ACOUSTIC SCENE CLASSIFICATION WITH MISMATCHED RECORDING DEVICES

Technical Report

Paul Primus, David Eitelsebner

Institute of Computational Perception (CP-JKU)
Johannes Kepler University Linz, Austria
primus_paul@yahoo.de, david@eitelsebner.net

ABSTRACT

This technical report describes CP-JKU Student team's approach for Task 1 - Subtask B of the DCASE 2019 challenge. In this context, we propose two loss functions for domain adaptation to learn invariant representations given time-aligned recordings. We show that these methods improve the classification performance on our cross-validation, as well as performance on the Kaggle leader board, up to three percentage points compared to our baseline model. Our best scoring submission is an ensemble of eight classifiers.

Index Terms— Acoustic Scene Classification, Domain Adaptation, DCASE

1. INTRODUCTION

Convolutional Neural Networks (CNNs) have emerged to become state of the art tools for audio related machine learning tasks, such as acoustic scene classification, audio tagging and sound event localization. While CNNs are known to generalize well if the recording conditions for training and unseen data remain the same, the generalisation of this class of models degrades when there is a distribution mismatch between the training and the testing data [1]. Subtask 1b of 2019's IEEE AASP Challenge on Detection and Classification of Acoustic Scenes and Events [2] is concerned with this problem: Participants are asked to create an acoustic scene classification system for ten different acoustic scene classes. A set of labelled audio snippets recorded with a high-quality microphone (known as Device A) is provided for training. Additionally, for a small subset of samples from device A, parallel recordings from two lower quality microphones (devices B and C) are given. Fig. 1 shows three time-aligned recordings, for which we can observe the device-specific characteristics. Ranking of submissions is done based on the overall accuracy on unseen samples from devices B and C. The acoustic scene, the city, and the device labels are provided for samples of the development set only. The main challenge of task 1b is to develop a model that although trained mostly on samples from device A, is able to generalize well to samples from devices B and C. Since this problem is related to the field of Domain Adaptation (DA) we refer to the distribution of A samples as the source domain ($x^s \sim X^s$), and the distribution of B and C samples as the target domain ($x^t \sim X^t$). This technical report explains how a state-of-the-art CNN architecture which by itself achieves a high rank on the public Kaggle leader board[1] can be further improved by using simple DA techniques. For reproducibility, we make our source code available on GitHub[2].

2. THE PROPOSED METHOD

The following section describes our experimental setup and domain adaptation techniques in detail.

2.1. Data & Cross Validation Setup

Preprocessing is done similar to [3]: We resample the audio signals to 22050Hz and compute a mono-channel Short Time Fourier Transform using 2048-sample windows and a hop-size of 512 samples. We apply perceptual weighting to the individual frequency bands of the power spectrogram and a mel-scaled filterbank for frequencies between 40 and 11025Hz, yielding 431-frame spectrograms with 256 frequency bins. Samples are normalized during training by subtracting the training set mean and dividing by training set standard deviation. For the cross-validation splits, all parallel recorded samples are divided into four folds such that triplets of time-aligned samples are in the same fold. Samples from device A are distributed between folds such that samples recorded at the same location are placed into the same fold. The remaining samples are shared between all folds.

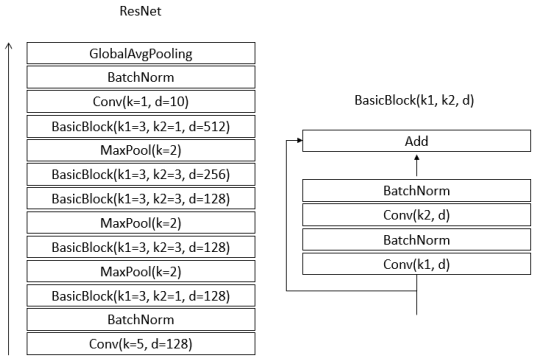

Figure 2: Model Architecture [4]. $k1$ and $k2$ are the kernel sizes of the first and the second layer, respectively. d is the number of channels.

[1] https://www.kaggle.com/c/dcase2019-task1b-leaderboard/ [2] https://github.com/OptimusPrimus/dcase2019_task1b

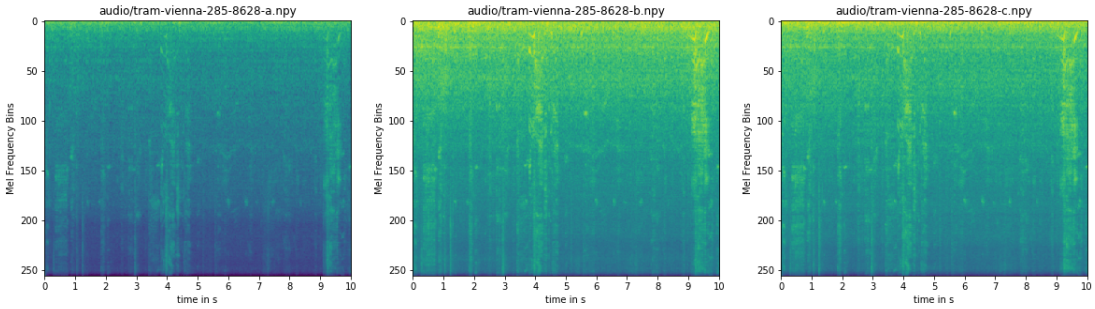

Figure 1: From left to right: Time-aligned recordings from devices A (Soundman OKM II Klassik/Studio A3 Microphone & Zoom F8 Recorder), B (Samsung Galaxy S7) and C (iPhone SE). Spectrograms show microphone-specific patterns, e.g. samples from devices B and C have more noise in lower Mel bins, compared to those from device A.

2.2. Network Architecture

We use the model architecture introduced by Koutini et al. [4], a receptive-field-regularized, fully convolutional, residual network (ResNet) with five residual blocks (Fig. 2.1). The receptive field of this architecture is tuned to achieve the best performance in audio-related tasks using spectrograms, as discussed in [4].

2.3. Domain Adaptation

We propose two distance and entropy-based loss functions to encourage the CNN to learn device-invariant representations for parallel samples from the source (X^s), and the target domain (X^t). Both of these losses exploit the fact that the time-aligned spectrograms (x^s, x^t) contain the same acoustic information about the acoustic scenes and differ only due to device characteristics. To achieve the best performance, we use a combination of classification loss and a DA loss. We use Categorical Cross Entropy (CCE) for classification, and combine it with the DA loss as follows:

$$\mathcal{L} = \mathcal{L}_{CCE} + \lambda \mathcal{L}_{l,DA} \tag{1}$$

λ controls the influence of the DA loss during training and l specifies to which (hidden) representation it is applied.

2.3.1. Mean Squared Error Loss

Let $\phi_l(x^s)$ and $\phi_l(x^t)$ be hidden layer activations for samples from the source and the target domains, respectively. We define the Mean Squared Error (MSE) loss between the (hidden) representations of size f as follows:

$$\mathcal{L}_{MSE} = \frac{1}{f \cdot n} \sum_{i=0}^{n-1} \sum_{j=0}^{f-1} (\phi_l(x_i^s)_j - \phi_l(x_i^t)_j)^2 \tag{2}$$

where n is the DA mini-batch size. For our experiments l is set to correspond to the output layer before applying the softmax activation.

2.3.2. Mutual Information Loss

According to Ji et al. [5], Mutual Information (MI) I can be used to learn a representation $\phi(\cdot)$ which preserves what is common between two different images containing the same object.

We apply the same idea to learn a representation which discards device-specific characteristics of features, by maximizing the MI between parallel representations:

$$max_{\psi_l} I(\phi_l(x^s), \phi_l(x^t)) \tag{3}$$

Note that MI is maximized if, given one representation the other one is predictable, i.e. they are the same. To compute this quantity, we introduce a layer with output size c parallel to the classification output layer. The MI-Loss between the parallel representations can then be computed as described in [5]. We tried different numbers of clusters, and setting c to 20 provided the best results for our experiments.

2.4. Training

All models are trained for 250 epochs with mini-batch updates of size 32 and ADAM [6] update rule. The initial learning rate is set to 10^{-3} and decreased by a factor 0.5 if the mean accuracy for devices B and C does not increase for 15 epochs. If the learning rate is decreased, we also reset the model parameters to the best model in terms of mean accuracy of device B and C up to the last epoch. We further use MixUp augmentation [7] with parameters of the beta-distribution set to $\alpha = \beta = 0.2$.
For each fold, the model that scores the highest averaged device B and C accuracy acc_{bc} is selected for prediction on future data. acc_{bc} is calculated by averaging the individual device accuraccies as shown in Equation (4).

$$acc_{bc} = \frac{acc_b + acc_c}{2} \tag{4}$$

2.5. Model Ensembling

As we train every model on 4 folds, our final submission models are an ensemble of the outputs of the 4 folds. For submission 1, 2 and 3, we average the softmax predictions of each fold's best scoring model and select the class with the highest score. Submission 4 combines two independently trained submission models, again by averaging each of their 4 folds softmax outputs.

	Kaggle	Fold 0	Fold 1	Fold 2	Fold 3	4-Fold-CV	Prov. Split	Sub. ID
No DA	0.7366	0.6977	**0.6899**	0.6459	0.6429	0.6691	0.6120	1
MSE ($\lambda = 0.1, n = 4$)	0.7583	0.7186	0.6846	**0.7042**	**0.6795**	**0.6967**	**0.6435**	2
MI ($\lambda = 0.1, n = 4, c = 20$)	0.7333	**0.7338**	0.6742	0.6362	0.6538	0.6745	0.6250	3
Ensemble	**0.7666**	-	-	-	-	-	-	4

Table 1: Result table shows validation results of all submitted systems in terms of accuracy for different validation setups: Kaggle public leaderboard, folds 0 to 3, cross validation, and provided split. The last column shows the submission ID of the corresponding method.

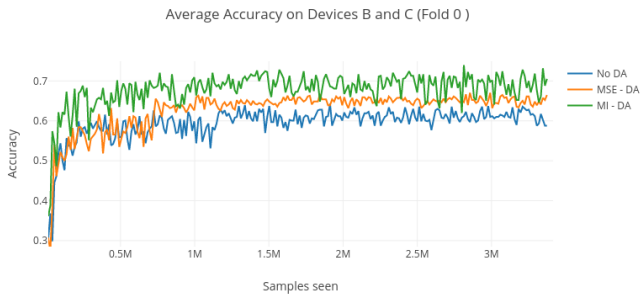

Figure 3: Average accuracy of device B and C during training on fold 0 for no domain adaptation, mutual information and MSE.

3. RESULTS

The results of the models can be seen in Table 2.2. All scores are the averaged accuracy of device B and C. The column *Sub. ID* is a reference to the submission ID for the final challenge submission. The column *Kaggle* refers to the score the model achieved on the public leader board on Kaggle. The columns *Fold 0 - 3* show the accuracy for each individual fold and the avg-column the averaged accuracy over all folds. The last column provides the accuracy of the model on the provided train-test split. For this score, the complete model was retrained on the proposed train/test split.

Our base model (Section 2.1), the convolutional residual network (ResNet) introduced by Koutini et al. [4] achieved a BC-accuracy of 0.7366 on the leader board, training with the provided split achieved an accuracy of 0.612.

The model with mutual information as additional loss for domain adaptation (Section 2.3.2) shows an improvement on the provided split of 1.3 p.p. over the base model, however on the public leader board it performed slightly worse (0.7333 compared to 0.7366 of the base model). Although it did not improve the performance as much as MSE, MI showed in some folds promising results. As we can see in Figure 3, which shows acc_{bc} during the training of fold 0, the model scored with an accuracy of 0.7338 the top score for a single fold over all of our trained models.

The third model uses MSE loss (Section 2.3.1) for domain adaptation. On the public leader board, it gained an additional 2.17 p.p. over the base model, resulting in an accuracy of 0.7583. The gain is even larger on the proposed split, as with 0.6435 it performed 3.15 p.p. better than our base model. In our experiments, this model also produced on average the most balanced accuracy between all three devices, however as we have chosen the model with the best accuracy for device B and C for our final submission, accuracy on device A is slightly worse.

Our ensemble of eight predictors achieves the best results: With an accuracy of 0.7666, it is the top scoring submission on the public Kaggle leader board.

4. CONCLUSION

In this report, we have shown how an already good performing ResNet-like model [4] can be further improved for DCASE 2019 task 1b by using DA techniques. Our DA losses are designed to enforce equal hidden layer representations for different devices by exploiting time-aligned recordings. Notably, the MSE domain adaptation increased the performance by 2.17 p.p. on the public leader board and by 3.15 p.p. on the proposed split on the train set. Mutual information did not increase the performance on the leader board, but it increased performance on the development-dataset compared to our baseline and even showed the top result for fold 0 over all of our models.

5. ACKNOWLEDGEMENTS

We thank Andreas Arzt, Hamid Eghbal-zadeh, William Freilinger, Rainer Kelz, Khaled Koutini, and Gerhard Widmer for Advice, Supervision & Support. This work has been supported by the COMET-K2 Center of the Linz Center of Mechatronics (LCM), funded by the Austrian federal government and the federal state of Upper Austria.

6. REFERENCES

[1] H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "Deep within-class covariance analysis for robust deep audio representation learning," in *Neural Information Processing Systems, Interpretability and Robustness in Audio, Speech, and Language Workshop*, 2018.

[2] A. Mesaros, T. Heittola, and T. Virtanen, "A multi-device dataset for urban acoustic scene classification," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, November 2018, pp. 9–13. [Online]. Available: https://arxiv.org/abs/1807.09840

[3] M. Dorfer, B. Lehner, H. Eghbal-zadeh, H. Christop, P. Fabian, and W. Gerhard, "Acoustic scene classification with fully convolutional neural networks and I-vectors," DCASE2018 Challenge, Tech. Rep., September 2018.

[4] K. Koutini, H. Eghbal-zadeh, M. Dorfer, and G. Widmer, "The Receptive Field as a Regularizer in Deep Convolutional Neural Networks for Acoustic Scene Classification," in *Proceedings*

of the European Signal Processing Conference (EUSIPCO), A Coruña, Spain, 2019.

[5] X. Ji, J. F. Henriques, and A. Vedaldi, "Invariant information distillation for unsupervised image segmentation and clustering," *CoRR*, vol. abs/1807.06653, 2018. [Online]. Available: http://arxiv.org/abs/1807.06653

[6] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[7] H. Zhang, M. Cissé, Y. N. Dauphin, and D. Lopez-Paz, "mixup: Beyond empirical risk minimization," *CoRR*, vol. abs/1710.09412, 2017. [Online]. Available: http://arxiv.org/abs/1710.09412