities while for samples, the musical background can vary a lot. The feature learning approach improves over the MFCC features in overall results, and the improvements are seen within every genre. Much like in the sample searching task, the highest performance is seen in the House genre, and Dubstep is an outlier in being significantly harder than other genres.

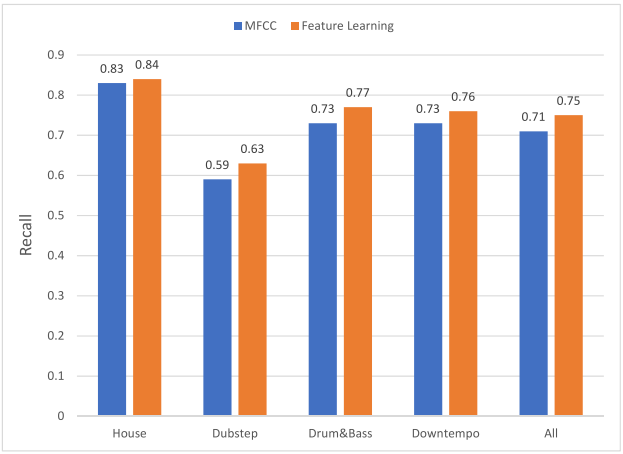

Fig. 2. Results in the *loops* category

C. BYOL vs. Autoencoder

Fig. 3 shows the comparison of results depending on the chosen loss function in both samples and loop categories, which motivated us to choose a combination of BYOL and autoencoder loss. While the BYOL loss alone is insufficient and autoencoder is enough to outperform MFCC features, the best performance is achieved when using a combination of both approaches.

D. Augmentation Choice

Fig. 4 shows the comparison of results depending on the choice of augmentation in BYOL training. The augmentation selection for BYOL ended up being less crucial than expected. This is especially seen for Harmonic-Percussive Separation, which we expected to improve the results by helping the extractor to focus on respectively percussive (more significant for sound samples) or melodic (more significant for loops) components of the sound. In practice, simple augmentations such as dropping frequency bins or adding Gaussian noise are enough to improve the results over autoencoder alone and the use of highly computationally complex HPSS isn't justified by the results.

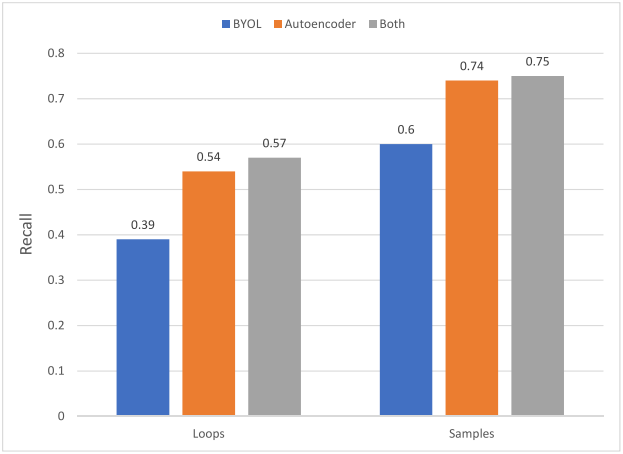

Fig. 3. Reuslts deppending on feature learning approach

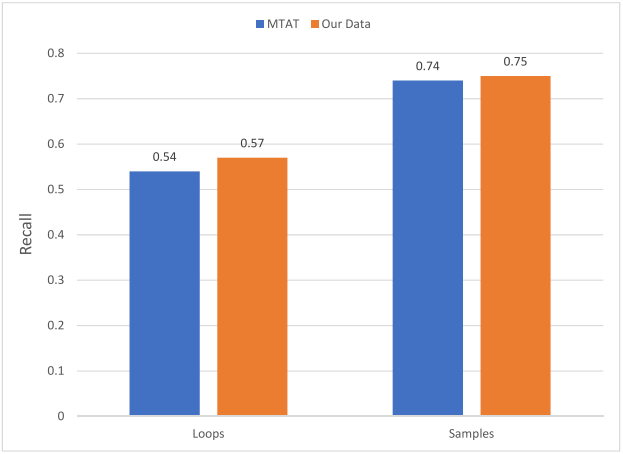

Fig. 5. Comparison of training the feature extractor on MTAT and our dataset

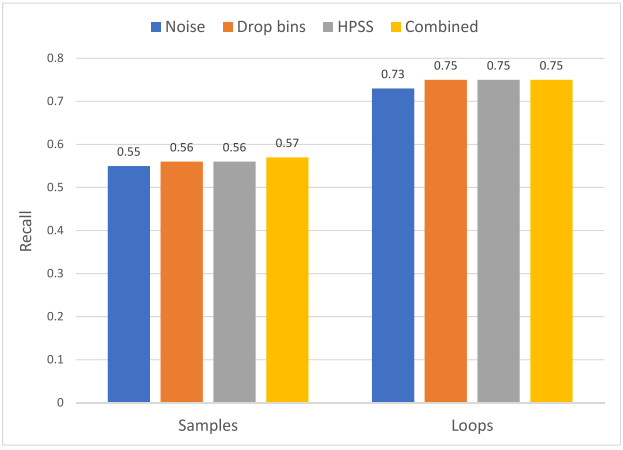

Fig. 4. Results depending on augmentations used in BYOL

E. More Representative vs. Bigger Training Data

As the key problem that motivates the use of feature learning is limited data availability, we also compare the results of training on a larger available dataset against the use of a small, but representative dataset. For this comparison, we used our sampleswap.org evaluation data (without labels) to train the "representative" model, while the "bigger" model is trained on a set of 15000 songs from the MagnaTagATune dataset. MagnaTagATune includes electronic music, but is not focused on it, and contains many genres irrelevant for our evaluation set such as classical and folk music. In Fig. 5, we compare the results. As can be seen, we achieve a similar retrieval quality with both approaches, but hand-selected representative data slightly outperforms training on a larger, but non-representative dataset.

V. Conclusions And Future Work

We have demonstrated a feature learning approach improving on our earlier work on the retrieval of subjectively interesting sound excerpts. The task concerns using active learning and user interaction to find multiple occurrences of an interesting sound in a music piece.

To facilitate a better evaluation of our results, we have developed our dataset to separate two categories of potential interesting "sound components" of electronic music. *Loops* are repeating melodic and rhythmic patterns commonly used by electronic music composers, while *sound samples* are sounds that stand out against the musical background and do not have to be melodic or rhythmic in nature. We have found that *sound samples* are significantly harder to find using our approach and may require further effort to separate well from the musical background.

The learneble feature extractor we used was a neural network trained in a fully unsupervised manner, using the principles of autoencoding and contrastive learning. We have found that this approach improves results when compared to a MFCC representation. A more detailed examination of the method's performance shows a number of conclusions. For best performance, autoencoding and BYOL approaches fto feature learning can be combined. Neither of these achieves the best results alone. In BYOL, an agumentation-based contrastive learning approach, the choice of augmentation affected the results, but the effect was not crucial to achieving good results. We have found that training on a small, but representative dataset was better than using a larger dataset with wide variety of music.

Future directions of development could include improvements of the neural network architecture, and development of the unsupervised learning method to achieve a representation that better separates ditinct sound components.

VI. Acknowledgments

This research was carried out in cooperation with Scalac sp. z o.o. and Vixa Games game development team as a part of the *Elaboration and implementation of audio modules (audio tracks indexing and analysis) and video modules (visualization of processed data) involving unique sound and*

graphic design use for the multimedia applications requirements RPPM.01.01.01-22-0011/17 project co-funded by the European Union.

REFERENCES

[1] E. J. Humphrey, J. P. Bello, Y. LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics," in *ISMIR 2012*, pp. 403-408.

[2] M. Defferrard, K. Benzi, P. Vandergheynst, X. Bresson, "FMA: A dataset for music analysis," arXiv preprint arXiv:1612.01840. 2017, https://doi.org/10.48550/arXiv.1612.01840

[3] Y. A. Chen, Y. H. Yang, J. C. Wang, H. Chen, "The AMG1608 dataset for music emotion recognition," in *ICASSP 2015*, pp. 693-697, https://doi.org/0.1109/ICASSP.2015.7178058

[4] J. W. Kim, J. Salamon, P. Li, J. P. Bello, "Crepe: A convolutional representation for pitch estimation," in *ICASSP 2018*, pp. 161-165, https://doi.org/10.1109/ICASSP.2018.8461329

[5] J. Jakubik, "Retrieving Sound Samples of Subjective Interest With User Interaction," in *Proc. of the 2020 Federated Conference on Computer Science and Information Systems*, 2020, pp. 387-390, https://doi.org/10.15439/2020F82

[6] B. McFee, D. Ellis, "Analyzing Song Structure with Spectral Clustering," in *ISMIR 2014*, pp. 405-410, https://doi.org/10.5281/zenodo.1415778

[7] Kothinti, S., Imoto, K., Chakrabarty, D., Sell, G., Watanabe, S., Elhilali, M. (2019, May). "Joint acoustic and class inference for weakly supervised sound event detection," in *ICASSP 2019*, pp. 36-40, https://doi.org/10.1109/ICASSP.2019.8682772

[8] H. Xie, T. V. Huang, "Zero-Shot Audio Classification via Semantic Embeddings," in *IEEE/ACM Transactions on Audio, Speech, and Language Processing,* vol. 29, 2021, pp. 1233-1242, https://doi.org/10.48550/arXiv.2011.12133

[9] S. Makino, "Audio source separation," Springer, 2018.

[10] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, M. B. Sandler, "A tutorial on onset detection in music signals," in *IEEE Transactions on speech and audio processing*, vol. 13, no. 5, 2005, pp. 1035-1047, https://doi.org/10.1109/TSA.2005.851998

[11] R. Marxer, J. Janer, "Study of Regularizations and Constraints in NMF-Based Drums Monaural Separation", in *Proc. of the 7th Int. Conference on Digital Audio Effects (DAFx'13)*. Maynooth, Ireland, 2013.

[12] L. Lu, M. Wang, H. J. Zhang, "Repeating pattern discovery and structure analysis from acoustic music data," in *Proc. of the 6th ACM SIGMM Int. Workshop on Multimedia Information Retrieval*, 2016, pp. 275-282, https://doi.org/10.1145/1026711.1026756

[13] P. López-Serrano, C. Dittmar, J. Driedger, M. Müller, "Towards Modeling and Decomposing Loop-Based Electronic Music," in *ISMIR 2016*, pp. 502-508.

[14] J. B. L. Smith, M. Goto, "Nonnegative tensor factorization for source separation of loops in audio," in *ICASSP 2018*, Calgary, Canada, pp. 171–175, https://doi.org/10.1109/MSP.2018.2877582

[15] J. B. L. Smith, Y. Kawasaki, M. Goto, "Unmixer: An interface for extracting and remixing loops," in *ISMIR 2019*, Delft, Nethedlands, pp. 824–831, https://doi.org/10.5281/zenodo.3527938

[16] C. Chen, S. Xin, "Combined Transfer and Active Learning for High Accuracy Music Genre Classification Method," in *2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, IEEE, 2021, https://doi.org/10.1109/ICBAIE52039.2021.9390062

[17] A. Sarasúa, C. Laurier, P. Herrera, "Support vector machine active learning for music mood tagging," in *9th International Symposium on Computer Music Modeling and Retrieval (CMMR)*, London, 2012, https://doi.org/10.1007/s00530-006-0032-2

[18] W. Li, X. Feng, M. Xue, "Reducing manual labeling in singing voice detection: An active learning approach," in *2016 IEEE International Conference on Multimedia and Expo (ICME)* IEEE, 2016, https://doi.org/10.1109/ICME.2016.7552987

[19] Fu, Yifan, Xingquan Zhu, and Bin Li. "A survey on instance selection for active learning," in *Knowledge and information systems*, vol. 35.2, pp. 249-283, 2013, https://doi.org/10.1007/s10115-012-0507-8

[20] T. H. Hsieh, L. Su, Y. H. Yang, "A streamlined encoder/decoder architecture for melody extraction," in *ICASSP 2019*, pp. 156-160, https://doi.org/10.1109/ICASSP.2019.8682389

[21] J. Spijkervet, J. A.Y. Burgoyne, "Contrastive Learning of Musical Representations." arXiv preprint arXiv:2103.09410, 2021, https://doi.org/10.48550/arXiv.2103.09410

[22] Grill, J. B., Strub, F., Altché, F., Tallec, C., Richemond, P., Buchatskaya, E., Valko, M. (2020). Bootstrap your own latent-a new approach to self-supervised learning. Advances in Neural Information Processing Systems, 33, 21271-21284, https://doi.org/10.48550/arXiv.2006.07733

[23] Nguyen, K., Nguyen, Y., & Le, B. (2021). Semi-Supervising Learning, Transfer Learning, and Knowledge Distillation with SimCLR. arXiv preprint arXiv:2108.00587, https://doi.org/10.48550/arXiv.2108.00587

[24] B. McFee, C. Raffel, D. Liang, D. P. W. Ellis, M. McVicar, E. Battenberg, O. Nieto, "librosa: Audio and music signal analysis in python," in *Proc. of the 14th python in science conference*, pp. 18-25, 2015.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 8024-8035, https://doi.org/10.48550/arXiv.1912.01703

[26] C.R. Harris, K.J. Millman, S.J. van der Walt, "Array programming with NumPy," *Nature* vol. 585, pp. 357–362, 2020. DOI: 0.1038/s41586-020-2649-2, https://doi.org/10.1038/s41586-020-2649-2

[27] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," in *Hournal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011, https://doi.org/10.48550/arXiv.1201.0490