

# Unsupervised Adversarial Domain Adaptation Based On The Wasserstein Distance For Acoustic Scene Classification

Konstantinos Drossos, Paul Magron, and Tuomas Virtanen  
Audio Research Group, Tampere University, Tampere, Finland

## Abstract

A challenging problem in deep learning-based machine listening field is the degradation of the performance when using data from unseen conditions. In this paper we focus on the acoustic scene classification (ASC) task and propose an adversarial deep learning method to allow adapting an acoustic scene classification system to deal with a new acoustic channel resulting from data captured with a different recording device. We build upon the theoretical model of  $\mathcal{H}\Delta\mathcal{H}$ -distance and previous adversarial discriminative deep learning method for ASC unsupervised domain adaptation, and we present an adversarial training based method using the Wasserstein distance. We improve the state-of-the-art mean accuracy on the data from the unseen conditions from 32% to 45%, using the TUT Acoustic Scenes dataset.

**Keywords:** Acoustic scene classification, unsupervised domain adaptation, Wasserstein distance, adversarial training

## 1 Introduction

The task of acoustic scene classification (ASC) consists in classifying a sound segment  $\mathbf{x} \sim \mathcal{X}$  into one of the predefined classes  $y \sim \mathcal{Y}$  representing different acoustic scenes (e.g., “urban area”, “metro station”), where  $\mathcal{X}$  and  $\mathcal{Y}$  are the underlying distributions of the sound segments and predefined classes, respectively. ASC has recently been tackled with deep learning methods [1, 2, 3, 4]. These methods can be viewed as consisting of a feature extractor that outputs a latent representation  $\mathbf{z}$  from  $\mathbf{x}$  and a label classifier that assigns a label  $\mathbf{y}$  to  $\mathbf{z}$ .

Training and testing data can exhibit different capturing conditions (e.g., different recording devices) and subjective and noisy labels (e.g., an “urban area” scene labeled as “residential area” and “town”), resulting in the degradation of the performance of the method [5, 6]. One promising way for tackling the above mentioned problems is *domain adaptation* (DA) [7, 8, 9, 10] and, in the relevant terminology, the above mentioned problems are the capture and

label biases, respectively [8, 11].

Domain adaptation is a sub-space alignment/divergence minimization problem which consists in optimizing a system on the data from the *source domain* and then adapting it to the data from the *target domain* [7, 10, 12]. The data from the source and target domains exhibit any combination of the capturing and labeling biases, leading to what is known as domain shift or dataset bias phenomenon. This phenomenon manifests as differences between the distributions  $\mathcal{X}_S$  and  $\mathcal{X}_T$ , and/or  $\mathcal{Y}_S$  and  $\mathcal{Y}_T$ , where the indices  $S$  and  $T$  refer to the source and target domains, respectively. The goal of DA is to minimize the error of a classifier when classifying data from the target domain. This is usually achieved by training a system which minimizes the discrepancy between the distributions  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$  of the learned latent representations from the source and target domains,  $\mathbf{z}_S$  and  $\mathbf{z}_T$  respectively. All the labels for the source data are available but only some or none labels are available for the target data. This results in semi- or

unsupervised DA, respectively.

We consider here the unsupervised adversarial DA, focused on the capturing bias problem. Previous approaches can be organized in two categories, depending on when DA is performed. Methods in the first category such as [7, 13, 14] perform DA jointly with the optimization on the source domain data. Methods in the other category such as [10, 15, 16] are two-stage approaches that first perform optimization on the source domain data and then do DA. The previous ASC unsupervised adversarial DA state-of-the-art (SOTA) method [5] falls in this second category. However, this method suffers from vanishing gradients and slow learning process, during the minimization of the discrepancy between the distributions of the source and target domains.

In this paper we present a deep learning method for ASC unsupervised adversarial DA which is based on the theoretical analysis of domain adaptation based on optimal transport, presented in [17]. We replace the adversarial adaptation process SOTA method [5] with the Wasserstein generative adversarial networks (WGAN) formulation [18], tackling the computational problems that hampered the performance of the previous SOTA method. Furthermore, we enhance the WGAN algorithm by using an extra loss for preventing the performance from dropping on the source domain when the system is adapted to the target domain. We achieve an increase of 13% in the mean accuracy on the target data compared to previous SOTA approach [5] using the TUT Acoustic Scenes dataset [19]. The contributions of our work are:

1. Providing the first theoretical based approach for deep learning DA on general audio;
2. Presenting the first work adopting the Wasserstein distance for ASC unsupervised adversarial DA;
3. Significantly improving the performance of ASC unsupervised adversarial DA over the previous SOTA method [5].

The rest of the paper is structured as follows. In Section 2 we present the most widely used theoretical background in DA. In Section 3 we introduce

our proposed method. The setup for evaluating our method is presented in Section 4 and the results are discussed in Section 5. Finally, the paper is concluded in Section 6.

## 2 Background

We present here a summary of the most widely used theoretical background for understanding adversarial deep learning based unsupervised DA methods, and the interested reader can refer to [12, 20] for further details.

### 2.1 Classifier error and target error bound

A domain  $\mathcal{D}$  is defined as the pair of a distribution  $\mathcal{Z}$  and a labeling process  $f$ .  $\mathcal{Z}$  is the distribution of  $\mathbf{z}$ , i.e.  $\mathbf{z} \sim \mathcal{Z}$ , and  $\mathbf{z}$  is used as an input to  $f$ . The output of  $f$  is considered as the ground truth that a classifier  $h$  should predict when performing classification over  $\mathbf{z}$ . It follows that  $\mathcal{D}_S = \langle \mathcal{Z}_S, f_S \rangle$  and  $\mathcal{D}_T = \langle \mathcal{Z}_T, f_T \rangle$  are the source and target domains, respectively. The expected error of an  $h$  over its input  $\mathbf{z}$  is

$$\epsilon(h, f) = \mathbb{E}_{\mathbf{z}}[\mathcal{L}(h(\mathbf{z}), f(\mathbf{z}))], \quad (1)$$

where  $\mathcal{L}$  is a loss function.  $\epsilon(h, f)$  indicates the average disagreement between the prediction of the classifier  $h$  and the output of the labeling function  $f$ .

If  $\mathbf{z}$  comes from the source (resp. target) domain, that is,  $\mathbf{z}_S \sim \mathcal{Z}_S$  (resp.  $\mathbf{z}_T \sim \mathcal{Z}_T$ ), then the error is the source error  $\epsilon_S(h, f_S)$  (resp. the target error  $\epsilon_T(h, f_T)$ ). The aim of unsupervised DA is to obtain a classifier  $h$  that yields a low value for  $\epsilon_S$  and adapt it to yield a low value for  $\epsilon_T$ , without employing the labels from  $\mathcal{D}_T$  during the adaptation process.

We can obtain a classifier  $h$  that yields a low value for  $\epsilon_S$  by following classical supervised training approaches. However, we cannot optimize  $h$  on  $\mathbf{z}_T$  from the target domain  $\mathcal{D}_T$  in a supervised manner, since we do not have access to the labels from  $\mathcal{D}_T$ . Intuitively, if there was a low discrepancy between  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$ , then the classifier  $h$  would also yield a low value for  $\epsilon_T$ . Calculating or estimating this discrepancy allows us to obtain a generalization bound for

$\epsilon_T$ , and reducing this bound will consequently reduce  $\epsilon_T$  as shown in [12, 17, 21].

The discrepancy between  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$  can be measured with the  $\mathcal{H}\Delta\mathcal{H}$ -distance [12, 20]. In the case of binary classifiers  $\mathcal{H}$ , the  $\mathcal{H}\Delta\mathcal{H}$ -distance is defined as

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{Z}_S, \mathcal{Z}_T) = 2 \sup_{h, h' \in \mathcal{H}} |\mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_S}[h(\mathbf{z}) \neq h'(\mathbf{z})] - \mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_T}[h(\mathbf{z}) \neq h'(\mathbf{z})]|. \quad (2)$$

In a nutshell,  $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{Z}_S, \mathcal{Z}_T)$  returns the highest prediction difference between two classifiers in  $\mathcal{H}$  under the two distributions  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$ . We use  $d_{\mathcal{H}\Delta\mathcal{H}}$  to bound the target error for DA [12, 20, 13] as

$$\epsilon_T(h, f_T) \leq \epsilon_S(h, f_S) + \frac{1}{2}d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{Z}_S, \mathcal{Z}_T) + \lambda, \quad (3)$$

where

$$\lambda = \epsilon_S(h^*, f_S) + \epsilon_T(h^*, f_T), \text{ and} \quad (4)$$

$$h^* = \underset{h \in \mathcal{H}}{\operatorname{argmin}} (\epsilon_S(h, f_S) + \epsilon_T(h, f_T)). \quad (5)$$

$h^*$  and  $\lambda$  are termed as the ideal joint classifier and combined error of the ideal joint classifier, respectively [12, 20]. Eq. 3 shows that the upper bound of the target error  $\epsilon_T(h, f_T)$  is affected by three factors: the error on the source domain (i.e.,  $\epsilon_S(h, f_S)$ ); the  $\mathcal{H}\Delta\mathcal{H}$ -distance between the source and target distributions; and the combined error of the ideal joint classifier (i.e.,  $\lambda$ ). In DA it is safely assumed that there is a classifier that performs well on both the source and the target domains, thus yielding a small value of  $\lambda$  and therefore allowing to neglect its effect [12]. Under this assumption, the problem of DA reduces to obtaining a classifier with a good performance on the source domain (i.e., minimizing  $\epsilon_S(h, f_S)$ ) and trying to minimize the discrepancy between the distributions of the two domains (i.e.,  $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{Z}_S, \mathcal{Z}_T)$ ).

## 2.2 Adversarial formulation

Consider a set of domain classifiers  $\mathcal{H}_d$  (i.e., classifiers that predict whether  $\mathbf{z}$  is from  $\mathcal{D}_S$  or  $\mathcal{D}_T$ ). Then,

according to [22],

$$\begin{aligned} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{Z}_S, \mathcal{Z}_T) &= 2 \sup_{h, h' \in \mathcal{H}} |\mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_S}[h(\mathbf{z}) \neq h'(\mathbf{z})] - \mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_T}[h(\mathbf{z}) \neq h'(\mathbf{z})]| \\ &\leq 2 \sup_{h_d \in \mathcal{H}_d} |\mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_S}[h_d(\mathbf{z}) = 1] - \mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_T}[h_d(\mathbf{z}) = 1]| \\ &= 2 \sup_{h_d \in \mathcal{H}_d} |\mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_S}[h_d(\mathbf{z}) = 0] + \mathbb{P}_{\mathbf{z} \sim \mathcal{Z}_T}[h_d(\mathbf{z}) = 1] - 1|. \end{aligned} \quad (6)$$

Thus, the ideal  $h_d$  yields as an upper bound for  $d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{Z}_S, \mathcal{Z}_T)$  [7]. This can be exploited in an adversarial setting, where the focus is to obtain a domain classifier  $h_d$  good enough to predict the domain of  $\mathbf{z}$  and a feature extractor that confuses the domain classifier. In this setting, instead of directly minimizing  $d_{\mathcal{H}\Delta\mathcal{H}}$ , one aims at obtaining a feature extractor  $M$  that is able to fool a good domain classifier  $h_d$ .

## 2.3 Previous SOTA

The previous SOTA approach for ASC unsupervised adversarial DA [5] falls in the second category of approaches mentioned in the introduction (i.e. first performing optimization on source domain and then doing DA). In the first stage, a feature extractor  $M_S$  is obtained during the optimization of the label classification, and in the second stage, a copy  $M_T$  of  $M_S$  is further optimized during the adversarial training. Following the framework presented in Section 2.2, the second stage of this approach consists of an adversarial training that aims at minimizing  $d_{\mathcal{H}\Delta\mathcal{H}}(M_S(\mathbf{x}_S), M_T(\mathbf{x}_T))$ . The implementation of the adversarial training follows the typical formulation initially presented in [23], using two losses. The first one is used for optimizing  $h_d$  over the output of  $M_T$  and the second loss is utilized for making  $M_T$  to produce an output that maximizes the domain classification error. In the field of generative adversarial neural networks (GANs), these are the losses associated with the discriminator (i.e.,  $h_d$ ) and the generator (i.e.,  $M_T$ ). The GAN loss  $\mathcal{L}_{\text{GAN}}$  used in the previous SOTA approach can be formulated [5, 10, 23]

as

$$\mathcal{L}_{\text{GAN}}(h_d, M) = \mathbb{E}_{\mathbf{x}_S}[\log \sigma(h_d(M_S(\mathbf{x}_S)))] + \mathbb{E}_{\mathbf{x}_T}[\log(1 - \sigma(h_d(M_T(\mathbf{x}_T))))], \quad (7)$$

where  $\sigma$  is the sigmoid function. The adversarial process aims [23] at solving the optimization problem

$$\min_{h_d} \max_{M_T} \mathcal{L}_{\text{GAN}}(h_d, M_T). \quad (8)$$

However, solving the optimization problem described in Eq. (8) is known to introduce computational issues, such as vanishing gradients and slow learning process for  $h_d$  [18]. These problems hamper the performance of the adversarial training [18] and thus limit the performance of the method proposed in [5]. The method proposed in this paper aims at tackling these problems for the ASC unsupervised DA task.

### 3 Proposed method

Our proposed method builds upon the framework presented in Section 2 and it can be applied with any deep neural network (DNN) that performs ASC. We employ a DNN and the Wasserstein generative adversarial networks (WGAN) formulation and algorithm, presented in [18]. Our DNN consists of a feature extractor  $M$ , a label classifier  $h$ , and a domain classifier  $h_d$ . We consider as  $\mathbf{z}$  the output of our feature extractor  $M$ , which is used as an input to our  $h$ . The output of  $h$  is a vector and the output of  $h_d$  is a scalar.

#### 3.1 Source domain training

As a first step, we optimize  $M$  and  $h$  using the labeled data  $(\mathbf{x}_s, \mathbf{y}_s)$  from source domain  $(\mathbb{X}_S, \mathbb{Y}_S)$ , where  $\mathbf{y}_S$  is an 1-hot encoding of the available classes. We employ the binary cross-entropy as the source domain loss function  $\epsilon_S(h, f_S)$ :

$$\mathcal{L}_{\text{labels}}(h, M) = - \sum_{(\mathbf{x}, \mathbf{y}) \in (\mathbb{X}_S, \mathbb{Y}_S)} \mathbf{y}^T \log(h(M(\mathbf{x}))), \quad (9)$$

and we obtain the classifier  $h^*$  and the source domain feature extractor  $M_S$  by

$$h^*, M_S = \underset{h, M}{\operatorname{argmin}} \mathcal{L}_{\text{labels}}(h, M). \quad (10)$$

---

#### Algorithm 1 WGAN based training algorithm

---

**Require:** the learning rate  $\alpha$ , the clipping parameter  $c$ , the batch size  $m$ , the number of iterations  $n_d$ , the domain classifier  $h_d$  with parameters  $w_d$ , the feature extractor  $M_S$  with parameters  $w_{M_S}$ .

```

1: Initialization:  $w_{M_T} = w_{M_S}$ 
2: while  $w_{M_T}$  not converged do
3:   for  $i = 1, \dots, n_d$  do
4:     Get  $m$  samples  $\{\mathbf{x}_n^S\}_{n=1}^m$  from  $\mathbb{X}_S$ 
5:     Get  $m$  samples  $\{\mathbf{x}_n^T\}_{n=1}^m$  from  $\mathbb{X}_T$ 
6:      $g_{w_d} \leftarrow \nabla_{w_d} [\frac{1}{m} \sum_{n=1}^m h_d(M_S(\mathbf{x}_n^S)) - \frac{1}{m} \sum_{n=1}^m h_d(M_T(\mathbf{x}_n^T))]$ 
7:      $w_d \leftarrow w_d - \alpha \operatorname{Opt}(w_d, g_{w_d})$ 
8:      $w_d \leftarrow \operatorname{clip}(w_d, c)$ 
9:   end for
10:  Get  $m$  samples  $\{\mathbf{x}_n^S, \mathbf{y}_n\}_{n=1}^m$  from  $(\mathbb{X}_S, \mathbb{Y}_S)$ 
11:  Get  $m$  samples  $\{\mathbf{x}_n^T\}_{n=1}^m$  from  $\mathbb{X}_T$ 
12:   $g_{w_{M_T}} \leftarrow \nabla_{w_{M_T}} [\frac{1}{m} \sum_{n=1}^m h_d(M_T(\mathbf{x}_n)) - \frac{1}{m} \sum_{n=1}^m \mathbf{y}_n^T \log(h^*(M_T(\mathbf{x}_n)))]$ 
13:   $w_{M_T} \leftarrow w_{M_T} - \alpha \operatorname{Opt}(w_{M_T}, g_{w_{M_T}})$ 
14: end while
15: return  $M_T$ 

```

---

The parameters of the feature extractor  $M_S$  are denoted  $w_{M_S}$  and will be used as initial values for the adapted feature extractor  $M_T$ .

#### 3.2 Wasserstein adversarial formulation

As a second step, we aim at adapting  $M_S$  to the target domain using an adversarial training process as described in Section 2.2. However, as pointed out in Section 2.3, using  $d_{\mathcal{H}\Delta\mathcal{H}}$  as a discrepancy measure between distributions yields computational problems. To alleviate this issue, we propose to employ the order-1 Wasserstein distance (called Wasserstein distance from now on)  $W$  [18, 24] as a metric for the discrepancy between  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$ .  $W$  comes as a natural and intuitive candidate for our method since its

minimization does not suffer from the problems mentioned in Section 2.3. Additionally, the usage of a binary label classifier  $h$  by  $\mathcal{H}\Delta\mathcal{H}$ -distance introduces an intractable problem in practice, but  $W$  distance does not suffer from this problem as it does not require  $h$  to be a binary classifier [17]. Furthermore,  $W$  is proven to enhance the results of adversarial training, and it is a weak topology over the space of  $\mathcal{Z}$  allowing important convergence modes such as smooth convergence and point-wise convergence [12, 18, 17]. Finally,  $W$  accounts for the geometry of the space of  $\mathcal{Z}$ , therefore being an appropriate distance for measuring the discrepancy between the two distributions  $\mathcal{Z}_T$  and  $\mathcal{Z}_S$  [17].  $W$  is defined [18] as

$$W(\mathcal{Z}_S, \mathcal{Z}_T) = \inf_{(\mathbf{z}_S, \mathbf{z}_T) \sim \prod(\mathcal{Z}_S, \mathcal{Z}_T)} \mathbb{E}_{(\mathbf{z}_S, \mathbf{z}_T)} [\|\mathbf{z}_S - \mathbf{z}_T\|], \quad (11)$$

where  $\prod(\mathcal{Z}_S, \mathcal{Z}_T)$  is the set of all joint distributions whose respective marginals are  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$ . It is proven [17] that we can use  $W$  as a divergence metric to upper bound the target error  $\epsilon_T$  using an expression similar to Eq. (3):

$$\epsilon_T(h, f_T) \leq \epsilon_S(h, f_S) + W(\mathcal{Z}_S, \mathcal{Z}_T) + \lambda. \quad (12)$$

In Eq. (12) the factors corresponding to the source error and the ideal joint classifier (i.e.,  $\epsilon_S(h, f_S)$  and  $\lambda$ ) are the same as in Eq. (3). The difference is the factor representing the discrepancy between the two distributions,  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$ , i.e.,  $W(\mathcal{Z}_S, \mathcal{Z}_T)$  instead of  $d_{H\Delta H}(\mathcal{Z}_S, \mathcal{Z}_T)$ . Thus, we can use the WGAN training algorithm presented in [18], which yields the adapted feature extractor  $M_T$ . The process of the adaptation of  $M_T$  is performed by the iterative minimization of the losses

$$\sum_{\mathbf{x} \in \mathbb{X}_S} h_d(M_S(\mathbf{x})) - \sum_{\mathbf{x} \in \mathbb{X}_T} h_d(M_T(\mathbf{x})) \text{ and} \quad (13)$$

$$\sum_{\mathbf{x} \in \mathbb{X}_T} h_d(M_T(\mathbf{x})) + \mathcal{L}_{\text{labels}}(h^*, M_T). \quad (14)$$

Eq. (13) and (14) (without the  $\mathcal{L}_{\text{labels}}(h^*, M_T)$  term) are proposed in the original WGAN algorithm and their minimization is shown to minimize the  $W$  distance in Eq. (11) [18]. We enhance the original WGAN losses and training algorithm by including an

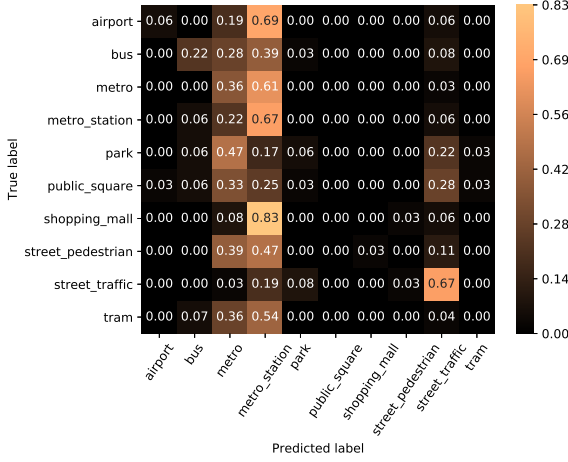
extra cross-entropy loss (the second term in Eq. (14)), similarly to [5]. The goal of this addition is to prevent the adapted model from deteriorating the performance on the source domain  $\mathcal{D}_S$ . Our method is summarized in Algorithm 1 and our enhancement to the original WGAN algorithm is at line 12 of this algorithm. Lines 6 and 12 are the calculations over a minibatch of the gradients of Eq. (13) and (14), respectively. Note that Algorithm 1 uses an optimizer *Opt* chosen as the RMSProp optimizer [25] and a clipping function *clip* defined as:

$$\text{clip}(x, c) = \max(\min(x, c), -c). \quad (15)$$

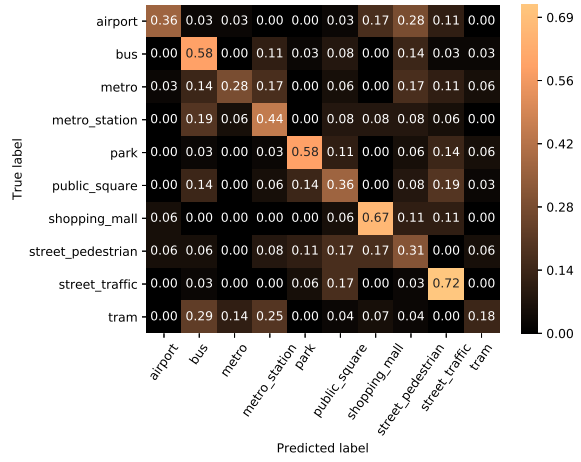
## 4 Evaluation setup

To evaluate our method and obtain comparable results, we use the same data and models for  $M$  and  $h$  as in the previous SOTA method for ASC unsupervised DA [5]. The data that we use are from the TUT Acoustic Scenes dataset used in the previous SOTA approach [5, 19], and consist of 64 log mel-bank energies, extracted from audio recordings of 10 different acoustic scenes, namely: “airport”, “bus”, “metro”, “metro station”, “park”, “public square”, “shopping mall”, “street pedestrian”, “street traffic”, and “tram”. The recordings are performed with three different devices, denoted A, B, and C. Device A is regarded as a high quality recording device and the other two as consumer devices. Data from device A are 24 hours long, and data from devices B and C are two hours long, each. We follow the splitting of the data in training, validation, and testing as in [5, 19], leading to training and validation set consisting of approximately 15 hours of data from device A, 1.35 hours from device B, and 1.35 hours from device C. Consequently, the testing set consists of approximately 7 hours of data from device A, 0.5 hours from device B, and 0.5 hours from device C. We also adopt the definition of source and target domain data as in [5, 19]. That is, we regard the data from device A as our source domain data, and the data from the consumer devices as our target domain data.

We employ the non-adapted, pre-trained model referred to as “Kaggle model” in [5] as our  $M_S$ , and the



(a) Non adapted model



(b) Adapted model

Figure 1: Normalized confusion matrices for the non- and adapted models on the target domain data  $\mathbf{x}_T \in \mathbb{X}_T$ .

pre-trained classifier as our  $h$ . Both  $M_S$  and  $h$  are freely available at [19].  $M_S$  consists of five 2D convolutional neural network (CNN) layers, with square kernel shapes of widths  $\{11, 5, 3, 3, 3\}$ , and amount of channels  $\{48, 128, 192, 192, 128\}$ . The utilized stride is  $(2, 3)$  for the first two CNNs and  $(1, 1)$  for the others. All CNNs are followed by a rectified linear unit (ReLU) non-linearity, and the first two and last CNNs also use batch normalization and max pooling, with square kernels of shape 3 and a stride of  $\{(1, 2), (2, 2), (1, 2)\}$ .  $h$  consists of three feed-forward layers, each one followed by a ReLU. The output non-linearity of  $h$  is the softmax function. The method was implemented using the PyTorch framework [26] for Python programming language. For the adaptation stage, the RMSProp optimizer is used with a learning rate of  $5 \times 10^{-5}$  and have its rest parameters at the proposed default values. We use a batch size of 16 and the adapted  $M_T$  was obtained after the saturation of the first term in Eq. 14 (after approximately 300 epochs). For a reproducible research purpose, we offer all of our code<sup>1</sup> and adapted  $M_T$  model<sup>2</sup> in online repositories.

<sup>1</sup><https://github.com/dr-costas/undaw>

<sup>2</sup><https://doi.org/10.5281/zenodo.2649151>

## 5 Results and discussion

We present the mean accuracy and the normalized classification results per class as a confusion matrix, for adapted and non-adapted models. Table 1 presents the obtained mean accuracy of the adapted and non-adapted models from our method and the previous SOTA approach. As it can be seen from Table 1, our adaptation method provides a significant increase of 13% in mean accuracy for the target domain compared to the previous SOTA. The 1% decrease in accuracy of the adapted model on the source domain and the 1% increase in accuracy of the non-adapted model on the target domain, are considered insignificant and can be attributed to number rounding processes. In Figure 1 are the confusion matrices for the non-adapted and adapted models on the target domain data  $\mathbf{x}_T \in \mathbb{X}_T$ . In Figure 1b it can be seen that there is an increase of the values on the diagonal, compared to Figure 1a. This indicates that with the proposed approach, the discrepancy between  $\mathcal{Z}_S$  and  $\mathcal{Z}_T$  decreased and  $h$  is able to perform classification up to an extent. There are classes where the adapted model yielded an accuracy above 50%, like street traffic, shopping mall, bus, and park. For the same classes, the non-adapted model could only

Table 1: Mean accuracy of the adapted and non adapted models on the source  $\mathcal{D}_S$  and target  $\mathcal{D}_T$  domains.

	Non adapted		Adapted	
	$\mathcal{D}_S$	$\mathcal{D}_T$	$\mathcal{D}_S$	$\mathcal{D}_T$
Previous SOTA [5]	0.65	0.20	0.65	0.32
Proposed approach	0.65	0.21	0.64	<b>0.45</b>

classify the street traffic class, while for the others it yielded an accuracy below 40%.

## 6 Conclusions and future work

In this work we presented a first approach for acoustic scene classification unsupervised domain adaptation that is based on the Wasserstein distance, along with the underlying theoretical framework. The presented method is evaluated on the TUT Acoustic Scenes dataset and the obtained results surpassed the previous state-of-the-art mean accuracy on the target domain by 13%. Future work on ASC DA will include the evaluation of the method on a larger dataset, along with using better ASC models. Besides, we will also apply these methods to alternative problems such as speech enhancement, where DA may help accounting for very adverse recording conditions.

## 7 Acknowledgement

Part of the computations leading to these results were performed on a TITAN-X GPU donated by NVIDIA to K. Drossos. The authors wish to acknowledge CSC-IT Center for Science, Finland, for computational resources. The research leading to these results has received funding from the European Research Council under the European Union’s H2020 Framework Programme through ERC Grant Agreement 637422 EVERYSOUND. P. Magron is supported by the Academy of Finland, project no. 290190.

## References

- [1] Y. Han and J. Park, “Convolutional neural networks with binaural representations and back-

ground subtraction for acoustic scene classification,” DCASE2017 Challenge, Tech. Rep., Sep 2017.

- [2] Z. Weiping, Y. Jiantao, X. Xiaotao, L. Xiangtao, and P. Shaohu, “Acoustic scene classification using deep convolutional neural network and multiple spectrograms fusion,” DCASE2017 Challenge, Tech. Rep., Sep 2017.
- [3] Y. Sakashita and M. Aono, “Acoustic scene classification by ensemble of spectrograms based on adaptive temporal divisions,” DCASE2018 Challenge, Tech. Rep., Sep 2018.
- [4] 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., Sep 2018.
- [5] S. Gharib, K. Drossos, E. Çakir, D. Serdyuk, and T. Virtanen, “Unsupervised adversarial domain adaptation for acoustic scene classification,” in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2018 Workshop (DCASE2018)*, Nov 2018, pp. 138–142.
- [6] C. Cortes, M. Mohri, M. Riley, and A. Rostamizadeh, “Sample selection bias correction theory,” in *Algorithmic Learning Theory*, Y. Freund, L. Györfi, G. Turán, and T. Zeugmann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 38–53.
- [7] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, “Domain-adversarial training of neural networks,” *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2096–2030, 2016.
- [8] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *IEEE Computer Vision and Pattern Recognition (CVPR)*, Jun 2011, pp. 1521–1528.
- [9] A. Gretton, A. Smola, J. Huang, M. Schmittfull, K. Borgwardt, and B. Schölkopf, *Covariate*

- shift and local learning by distribution matching*. Cambridge, MA, USA: MIT Press, 2009, pp. 131–160.
- [10] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, “Adversarial discriminative domain adaptation,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jul 2017, pp. 2962–2971.
  - [11] T. Tommasi, N. Patricia, B. Caputo, and T. Tuytelaars, “A deeper look at dataset bias,” in *German Conference on Pattern Recognition*, ser. Lecture Notes in Computer Science, vol. 9358. Springer International Publishing, Oct 2015, pp. 504–516.
  - [12] 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, pp. 151–175, May 2010.
  - [13] H. Zhao, S. Zhang, G. Wu, J. M. F. Moura, J. P. Costeira, and G. J. Gordon, “Adversarial multiple source domain adaptation,” in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 8559–8570.
  - [14] Z. Pei, Z. Cao, M. Long, and J. Wang, “Multi-adversarial domain adaptation,” in *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 2018.
  - [15] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell, “CyCADA: Cycle-consistent adversarial domain adaptation,” in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, J. Dy and A. Krause, Eds., vol. 80. PMLR, Jul 2018, pp. 1989–1998.
  - [16] A. Chadha and Y. Andreopoulos, “Improving adversarial discriminative domain adaptation,” *CoRR*, vol. abs/1809.03625, 2018. [Online]. Available: <http://arxiv.org/abs/1809.03625>
  - [17] I. Redko, A. Habrard, and M. Sebban, “Theoretical analysis of domain adaptation with optimal transport,” in *Machine Learning and Knowledge Discovery in Databases*, M. Ceci, J. Hollmén, L. Todorovski, C. Vens, and S. Džeroski, Eds. Cham: Springer International Publishing, 2017, pp. 737–753.
  - [18] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, D. Precup and Y. W. Teh, Eds., vol. 70. International Convention Centre, Sydney, Australia: PMLR, Aug 2017, pp. 214–223.
  - [19] S. Gharib, K. Drossos, E. Çakir, D. Serdyuk, and T. Virtanen, “Adversarial unsupervised domain adaptation for acoustic scene classification,” online: <https://doi.org/10.5281/zenodo.1401995>, Aug 2018. [Online]. Available: <https://doi.org/10.5281/zenodo.1401995>
  - [20] J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Wortman, “Learning bounds for domain adaptation,” in *Advances in Neural Information Processing Systems 20*, J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, Eds. Curran Associates, Inc., 2008, pp. 129–136.
  - [21] V. N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed., ser. Statistics for Engineering and Information Science. Springer, 1999.
  - [22] Y. Ganin and V. Lempitsky, “Unsupervised domain adaptation by backpropagation,” in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, Nov 2015.
  - [23] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems 27*, Z. Ghahramani, M. Welling, C. Cortes,



- N. D. Lawrence, and K. Q. Weinberger, Eds.  
Curran Associates, Inc., 2014, pp. 2672–2680.
- [24] D. Edwards, “On the kantorovichrubinstein theorem,” *Expositiones Mathematicae*, vol. 29, no. 4, pp. 387–398, 2011.
- [25] S. Ruder, “An overview of gradient descent optimization algorithms,” *CoRR*, vol. abs/1609.04747, 2016. [Online]. Available: <http://arxiv.org/abs/1609.04747>
- [26] A. Paszke, S. Gross, and A. Lerer, “Automatic differentiation in PyTorch,” in *NIPS-W*, 2017.