TRANSFER LEARNING USING MUSICAL/NON-MUSICAL MIXTURES FOR MULTI-INSTRUMENT RECOGNITION

*H. Bradl, F. Pernkopf**

Graz University of Technology
Signal Processing and Speech Communication Lab.

M. Huber

sonible GmbH
Graz, Austria

ABSTRACT

Datasets for most music information retrieval (MIR) tasks tend to be relatively small. However, in deep learning, insufficient training data often leads to poor performance. Typical ways to overcome this problem are transfer learning (TL) as well as data augmentation. In this work, we compare various of these methods for the task of multi-instrument recognition. The classifier, a convolutional neural network (CNN) operating on mel-spectrogram representations of short audio chunks, is able to identify eight instrument families and seven specific instruments from polyphonic music recordings. Training is conducted in two phases: After pre-training with a music tagging dataset, the CNN is retrained using three multi-track datasets. Experimenting with different TL methods suggests that training the final fully-connected layers from scratch while fine-tuning the convolutional backbone yields the best performance. Training examples are produced on-the-fly by mixing of single-instrument tracks from the multi-track datasets. Two different mixing strategies – musical and non-musical mixing – are investigated. It turns out that a blend of both mixing strategies works best for multi-instrument recognition. The final classifier shows state-of-the-art F1-scores for classes with sufficient training data.

Index Terms— Multi-Instrument Recognition, Transfer Learning, Convolutional Neural Network

1. INTRODUCTION

Instrument recognition is an important task in MIR. The goal of early research was to identify single instruments in monophonic recordings. This task was usually approached using hand-crafted features and traditional machine learning algorithms [1, 2]. Later, research shifted to predominant instrument recognition in polyphonic recordings. Only recently, people started to think about the ultimate goal of instrument recognition; namely multi-instrument recognition – whose aim is to identify all instruments in a polyphonic mixture. Formally, this is a multi-label classification problem since multiple mutually non-exclusive labels can be assigned to each instance. Lately, deep learning became the preferred way to tackle multi-instrument recognition and most other problems in MIR. Commonly, music classification tasks – including multi-instrument recognition – are approached with CNNs using mel-spectrograms as input. Macro F1-scores of 85 % have been reported for multi-instrument recognition in classical music recordings [3]. On the *MedleyDB* [4] dataset, class-wise F1-scores up to 90 % were achieved using CNNs operating on mel-spectrogram representations [5]. Deep learning approaches using raw audio as input [6] are usually inferior to those which use 2D time-frequency representations. However, this might change in the future if datasets grow and computing power increases. Besides CNNs, other architectures have also been explored. For instance, an attention model [7] obtained a macro F1-score of 81 % on the *OpenMIC* [8] dataset.

Training of deep neural networks (DNNs) requires huge amounts of data and '. . . the amount of skill required reduces as the amount of training data increases' [9, p. 19]. However, even today's biggest datasets for multi-instrument recognition, such as *OpenMIC*, are small compared to datasets for other audio signal processing tasks. This is mainly due to copyright protection of music. TL is a good option to boost generalization of DNNs for MIR tasks with insufficient data. For pre-training, there are several large-scale audio and music datasets available [10, 11]. Promising results have been obtained with transfer of DNNs to various audio analysis tasks, after pre-training on a huge audio dataset [12]. For TL to be most effective, source and target task should be similar.

In this paper, we focus on multi-instrument recognition as a target task. For pre-training, we use a major music tagging dataset [13]. Music tagging is perfectly suitable as a source task, as it is highly related to multi-instrument recognition; in music tagging, labels typically include genre, instrumentation, mood etc., whereas in multi-instrument recognition labels are limited to instrumentation. Besides TL, we use data augmentation to overcome data scarcity. Fortunately, three major multi-track datasets have been released lately [4, 14, 15]. Although these multi-track datasets are much smaller than other audio datasets, they allow extensive data augmentation by mixing of the single-instrument tracks. In music source separation, good results have been obtained by randomly combining sources from different songs [16]. However, this data augmentation technique was never explored for multi-instrument recognition. Moreover, at least for humans, source separation and multi-instrument recognition are significantly easier in case of such random, non-musical mixtures, i.e. each instrument plays in a different key and tempo. In this work, we investigate if augmenting musical mixtures with non-musical mixtures while training DNNs results in performance gains for multi-instrument recognition. Furthermore, results on various non-musical and musical blending compositions are reported.

The paper is organized as follows. In Section 2 the DNN model and the data resources are introduced. Furthermore, details about data mixing and TL are discussed. In Section 3 experimental results for various TL methods and mixing strategies are presented. Section 4 concludes the paper.

*Thanks to the Austrian Research Promotion Agency (FFG) for partial funding.

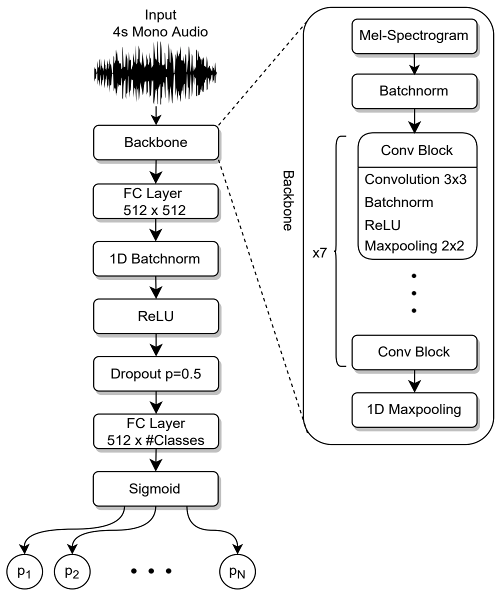

Fig. 1. Architecture of the CNN used for multi-instrument recognition.

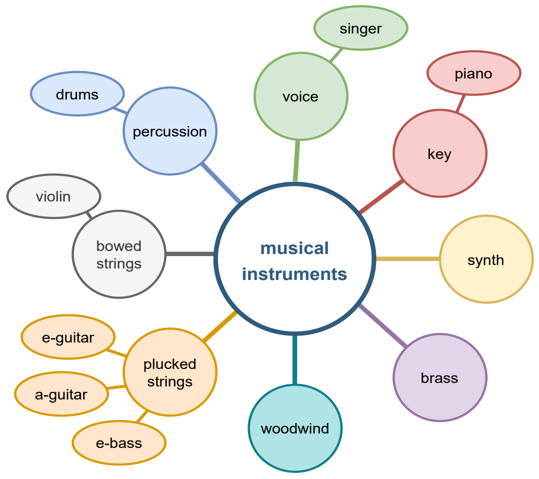

Fig. 2. Two-level instrument taxonomy.

2. METHOD

2.1. Model Architecture

The classifier's architecture (see Fig. 1) was adopted from [17] and is similar to a VGG-Net [18] commonly used for image classification. A stack of convolutional layers, hereafter referred to as *backbone*, is followed by fully connected layers. In the backbone, the time domain signal is converted to a mel-spectrogram with 128 frequency bins. A window length of 512 and a hop size of 256 are used to compute the spectrogram representation. After that, seven convolutional blocks are employed to find relevant patterns in this time-frequency representation. Each block consists of a 2D convolutional layer with a kernel size of 3×3 followed by batch normalization, ReLU activation and 2×2 max pooling. After seven convolutional blocks, the frequency dimension is reduced to one. Finally, 1D max pooling is applied over the time axis to shrink the time dimension to one as well. Since the last convolutional layer has 512 kernels, the resulting output representation of the backbone has a size of 512. The backbone extracts suitable representations from the mel-spectrograms. Subsequent fully connected layers learn non-linear combinations of these features to support the decision whether a certain class is active or not. Between the fully connected layers, batchnorm and dropout are applied. In the last layer of the classifier, a sigmoid activation function for each instrument class maps the model's predictions to values between zero and one, hence the outputs can be interpreted as probabilities.

2.2. Data Resources

For TL, a combination of three multi-track datasets is used – *MedleyDB* [4] (including *MedleyDB 2.0* [19]), *Mixing Secrets* [14] and *Slakh* [15]. After discarding songs which exhibit severe crosstalk between the individual instrument tracks, the first two datasets both feature approximately 130 songs and the last one contains 2100 songs. Note that, although the *Slakh* dataset is superior when it comes to size, several instruments sound quite unrealistic because all audio was synthesized from MIDI files using virtual instruments. In addition to the three multi-track datasets, *MTG-Jamendo* [13], a large music tagging dataset is utilized for pre-training of all models. This music tagging dataset already comes with a predefined split. However, each multi-track dataset was randomly divided into training, validation and test set containing 65 %, 17.5 % and 17.5 % of all songs, respectively. We made the decision to split the multi-track datasets at random for the sake of simplicity. This practice can potentially lead to overoptimistic results if songs from the same artist or album are contained in multiple splits. In order to reduce computation time, all audio data was converted to mono and downsampled to 32 kHz.

Since the three multi-track datasets used in this work exhibit different class structures, a unified instrument taxonomy has to be developed. Moreover, we want this taxonomy to be hierarchical in order to leverage relationships between instruments for learning broader concepts of musical instruments [20]. As a general rule for training DNNs, a sufficient number of examples for every class is required. In other words, classes with insufficient number of examples have to be discarded, which basically means that some information gets lost. However, our multi-track datasets are highly imbalanced and for certain instruments only few examples are available. To still utilize these underrepresented classes for training, our two-level hierarchy was defined as shown in Fig. 2. It is inspired by the *Hornbostel-Sachs* [21] system, which classifies musical instruments based on their underlying sound production mechanisms. As the physical principle of an instrument strongly correlates with its sound, such a taxonomy seems appropriate for our use case. The first level of our proposed taxonomy represents eight instrument families: voice, percussion, bowed string, plucked string, woodwind, brass, key and synth. The second level embodies seven specific instruments for which abundant data was available: singer, drums, violin, electric guitar, acoustic guitar, electric bass and piano. In total we have 15 classes.

2.3. Musical and Non-Musical Mixing

Our main data augmentation approach is to create new unique mixtures out of the single-instrument tracks (sources) from the multi-track datasets. Therefore, we propose two mixing strategies – *musical mixing* and *non-musical mixing*. Musical mixing means that only sources from the same song are combined to generate a new mixture, whereas non-musical mixing signifies that each source in a mix originates from a different song. The latter, which was successfully used in music source separation [16], obviously results in "non-musical", strange sounding mixes, as each instrument plays in a different key and tempo. Nevertheless, the number of training examples can be increased massively by this mixing strategy, which in turn improves generalization. Musical mixing, on the other hand, naturally produces well-sounding results, since all sources originate from the same song and are therefore uniform in tempo and key. In order to investigate the performance of the two proposed mixing strategies, an experiment is conducted in Section 3.3. Note that for validation and testing, musical mixing was used exclusively, because in real-world recordings, instruments will almost always play in the same key and tempo. In the following paragraph, the data mixing approach is explained in more detail, illustrating how a polyphonic mixture is constructed from individual sources.

Initially, one of the three multi-track datasets is selected – each one with the same probability of $1/3$. This step is necessary, because the *Slakh* dataset is much larger than the other two multi-track datasets. Simply merging them and sampling from the combined dataset would lead to a domination of *Slakh* samples. However, this should be avoided, because, as already mentioned, the *Slakh* dataset is synthesized from MIDI files and is therefore not the best to represent real-world music recordings. In the next step, one of the two proposed mixing strategies is selected. Musical mixing is applied with probability $p_{musical}$ and non-musical mixing is used with probability $1-p_{musical}$. Then we choose the number of sources N in the mixture. As our models should be able to cope with solo-instrument recordings as well, we load single sources ($N = 1$) with a certain probability $p_{single-source}$. After that, the following steps are performed N times for the non-musical mixing method: select a song, select a source from this song, select a four-second chunk from this source, apply some digital audio effects on the audio and add it to the mixture. Four digital audio effects from the *audiomentations* [22] library are utilized – each one with a certain probability: amplitude scaling (gain), three types of filters (high shelf, low shelf and peaking filter), pitch shifting and time stretching. The order of these effects as well as all their parameters are random. Note that, in the musical mixing case, all sources originate from the same song, hence a song is selected only once and not N times (at every loop iteration). Furthermore, the time position of all chunks has to be identical in order to obtain musical sounding results. Finally, pitch shifting and time stretching are omitted for the same reason when musical mixing is applied.

2.4. Transfer Learning (TL)

TL is the procedure of applying knowledge gained from solving one task to another, related task [23]. CNNs typically learn general concepts – like detecting edges or simple shapes – in the earlier layers and increasingly more complex, task-specific concepts in the later layers. Therefore, the first convolutional layers can easily be reused for a similar task. If source and target tasks are related, such as music tagging and multi-instrument recognition, even a complete transfer of all convolutional layers can be considered [24] as for both tasks, the CNN has to identify similar patterns in mel-spectrograms.

In order to train our classifier, we experimented with two TL methods. For both approaches, we first replaced the fully connected layers of the pre-trained model with new, randomly initialized ones. Furthermore, the size of the output layer was adjusted to 15 to suit the target task of predicting eight instrument families and seven explicit instruments, according to the taxonomy proposed in Section 2.2. During training with the multi-track data, we either (1) froze the whole backbone network and only retrained the fully connected layers or (2) allowed the backbone to be learnable as well, but with a smaller learning rate than the fully connected layers. As a third method (3) the entire model is trained with the multi-track data from scratch, i.e. not utilizing the pre-trained weights obtained by training with *MTG-Jamendo*. Strictly speaking, method (3) has nothing to do with TL, nevertheless we use it as a reference to show the benefits of methods (1) and (2). The results of these three experiments are discussed in Section 3.2.

3. EXPERIMENTS

3.1. Experimental Setup

Prior to all experiments, a CNN is trained on the 50 most popular tags from the *MTG-Jamendo* dataset. The following experiments use this pre-trained model as a starting point for TL. Since we generate training examples on-the-fly when working with the multi-track data, the concept of epochs – a complete pass over the training set – does not exist. Therefore, we defined the transit of 1920 samples (120 batches of size 16) as one epoch. After each epoch, the model is evaluated on the validation set and at the end of training (after 150 epochs), the model with the lowest validation loss is retained. For all experiments, the ADAM optimizer in combination with a step-wise learning rate scheduler was employed. Every 20 epochs the learning rate was reduced by a constant factor of 0.3. Binary cross-entropy served as a loss function. For testing, 120 *musical* mixtures were created on-the-fly from the individual instrument tracks contained in the test splits of the three multi-track datasets.

3.2. Transfer Learning (TL) Results

We investigate the three TL methods proposed in Section 2.4. Each model was evaluated on the testing data and area under the receiver operating characteristic curve (ROC-AUC), area under the precision-recall curve (PR-AUC) [25] and test loss were computed. For all three TL methods, 0.01 was chosen as an initial learning rate for the fully connected layers. When training from scratch, the same learning rate was used for the backbone as well. For the fine-tuning approach, the learning rate of the backbone was initialized with 0.0001. The ratio between muscial and non-musical mixes $p_{musical}$ was set to 0.6 for this experiment.

Table 1 contains the results of the three TL experiments. Unsurprisingly, the performance is worst when no pre-training is exploited and the model is trained from scratch (i.e. *From Scratch*) on the multi-track data. Utilizing a pre-trained backbone with frozen parameters (i.e. *Frozen Backbone*) significantly increases the performance, indicating that pre-training on a large-scale dataset is beneficial indeed. However, fine-tuning the weights of the backbone (i.e. *Fine-tuned*) to fit the needs of the target task yields additional performance gains.

	ROC-AUC	PR-AUC	Test Loss
From Scratch	0.8488	0.6812	0.4459
Frozen Backbone	0.8908	0.7623	0.4160
Fine-tuned	0.9429	0.8334	0.3618

Table 1. ROC-AUC, PR-AUC and test loss for different TL approaches.

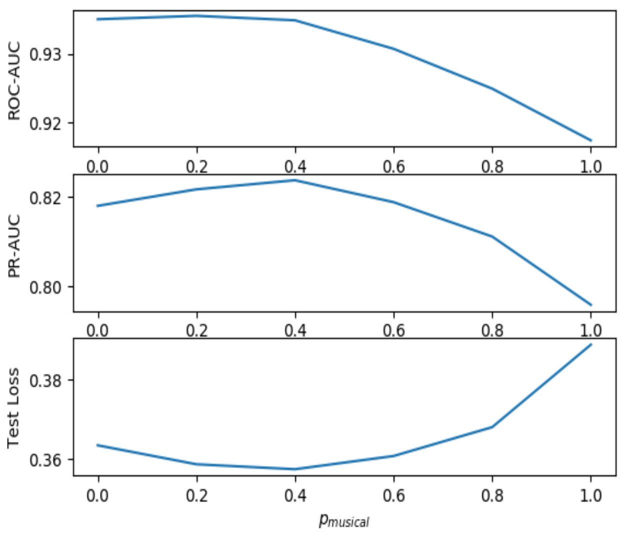

Fig. 3. ROC-AUC, PR-AUC and test loss for different values of $p_{musical}$.

3.3. Mixing Strategy

Our data augmentation technique (see Section 2.3), is based on mixing of individual sources to create new examples for training. For this reason, the two mixing strategies musical mixing with probability $p_{musical}$ and non-musical mixing with probability $1 - p_{musical}$ are used. To determine the best ratio between these two methods, we experimented with different values for the hyperparameter $p_{musical}$. As depicted in Fig. 3, the best performance was obtained for $p_{musical} = 0.4$. The result of this investigation suggests, that neither mixing strategy is superior, but rather a combination of both works best. Using only the non-musical mixing approach results in a larger variation of the training examples. However, the model does not get the chance to learn how to identify sources in "real" music where all instruments play in the same key and tempo. In this case, instrument recognition is usually more difficult though, at least for us humans, since masking of individual sources is more pronounced due to synchronous note onsets and overlapping partials. Although overlapping of partials results in consonant sounds, it is substantially harder to distinguish individual sources in this case. On the other hand, if only the musical mixing approach is utilized, fewer variation in the training examples can be created, as the number of possible combinations of sources from a single song is limited. This in turn can lead to poorer generalization of the model. We infer from this experiment, that a blend of both mixing strategies is optimal for multi-instrument recognition.

3.4. Performance of the Classifier

After setting the classification threshold for the sigmoid activations of the CNN outputs to 0.35 – this value maximizes the F1-score – we are able to report threshold-dependent metrics. Fig. 4 shows the class-wise F1-scores and accuracies of our final model. Unsurprisingly, classes with abundant training data, such as percussion, drums or plucked strings, exhibit the best F1-scores, while identification of instruments or families with insufficient data, like brass, woodwind or violin, does not work well. Note that the F1-score for the brass class is zero because the number of true positives is zero. In addition to F1-scores, we also report the class-wise accuracy. However, accuracy has to be used with caution, since it is highly dependent on the distribution of the dataset. For example, a high accuracy for the brass class is obtained, completely hiding the fact that not a single brass instrument was identified correctly. Instead, the model always predicts the absence of the brass family, which is true most of the time, since the brass class is very poorly represented in the data. Apart from that, we obtained state-of-the-art performance for the majority of classes. With F1-scores above $95\,\%$ for some classes, our classifier can easily compete with other recent multi-instrument recognition systems from the literature [7, 3, 5]. Unfortunately, performances reported by different researchers are generally hard to compare since there is no benchmark dataset for multi-instrument recognition.

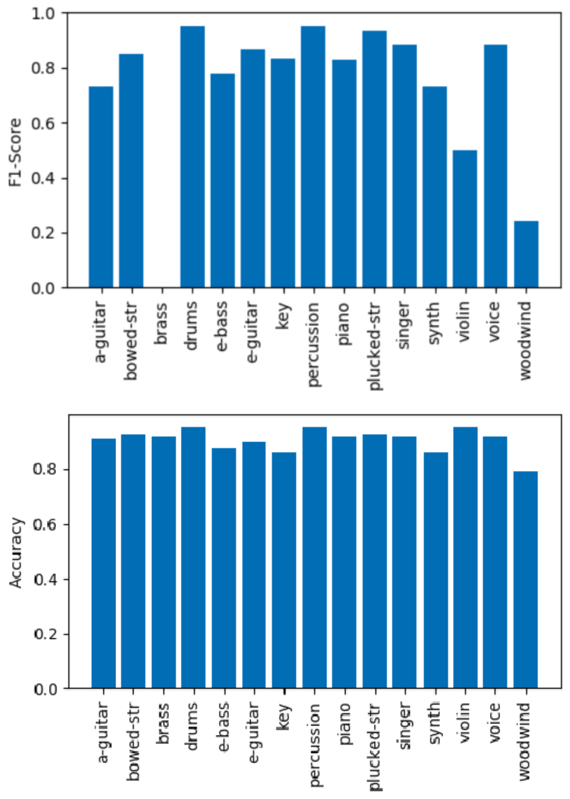

Fig. 4. Class-wise F1-score and accuracy of the classifier.

4. CONCLUSION

In this paper, we investigated how different TL methods perform when transferring pre-trained CNNs to the task of multi-instrument recognition. We show that pre-training on a music tagging dataset yields significant performance gains. For TL, training the fully connected layers from scratch and fine-tuning the convolutional layers worked best. Furthermore, we explored musical and non-musical mixing as strategies to generate training examples. For the task of multi-instrument recognition, a combination of both mixing strategies – using $40\,\%$ musical mixtures and $60\,\%$ non-musical mixtures – turned out to be ideal. In the end, we evaluated our final model and obtained class-wise F1-scores above $95\,\%$ for some classes, which is state-of-the-art for instrument recognition in polyphonic recordings.

5. REFERENCES

[1] Antti Eronen and Anssi Klapuri, "Musical instrument recognition using cepstral coefficients and temporal features," in *2000 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No. 00CH37100)*. IEEE, 2000, vol. 2, pp. II753–II756.

[2] Bertrand David, Gaël Richard, and Slim Essid, "Efficient musical instrument recognition on solo performance music using basic features," in *Audio Engineering Society Conference: 25th International Conference: Metadata for Audio*. Audio Engineering Society, 2004.

[3] Fabian Seipel, "Music instrument identification using convolutional neural networks," *TU Berlin, Institut für Kommunikation und Sprache, Fachgebiet Audiotechnologie, Masterarbeit*, 2018.

[4] Rachel M Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Pablo Bello, "Medleydb: A multitrack dataset for annotation-intensive mir research.," in *ISMIR*, 2014, vol. 14, pp. 155–160.

[5] Venkatesh Shenoy Kadandale, "Musical instrument recognition in multi-instrument audio contexts," M.S. thesis, MSc thesis, Universitat Pompeu Fabra, 2018.

[6] Peter Li, Jiyuan Qian, and Tian Wang, "Automatic instrument recognition in polyphonic music using convolutional neural networks," *arXiv preprint arXiv:1511.05520*, 2015.

[7] Siddharth Gururani, Mohit Sharma, and Alexander Lerch, "An attention mechanism for musical instrument recognition," *arXiv preprint arXiv:1907.04294*, 2019.

[8] Eric Humphrey, Simon Durand, and Brian McFee, "Openmic-2018: An open data-set for multiple instrument recognition.," in *ISMIR*, 2018, pp. 438–444.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, *Deep learning*, MIT press, 2016.

[10] 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 *2017 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2017, pp. 776–780.

[11] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere, "The million song dataset," 2011.

[12] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[13] Dmitry Bogdanov, Minz Won, Philip Tovstogan, Alastair Porter, and Xavier Serra, "The mtg-jamendo dataset for automatic music tagging," 2019.

[14] Siddharth Gururani and Alexander Lerch, "Mixing secrets: A multi-track dataset for instrument recognition in polyphonic music," *Proc. ISMIR-LBD*, pp. 1–2, 2017.

[15] Ethan Manilow, Gordon Wichern, Prem Seetharaman, and Jonathan Le Roux, "Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019, pp. 45–49.

[16] Stefan Uhlich, Marcello Porcu, Franck Giron, Michael Enenkl, Thomas Kemp, Naoya Takahashi, and Yuki Mitsufuji, "Improving music source separation based on deep neural networks through data augmentation and network blending," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 261–265.

[17] Minz Won, Andres Ferraro, Dmitry Bogdanov, and Xavier Serra, "Evaluation of cnn-based automatic music tagging models," *arXiv preprint arXiv:2006.00751*, 2020.

[18] Karen Simonyan and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[19] Rachel M Bittner, Julia Wilkins, Hanna Yip, and Juan P Bello, "Medleydb 2.0: New data and a system for sustainable data collection," *ISMIR Late Breaking and Demo Papers*, p. 36, 2016.

[20] Hugo Flores Garcia, Aldo Aguilar, Ethan Manilow, and Bryan Pardo, "Leveraging hierarchical structures for few-shot musical instrument recognition," *arXiv preprint arXiv:2107.07029*, 2021.

[21] Erich M Von Hornbostel and Curt Sachs, "Systematik der musikinstrumente. ein versuch," *Zeitschrift für Ethnologie*, vol. 46, no. H. 4/5, pp. 553–590, 1914.

[22] Iver Jordal, Araik Tamazian, Emmanouil Theofanis Chourdakis, Céline Angonin, askskro, Nikolay Karpov, Omer Sarioglu, kvilouras, Enis Berk Çoban, Florian Mirus, Jeong-Yoon Lee, Kwanghee Choi, MarvinLvn, SolomidHero, and Tanel Alumäe, "iver56/audiomentations: v0.25.0," May 2022.

[23] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu, "A survey on deep transfer learning," in *International conference on artificial neural networks*. Springer, 2018, pp. 270–279.

[24] Ricardo Ribani and Mauricio Marengoni, "A survey of transfer learning for convolutional neural networks," in *2019 32nd SIBGRAPI conference on graphics, patterns and images tutorials (SIBGRAPI-T)*. IEEE, 2019, pp. 47–57.

[25] Jesse Davis and Mark Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 233–240.