M-ADDA: Unsupervised Domain Adaptation with Deep Metric Learning

Issam H. Laradji and Reza Babanezhad

Department of Computer Science, University of British Columbia
Vancouver, British Columbia, Canada
{issamou,rezababa}@cs.ubc.ca

Abstract. Unsupervised domain adaptation techniques have been successful for a wide range of problems where supervised labels are limited. The task is to classify an unlabeled 'target' dataset by leveraging a labeled 'source' dataset that comes from a slightly similar distribution. We propose metric-based adversarial discriminative domain adaptation (M-ADDA) which performs two main steps. First, it uses a metric learning approach to train the source model on the source dataset by optimizing the triplet loss function. This results in clusters where embeddings of the same label are close to each other and those with different labels are far from one another. Next, it uses the adversarial approach (as that used in ADDA [34]) to make the extracted features from the source and target datasets indistinguishable. Simultaneously, we optimize a novel loss function that encourages the target dataset's embeddings to form clusters. While ADDA and M-ADDA use similar architectures, we show that M-ADDA performs significantly better on the digits adaptation datasets of MNIST and USPS. This suggests that using metric-learning for domain adaptation can lead to large improvements in classification accuracy for the domain adaptation task. The code is available at https://github.com/IssamLaradji/M-ADDA.

1 Introduction

Convolutional neural networks (CNN) [19] allow us to extract powerful features that can be used for tasks such as image classification and segmentation. However, these features are usually domain specific in that they are not discriminative enough for datasets coming from other domains, resulting in poor classification performance. Consequently, unsupervised domain adaptation techniques have emerged [9,35,21,38] to address the domain shift phenomenon between a source dataset and a target dataset. Common techniques use adversarial learning in order to make extracted features from the source and target datasets indistinguishable. The extracted features from the target dataset are then passed through a trained classifier (pre-trained on the source dataset) to predict the labels of the target test-set [34].

Recently, metric-based methods were introduced to address the problem of unsupervised domain adaptation [14,24]. Namely, classifying an example is performed by computing its similarity to prototype representations of each category [24]. Further, a category-agnostic clustering network was proposed by [14] to cluster new datasets through transfer learning. In this paper, we introduce M-ADDA, a metric-based adver-

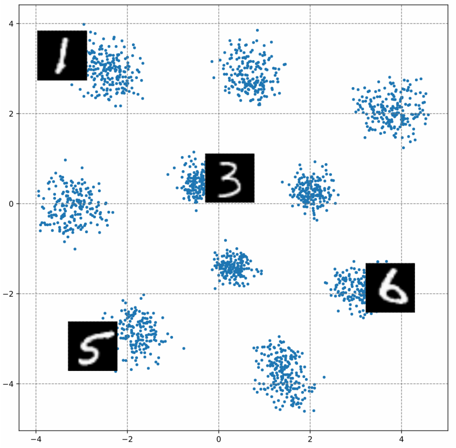

Fig. 1: **Metric Learning.** The result of minimizing the triplet loss on the MNIST dataset. Each cluster corresponds to examples belonging to a single digit label.

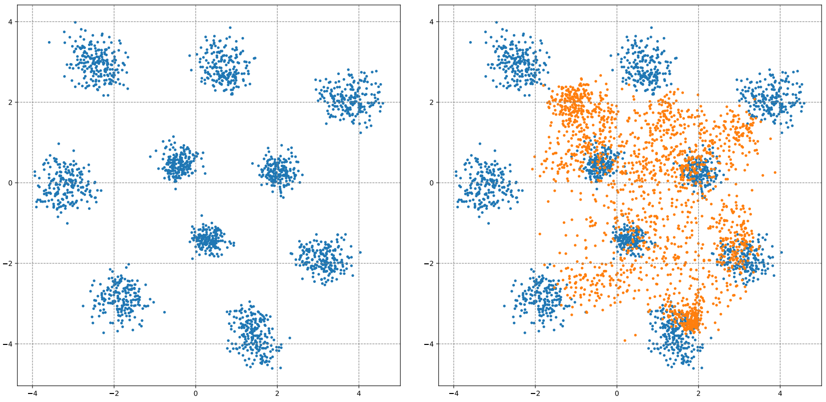

Fig. 2: **Domain Adaptation.** The **blue** dots represent the MNIST embeddings after optimizing Eq. (1). The **orange** dots represent the USPS embeddings. The center image shows the USPS embeddings before minimizing the domain shift adverbially by Eq. (3). The right-most image shows the USPS embeddings after optimizing Eq. (2).

sarial discriminative domain adaptation framework. First, M-ADDA trains our source model using metric learning by optimizing the triplet loss [13] on the source dataset. As a result, if K is the number of classes then the dataset is clustered into K clusters where each cluster is composed of examples having the same label (see Fig. 1). The goal is to obtain an embedding of the target dataset where the k-nearest neighbors (kNN) of each example belong to the same class and where examples from different classes are separated by a large margin. A major strength in this approach is its non-parametric

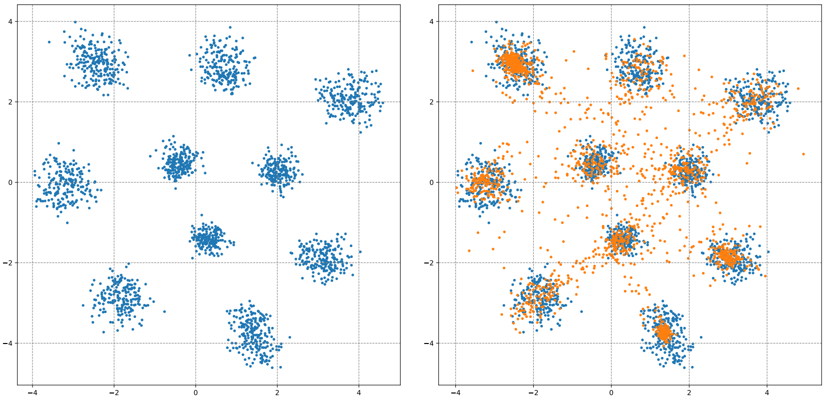

Fig. 3: **Domain Adaptation.** The **blue** dots represent the MNIST embeddings after optimizing Eq. (1). The **orange** dots represent the USPS embeddings. The center image shows the USPS embeddings before minimizing the domain shift adverbially by Eq. (3). The right-most image shows the USPS embeddings after optimizing Eq. (2).

nature [39] as it does not implicitly make parametric (possibly limiting) assumptions about the input distributions.

Next we adapt the distributions between the source and target extracted features using the adversarial learning method used by ADDA [34]. This addresses the domain discrepancy between the datasets. Early methods for domain adaptation are based on minimizing correlation distances and minimizing the maximum mean discrepancy to ensure both datasets have a common feature space [37,22,31,32]. However, adversarial learning approaches showed state-of-the-art performance for domain adaptation. While the features' distributions become more similar during training, we also train a network that maps the extracted features to embeddings such that they are clustered into K clusters. Concurrently, we encourage the clusters to have large margins between them. Therefore, the network is trained by minimizing the distance between each target example embedding and its closest cluster center corresponding to the source embedding. This approach is simple to implement and achieves competitive results on digit datasets such as MNIST [18], and USPS [17].

To summarize our contributions, (1) we propose a novel metric-learning framework that uses the triplet loss to cluster the source dataset for the task of domain adaptation; (2) we propose a new loss function that regularizes the embeddings of the target dataset to encourage them to form clusters; and (3) we show a large improvement over ADDA [34] on a standard unsupervised domain adaptation benchmark. Note that ADDA uses a similar architecture but a different loss function than M-ADDA.

In section 2, we review the related works and other similar approaches. In section 3, we introduce our framework and the new loss terms for domain adaptation. In section 4, we present experimental results illustrating the efficacy of our approach on the digits dataset. Finally, we conclude the paper in section 5.

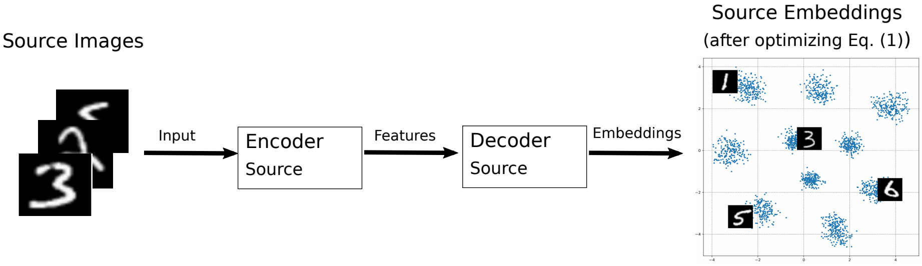

Fig. 4: **Training the source model.** We pre-train the source encoder and decoder by optimizing the triplet loss in Eq. (1). The source encoder extracts the features from the source dataset and the decoder maps the features to the embedding space where clusters are formed.

2 Related Work

Metric learning has shown great success in many visual classification tasks [39,30,13]. The goal is to learn a distance metric such that examples belonging to the same label are close as possible in some embedding space and samples from different labels are as far from one another as possible. It can be used for unsupervised learning such as clustering [40] and supervised learning such as k-nearest neighbor algorithms [12,39]. Recently, triplet networks [13] and Siamese networks [1] were proposed as powerful models for metric learning which have been successfully applied for few-shot learning and learning with few data. However, to the best of our knowledge we are the first to apply metric learning that is based on triplet networks for domain adaptation.

A close topic to domain adaptation is *transfer learning* which has received tremendous attention recently. It allows us to solve tasks where labels are scarce by learning from relevant tasks for which labels are abundant [4,28,5] by identifying a common structure between multiple tasks [6]. A common transfer learning strategy is to use pre-trained networks such as those trained on imagenet [15] and fine-tune them on new tasks. While this approach can significantly improve performance for many visual tasks, it performs poorly when the pre-trained network is used on a dataset which comes from a different distribution than the one it trained on. This is because the model has learned features that are specific to one domain that might not be meaningful for other domains.

To address this challenge, a large set of domain adaptation methods were proposed over the years [34,9,35,21] whose goal is to determine a common latent space between two domains often referred to as a source dataset and a target dataset. The general setting is to use a model that trains to extract features from the source dataset, and then encourage features extracted from the target dataset to be similar to the source features [10,3,8,36,26]. Auto-encoder based methods [10,2] train one or a variety of auto-encoders for the source and target datasets. Then, a classifier is trained based on the latent representation of the source dataset. The same classifier is then used to label the target dataset. Adversarial networks [11] based approaches use a generator model to transform the examples' feature representations from one domain to another [2,29,25].

Another group of domain adaptation methods [36,37,20,32] minimize the difference between the distributions of the features extracted from the source and target data. They achieve this by minimizing point estimates of a given metric between the source and target distributions by using maximum or mean discrepancy metrics. Current state-of-the-art techniques use the adversarial learning approach to encourage the feature representations from the two datasets to be indistinguishable (i.e. have a common distribution) [34]. Close to our method are the recent similarity based approaches proposed by [14,24], which transfer class-agnostic prior to new datasets, and classify examples by computing their similarity to prototype representation of each category, respectively. Our approach uses a regularized metric learning method with the help of k-nearest neighbors as a non-parametric framework. This can be more powerful than ADDA which uses a model that makes parametric assumptions (introducing limitations) about the input distribution [39].

Another class of domain adaptation methods are self-ensembling methods which augment the source dataset by applying various label preserving transformations on the images [16,33,7,27]. Using the augmented dataset they train several deep network models and use an ensemble of those networks for the domain adaptation task. Laine et. al. [16] have two networks in their model: the Π-model and temporal model. In the Pi-model, every unlabelled sample feeds to a classifier twice with different dropout, noise and image translation parameters. Their temporal model records the average of the historical network prediction per sample and forces the subsequent predictions to be close to the average. Travainen et.al [33] improve the temporal network by recording the average of the network weights rather than class prediction. This results in two networks: the student and the teacher network. The student network is trained via gradient descent and the weights of the teacher are the historical exponential moving average of the weights of the student network. The unsupervised loss is the mean square difference between the prediction of the student and the teacher under different dropout, noise and image translation parameters. French et. al. [7] combine the previous two methods with adding extra modifications and engineering and gets state of the art results in many domain adaptation tasks for image datasets. However, this method uses heavy engineering with many label preserving transformations to augment the data. In contrast, we show that our method significantly improves results over ADDA by making simple changes to their framework.

3 Proposed Approach: M-ADDA

We propose M-ADDA which performs two main steps:

1. train a source model on the source dataset using metric learning (as in Figure 4) using the Triplet loss function; then
2. simultaneously, adapt the distributions between the extracted source and target dataset features and regularize the predicted target dataset embeddings to form clusters (see Figure 5).

Our M-ADDA framework consists of a source model and a target model. The two models have the same architecture, and they both have an encoder that extracts features from the input dataset and a decoder to map the extracted features to embeddings. Consider a source dataset (X_S, Y_S), and a target dataset (X_T, Y_T) where the data X_S and X_T are drawn from two different distributions.

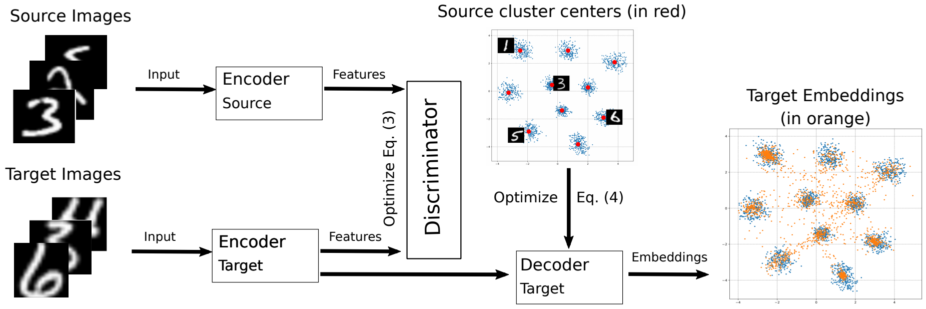

Fig. 5: **Training the target model.** We adversarially adapt the encoded features' distributions between the source and target encoder using Eq. (3) while using the source cluster centers to optimize Eq. (4). The label of each target embedding is the mode of the labels of the nearest source embedding neighbors.

Training the source model. The source model $f_{\theta_S}(\cdot)$, parameterised by θ_S, is first trained on the source dataset by optimizing the following triplet loss:

$$\mathcal{L}(\theta_S) = \sum_{(a_i,p_i,n_i)} \max(||f_{\theta_S}(a_i) - f_{\theta_S}(p_i)||^2 - \\ ||f_{\theta_S}(a_i) - f_{\theta_S}(n_i)||^2 + m, 0) \tag{1}$$

where a_i is an anchor example (picked randomly), p_i is an example with the same label as the anchor and n_i is an example with a different label from the anchor. Optimizing Eq. (1) encourages the embedding of a_i to be closer to p_i than to n_i by at least margin m. If the anchor example is close enough to the positive example p_i, and far from the negative example n_i by a margin of at least m, the max function returns zero; therefore, the corresponding triplet (a_i, p_i, n_i) does not contribute to the loss function. If the margin is smaller than m, then the max function returns $||f_{\theta_S}(a_i) - f_{\theta_S}(p_i)||^2 - ||f_{\theta_S}(a_i) - f_{\theta_S}(n_i)||^2 + m$. Minimizing this term results in moving a_i towards p_i and moving it away from n_i in the embedding feature space. After optimizing the loss term long enough, the samples with the same label are pulled together and those with different labels are pushed away from each other. As a result, points of the same label form a single cluster which allows us to efficiently classify examples using k-nearest neighbors (see Figure 4).

Algorithm 1 shows the procedure of training the source model on the source dataset for one epoch. Given a batch (X_B, Y_B), for each unique element y_i in Y_B, we obtain an anchor a_i whose label is y_i, a positive example p_i whose label is y_i, and a negative example n_i whose label is not y_i. Note that set(Y_B) returns the unique elements of Y_B. In our experiments, we obtained the negative example uniformly at random. However, other methods are possible such as greedily picking the triplet with the largest loss (as computed by Eq. (1)), and non-uniformly picking triplets based on their individual loss values. Finally, for each triplet, we compute the loss and update the parameters of the source model to minimize Eq. (1).

Training the target model. Next, we define C as the set of centers corresponding to the source embedding clusters (represented as red dots in Figure 5). Each center in C corresponds to a single label in the source dataset. A center is computed by taking the mean of the source embeddings belonging to that center's label. Then, we train the target model, parametrized by θ_T by optimizing the following two loss terms:

$$\mathcal{L}(\theta_T, \theta_D) = \underbrace{\mathcal{L}_A(\theta_{T_E}, \theta_D)}_{\text{Adapt}} + \underbrace{\mathcal{L}_C(\theta_T)}_{\text{C-Magnet}} \tag{2}$$

where θ_{T_E} correspond to the parameters of the target model's encoder; and θ_D is the parameter set for a discriminator model we use to adapt the distributions of the extracted features between the source (S) and target (T) datasets. We achieve this by optimizing:

$$\begin{aligned}\mathcal{L}_A(\theta_{T_E}, \theta_D) = \min_{\theta_D} \max_{\theta_{T_E}} - \sum_{i \in S} \log D_{\theta_D}(E_{\theta_S}(X_{S_i})) - \\ \sum_{i \in T} \log\left(1 - D_{\theta_D}(E_{\theta_{T_E}}(X_{T_i}))\right),\end{aligned} \tag{3}$$

where θ_{S_E} is the source model encoder's set of parameters; and $D(\cdot)$ is the discriminator model which is trained to maximize the probability that the features extracted by the source model's encoder come from the source dataset and that the features extracted by the target model's encoder come from the target dataset. In other words, the discriminator $D(.)$ tries to distinguish between the features extracted from the source dataset and the features from the target dataset by giving higher value (close to one) to a source dataset feature vector and a lower value (close to zero) to a target dataset feature vector. Simultaneously, the encoder of the target model is trained to confuse the discriminator into predicting the target features as coming from the source dataset. This adversarial learning approach encourages the features extracted by $E_{\theta_{S_E}}(X_{S_i})$ and $E_{\theta_{T_E}}(X_{T_i})$ to be indistinguishable in their distributions. For the sake of brevity, note that we show the loss functions in terms of a single source example X_{S_i} and target example X_{T_i}.

In parallel, we minimize the center magnet loss term defined as,

$$\mathcal{L}_C(\theta_T) = \sum_{i \in T} \min_j ||f_{\theta_T}(x_i) - C_j||^2, \tag{4}$$

which pulls the embeddings of example X_i to the closest cluster center defined in C (see Figure 5). The cluster center for a class is obtained by taking the Euclidean mean of all samples belonging to that class. Since we have 10 classes in MNIST and USPS, $|C| = 10$. This regularization term allows the target dataset embeddings to form clusters that are similar to the clusters formed by the source dataset embeddings. This is useful when minimizing $\mathcal{L}(\theta_T, \theta_D)$ fails to make the target embedding clustered in a similar way as the source embeddings. For example, in Fig. 2(b) we see that the target embeddings become scattered around the center when minimizing $\mathcal{L}_A(\theta_T, \theta_D)$ only. However, by simultenously minimizing $\mathcal{L}_C(\theta_T)$ we get a better formation of clusters as seen in Fig. 3(b).

Algorithm 2 shows the procedure for training the target model on the target dataset. Lines 4-5 use Eq. (3) to make the target features and the source features indistinguishi-

Algorithm 1 Training the source model on the source dataset (single epoch).

```
inputs
    Source model $f_{\theta_S}(\cdot)$, and source images and labels $(X_S, Y_S)$.
for $\{X_B, Y_B\} \in (X_S, Y_S)$ do
    for $y_i \in$ set( $Y_B)$ do
        $AP \leftarrow$ All image pairs whose label is $y_i$.
        for each $\{a_i, p_i\} \in AP$ do
            $n_i \leftarrow$ A random sample in $X_B$ whose label is not $y_i$.
            $L \leftarrow$ The loss in Eq (1) using $\{a_i, p_i, n_i\}$ and $f_{\theta_S}(\cdot)$.
            Update the parameters $\theta_S$ by backpropagating through $L$.
        end for
    end for
end for
```

Algorithm 2 Training the target model on the target dataset (single epoch).

```
inputs
    Target model $f_{\theta_T}(\cdot)$, and source and target images and labels $(X_S, Y_S, X_T, Y_T)$.
for $\{X_{S_B}, Y_{S_B}, X_{T_B}, Y_{T_B}\} \in (X_S, Y_S, X_T, Y_T)$ do
    Maximize Eq. (3) w.r.t. $\theta_D$ using $\{X_{S_B}, Y_{S_B}, X_{T_B}, Y_{T_B}\}$
    Minimize Eq. (3) w.r.t. $\theta_T$ using $\{X_{S_B}, Y_{S_B}, X_{T_B}, Y_{T_B}\}$
end for
$E_S \leftarrow$ The embeddings of the source dataset extracted by$f_{\theta_S}(\cdot)$
$C \leftarrow$ The cluster centers of $E_S$ are obtained by taking the Euclidean mean for each class.
for $\{X_{T_B}, Y_{T_B}\} \in (X_T, Y_T)$ do
    $L \leftarrow$ The loss computed using Eq. 4 and cluster centers $C$
    Update parameters $\theta_T$ by backpropagating through $L$.
end for
```

Algorithm 3 Predicting the labels of the test images.

```
inputs
    Target model $f_{\theta_T}(\cdot)$, Source model $f_{\theta_T}(\cdot)$, and source and target images and labels.
$E_S \leftarrow$ The embeddings of the source dataset extracted by$f_{\theta_S}(\cdot)$
for $\{X_{T_B}, Y_{T_B}\} \in (X_T, Y_T)$ do
    $E_{T_B} \leftarrow$ The embeddings of $X_{T_B}$ extracted by$f_{\theta_T}(\cdot)$
    $P_{T_B} \leftarrow$ The mode label of the k-nearest $E_S$ samples.
end for
```

Table 1: **Digits Adaptation**. We evaluate our method on the unsupervised domain adaptation task on the digits datasets, using the setup in [34].

Method	MNIST → USPS	USPS → MNIST
Source only (ADDA [34])	0.752	0.571
Source only (Ours)	0.601	0.679
Gradient reversal [9]	0.771	0.730
Domain confusion [35]	0.791	0.665
CoGAN [21]	0.912	0.891
ADDA [34]	0.894	0.901
M-ADDA (Ours)	**0.952**	**0.940**

Fig. 6: **Dataset.** Example images taken from the 2 digit domains we used in our benchmark.

ble. Lines 7-12 update the target model parameters by encouraging the target embeddings to move to the closest source cluster center. As shown in Algorithm 3, the prediction stage consists of two steps. First we extract the embeddings of the source dataset examples using the pre-trained source model. Then, the label of an example X_{T_i} is the mode label of the k-nearest source embeddings. This non-parametric approach allows us to implicitly learn powerful features that are used to compute the similarities between the examples.

4 Experiments

To illustrate the performance of our method for the unsupervised domain adaptation task, we apply it on the standard digits dataset benchmark using accuracy as the evaluation metric. We consider 2 domains: MNIST, and USPS. They consist of 10 classes representing the digits between 0 and 9 (we show some digit examples in Figure 6). We follow the experimental setup in [34] where 2000 images are sampled from MNIST and 1800 from USPS for training. Since our task is unsupervised domain adaptation, all the images in the target domain are unlabeled. In each experiment, we ran Algorithm 1 for 200 epochs to train our source model. Then, we report the accuracy on the target test set after running Algorithm 2 for 200 epochs.

We also use similar architectures for our models as those in [34]. The encoder module is the modified LeNet architecture provided in the Caffe source code [19]. The decoder is a simple linear model that transforms the encoded features into 256-unit embedding vectors. The discriminator consists of 3 fully connected layers: two layers with

Table 2: **Digits Adaptation**. We evaluate our method using the setup in [3,2].

Method	MNIST → USPS	USPS → MNIST
Source only (Ours)	0.60	0.68
DSN [3]	0.91	-
PixelDA [2]	0.96	-
SimNet [24]	0.96	0.96
M-ADDA (Ours)	**0.98**	**0.97**

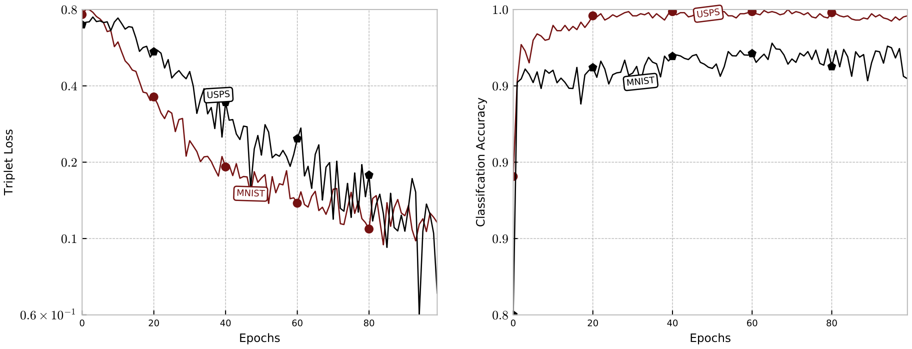

Fig. 7: **Optimizing the triplet loss.** (left) the Triplet loss value during the training of the source model on the USPS and MNIST datasets; (right) The classification accuracy obtained on the target datasets.

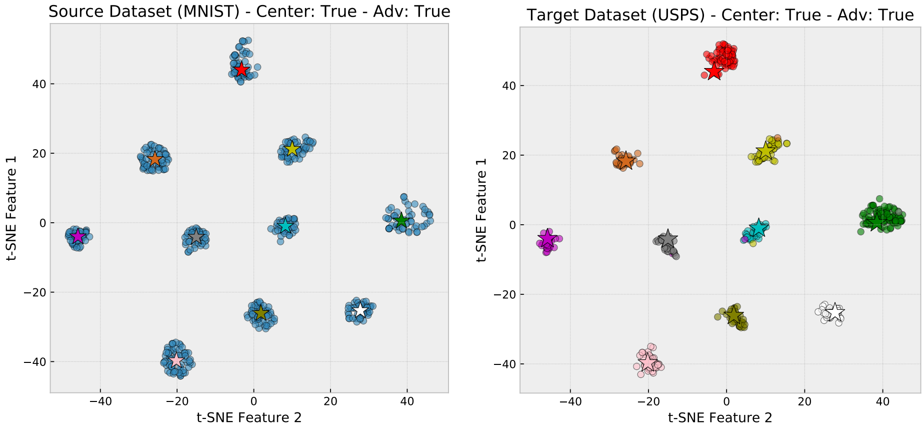

Fig. 8: **M-ADDA results.** (left) The t-SNE components of the source embeddings on the MNIST dataset after training the source model. (right) The t-SNE components of the target embeddings of the USPS dataset after training the target model. The stars represent the cluster centers of the source embeddings. The colors represent different labels.

Table 3: **Ablation studies**. Impact of the loss terms on the classification accuracy of the target model.

Method	MNIST → USPS	USPS → MNIST
Center Magnet Only	0.77	0.85
Adversarial Adaptation Only	0.93	0.92
M-ADDA	**0.98**	**0.97**

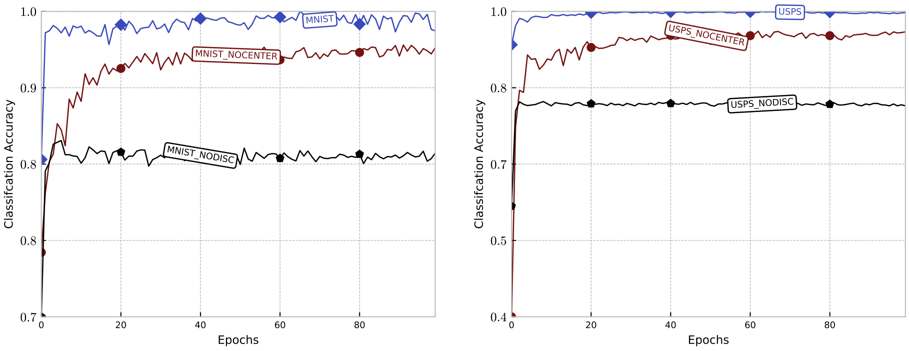

Fig. 9: **Ablation studies.** (left) The classification accuracy on MNIST using variations of the loss function (2); (right) The classification accuracy on USPS using variations of the loss function (2). NOCENTER refers to optimizing Eq. (3) only, and NODISC refers to optimizing Eq. (4) only. The blue lines refer to the result of optimizing Eq. (2).

500 hidden units followed by the final discriminator output. Each of the 500-unit layers uses a ReLU activation function.

Table 1 shows the results of our experiments on the digits datasets. We see that our method achieves competitive results compared to previous state-of-the-art methods, ADDA [34]. This suggests that metric learning allows us to achieve good results for domain adaptation. Further, Table 2 shows the results of our experiments using the setup in [3,2] where the full training set was used for both MNIST and USPS. We see that our method beats recent state-of-the-art methods in the USPS, MNIST domain adaptation challenge. However, it would be interesting to see the efficacy of M-ADDA in more complicated tasks such as the VisDA dataset challenge [23]. We show in Fig. 7 (left) the Triplet loss value during the training of the source model on the USPS and MNIST datasets. Further, Fig. 7 (right) shows the classification accuracy obtained on the target datasets with respect to the number of epochs. Higher accuracy was obtained for USPS when the model was trained on MNIST, which is expected since MNIST consists of more training examples.

In Table 3, we compare between two main variations for training the target model. Center Magnet only updates the target model using only Eq. (4); therefore, it ignores the adversarial training part of Eq. (3). Using Center Magnet only to train the target model results in poor performance. This is expected since the performance highly depends on the initial clustering. We see in Fig. 10 (right) that several source cluster centers

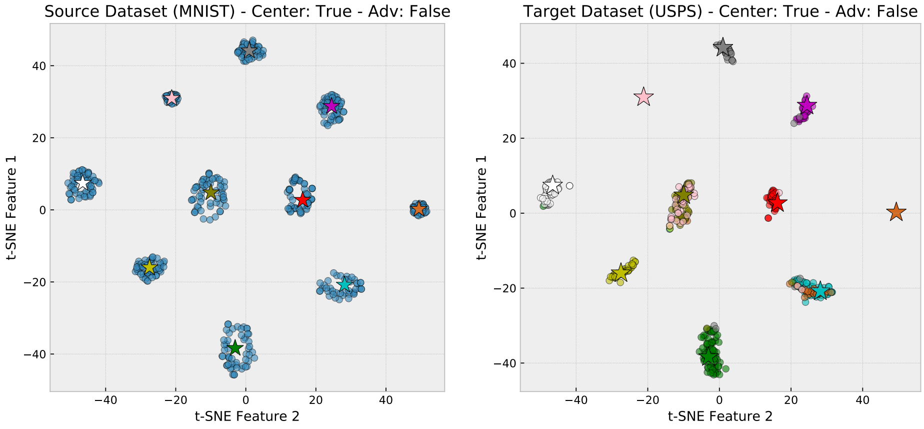

Fig. 10: **Center magnet optimization only.** The stars represent the cluster centers of the source embeddings.

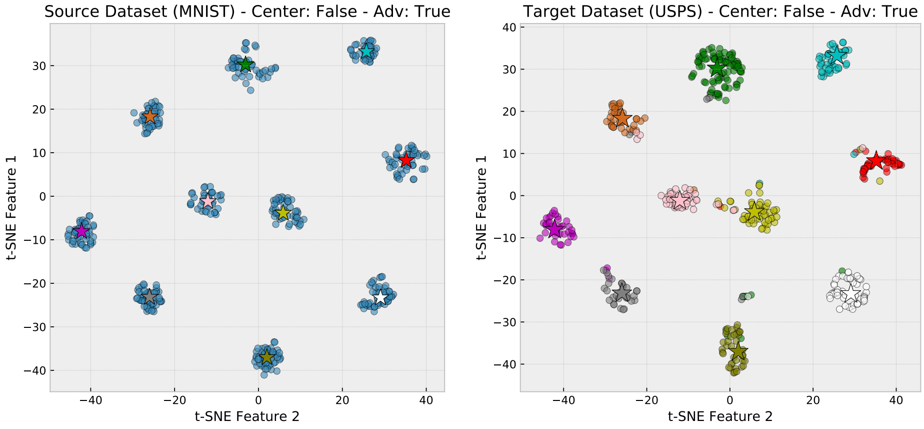

Fig. 11: **Adversarial optimization only.** The stars represent the cluster centers of the source embeddings.

(represented as stars) contain samples corresponding to different labels. For example, the samples with the pink label are clustered with those of the green label. Similarly, those with the orange label are clustered with those of the teal label. This is expected since the target model is encouraged to move the embeddings to the nearest cluster centers without having to match the extracted feature distributions between the source and target datasets.

Using only the adversarial adaptation loss improves the results significantly, since having the extracted features distribution between the source and target similar is crucial. However, we see in Fig. 11 (right) that some samples are far from any cluster center which makes their class labels ambiguous. Namely, the pink and yellow samples that are in the center between the yellow and pink cluster centers. To address these ambiguities, the center magnet loss helps the model to regularize against them. As a result, we see in Fig. 8 (right) that better clusters are formed when we optimize the whole loss function defined in Eq. 2. This suggests that M-ADDA has strong potential in addressing the task of unsupervised domain adaptation.

5 Conclusion

We propose M-ADDA, which is a metric-learning based method, to address the task of unsupervised domain adaptation. The framework consists of two main steps. First, a triplet loss is used to pre-train the source model on the source dataset. Then, we adversarialy train a target model to adapt the distributions of its extracted features to match those of the source model. In parallel, we optimize a center magnet loss to regularize the output embeddings of the target model so that they form clusters that have similar structure as that of the source model's output embeddings. We showed that this approach can perform significantly better than ADDA [34] on the digits adaptation dataset of MNIST and USPS. For future work, it would be interesting to apply these methods on more complicated datasets such as those in the VisDA challenge.

References

1. Bertinetto, L., Valmadre, J., Henriques, J.F., Vedaldi, A., Torr, P.H.: Fully-convolutional siamese networks for object tracking. ECCV (2016)
2. Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., Krishnan, D.: Unsupervised pixel-level domain adaptation with generative adversarial networks. CVPR (2017)
3. Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D.: Domain separation networks. NIPS (2016)
4. Cao, X., Wipf, D., Wen, F., Duan, G., Sun, J.: A practical transfer learning algorithm for face verification. ICCV (2013)
5. Deselaers, T., Alexe, B., Ferrari, V.: Weakly supervised localization and learning with generic knowledges. IJCV (2012)
6. Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. arXiv (2017)
7. French, G., Mackiewicz, M., Fisher, M.: Self-ensembling for visual domain adaptation. ICLR (2018)
8. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. arXiv (2014)
9. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. JMLR (2016)
10. Ghifary, M., Kleijn, W.B., Zhang, M., Balduzzi, D., Li, W.: Deep reconstruction-classification networks for unsupervised domain adaptation. ECCV (2016)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. NIPS (2014)
12. Han, E.H.S., Karypis, G., Kumar, V.: Text categorization using weight adjusted k-nearest neighbor classification. PAKDD (2001)
13. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. International Workshop on Similarity-Based Pattern Recognition (2015)

14. Hsu, Y.C., Lv, Z., Kira, Z.: Learning to cluster in order to transfer across domains and tasks. arXiv (2017)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. NIPS (2012)
16. Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. arXiv (2016)
17. Le Cun, Y., Jackel, L., Boser, B., Denker, J., Graf, H., Guyon, I., Henderson, D., Howard, R., Hubbard, W.: Handwritten digit recognition: Applications of neural network chips and automatic learning. IEEE Communications Magazine (1989)
18. LeCun, Y.: The mnist database of handwritten digits. http://yann. lecun. com/exdb/mnist/
19. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. IEEE (1998)
20. Li, Y., Wang, N., Shi, J., Liu, J., Hou, X.: Revisiting batch normalization for practical domain adaptation. arXiv (2016)
21. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. NIPS (2016)
22. Long, M., Cao, Y., Wang, J., Jordan, M.I.: Learning transferable features with deep adaptation networks. arXiv (2015)
23. Peng, X., Usman, B., Kaushik, N., Hoffman, J., Wang, D., Saenko, K.: Visda: The visual domain adaptation challenge. arXiv (2017)
24. Pinheiro, P.O.: Unsupervised domain adaptation with similarity learning. arXiv (2017)
25. Russo, P., Carlucci, F.M., Tommasi, T., Caputo, B.: From source to target and back: symmetric bi-directional adaptive gan. arXiv (2017)
26. Saito, K., Ushiku, Y., Harada, T.: Asymmetric tri-training for unsupervised domain adaptation. arXiv (2017)
27. Sajjadi, M., Javanmardi, M., Tasdizen, T.: Regularization with stochastic transformations and perturbations for deep semi-supervised learning. NIPS (2016)
28. Shi, Z., Siva, P., Xiang, T.: Transfer learning by ranking for weakly supervised object annotation. arXiv (2017)
29. Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R.: Learning from simulated and unsupervised images through adversarial training. CVPR (2017)
30. Song, H.O., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. CVPR (2016)
31. Sun, B., Feng, J., Saenko, K.: Return of frustratingly easy domain adaptation. AAAI (2016)
32. Sun, B., Saenko, K.: Deep coral: Correlation alignment for deep domain adaptation. ECCV (2016)
33. Tarvainen, A., Valpola, H.: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. NIPS (2017)
34. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial Discriminative Domain Adaptation. arXiv (2017)
35. Tzeng, E., Hoffman, J., Darrell, T., Saenko, K.: Simultaneous deep transfer across domains and tasks. ICCV (2015)
36. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. CVPR (2017)
37. Tzeng, E., Hoffman, J., Zhang, N., Saenko, K., Darrell, T.: Deep domain confusion: Maximizing for domain invariance. arXiv (2014)
38. Wang, M., Deng, W.: Deep visual domain adaptation: A survey. Neurocomputing (2018)
39. Weinberger, K.Q., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. JMLR (2009)
40. Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y.: Distance metric learning with application to clustering with side-information. NIPS (2003)