Deep Learning-Based Music Instrument Recognition: Exploring Learned Feature Representations

Michael Taenzer*, Stylianos I. Mimilakis*, and Jakob Abeßer

Fraunhofer IDMT, Semantic Music Technologies Group, Ilmenau, Germany
{tzr, mis, abr}@idmt.fraunhofer.de

Abstract. In this work, we focus on the problem of automatic instrument recognition (AIR) using supervised learning. In particular, we follow a state-of-the-art AIR approach that combines a deep convolutional neural network (CNN) architecture with an attention mechanism. This attention mechanism is conditioned on a learned input feature representation, which itself is extracted by another CNN model acting as a feature extractor. The extractor is pre-trained on a large-scale audio dataset using discriminative objectives for sound event detection. In our experiments, we show that when using log-mel spectrograms as input features instead, the performance of the CNN-based AIR algorithm decreases significantly. Hence, our results indicate that the feature representations are the main factor that affects the performance of the AIR algorithm. Furthermore, we show that various pre-training tasks affect the AIR performance in different ways for subsets of the music instrument classes.

Keywords: music instrument recognition, deep learning, representation learning

1 Introduction

Real world music recordings often consist of multiple music instruments that can be active simultaneously. Detecting individual instruments or instrument families is an important research problem in areas such as machine listening, music information retrieval (MIR), and (music) source separation. The problem of detecting and categorizing the active instruments is often referred to as automatic instrument recognition (AIR). Recent approaches to AIR are mostly based on deep convolutional neural networks (CNNs) [1–5].

One commonality in deep learning approaches for AIR is that they consist of three modules [1, 3–5], namely the pre-processing, embedding, and classification modules. The first module pre-processes and transforms the respective input music waveform into a compact signal representation. The most common transforms are the short-time Fourier transform (STFT) and related filter-banks such as Mel-bands [1, 3, 4, 6] and the constant-Q transform (CQT) [7]. Common operations as pre-processing steps are harmonic and percussive separation [3] as well as logarithmic magnitude compression and data normalization or standardization [4].

* Equally contributing authors

The second module, referred to as embedding, accepts as input the pre-processed and transformed music waveform from the first module. It yields a feature *representation* that is used to condition the last module, i. e., the classification module, which is responsible for computing the posterior, i.e., the label probability, of the corresponding instrument classes (e.g., "electric guitar" or "piano"). Most often, the embedding and classification modules are learned jointly during a training procedure that is based on supervised learning, in which the class labels for each recording are given from a curated dataset [1, 4].

Regarding the embedding module, a common ingredient in the related literature is the usage of CNNs [1, 3, 4] and, more recently, CNN-based attention mechanisms [5, 8]. The approaches employing attention mechanisms are experimentally shown to yield state-of-the-art results, and it is assumed that the attention mechanism is responsible for the success of the methods. However, the studies presented in [5] and [8] condition the attention mechanism on a feature representation that is computed using a *pre-trained* CNN: That CNN, in particular the VGGish network [9], is initially trained for audio event detection (AED) in a supervised way using general audio signals and classes obtained from Audio Set [10], before being applied on the task of AIR. This means that the attention-based approaches to AIR make use of transfer learning [11, 12]. This differs from other approaches, which learn the representations jointly for the task of AIR [1, 3, 4]. Therefore, it could be argued that the observed increase in performance of such attention-based models rather needs to be attributed to the discriminative power of the feature representations from the CNN, which was previously learnt from more general audio signals instead of solely music signals [13].

In this work, we analyze the impact of the role of learning feature representations for an attention mechanism for music instrument classification performance. It should be noted that it is not our intention to conduct a comparative study on attention mechanisms versus representation learning, as we believe that both are equally beneficial for the task at hand. Instead, we aim to show that deep learning approaches to AIR can substantially benefit from employing representations that are learned using reconstruction or alignment optimization objectives [14] as well as datasets that contain general purpose audio signals [10].

To answer our research question, we focus on the attention-based model presented in [5], which is trained and tested on the respective subsets of the OpenMIC dataset [15]. To investigate the influence of different feature representations, we experiment with various commonly used filter-banks, such as (log) Mel-spectrograms, and learned representations. For the latter case, different datasets and optimization objectives are used to pre-train the CNN responsible for yielding the feature representations. These are described in sections 3 and 4.1, respectively.

2 Attention-based Model

The attention-based model for AIR from the work presented in [5] is illustrated in Fig. 1 embedded into our general experimental pipeline as described in the following.

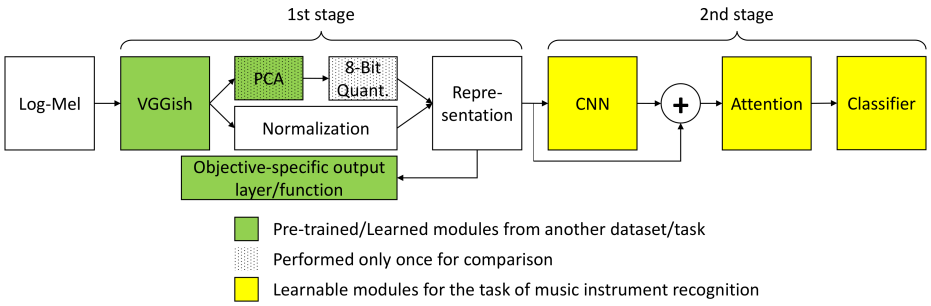

Fig. 1. An illustration of our experimental pipeline and the method presented in [5] that employs an attention mechanism and a pre-trained CNN (VGGish) for computing the feature representation(s).

2.1 Input pre-processing

An input time-domain music signal is first re-sampled at a sampling frequency of 16 kHz and then transformed into a time-frequency representation using the short-time Fourier transform (STFT). The parameters for the STFT are a window size of 25 ms using the Hann function and a hop-size of 10 ms. Each windowed segment is zero-padded to 512 samples. From the magnitude of the STFT, a Mel-spectrogram with 64 mel bands is computed. We apply log-magnitude scaling to the Mel-spectrogram, yielding a final input spectral representation denoted as "Log-Mel" in Fig. 1.

2.2 Post-processing & Representation

The Log-Mel is used to condition the VGGish network presented in [16]. This network comprises six convolutional (conv) blocks followed by three fully-connected feed-forward (dense) layers (FC). Each conv block consists of a two-dimensional conv layer (2Dconv), the rectified linear unit (ReLU) activation function, and a two-dimensional max-pooling operation. The numbers of kernels across the conv blocks are $\{64, 128, 256, 256, 512, 512\}$. The kernel sizes for the conv and max-pooling operations in all conv blocks are 3×3 and 2×2, respectively, and the stride size is set to 1. Furthermore, zero-padding is applied to preserve the size of the intermediate latent representations (activation maps), which are computed using the convolutions.

The outputs of each kernel in the last conv block are concatenated to a vector and then given as input to the first FC. The number of output units in each FC is $\{4096, 4096, 128\}$, respectively. The ReLU activation function is used after each FC. The output of the VGGish is a feature representation that summarizes approximately one second of spectral information into a single embedding vector [16].

This output representation is then post-processed by applying a whitening transform using principal component analysis (PCA). The bases for the PCA are *pre-computed* from the audio signals' corresponding representation obtained by the VGGish [15, 16] using the training subset of Audio Set [10]. This whitened feature representation is 8-bit quantized and mapped to the range of $[0, 1]$.

2.3 Additional CNN, Attention Mechanism & Classification

The above representation is processed by a block of 2D CNNs, which precede the attention module. It consists of three 2Dconv layers with unit stride and a group-normalization layer. Each layer employs 128 1×1-kernels. The output of the group-normalization layer is then updated by means of residual connections using the information of the post-processed representation.

The output of the residual connections is given to the attention module that consists of two 2Dconv layers with kernel size 1×1. The number of kernels in each 2Dconv layer is equal to the number of classes. The representation is fed to each 2Dconv layer in the attention module, followed by the application of the element-wise sigmoid activation function. The output of the conv layer responsible for decoding the attention embedding is normalized to unit sum with respect to the time-frame information. The output of the conv layer responsible for the class activity is used to gate the normalized output of the other conv layer.

Using this attention mechanism, the posterior can be computed by aggregating the time-information of the output of the attention mechanism, followed by the application of the hard-tanh function linear in the range of $[0, 1]$, equal to 1 for values > 1, and 0 for negative values. The aggregation is performed due to the weakly annotated labels contained in OpenMIC [5].

3 Datasets

To optimize the overall model parameters contained in each module described in Section 2, a two-stage training scheme is employed. In the first stage, the modules that are used to compute the feature representation, i. e., as illustrated in green color in Fig. 1, are pre-trained on a task different from AIR. The second stage uses the pre-trained modules from the previous stage, and optimizes the rest of the modules, i. e., the yellow modules illustrated in Fig. 1, using the labels associated with the task of AIR.

Table 1. Usage of different datasets for the respective training stages and objectives. Denoising from additive noise is indicated by $+$, and from multiplicative noise as $\odot$.

Datasets		
1st stage	**2nd stage**	**1st stage objectives**
Audio Set	OpenMIC	General purpose AED
NSynth	OpenMIC	Textual description, Denoising $+/\odot$
Freesound	OpenMIC	Tag Alignment

For the first stage, we employ the already optimized VGGish embeddings [15][1] pre-trained on the Audio Set [10], and the NSynth [17] and Freesound [18] datasets. Depending on the dataset for this stage, various pre-training objectives are used (see Section 4.1). For the second stage, we utilize only OpenMIC [15] for both training and testing, with the respective subsets used in [5]. Table 1 provides an overview of this.

[1] Publicly available under `https://github.com/cosmir/openmic-2018`

4 Experimental Procedure

Since the PCA and 8-bit quantization steps in the post-processing of the feature representation from the VGGish are irrelevant to the scope of our work, we have excluded them from our experiments. Instead, a simple normalization to $[0, 1]$ is applied to the representation during the second stage of training to avoid any crucial performance discrepancies due to the inductive biases of the attention-based method for AIR.

4.1 Pre-training Objectives (First Stage)

This section provides technical details regarding the experimental setup for each employed objective in the first training stage for optimizing the parameters of the VGGish. Table 1 gives a summary and overview. For all pre-training learning objectives, the Adam optimizer is used with a fixed learning rate of $1e^{-4}$. Furthermore, the batch size is set to 64 and an early stopping mechanism is used, which terminates the training procedure if the used criterion (validation loss) has not improved for five consecutive iterations throughout the entire training data. The maximum number of training epochs is 50. All parameters are initialized randomly with samples drawn from a uniform distribution and scaled using the method presented in [19].

Textual description One investigated pre-training objective is the prediction of the textual description of a music recording. This objective draws inspiration from the field of audio captioning, which aims at generating a textual description of an audio signal. Subject to the goal of this work, we employ the NSynth dataset [17] that contains the following textual descriptions of the musical notes for every recording in the dataset: {'bright', 'dark', 'distortion', 'decay', 'presence', 'multiphonic', 'modulation', 'percussive', 'reverb', 'rhythmic'}[2]. For using the textual descriptions of the music files to train the VGGish, we employ the Word2Vec language model, presented in [20] and pre-trained on an English vocabulary, to yield a vector representation of each description.

The Word2Vec model encodes each input word, in our case the textual description, into a 300-dimensional vector embedding. That vector is used as the target to learn the parameters of the VGGish. To do so, the output of the VGGish is given as input to a trainable batch-norm layer and two fully-connected feed-forward (dense) layers (FC). The first FC employs the non-linear tanh activation function, whereas the second does not employ any element-wise non-linear functions. The number of units in the FCs is set to 300. The VGGish network applied on an audio example from the NSynth dataset yields a single vector, because every NSynth example has a length of one second. Therefore, it is not necessary to aggregate over temporal information. The parameters of the VGGish and the following block of batch-norm and FCs are jointly optimized by minimizing the cosine loss between the predicted and target word-vector embeddings. The margin hyperparameter in the computation of the cosine loss is set to 0.5.

[2] We replaced the original NSynth descriptions {'fast decay', 'long release', 'nonlinear env' 'tempo-synced'} with {'decay', presence', 'modulation' 'rhythmic'}, based on the additional description contained in the dataset. This was due to the fact that the original descriptions could not be fully encoded by the employed language model.

Signal Recovery Another training objective we investigate is the recovery of the original Log-Mel spectrogram from a corrupted version of the signal. The goal of this objective is to enforce the representation from the VGGish to encode the relevant information contained in the Log-Mel. To do so, we employ denoising auto-encoders (DAEs) [21] and corrupt the Log-Mel in two different ways before it is input to the VGGish: a) with element-wise additive noise ($+$) drawn from a Gaussian distribution with zero mean and 0.1 standard deviation, and b) with element-wise multiplicative noise ($\odot$). For the latter case, random values are drawn from a Bernoulli distribution with $p = 0.5$.

To decode the representation of the corrupted Log-Mel obtained by the VGGish, we employ a single block of conv layers containing four transposed one-dimensional conv (1Dconv) layers. We use transposed 1Dconv layers to be able to recover (upsample back to) the original dimensionality regarding time-frames, which the VGGish reduced. The number of kernels and their size in each layer are $\{128, 64, 64, 64\}$ and $\{10, 21, 31, 37\}$, respectively. No zero-padding is applied between each convolution. Furthermore, the first three 1Dconv layers employ the leaky ReLU activation function with a leaky-factor of $\{0.1, 0.25, 0.5\}$, respectively. The last conv layer uses a linear activation function. These hyperparameters are chosen experimentally so that a reasonable convergence is achieved using a random and smaller subset of NSynth.

Audio & Tag Alignment We also explore the objective of aligning audio and associated tags. The alignment is achieved by maximizing the agreement of the computed audio and tag representations using a contrastive loss. We employ the tag encoder and the corresponding hyperparameters following the method presented in [14], whose goal is to compute representations that reflect acoustic and semantic characteristics of audio signals. For the audio encoder, we use the VGGish as discussed above. To match the dimensionality used by the audio tag encoder, we apply an affine transformation after the VGGish. The optimization hyperparameters for this configuration are taken from [14].

4.2 Downstream Instrument Recognition (Second Stage)

After optimizing the parameters of the VGGish with one of the above pre-training objectives, the VGGish computes the representations of the audio files contained in OpenMIC. Together with the corresponding labels within OpenMIC, these are then used to optimize the parameters of the CNN and the attention mechanism. To that aim, we use the existing splits of OpenMIC for training and validation as employed in [5].

For training, we use the binary cross-entropy loss function. The hyperparameters for optimization are the Adam algorithm with a learning rate of $5e^{-4}$, a batch size equal to 128 data points and a total number of 350 training epochs, following [5]. After every iteration over the entire training subset, we evaluate the model performance on the validation subset. After training, we select the best set of parameters based on the obtained evaluation score calculated in every iteration.

5 Evaluation

While the total number of audio files per instrument in OpenMIC is balanced, the number of positive and negative examples varies from one instrument class to another. For

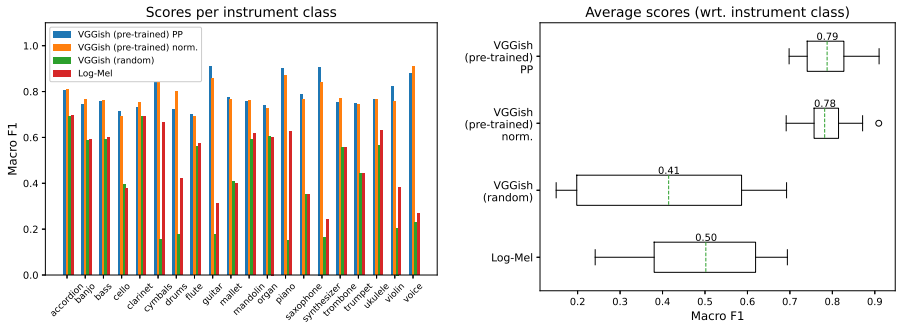

Fig. 2. F1-scores per instrument class (left) and overall (right) for the attention-based model [5], conditioned on feature representations computed using the pre-trained VGGish and a randomly initialized VGGish, and the Log-Mel representation. Note the marginal differences between post-processing (PP) and normalization (norm.) For VGGish (pre-trained), the Audio Set is used in the first training stage. For VGGish (random) and Log-Mel, no first training stage is performed.

this reason, we compute the macro-average F1-score (F1-macro) explicitly for positive and negative classes of every instrument class to evaluate both the parameters of the attention-based model and the pre-training stages, during both training and validation phases. During evaluation, the outputs of the classifier are subject to a post-processing operation that thresholds to zero values below 0.5 and unity values otherwise. Finally, to determine the benefits of each objective, we test the attention mechanism each time it has been trained with a different feature representation on the test subset of OpenMIC.

6 Results & Discussion

6.1 Representation Post-processing: Impact on performance

First, we examine the impact of the post-processing steps (see Section 2.2) versus normalization on classification performance, illustrated in Fig. 2. From the F-score, it can be seen that a simple scaling of the feature representation induces only a marginal performance drop. This allows us to omit further data-dependent post-processing stages that are irrelevant to our research question, yet might impose some performance discrepancies. From the barplot it can also be observed that without the PCA and quantization steps the performance increases marginally for the banjo, clarinet, drums, mandolin, trombone, and voice instrument classes.

6.2 Learned Representations: Impact on Performance

To highlight the impact of the learned representations on the performance for classifying musical instruments using the discussed attention mechanism, Fig. 2 shows the results from the attention-based model conditioned on three feature representations. These are computed from the pre-trained VGGish, a randomly initialized VGGish, and using the Log-Mel representation directly, i. e., the VGGish acts as an identity operator.

The Fig. 2 boxplot highlights two observations: First, regarding F1-macro, the discriminative power of the pre-trained VGGish is responsible for obtaining the best classification performance. Secondly, even an unoptimized (randomly initialized) VGGish can be used to compute a feature representation that yields a classification performance comparable to Log-Mel, which is a common feature representation for audio classification tasks. However, the barplot demonstrates that Log-Mel outperforms the representation from the randomly initialized VGGish for a few musical instruments classes including cymbals, ukulele, violin, and voice. These two tendencies suggest that the performance of the attention-based model may be accredited mostly to the discriminative power of the representation yielded by the VGGish. They also imply that different objectives or datasets may be used to pre-train the VGGish and yield different results.

Fig. 3 explores this observed direction with the results of the classification performance using the various objectives described in Section 4.1. As can be seen in the boxplot, the pre-training tasks of signal recovery and textual description provide significant improvements by 0.11 in the F-score over the randomly initialized VGGish. Compared to the Log-Mel features, marginal improvements of approximately 0.02 are observed. The barplot shows that each pre-training objective seems to be beneficial for different musical instrument classes. For example, Textual provides competitive results for the instrument classes mallet, mandolin, organ, ukulele, and voice, while Den $\odot$ seems to work well for piano, ukulele, and violin. Den $+$ provides improvements for percussive musical instruments such as cymbals and drums. In any case, the VGGish pre-trained on Audio Set (see Fig. 2) significantly outperforms the best performing models which employ NSynth.

Plausible explanations for these observed discrepancies lie in the amount of data and variability within Audio Set, and in the naiveness (in the sense of simple and not carefully devised) of the pre-training objectives, e.g. the signal recovery. This explanation is underlined when considering the results for the Align objective (Fig. 3), which employs Freesound and uses a more sophisticated mechanism to exploit the information of the audio tags.

Fig. 3 shows that the usage of larger corpora in conjunction with a more sophisticated objective (Align) can lead to significant improvements in attention-based AIR compared to the signal recovery and textual description objectives. Nonetheless, it is still sub-optimal compared to VGGish pre-trained on Audio Set. The discrepancy in performance between the two may be accredited to the data availability. However, this finding highlights the trend that using more general purpose audio datasets can improve the downstream task of AIR.

7 Conclusions

In this work, we investigated the importance of the role of learning representations w.r.t. an attention mechanism in music instrument classification algorithms. To that aim, we focused on the attention-based model for music instrument recognition presented in [5], and experimentally explored the impact of various feature representations on the performance of the attention-based model. Our experimental findings highlight the following trends: i) Discriminative objectives in conjunction with large scale and general purpose

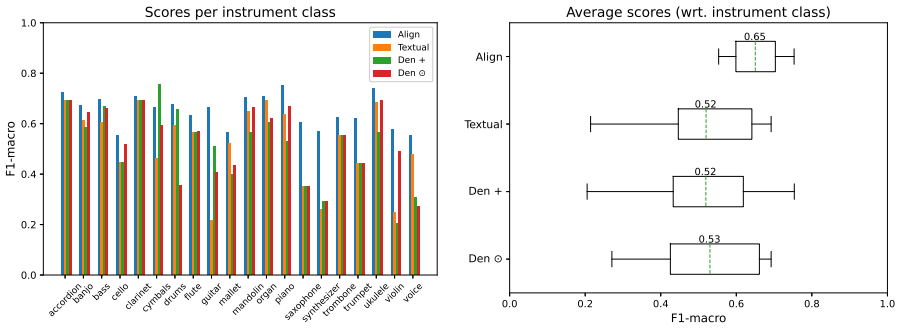

Fig. 3. F1-scores per instrument class (left) and overall (right) for the attention-based model [5], conditioned on feature representations computed from pre-training the VGGish for audio and tag alignment (Align), textual description (Textual), and signal recovery from additive noise (Den $+$) and multiplicative noise (Den $\odot$). The first training stage uses Freesound for Align, and NSynth for Textual, Den $+$ and Den $\odot$.

audio corpora are an important factor to be considered in AIR apart from the attention mechanism, ii) the usage of audio tags for computing representations is an attractive objective that yields competing performance, and iii) training objectives that take advantage of general audio and annotations or, respectively, the exploitation of multimodalities in a self-supervised manner are emerging directions for future research.

Acknowledgements This work has been supported by the German Research Foundation (AB 675/2-1).

References

1. Y. Han, J. Kim, and K. Lee. Deep Convolutional Neural Networks for Predominant Instrument Recognition in Polyphonic Music. *IEEE/ACM Transactions on Audio, Speech, and Language Processing (TASLP)*, 25(1):208–221, 2017.
2. T. Park and T. Lee. Musical Instrument Sound Classification with Deep Convolutional Neural Network Using Feature Fusion Approach. *arXiv preprint arXiv:1512.07370*, 2015.
3. J. Gomez, J. Abeßer, and E. Cano. Jazz Solo Instrument Classification with Convolutional Neural Networks, Source Separation, and Transfer Learning. In *Proceedings of the 19th International Society of Music Information Retrieval Conference (ISMIR)*, pages 577–584, Paris, France, 2018.
4. M. Taenzer, J. Abeßer, S. I. Mimilakis, C. Weiß, M. Müller, and H. Lukashevich. Investigating CNN-Based Instrument Family Recognition for Western Classical Music Recordings. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, pages 612–619, Delft, The Netherlands, 2019.
5. S. Gururani, M. Sharma, and A. Lerch. An Attention mechanism for musical instrument recognition. In *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, pages 83–90, Delft, The Netherlands, 2019.

6. S. I. Mimilakis, C. Weiß, V. Arifi-Müller, J. Abeßer, and M. Müller. Cross-Version Singing Voice Detection in Opera Recordings: Challenges for Supervised Learning. In *Proceedings of the 12th International Workshop on Machine Learning and Music (MML)*, pages 429–436, Würzburg, Germany, 2019.
7. Y.-N. Hung and Y.-H. Yang. Frame-Level Instrument Recognition by Timbre and Pitch. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 135–142, Paris, France, 2018.
8. K. Watcharasupat, S. Gururani, and A. Lerch. Visual Attention for Musical Instrument Recognition. *arXiv preprint arXiv:2006.09640*, 2020.
9. S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. J. Weiss, and K. Wilson. CNN architectures for large-scale audio classification. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 131–135, New Orleans, LA, USA, 2017.
10. J. F. Gemmeke, D. P. W. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. C. Moore, M. Plakal, and M. Ritter. Audio Set: An Ontology and Human-Labeled Dataset for Audio Events. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 776–780, 2017.
11. M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and Transferring Mid-level Image Representations Using Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1717–1724, 2014.
12. M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning Transferable Features with Deep Adaptation Networks. In *Proceedings of the 32nd International Conference on Machine Learning (ICML) - Volume 37*, pages 97–105, Lille, France, 2015.
13. Y. Bengio, A. Courville, and P. Vincent. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828, Aug. 2013.
14. X. Favory, K. Drossos, T. Virtanen, and X. Serra. Coala: Co-aligned autoencoders for learning semantically enriched audio representations. *arXiv preprint arXiv:2006.08386*, 2020.
15. E. J. Humphrey, S. Durand, and B. Mcfee. OpenMIC-2018: An Open Data-set for Multiple Instrument Recognition. In *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, pages 438–444, Paris, France, 2018.
16. A. Jansen, J. F. Gemmeke, D. P. W. Ellis, X. Liu, W. Lawrence, and D. Freedman. Large-Scale Audio Event Discovery in One Million YouTube Videos. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 786–790, 2017.
17. J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. *arXiv preprint arXiv:1704.01279*, 2017.
18. F. Font, G. Roma, and X. Serra. Freesound technical demo. In *Proceedings of the 21st ACM International Conference on Multimedia*, pages 411–412, New York, NY, USA, 2013.
19. X. Glorot and Y. Bengio. Understanding the Difficulty of Training Deep Feedforward Neural Networks. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS'10)*, pages 249–256, 2010.
20. J. Pennington, R. Socher, and C. Manning. GloVe: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Oct. 2014.
21. P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *Proceedings of the 25th International Conference on Machine Learning (ICML)*, pages 1096–1103, Helsinki, Finland, 2008. ACM.