TRAINING NEURAL AUDIO CLASSIFIERS WITH FEW DATA

Jordi Pons[⋆†] *Joan Serrà*[⋆] *Xavier Serra*[†]

[⋆] Telefónica Research, Barcelona
[†] Music Technology Group, Universitat Pompeu Fabra, Barcelona

ABSTRACT

We investigate supervised learning strategies that improve the training of neural network audio classifiers on small annotated collections. In particular, we study whether (i) a naive regularization of the solution space, (ii) prototypical networks, (iii) transfer learning, or (iv) their combination, can foster deep learning models to better leverage a small amount of training examples. To this end, we evaluate (i–iv) for the tasks of acoustic event recognition and acoustic scene classification, considering from 1 to 100 labeled examples per class. Results indicate that transfer learning is a powerful strategy in such scenarios, but prototypical networks show promising results when one does not count with external or validation data.

***Index Terms*—** prototypical networks, transfer learning, audio classification, small data.

1. INTRODUCTION

It exists a prominent corpus of research assuming that sizable amounts of annotated audio data are available for training end-to-end classifiers [1, 2, 3, 4]. These studies are mostly based on publicly-available datasets, where each class typically contains more than 100 audio examples [5, 6, 7, 8, 9]. Contrastingly, only few works study the problem of training neural audio classifiers with few audio examples (for instance, less than 10 per class) [10, 11, 12, 13]. In this work, we study how a number of neural network architectures perform in such situation. Two primary reasons motivate our work: (i) given that humans are able to learn novel concepts from few examples, we aim to quantify up to what extent such behavior is possible in current neural machine listening systems; and (ii) provided that data curation processes are tedious and expensive, it is unreasonable to assume that sizable amounts of annotated audio are always available for training neural network classifiers.

The challenge of training neural networks with few audio data has been previously addressed. For example, Morfi and Stowell [12] approached the problem via factorising an audio transcription task into two intermediate sub-tasks: event and tag detection. Another way to approach the problem is by leveraging additional data sources, like in unsupervised and semi-supervised frameworks where non-labelled data is also utilized [14, 15, 16]. Transfer learning is a popular way to exploit such additional data sources [17, 18], and it has been used to construct acoustic models for low-resource languages [19, 20], to adapt generative adversarial networks to new languages and noise types [21], or to transfer knowledge from the visual to the audio domain [22]. An additional alternative is to use data augmentation, which has proven to be very effective for audio classification tasks [2, 23]. However, in this work, we center our efforts into exploiting additional data resources with transfer learning. This, according to our view, has three main advantages: (i) differently to data augmentation, it allows leveraging external sources of data; (ii) it exists a rich set of techniques for learning transferable representations [16, 17, 18, 20, 22]; and (iii) transfer learning can always be further extended with data augmentation.

In parallel to previous works, the machine learning community has been developing methods for learning novel classes from few training instances, an area known as few-shot learning [11, 24, 25, 26]. These methods aim to build a classifier that generalizes to new classes not seen during training, given only a small number of training examples for each new class. Differently to few-shot learning, the models we study do not generalize to new classes. Instead, we assume a fixed taxonomy during both training and prediction. Still, we derive inspiration from few-shot learning for their capacity to learn from few training data. A popular approach to few-shot learning is metric learning, which aims to learn representations that preserve the class neighborhood structure so that simple distances can be measured in a learnt space [24, 26]. Such methods have been mostly used for image classification, and are very appealing due to their simplicity and yet powerful performance on several benchmarks.

In our study we consider prototypical networks [24], a metric learning approach that is based on computing distances against class-based prototypes defined in a learnt embedding space (one can think of it as a nearest-neigbour classifier trained end-to-end). Our aim is to study if prototypical networks are capable to generalize better than raw deep learning models when trained on small data. To the best of our knowledge, only Tilk [11] has explored metric learning methods for constructing neural audio classifiers. He used the last layer features of a siamese network [27], an alternative metric learning approach, as input to an SVM classifier. Therefore, our work can be considered the first one to employ prototypical networks for audio. Besides, back in the 80's, Kohonen [28] proposed a distance-based model for speech recognition called learning vector quantization (LVQ), which is closely related to prototypical networks. However, LVQ is not designed to learn from few data and, furthermore, does not exploit the powerful non-linear mapping that neural networks can provide.

In this work, we investigate which strategies can provide a performance boost when neural network audio classifiers are trained with few data. These are evaluated under different low-data situations, which we describe in section 2. Firstly, we consider the regularization of the traditional deep learning pipeline (section 3.1). Next, we consider prototypical networks, with the aim to showcase the potential of metric learning-based classifiers coming from the few-shot learning literature (section 3.2). Finally, we consider transfer learning as a canonical way to leverage external sources of audio data (section 3.3). Results are presented in section 4 and, to conclude, we provide further discussion in section 5. We use publicly-available data sets, and share the code to reproduce our experiments at github.com/jordipons/neural-classifiers-with-few-audio.

Work partially funded by the Maria de Maeztu Programme (MDM-2015-0502). J. Pons and X. Serra are grateful to NVidia for the donated GPUs.

2. METHODOLOGY

2.1. Data: Where is the validation set?

The focus of this work is to investigate which neural network-based strategies perform best in the low-data regime. To do so, we simulate classification scenarios having only n randomly selected training audios per class, $n \in \{1, 2, 5, 10, 20, 50, 100\}$. Since results of the same repeated experiment might vary depending on which audios are selected, we run each experiment m times per fold of data, and report average accuracy scores across runs and folds. Specifically: $m = 20$ when $n \in \{1, 2\}$, $m = 10$ when $n \in \{5, 10\}$, and $m = 5$ when $n \in \{20, 50, 100\}$.

We run the study for both the tasks of acoustic event recognition and acoustic scene classification. For acoustic event recognition, we employ the UrbanSound8K dataset (US8K) [8], featuring 8,732 urban sounds divided into 10 classes and 10 folds (with roughly 1000 instances per class). For acoustic scene classification, we resort to the TUT dataset (ASC-TUT) [7, 29], featuring 4,680 audio segments for training and 1,620 for evaluation, of 10 s each, divided into 15 classes (with 312 instances per class). Furthermore, and we consider this a crucial aspect of our work, we assume that data is so scarce that it is unreasonable to presuppose the existence of a validation set for deciding when to stop training. As a result of such constraint, the following sections also describe the rules we use to decide when to stop training. Besides reducing the train set size and not utilizing any validation set, we keep the original partitions to compare our results with previous works.

2.2. Baselines

To put results into context, we employ several baselines aiming to describe lower and upper bounds for the two tasks we consider:

- **Random guess:** A model picking a class at random, which scores 9.99% accuracy for US8K and 6.66% for ASC-TUT.
- **Nearest-neighbor MFCCs:** A model based on a simple nearest-neighbor classifier using the cosine distance over MFCC features. The feature vector is constructed from 20 MFCCs, their Δs, and $\Delta\Delta$s. We compute their mean and standard deviation through time, with the resulting feature vector per audio clip being of size 120.
- **Salamon and Bello [2] (SB-CNN):** This model achieves state-of-the-art results for US8K. When trained with all US8K training data it achieves an average accuracy score of 73% across folds (79% with data augmentation). SB-CNN is an AlexNet-like model that consists of 3 convolutional layers with filters of 5×5, interleaved with max-pool layers. The resulting feature map is connected to a softmax output via a dense layer of 64 units. To assess how standard deep learning models would perform when learning their weights from scratch with small data, we train this baseline for all n.
- **Han et al. [30] and Mun et al. [23]:** These models achieve state-of-the-art performance for ASC-TUT. When trained with all ASC-TUT training data, they achieve an accuracy score of 80.4% using an ensemble [30], and 83.3% using an ensemble trained with GAN-based data augmentation [23].

2.3. Training details

Unless stated otherwise, the sections below follow the same experimental setup. Following common practice [2, 31], inputs are set to be log-mel spectrogram patches of 128 bins $\times$ 3 s (128 frames)[1]. For US8K, when audio clips are shorter than 3 s, we 'repeat-pad' spectrograms in order to meet the model's input size. That is, the original short signal is repeated up to create a 3 s signal. During training, data are randomly sampled from the original log-mel spectrograms following the previous rules. However, during prediction, if sounds are longer than 3 s, several predictions are computed by a moving window of 1 s and then averaged. We use ReLUs and a batch size of 256. Learning proceeds via minimizing the cross-entropy loss with vanilla stochastic gradient descent (SGD) at a rate of 0.1. We stabilize learning with gradient clipping, that is, we rescale the gradients so that their L2 norm does never exceed a threshold of 5.

3. AUDIO CLASSIFICATION WITH FEW DATA

3.1. Regularized models

In a first set of experiments, we showcase the limitations of the commonly used deep learning pipeline when training data are scarce. An ordinary approach to avoid overfitting in such cases is to use regularization. Following this idea, we consider two architectures. The first one, VGG, is meant to keep the model highly expressive while introducing as much regularization as possible. The second one, TIMBRE, strongly regularizes the set of possible solutions via domain-knowledge informed architectural choices.

- **VGG [1, 18]:** This is a computer vision architecture designed to make minimal assumptions regarding which are the local stationarities of the signal, so that any structure can be learnt via hierarchically combining small-context representations. To this end, it is common to utilize a deep stack of small 3×3 filters (in our case 5 layers, each having only 32 filters), combined with max-pool layers (in our case of 2×2). We further employ a final dense layer with a softmax activation that adapts the feature map size to the number of output classes, ELUs as non-linearities [18], and batch norm.
- **TIMBRE [3, 31]:** This model is designed to learn timbral representations while keeping the model as small as possible. We use a single-layer convolutional neural network (CNN) with vertical filters of 108 bins $\times$ 7 frames. A softmax output is computed from the maximum values present in each CNN feature map and, therefore, the model has as many filters as output classes. TIMBRE is possibly the smallest CNN one can imagine for an audio classification task, provided that it only has a single 'timbral' filter per class [3].

Note that the studied VGG and TIMBRE models, of approximately 50 k and 10 k parameters, respectively, are much smaller than the state-of-the-art SB-CNN model, which has approximately 250 k parameters. Besides the regularization we introduce via minimizing the model size, we employ L2-regularization, with a penalty factor of 0.001, and make use of 50% dropout whenever a dense layer is present (only VGG and SB-CNN models have dense layers). Finally, and given that no validation set is available, we empirically find that (early) stopping training after 200 epochs provides good results for all studied models that are not based on prototypical networks.

3.2. Prototypical networks

In the next set of experiments, we study how prototypical networks can be exploited for audio classification with few examples. As mentioned, prototypical networks are based on learning a latent metric

[1] STFT parameters: *window_size*=*hop_size*=1024 and *fs*=44.1 kHz.

space in which classification can be performed by computing distances to prototype representations of each class. Prototypes μ_k are mean vectors of the embedded support data belonging to class k. That is, given input samples x_i:

$$\mu_k = \frac{1}{|S_k|} \sum_{x_i \in S_k} f_\phi(x_i),$$

where f_ϕ is parametrized by a neural network. In our case, we use a VGG as in section 3.1, but with 128 filters per layer and a final linear layer instead of a softmax. The prototype vector $\mu_k \in \mathbb{R}^D$ is set to have an embedding size of $D = 10$. In this work, the support set S_k to compute each class' prototype is conformed by 5 randomly selected patches from the train set belonging to class k. These same sounds will be then reused to train f_ϕ.

Prototypical networks produce a distribution over classes for a query point x_i based on a softmax over distances to the prototypes in the embedding space:

$$p_k(x_i) = \frac{e^{-d(f_\phi(x_i), \mu_k)}}{\sum_{k'} e^{-d(f_\phi(x_i), \mu_{k'})}},$$

where d is any suitable distance measure. We here use the Euclidean distance as we found it to outperform the cosine distance for our tasks, see Appendix 1 in [32]. The training of f_ϕ is based on minimizing the negative log-probability of the true class via SGD. Training epochs are formed by batches of 5 random patches per class, and we backpropagate until the train set accuracy does not improve for 200 epochs. Note that this stop criteria does not utilize a validation set, only the train set. Differently from the common supervised learning pipeline, we found overfitting not to be an issue with prototypical networks, which is an important point to consider when training models with few data. Actually, monitoring how well the model is able to separate the training data in the embedding space, through measuring train set accuracy, was an effective way to assess how discriminative such space is. Although this stop criteria could promote overfitting the train set, in section 4 we show that prototypical networks' generalization capabilities are still above the ones of the raw deep learning pipeline. Further discussion on the generalization abilities of prototypical networks is available at Appendix 2 in [32].

3.3. Transfer learning

In our final set of experiments, we assess the effectiveness of canonical transfer learning strategies. For that, we use a VGG model pre-trained with Audioset [1, 9], a dataset conformed by 2 M YouTube audios that was designed for training acoustic event recognition models. As a result, note that our source and target tasks are the same for US8K (all US8K classes have a direct correspondence in Audioset), but are different for ASC-TUT (only 5 out of 15 classes resemble Audioset classes). The pre-trained Audioset model[2] is composed of 6 convolutional layers with filters of 3×3, interleaved with max-pooling layers of 2×2, followed by 3 dense layers of 4096, 4096, and 128 units, respectively. Inputs are log-mel spectrogram patches of 64 bins $\times$ 1 s (96 frames)[3]. In order to match the same conditions as previous experiments for inputs longer than 1 s, we compute several predictions by a non-overlapping moving window of 1 s that are finally averaged. When audios are shorter than 1 s, we use 'repeat-pad' as in previous experiments (see section 2.3).

In order to study how the pre-trained Audioset model transfers to our tasks, we consider three alternatives:

- **Nearest-neighbor with Audioset features:** This baseline classifier serves to study how discriminative are the Audioset features alone. It is based on the cosine distance, and utilizes majority voting to aggregate the different predictions of the model through time (one per second of audio).
- **Transfer learning (fine-tuning):** The pre-trained Audioset model is fine-tuned, together with a dense softmax layer that acts as the final classifier.
- **Prototypical networks + transfer learning:** We experiment with the idea of using transfer learning in the context of prototypical networks, and we fine-tune the pre-trained model together with a dense linear layer (10 units) that defines the embedding space where the distance-based classifier operates.

For all transfer learning experiments, in order to avoid pre-trained layers to quickly overfit the train set, fine-tuning occurs at a slower pace than training the classification or embedding layer. We use a learning rate of 0.00001 for the pre-trained layers, and a learning rate of 0.1 for randomly initialized layers.

4. RESULTS

Fig. 1 summarizes the results we obtain for the two datasets considered in this work: US8K (first row) and ASC-TUT (second row). On the left-hand side, we compare the results of the regularized models with the ones of prototypical networks. On the right-hand side, we compare the results of transfer learning with the ones of prototypical networks. For each study case, we depict a lower bound (random guess) and an upper bound (previous works using all the train set, see section 2.2). In addition, we include a basic baseline, consisting of a nearest neighbor classifier. All baselines and references are depicted with dashed and dotted lines.

First of all, we elaborate on the results obtained by the standard and regularized deep learning models, namely SB-CNN, VGG, and TIMBRE (Fig. 1, left). All these models perform similarly when few training data are available ($n < 50$). Even so, it is remarkable the performance of the strongly-regularized TIMBRE model for the US8K dataset, as this outperforms the other two and the MFCC baseline when $n \leq 10$. Notice, however, that the trend changes when more data becomes avilable ($n > 20$). Under this regime, the expressive VGG model seems to better exploit the available data. Finally, it is also interesting to observe the limitations of the commonly used deep learning pipeline when few training data are available, as the SB-CNN and regularized models struggle to clearly outperform a simple nearest-neighbor MFCC baseline for $n \leq 20$.

In order to overcome the abovementioned limitations of standard and regularized deep learning models, we investigate the use of prototypical networks. Fig. 1 (left) depicts how they consistently outperform raw deep learning models, both for US8K and ASC-TUT. Although performance gains are moderate for $n < 5$, prototypical networks clearly outperform regularized models for $5 < n < 50$. Interestingly, though, the performance of prototypical networks can saturate for $n > 50$, as for US8K results. This suggests that prototypical networks may not be as competitive as regular deep learning architectures when sizable amounts of training data are available. However, such tendency could be data-dependent, as it is not observed for ASC-TUT.

Note that the prototypical networks' embedding space is defined by a larger VGG than the regularized VGG model. Hence, prototypical networks could seem to be more prone to overfitting than regularized models. However, we find that prototypical networks do generalize better than standard and regularized models in the low

[2] github.com/tensorflow/models/tree/master/research/audioset

[3] STFT parameters: *window_size*=400, *hop_size*=160 and *fs*=16 kHz

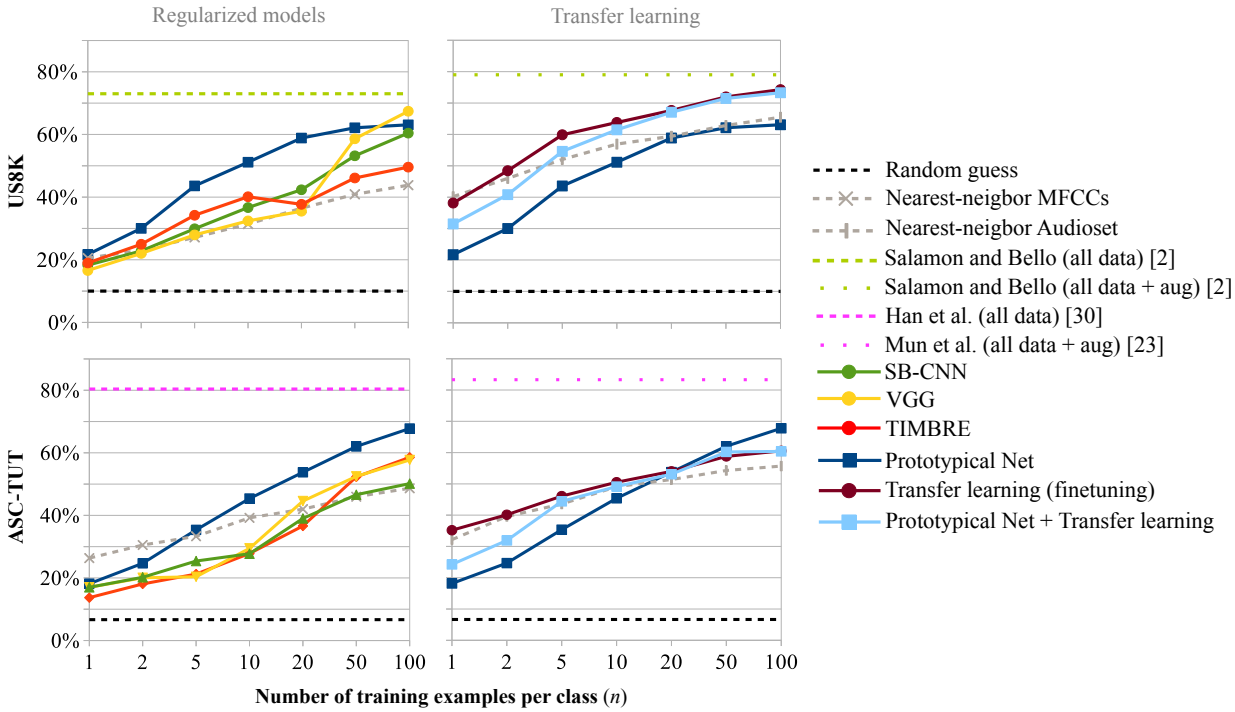

Fig. 1. Accuracy (%) of the studied strategies when compared to prototypical networks. Dashed and dotted lines represent baselines, and strong lines represent the considered strategies. For comparison, we repeat the curve for prototypical networks in both left and right plots.

data regime (Fig. 1, left). Since overfitting seems not to dramatically affect prototypical networks' results, we find that highly expressive models, like a large VGG, can deliver good results. We speculate that this might be caused because the resulting latent space is competent enough to discriminate each of the classes, whereas for smaller and less expressive models this might not be the case.

In our study, we also investigate how transfer learning approaches compare with the previous solutions (Fig. 1, right). We observe that, for $n \geq 10$, transfer learning with basic fine-tuning performs equivalently to prototypical networks + transfer learning. However, for $n < 10$, the former outperforms the latter. We speculate that this effect emerges when prototypes trained with small data are not representative enough of their corresponding classes. For such small data, $n < 10$, the support set to compute each class' prototype is also conformed by 5 randomly selected patches from the train set. As a result, for example, when $n = 1$ these 5 patches are sampled from the same spectrogram and then reused for training f_ϕ. Consequently, the variety of the training examples used for computing the prototypes can be very limited for $n < 10$, what might be harming the results of prototypical networks. Interestingly, though, for $n \geq 10$ we observe that prototypical networks + transfer learning start performing equivalently to transfer learning with fine-tuning. Note that $n = 10$ is the first scenario where prototypical networks can be trained with data being variate enough, since 5 examples can be used for computing the prototypes and 5 additional examples can be used for training f_ϕ (see section 3.2).

Finally, it is worth reminding that source and target tasks are the same for US8K but are different for ASC-TUT. Possibly for that reason, transfer learning consistently outperforms prototypical networks for US8K, but struggles to do so for ASC-TUT. For the latter dataset, we see that prototypical networks (trained from scratch with $n \leq 100$ instances) are able to outperform transfer learning-based approaches (pretrained with 2 M audios) for $n > 20$. This result denotes that transfer learning has a strong potential when small data are available ($n \leq 20$). However, it can be easily overthrown by prototypical networks if the number of training examples per class becomes large enough and target and source tasks do not match.

5. DISCUSSION

Among the strategies we have studied for training neural network classifiers with few annotated audios, we have found prototypical networks and transfer learning to be the ones providing the best results. However, choosing one or another might depend on the specificities of the use case. Transfer learning-based classifiers are generally a good choice when operating in low-data regimes, but they assume that a pre-trained model is readily available. Importantly, such model needs to be trained with data falling under a similar distribution to the few data samples we have available for solving the task. Otherwise, there is no guarantee for transfer learning to deliver better results than, for instance, prototypical networks. When data distributions do not match, we show that prototypical networks trained from scratch can be the right choice.

In order to restrict ourselves to a realistic low-data scenario, our results are computed without utilizing any validation set. As a result, the set of intuitions that generally help us deciding when to stop training no longer hold. We have found early stopping to be a valid approach when training regular deep learning classifiers. However, interestingly, we have found overfitting not to dramatically affect prototypical networks' results, possibly because these rely on a robust distance-based classifier. Since deciding in which epoch to 'early-stop' highly depends on many design choices, we have found the 'just overfit' criteria of prototypical networks to be very simple while delivering competitive results.

6. REFERENCES

[1] 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.

[2] Justin Salamon and Juan Pablo Bello, "Deep convolutional neural networks and data augmentation for environmental sound classification," *IEEE Signal Processing Letters*, vol. 24, no. 3, pp. 279–283, 2017.

[3] Jordi Pons, Olga Slizovskaia, Rong Gong, Emilia Gómez, and Xavier Serra, "Timbre analysis of music audio signals with convolutional neural networks," in *EUSIPCO*, 2017.

[4] Jordi Pons, Oriol Nieto, Matthew Prockup, Erik M Schmidt, Andreas F Ehmann, and Xavier Serra, "End-to-end learning for music audio tagging at scale," *ISMIR*, 2018.

[5] Eduardo Fonseca, Manoj Plakal, Frederic Font, Daniel P. W. Ellis, Xavier Favory, Jordi Pons, and Xavier Serra, "General-purpose tagging of freesound audio with audioset labels: task description, dataset, and baseline," *arXiv:1807.09902*, 2018.

[6] Eduardo Fonseca, Jordi Pons, Xavier Favory, Frederic Font Corbera, Dmitry Bogdanov, Andres Ferraro, Sergio Oramas, Alastair Porter, and Xavier Serra, "Freesound datasets: a platform for the creation of open audio datasets," in *ISMIR*, 2017.

[7] Annamaria Mesaros, Toni Heittola, and Tuomas Virtanen, "Tut database for acoustic scene classification and sound event detection," in *EUSIPCO*, 2016.

[8] Justin Salamon, Christopher Jacoby, and Juan Pablo Bello, "A dataset and taxonomy for urban sound research," in *ACM-Multimedia*, 2014.

[9] 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.

[10] Ivan Bocharov, A de Vries, and Tjalling Tjalkens, "K-shot learning of acoustic context," in *NIPS Workshop on Machine Learning for Audio Signal Processing*, 2017.

[11] Bence Tilk, "Make SVM great again with siamese kernel for few-shot learning," *Rejected ICLR submission*, 2018.

[12] Veronica Morfi and Dan Stowell, "Deep learning for audio transcription on low-resource datasets," *arXiv:1807.03697*, 2018.

[13] Veronica Morfi and Dan Stowell, "Data-efficient weakly supervised learning for low-resource audio event detection using deep learning," *arXiv:1807.06972*, 2018.

[14] Aren Jansen, Manoj Plakal, Ratheet Pandya, Daniel PW Ellis, Shawn Hershey, Jiayang Liu, R Channing Moore, and Rif A Saurous, "Unsupervised learning of semantic audio representations," *arXiv:1711.02209*, 2017.

[15] Honglak Lee, Peter Pham, Yan Largman, and Andrew Y Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *NIPS*, 2009.

[16] Yong Xu, Qiang Huang, Wenwu Wang, Peter Foster, Siddharth Sigtia, Philip JB Jackson, and Mark D Plumbley, "Unsupervised feature learning based on deep models for environmental audio tagging," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 6, pp. 1230–1241, 2017.

[17] Julius Kunze, Louis Kirsch, Ilia Kurenkov, Andreas Krug, Jens Johannsmeier, and Sebastian Stober, "Transfer learning for speech recognition on a budget," in *Workshop on Representation Learning for NLP*, 2017.

[18] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho, "Transfer learning for music classification and regression tasks," *ISMIR*, 2017.

[19] Arnab Ghoshal, Pawel Swietojanski, and Steve Renals, "Multilingual training of deep neural networks," in *ICASSP*, 2013.

[20] Jui-Ting Huang, Jinyu Li, Dong Yu, Li Deng, and Yifan Gong, "Cross-language knowledge transfer using multilingual deep neural network with shared hidden layers," in *ICASSP*, 2013.

[21] Santiago Pascual, Maruchan Park, Joan Serrà, Antonio Bonafonte, and Kang-Hun Ahn, "Language and noise transfer in speech enhancement generative adversarial network," in *ICASSP*, 2018, pp. 5019–5023.

[22] Yusuf Aytar, Carl Vondrick, and Antonio Torralba, "Soundnet: Learning sound representations from unlabeled video," in *NIPS*, 2016.

[23] Seongkyu Mun, Sangwook Park, David K Han, and Hanseok Ko, "Generative adversarial network based acoustic scene training set augmentation and selection using svm hyperplane," *DCASE Workshop*, 2017.

[24] Jake Snell, Kevin Swersky, and Richard Zemel, "Prototypical networks for few-shot learning," in *NIPS*, 2017.

[25] Sachin Ravi and Hugo Larochelle, "Optimization as a model for few-shot learning," *ICLR*, 2016.

[26] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al., "Matching networks for one shot learning," in *NIPS*, 2016.

[27] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah, "Signature verification using a" siamese" time delay neural network," in *Advances in neural information processing systems*, 1994, pp. 737–744.

[28] Teuvo Kohonen, "The 'neural' phonetic typewriter," *Computer*, vol. 21, no. 3, pp. 11–22, 1988.

[29] Annamaria Mesaros, Toni Heittola, Aleksandr Diment, Benjamin Elizalde, Ankit Shah, Emmanuel Vincent, Bhiksha Raj, and Tuomas Virtanen, "Dcase 2017 challenge setup: Tasks, datasets and baseline system," in *DCASE Workshop*, 2017.

[30] Yoonchang Han, Jeongsoo Park, and Kyogu Lee, "Convolutional neural networks with binaural representations and background subtraction for acoustic scene classification," *the Detection and Classification of Acoustic Scenes and Events (DCASE)*, pp. 1–5, 2017.

[31] Jordi Pons and Xavier Serra, "Randomly weighted CNNs for (music) audio classification," *arXiv:1805.00237*, 2018.

[32] Jordi Pons, Joan Serrà, and Xavier Serra, "Training neural audio classifiers with few data," *arXiv:1810.10274*, 2018.