INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Unsupervised Multi-Target Domain Adaptation for Acoustic Scene Classification

Dongchao Yang[1], *Helin Wang*[1], *Yuexian Zou*[1,2,*]

[1]ADSPLAB, School of ECE, Peking University, Shenzhen, China
[2]Peng Cheng Laboratory, Shenzhen, China
dongchao98@stu.pku.edu.cn, {wanghl15,zouyx}@pku.edu.cn

Abstract

It is well known that the mismatch between training (source) and test (target) data distribution will significantly decrease the performance of acoustic scene classification (ASC) systems. To address this issue, domain adaptation (DA) is one solution and many unsupervised DA methods have been proposed. These methods focus on a scenario of single source domain to single target domain. However, we will face such problem that test data comes from multiple target domains. This problem can be addressed by producing one model per target domain, but this solution is too costly. In this paper, we propose a novel unsupervised multi-target domain adaption (MTDA) method for ASC, which can adapt to multiple target domains simultaneously and make use of the underlying relation among multiple domains. Specifically, our approach combines traditional adversarial adaptation with two novel discriminator tasks that learns a common subspace shared by all domains. Furthermore, we propose to divide the target domain into the easy-to-adapt and hard-to-adapt domain, which enables the system to pay more attention to hard-to-adapt domain in training. The experimental results on the DCASE 2020 Task 1-A dataset and the DCASE 2019 Task 1-B dataset show that our proposed method significantly outperforms the previous unsupervised DA methods.
Index Terms: Unsupervised domain adaptation, mismatched recording devices, acoustic scene classification

1. Introduction

Acoustic scene classification (ASC) is the task of assigning a scene label (e.g., "Tram", "Park") to an audio recording. ASC has recently been tackled with many deep learning methods [1, 2, 3, 4, 5]. However, many ASC systems tend to be susceptible to the effects of domain shift, when training and test audio are recorded by different devices. Figure 1 shows that different recording devices lead to the change of data distribution. To address the problem of mismatched recording devices, many methods have been proposed, such as data augmentation [6, 7], spectrum correction [8, 9] and domain adaptation (DA) [10]. Although these methods got good performance, they trained with both labeled source- and target-domain samples. In this paper, we investigate the unsupervised domain adaptation (UDA) scenario, i.e., the acoustic scene labels of the target domain are not known during the adaptation part.

Many UDA methods [11, 12, 13] have been proposed in computer vision field, but only a few studies (such as [14, 15, 16, 17, 18]) have applied UDA techniques to ASC models. In [17], authors follow a unsupervised domain adaptation neural network [19], and introduces it to learn a common subspace for the ASC problem. In [16], authors follow maximum classifier discrepancy [20], which can properly consider distributions of each class within domains.

*Corresponding author

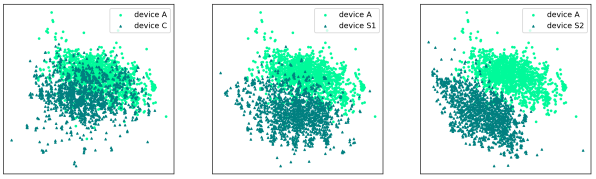

Figure 1: *Visualization of mel-spectrum of audio data from DCASE2020 dataset [21] by t-SNE. We choose small subset of samples from device A, and parallel recordings from three lower quality devices (devices C, S1 and S2).*

Table 1: *Comparison of accuracy on DCASE2020 dataset [21]. We use device A as source domain, devices B, C, S1 and S2 as target domains. DANN means that combining devices B, C, S1 and S2 as a single domain, then training once. DANN-respective means that we train (A → B), (A → C), (A → S1), (A → S2) separately.*

model	B(%)	C(%)	S1(%)	S2(%)
DANN [17]	47	53.3	35.8	28.5
DANN-respective	48.2	53.9	40.3	34.8

But all of these methods focus on pairwise adaptation settings from single source domain to single target domain. However, in reality, test data may come from multiple target domains, e.g., audio data is recorded by multiple devices. When test data consist of multiple target domains, there are two common solutions. One is combining all target domains as a single target domain [14, 15, 16, 17], and then applying once pairwise adaptation. The other is applying pairwise adaptation for each target domain separately. As Table 1 shows, experimental results indicate that combining all target domains as a single one will decrease performance, and in Section 2 we also give theoretical demonstration. Although we can produce one model per target domain, this approach becomes costly and impractical in applications with a growing number of target domains. In addition, applying pairwise adaptation approach may be suboptimal, as it ignore the underlying relation among multiple domains.

In this paper, we propose a novel unsupervised multi-target domain adaption (MTDA) method for ASC, which can adapt to multiple target domains simultaneously and make use of the underlying relation among multiple domains. Our approach is based on two technical insights. The first technical insight is to learn a common subspace shared by both the source and target domains, which enables all domains to have same data distribution in the feature space. Specifically, inspired by GAN [22] and DANN [17], we make use of the adversarial relationship

http://dx.doi.org/10.21437/Interspeech.2021-300

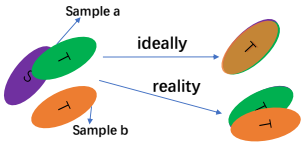

Figure 2: *Ideally, we want to align all domains. In fact, hard-to-adapt domain may not be aligned well.*

between modules Feature (F) and Discriminator (D) to learn domain-invariant feature in the feature space. Unlike previous methods [14, 17] which apply D in the binary classification task, we utilize D in the multi-classification or regression task. The second technical insight is to divide target domain into easy-to-adapt or hard-to-adapt domain. Intuitively, if the target domain is close enough to the source domain, the feature extracted from itself tends to result in accurate classification. Furthermore, we note that different devices cause different degrees of domain shift, and the target domain with severe domain shift has poor performance and is hard to adapt. Consequently, we should pay more attention to these hard-to-adapt domains during the training process. Our contributions are as follows: (1) We theoretically demonstrate why the pairwise adaptation methods cannot performance well when combining all target domains as a single one. (2) We propose to divide the target domain into the easy-to-adapt or the hard-to-adapt domain. (3) We propose the first unsupervised multi-target domain adaptation approach for ASC, which improves the performance of ASC unsupervised DA over the previous SOTA methods.

2. Vanilla method

In this section, we will theoretically demonstrate why the pairwise adaptation methods cannot perform well when combining different devices data as a single one. The following proof takes DANN [17] as an example.

In DANN [17], authors propose to learn an encoder E and a predictor C such that the distribution of the encodings $z = E(\boldsymbol{x})$ (where $\boldsymbol{x}$ denotes mel-spectrum of audio) from target and source domains are aligned so that all scene labels can be accurately predicted by the shared predictor C. This is achieved by the adversarial relationship between E and discriminator D. D predicts $\boldsymbol{z}$ coming from source or target domain. The domain label is defined as $\boldsymbol{d} = [0, 1]^T$ (which stands for source domain) or $\boldsymbol{d} = [1, 0]^T$ (which stands for target domain).

$$L_d = ||D(\boldsymbol{z}) - \boldsymbol{d}||_2^2 \tag{1}$$

$$L_e = L_y(C(\boldsymbol{z}), \boldsymbol{y}) - \lambda_d * L_d \tag{2}$$

Where L_d denotes the domain classification loss of discriminator, L_y denotes the scene classification loss of predictor, and y denotes the scene label of audio. L_e denotes the training objective function, which minimizes the scene classification loss and meanwhile maximizes the domain classification loss. The parameter λ_d controls the trade-off between L_y and L_d. In order to simplify our prove, we only consider the relationship between D and E. Formally, DANN performs a maximin optimization with the value function $V_d(E, D)$.

$$\max_E \min_D V_d(E, D) = \mathbb{E}(L_d(D(E(\boldsymbol{x})), \boldsymbol{d})) \tag{3}$$

Where $\mathbb{E}$ denotes math expectation. When E is fixed, the optimal D is shown as formula (4).

$$\begin{aligned} D_E^* &= \arg\min_D \mathbb{E}_{(\boldsymbol{z},\boldsymbol{d})\sim p(\boldsymbol{z},\boldsymbol{d})}[||D(\boldsymbol{z}) - \boldsymbol{d})||_2^2] \\ &= \arg\min_D \mathbb{E}_{\boldsymbol{z}\sim p(\boldsymbol{z})}\mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[||D(\boldsymbol{z}) - \boldsymbol{d})||_2^2] \end{aligned} \tag{4}$$

Formula (4) is equivalent to minimize $\mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[||D(\boldsymbol{z}) - \boldsymbol{d}||_2^2]$.

$$\begin{aligned} \mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[||D(\boldsymbol{z}) - \boldsymbol{d}||_2^2] = \mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[\boldsymbol{d}^2] - \\ -2D(\boldsymbol{z})\mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[\boldsymbol{d}] + D(\boldsymbol{z})^2 \end{aligned} \tag{5}$$

Formula (5) is a quadratic form of $D(\boldsymbol{z})$ which achieves the minimum at $D(\boldsymbol{z}) = \mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[\boldsymbol{d}]$.
Assuming that D always achieves its optimum, the maximin optimization in formula (3) can be reformulated as maximizing $C_d(E)$ where

$$C_d(E) \triangleq \min_D V_d(E, D) = V_d(E, D_E^*) \tag{6}$$

When D achieves its optimal, we fix it and then update E to maximize $C_d(E)$. Considering such scenario, there are one source domain and two target domains. As Figure 2 shows, two *samples* (a and b) are on different target domains, and *sample a* is easier to adapt because it is closer to source domain. But when updating parameters of E (denoted as θ) by gradient descent, we find that two samples have the same gradient $\frac{\partial L_d}{\partial \theta}$, because $\mathbb{E}_{\boldsymbol{d}\sim p(\boldsymbol{d}|\boldsymbol{z})}[\boldsymbol{d}]$ is a constant and they have same domain label. If a target domain is far from source domain, which needs a greater gradient. When combining all target domain as a single one, some difficult-adapt domains cannot be adapted well.

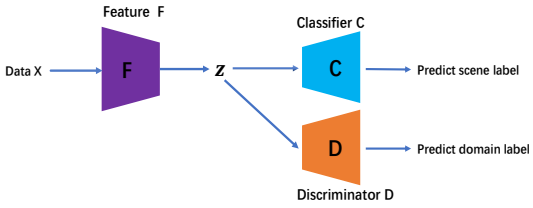

Figure 3: *Multi-target domain adaptation network.*

3. Proposed method

In this section, we formalize the problem of adaptation among multi-target domain, and describe our proposed methods.

3.1. Problem formulation

We consider a multi-class (K-class) classification problem for ASC. Let $(X, Y, D) = \{(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{d}_i)\}_{i=0}^N$ be a collection of M domains (a labeled source domain and $M-1$ unlabeled target domains). $\boldsymbol{x}_i$ denotes the i-th audio sample. $\boldsymbol{y}_i$ denotes scene label of the i-th audio, and $\boldsymbol{y}_i = [y_i^0, y_i^1, y_i^2, ..., y_i^K]$. $\boldsymbol{d}_i$ denotes domain label of the i-th audio, and $\boldsymbol{d}_i = [d_i^0, d_i^1, d_i^2, ..., d_i^{M-1}]$. Scene labels $\boldsymbol{y}_i$ and domain label $\boldsymbol{d}_i$ are both one-hot vectors. $\boldsymbol{y}_i$ is only available for the source samples, but $\boldsymbol{d}_i$ is available for all samples. Figure 3 shows the diagram of the proposed method. For any $\boldsymbol{x}_i$, Feature (F) is used to extract its features $\boldsymbol{z}_i$, and $\boldsymbol{z}_i = F(\boldsymbol{x}_i)$. The Classifier (C) tries to predict scene label, and the Discriminator (D) tries to predict domain label. F aims to project acoustic features from different domains into one subspace where the features are scene-discriminative and domain-invariant. D tries to discriminate which domains the input audio recording comes from.

3.2. Domain distance and domain index

Domain distance We can find two useful things by analysing Figure 1 and Table 1. Firstly, different devices cause different degrees of domain shift. Secondly, the greater the domain shift of the target domain, the poorer performance the target domain can obtain and the harder to adapt, e.g., device S1 and S2. Based on these two facts, we introduce domain distance to describe the extent of target domain shift. Domain distance is defined as the distance between the target domain and the source domain. To calculate the domain distance, the parallel data (it means these audios are recorded in the same environment simultaneously, but using different devices) are used. We assume that parallel samples contain the same information about the acoustic scenes and differ only due to device characteristics. To calculate domain distance conveniently, we reduce the dimension of the mel-spectrum of audio data ($\boldsymbol{x}$) by t-SNE algorithm [23]. We define $\boldsymbol{a}$ as the reduced dimension data, and $\boldsymbol{a} = tSNE(\boldsymbol{x})$. Then we use formula (7) to get domain distance.

$$Distance_i = \frac{1}{N}\sum_{j=0}^{N-1} ||\boldsymbol{a}_{i,j} - \boldsymbol{a}^*_{i,j}|| \tag{7}$$

Where $Distance_i$ denotes the distance from the i-th target domain to the source domain. $\boldsymbol{a}_{i,j}$ denotes the j-th data of the i-th target domain, and $\boldsymbol{a}^*_{i,j}$ is the data of source domain parallel to $\boldsymbol{a}_{i,j}$. N denotes the number of parallel data. We rank target domains according to their domain distance. If the target domain has high ranking, it is hard to adapt.

Domain index Domain index is used to quantify domain distance, which indicates the relative distance between the target and the source domain. So the ranking of target domain is used as their domain index. The index of source domain is set as 0.

3.3. Multi-classification task

According to previous methods [14, 15, 17], it is easy to consider letting discriminator to do a multi-classification task. The feed-forward process are summarized as formula (8) shows.

$$\boldsymbol{z} = F(\boldsymbol{x}), \tilde{\boldsymbol{y}} = C(\boldsymbol{z}), \tilde{\boldsymbol{d}} = D(\boldsymbol{z}) \tag{8}$$

The update process is using backpropagation algorithm. For C and D, loss function is cross-entropy function. For Feature, loss function is defined as formula (9) shows, where L_y and L_d denote the error of scene classification and domain classification, respectively. λ_d is a hyper-parameter.

$$L_f = L_y(C(\boldsymbol{z}), \boldsymbol{y}) - \lambda_d * L_d \tag{9}$$

Inspired by focal loss [24], we modify cross-entropy function of Discriminator according to domain index. Our motivation is paying more attention to the hard-to-adapt domain. The improved loss function is defined as formula (10) shows. Where u_i denotes the domain index of i-th sample, N denotes the number of data. $\boldsymbol{d}_i$ and $\tilde{\boldsymbol{d}}_i$ denote the domain label and the predicted result of i-th sample, respectively. T is a hyper-parameter, in our experiments, we set $T = 10$.

$$L_d = -\sum_{i=0}^{N-1} \frac{u_i + 1}{T} \boldsymbol{d_i} \log \tilde{\boldsymbol{d}}_{\boldsymbol{i}} \tag{10}$$

3.4. Regression task

We note that the domain index plays the role of a distance metric, i.e., it captures a similarity distance between the target domain and the source domain. So the domain index also can be viewed as domain label, and we consider letting discriminator to regress the domain index using a distance-base loss, such as L_2 loss. The feed-forward process and the update process are similar with multi-classification task in Section 3.3, the difference is that Discriminator just needs to predict domain index (denoted as u), and the loss function of Discriminator is defined as $L_d = ||D(\boldsymbol{z}) - u||_2^2$.

Table 2: *Comparison of accuracy on DCASE 2019 task1b dataset.*

model	A(%)	B(%)	C(%)	B&C(%)
DANN [17]	60.3	42.7	46.7	44.7
MTDA-R(ours)	**63.9**	40.6	52.1	46.4
MTDA-C1(ours)	62.4	42.4	50.3	46.3
MTDA-C2(ours)	63	**45.5**	**54.4**	**49.9**

4. Experiment

4.1. Datasets and metrics

Datasets The DCASE 2019 task1B dataset [25] and the DCASE 2020 task1A dataset [21] contain 10s segments, recorded at 48kHz and spanning 10 classes.

Evaluation metrics For all the experiments, we use the accuracy of classification as the evaluation metric, which is one of the most commonly used metrics for audio classification [26].

4.2. Experiments on DCASE2019 dataset

Our first experiment evaluates on DCASE 2019 task1B [25]. In the DCASE2019 task1B dataset, all data are recorded by device A, B and C. The data of device A is regarded as source domain and that of device B and C as two target domains.

Experimental setups To make a fair comparison, we use the same model structure and experimental setting as DANN [17]. See [17] for more details about DANN.

Experimental results and analysis Table 2 demonstrates the performance of our proposed methods and DANN on DCASE 2019 task1B dataset. We set three different experiments for MTDA. MTDA-R denotes that we make Discriminator do regression task. MTDA-C1 denotes that we make Discriminator do multi-classification task and choose cross-entropy as loss function. MTDA-C2 denotes loss function is improved cross-entropy according to formula (10). All of our three experiments perform better than DANN, which confirms the effectiveness of MTDA. MTDA-C2 performs better than MTDA-C1, which shows dividing target domain into easy-to-adapt and hard-to-adapt domain is very useful.

4.3. Experiments on DCASE2020 dataset

Our second experiment evaluates on DCASE 2020 task1A [21]. We take the data of device A as source domain and that of device B, C, S1, S2, S3 as target domains. Furthermore, we set the data of device S4, S5 and S6 as unseen domains, these data only available on test stage. For device A, 10215 segments audio are used to train. For device B, C, S1-S3, 750 segments audio are used to adapt, these audio are regarded as unlabeled data. In the test set, 330 segments audio for each device are used.

Previous methods We compare our method with previous state-of-the-art unsupervised DA methods for ASC including DANN [17], UADA [14], W-UADA [15], MCD [16], MMD [27]. To fairly compare with these methods, we choose two baseline models. One is DCASE model [28], which consists of 8 layers CNN. The other is Resnet14 model [29].

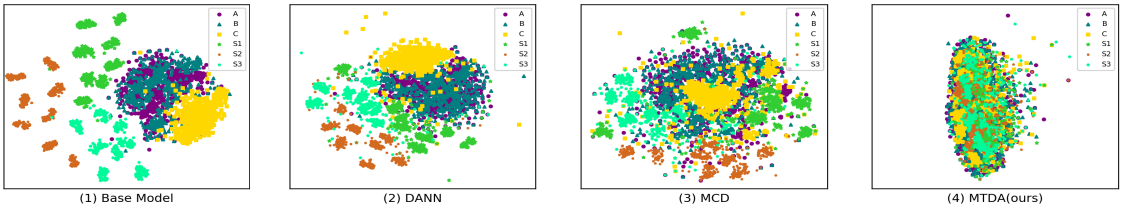

Figure 4: *Visualization of feature ($\boldsymbol{z}$) by t-SNE, which is extracted from audio data ($\boldsymbol{x}$) by module Feature.*

Preprocessing All the raw audios are resampled to 32kHz and fixed to a certain length of 10s. The short time Fourier transform (STFT) is then applied on the audio signals to calculate spectrograms, with a window size of 32ms and a hop size of 15.6ms. 64 mel filter banks are applied on the spectrograms followed by a logarithmic operation to extract the log mel spectrograms.

Experiment setting In the training phase, for UADA and W-UADA, the RMSProp [30] optimizer is used. For other methods, the Adam algorithm [31] is employed as the optimizer. All models are trained with an initial learning rate of 0.002. The batch size is set to 32 and training epoch is 200. λ_d is chosen from $\{0.2, 0.5, 1.0, 2.0, 5.0, 8.0, 10.0\}$, experimental results indicate $\lambda_d = 2.0$ is the best parameter. In our experiments, we never use any data augmentation methods.

Experimental results and analysis Table 3 and Table 4 demonstrate the performance of our proposed MTDA and other state-of-the-art methods. For base model [28, 29], we only train models on source data and do not use any domain adaptation. Due to the difference in the acquisition equipment greatly affects the characteristics of the audio signals, test data without domain adaptation is difficult to obtain good performance on the classifier trained in the source domain. DANN-respective denotes that we apply DANN for each target domain separately. For methods [27, 14, 15, 17, 16], we combine the data of device B, C, S1-S3 as one target domain. Although DANN and MCD can get good performance on easy-to-adapt domain, such as device B and C, they cannot perform well on hard-to-adapt domain, such as device S1-S3. On the contrary, our methods can get better performance on device S1-S3. Our methods perform better than DANN-respective, which shows MTDA can make better use of the underlying relation among multiple domains. Furthermore, our methods get better performance on unseen domains (device S4-S6), which shows our methods have good generalization. We note that previous methods [27, 14, 15, 16] cannot successfully classify source samples (device A) compared with baseline. It means that the loss of source domain information is influenced by adaptation process. Although our methods cannot overcome this problem completely, our methods also yield comparable results with baseline.

Comparison between MTDA-C1 and MTDA-C2 MTDA-C2 achieves higher accuracy than MTDA-C1 for the reason that MTDA-C2 pays more attention to hard-to-adapt domain.

Comparison between MTDA-C and MTDA-R MTDA-R also gets good performance, which validates the effectiveness of making Discriminator regress domain index. MTDA-R can get comparable results with MTDA-C2 for the reason that domain index plays the role of a distance metric, which means that regressing domain index by the Discriminator can also benefit the hard-to-adapt domain. Furthermore, MTDA-R achieves higher accuracy on unseen domain than MTDA-C.

Feature visualization T-SNE [23] is used to visualize adaptation results for different methods. Figure 4 (a) shows the results of base model [28]. Figure 4 (b) and Figure 4 (c) show adaptation results of DANN and MCD model, and they can adapt to easy-to-adapt domains, such as device B and C, but they cannot adapt to hard-to-adapt domains well (S1-S3). Figure 4 (d) shows adaptation results of MTDA-C2. We can find that almost all distribution of target domains align with the source domain.

Table 3: *Accuracy (%) comparison of different methods on DCASE 2020 task1a development set. The base model is DCASE model, and all methods are based on this model.*

model	A	B&C	S1-S3	S4-S6
Base(DCASE) [28]	**68.8**	41.1	23.6	24.9
MMD [27]	66.1	42.6	25.7	30.8
UADA [14]	59.4	44.7	37.4	32.8
W-UADA [15]	67.0	46.4	30.9	30.9
DANN [17]	68.2	51.5	37.1	36.7
DANN-respective	68.2	54.1	37.8	-
MCD [16]	63.9	52.1	33.0	36.8
MTDA-R(ours)	68.5	52.6	**41.7**	**41.5**
MTDA-C1(ours)	67.9	52.0	39.3	37.7
MTDA-C2(ours)	68.2	**54.5**	**41.7**	41.4

Table 4: *Accuracy (%) comparison of different methods on DCASE 2020 task1a development set. The base model is Resnet model, and all methods are based on this model.*

model	A	B&C	S1-S3	S4-S6
Base(Resnet14) [29]	**67.6**	39.8	22.4	25.6
MMD [27]	62.7	45.9	26.8	29.3
UADA [14]	63.3	47.3	37.6	39.6
W-UADA [15]	64.2	45.0	33.2	34.3
DANN [17]	65.9	50.2	35.1	37.9
DANN-respective	65.3	51.1	38.6	-
MCD [16]	65.2	51.2	30.2	33.1
MTDA-R(ours)	67.2	51.5	41.9	**39.9**
MTDA-C1(ours)	65.5	50.5	37.6	39.1
MTDA-C2(ours)	67.5	**53.2**	**42.3**	39.4

5. Conclusions

A novel multi-target domain adaption (MTDA) method for ASC has been proposed. Our method can adapt multiple target domains simultaneously and make use of the underlying relation among domains. Experimental results on the ASC tasks and visualization analysis validate the advantages of our method.

6. Acknowledgements

This paper was partially supported by the Shenzhen Science & Technology Fundamental Research Programs (No: JCYJ20180507182908274 & JSGG20191129105421211) and GXWD20201231165807007-20200814115301001.

7. References

[1] S. Hershey, S. Chaudhuri, D. P. Ellis, J. F. Gemmeke, A. Jansen, R. C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold *et al.*, "CNN architectures for large-scale audio classification," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 131–135.

[2] K. J. Piczak, "Environmental sound classification with convolutional neural networks," in *IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2015, pp. 1–6.

[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," in *Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2018.

[4] H. Wang, Y. Zou, D. Chong, and W. Wang, "Environmental sound classification with parallel temporal-spectral attention," in *Proc. Interspeech*, 2020, pp. 821–825.

[5] Q. Kong, Y. Cao, T. Iqbal, Y. Wang, W. Wang, and M. D. Plumbley, "PANNs: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2020.

[6] T. Nguyen and F. Pernkopf, "Acoustic scene classification with mismatched devices using cliquenets and mixup data augmentation," in *Proc. Interspeech*, 2019, pp. 2330–2334.

[7] H. Hu, C.-H. H. Yang, X. Xia, X. Bai, X. Tang, Y. Wang, S. Niu, L. Chai, J. Li, H. Zhu *et al.*, "A two-stage approach to device-robust acoustic scene classification," *arXiv preprint arXiv:2011.01447*, 2020.

[8] M. Kosmider, "Spectrum correction: Acoustic scene classification with mismatched recording devices," in *Proc. Interspeech*, 2020, pp. 4641–4645.

[9] T. Nguyen, F. Pernkopf, and M. Kosmider, "Acoustic scene classification for mismatched recording devices using heated-up softmax and spectrum correction," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 126–130.

[10] P. Primus, H. Eghbal-zadeh, D. Eitelsebner, K. Koutini, A. Arzt, and G. Widmer, "Exploiting parallel audio recordings to enforce device invariance in CNN-based acoustic scene classification," in *Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2019.

[11] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan, "A theory of learning from different domains," *Machine learning*, vol. 79, no. 1-2, pp. 151–175, 2010.

[12] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," *arXiv preprint arXiv:1608.06019*, 2016.

[13] H. Wang, H. He, and D. Katabi, "Continuously indexed domain adaptation," *arXiv preprint arXiv:2007.01807*, 2020.

[14] S. Gharib, K. Drossos, E. Cakir, D. Serdyuk, and T. Virtanen, "Unsupervised adversarial domain adaptation for acoustic scene classification," in *Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2018.

[15] K. Drossos, P. Magron, and T. Virtanen, "Unsupervised adversarial domain adaptation based on the wasserstein distance for acoustic scene classification," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019, pp. 259–263.

[16] S. Takeyama, T. Komatsu, K. Miyazaki, M. Togami, and S. Ono, "Robust acoustic scene classification to multiple devices using maximum classifier discrepancy and knowledge distillation," in *28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2020, pp. 36–40.

[17] R. Wang, M. Wang, X.-L. Zhang, and S. Rahardja, "Domain adaptation neural network for acoustic scene classification in mismatched conditions," in *Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. IEEE, 2019, pp. 1501–1505.

[18] A. I. Mezza, E. A. Habets, M. Müller, and A. Sarti, "Unsupervised domain adaptation for acoustic scene classification using band-wise statistics matching," in *28th European Signal Processing Conference (EUSIPCO)*. IEEE, 2020, pp. 11–15.

[19] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," in *International Conference on Machine Learning (ICML)*. PMLR, 2015, pp. 1180–1189.

[20] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada, "Maximum classifier discrepancy for unsupervised domain adaptation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 3723–3732.

[21] T. Heittola, A. Mesaros, and T. Virtanen, "Acoustic scene classification in dcase 2020 challenge: generalization across devices and low complexity solutions," *arXiv preprint arXiv:2005.14623*, 2020.

[22] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[23] L. Van der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of machine learning research*, vol. 9, no. 11, 2008.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[25] A. Mesaros, T. Heittola, and T. Virtanen, "Acoustic scene classification in dcase 2019 challenge: Closed and open set classification and data mismatch setups," in *Detection and Classification of Acoustic Scenes and Events (DCASE)*, 2019.

[26] T. Virtanen, M. D. Plumbley, and D. Ellis, *Computational Analysis of Sound Scenes and Events*. Springer, 2018.

[27] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *International Conference on Machine Learning (ICML)*. PMLR, 2015, pp. 97–105.

[28] Q. Kong, Y. Cao, T. Iqbal, Y. Xu, W. Wang, and M. D. Plumbley, "Cross-task learning for audio tagging, sound event detection and spatial localization: Dcase 2019 baseline systems," *arXiv preprint arXiv:1904.03476*, 2019.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[30] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural Networks for Machine Learning*, vol. 4, no. 2, pp. 26–31, 2012.

[31] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.