Report For CV Final Project: Domain Adaptation For Image Classification

Zi Wang
Shanghai Jiao Tong University
800 Dongchuan Rd, Shanghai, China
w4ngz1@sjtu.edu.cn

Abstract

Unsupervised domain adaptation aims to transfer the knowledge learnt from a labeled source domain to an unlabeled target domain. Previous works are mainly built upon ResNet or VGGnet. With the rapid development of large-scale pre-trained model, the capability of Vision Transformer is prominent in adapting cross-domain knowledge, thus we need to exploit this property. In the same time, previous works focus either on discrepancy measures or adversarial loss, no one pays attention to combining and balancing them in the same network. To bridge these two kinds of losses, I propose a feasible and reasonable solution to weight the proportion of each loss automatically. In addition, only seldom work considered the structure of network which can utilize different feature representations to align two variant domains. To remedy this, I also design a structure and use automatic weighting mechanism to adjust the losses. Also, I implement two effective and powerful methods on Vision Transformer, while these two methods were originally implemented on ResNet.

1. Introduction

With the development of big data, large amounts of images, voices are available to people. At the same time, industry and the research community put so much efforts on classification, segmentation tasks using deep learning, which requires labeled data in most cases. However, in the real world, we often have a serious problem that we don't have labeled data or the labeled data is not enough. In the beginning, a major assumption is that the training and test data have independent and identical distributions. But the assumption can be spoiled very easily due to the distortion and deformation in the background, quality or shape. Plus, there is no denying that annotating data manually is time-consuming and expensive. Therefore, it is of significance to leverage knowledge from an existing labeled domain to a similar but different domain without labels, which is called transfer learning. Due to the phenomenon of data bias or domain shift, deep learning methods don't always work, i.e., don't generalize well while transferring. The traditional approaches usually have bad performance if there exists a domain shift. Hence, more and more people dedicate to finding some methods to tackle this issue.

With deep learning, unsupervised domain adaptation mainly concentrate on the loss function and then can roughly be divided into two categories: Discrepancy based methods and Adversarial based methods.

Discrepancy based methods are one of the most popular deep network models, which aims to minimize the differences between the two domains and align data distributions. Variant distance losses are usually used after the last layer's activation function. Different distance losses represent different spaces where the hidden vectors are projected on. For instance, DAN [13] uses MMD(Maximum Mean Discrepancy) to embed the hidden vectors into reproducing kernel Hilbert space and then measure the squared distance. Deep CORAL [15] aims to align the second order statistics (covariances) between the different domain distributions. There are also some other distance measures, such as KL-divergence [17] , Jensen-Shannon Divergence [6], and so on.

Adversarial based methods have become an increasingly popular method in transfer learning recently because of the prominent performance of its discriminator. The domain discriminator aims to distinguish the source images and the target images. As the process of training going on, the feature representations in two different domains will be much similar than before. There are plenty of works to utilize GAN in transfer learning. The domain adversarial neural network (DANN) [7] is one of the first adversarial methods for adversarial based domain adaptation. Liu *et al*. [12] came up with a coupled generative adversatial networks consisted of several GANs. Cao *et al*. [3] proposed a partial transfer learning model.

Even though all the above methods have brilliant performance on transfer learning, no one considers how to balance discrepancy losses and adversarial losses efficiently and reasonably. To be specific, despite some people have combined

them, they just add them together instead of considering the weight of each loss. In fact, the proportion of losses matters while most researchers set the proportion as hyper-parameters. There is no denying that the weights cannot be adjusted adaptively in many cases. But in this situation, discrepancy losses and adversarial losses can be automatically adjusted according to their relative magnitudes. The idea is inspired by [10], which uses uncertainty to weigh losses. In addition, to utilize multiple discrepancy losses of different layers, I also apply the automatic weighting mechanism to balance discrepancy losses. The backbone of my framework is Vision Transformer(ViT) [5]. To validate the performance that above proposed method improves, I also implement the key parts of SHOT [11] and BNM [4] and combine them with ViT. This is for the fairness because both methods are bonded with ResNet originally, and the gap between the pre-trained models are quite large.

In a nutshell, the contributions of this project are as follows:

· Combining and balancing the discrepancy loss and adversarial loss using automatic weighting mechanism. Even though I only tested on MMD loss [13], but the theory shows that it can be extended to other distance measures easily.

· Utilizing several MMD loss [13] in different layers, i.e., different representational spaces to achieve better performance. Still, the proportion of each loss can be adjust adaptively.

· Above two methods both outperform SOTA. And I also incorporate SHOT [11] and BNM [4] into ViT [5] to validate whether they are still efficient on a new backbone and to verify the effectiveness of the proposed methods.

2. Related Works

2.1. Unsupervised Domain Adaptation

Transfer learning aims to learn transferable knowledge that are general across different domains with different distributions. Yosinski *et al*. [23] provided evidence that feature representations in deep neural networks are transferable. In the past few years, various methods have been proposed to address unsupervised domain adaptation problem, where no labels are available for the target domain. For example, DDC [16] tried to learn domain-invariant features by minimizing Maximum Mean Discrepancy (MMD) [2] of two features obtained from source and target, respectively. Long *et al*. [13] improved DDC by embedding two hidden vectors in a reproducing Hilbert space and using a multi-kernel MMD measure the discrepancy of two domains. Another group of people tend to minimize the difference between two domains via adversarial learning [8]. By introducing a discriminator and modeling the domain adaptation as a minimax problem [7], the network is able to make the representations of two domains as closer as possible. The original network is trained to deceive the discriminator which attempt to distinguish the features of source domain from that of target domain. Pseudo-labeling is another technique to address UDA problem. It also achieves substantial performance on this task. Pseudo-labeling typically generates pseudo-labels for the target domain based on the predicted class probability. Saito *et al*. [14] proposed an asymmetric tri-training method to generate pseudo labels for target samples using two networks, and the third can learn from them to obtain target discriminative representations. Liang *et al*. [11] proposed SHOT to get precise pseudo labels using a clustering method which is similar to weighted k-means. It is noteworthy that all of these methods completely used CNNs as the backbone. By contrast, only a few groups of people explores Vision Transformer [5] to tackle the UDA problem. However, because the potential and capability are so powerful, there is great value in studying how to fully exploit the unique structure of Transformer in this task.

2.2. Transformer for Vision

Transformers [18] was firstly proposed in NLP field and demonstrate outstanding performance on various tasks. Much of such impressive achievement is attributed to the power of capturing long-range dependencies through attention mechanism. Recently, some studies attempted to integrate attention into CNNs to augment feature maps [1,9,21]. A pioneering work of completely convolution-free architecture is Vision Transformer(ViT), which applied transformers on a sequence of image patches. It takes the advantage of large-scale pre-training data. ViT and its variants have proved their capability in object detection [20], segmentation [24], *etc*. Yang *et al*. [22] proved the transferability of ViT and proposed a novel UDA framework tailored for ViT by exploring its intrinsic merits.

To summarize, only a few methods focus on the transferability of Vision Transformer, hence it is of significance to fully exploit the potential of ViT. In this paper, I integrate both discrepancy based method and adversarial based method into ViT, and figure out an automatic weighting mechanism to balance two types of losses.

3. Methodology

In this section, I will firstly introduce the definition of unsupervised domain adaptation. Secondly, I will present the backbone——Vision Transformer [5]. Then, I will illustrate how to combine multiple MMD loss [13], and how to balance MMD loss and adversarial loss, respectively. At last, I am going to explain the key parts of SHOT and BNM [4] and how I incorporate them into ViT.

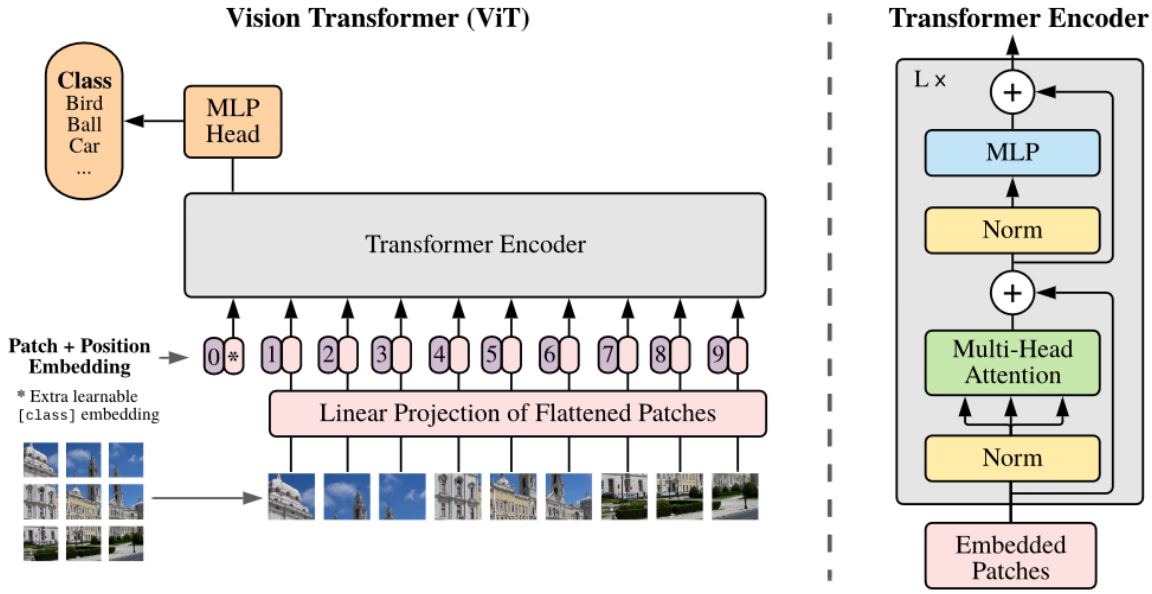


Figure 1. Framework of Vision Transformer

3.1. Definition

A domain $\mathcal{D}$ consists of a feature spaces $\mathcal{X}$ by considering the marginal probability $P(\mathcal{X})$, and the task is defined by the label space $\mathcal{Y}$. The conditional distribution is $P(\mathcal{Y}|\mathcal{X})$, and the joint distribution is denoted as $P(\mathcal{X}, \mathcal{Y})$. When it comes to unsupervised domain adaptation in classification, there is a source domain $\mathcal{D}_{\mathcal{S}} = \{\mathcal{X}_{\mathcal{S}}^i, \mathcal{Y}_{\mathcal{S}}^i\}_{i=1}^{\mathcal{N}_{\mathcal{S}}}$ of $\mathcal{N}_{\mathcal{S}}$ samples in C categories and a target domain $\mathcal{D}_{\mathcal{T}} = \{\mathcal{X}_{\mathcal{T}}^j\}_{j=1}^{\mathcal{N}_{\mathcal{T}}}$ of $\mathcal{N}_{\mathcal{T}}$ samples without any labels ($\mathcal{Y}_{\mathcal{T}}$ is unknown, but also in C categories). The goal of unsupervised s to learn features that are both discriminative and invariant to the domain discrepancy, and in turn guarantee accurate prediction on the unlabeled target data.

3.2. Transferable Vision Transformer

The framework of Vision Transformer is shown in Fig. 1. They reshape the iamge $\mathbf{x} \in \mathbb{R}^{H\times W\times C}$ into a sequence of flattened 2D patches $\mathbf{x} \in \mathbb{R}^{N\times(P^2\cdot C)}$, where (H, W) is the resolution of the image, and P represents the patch size. Then they flatten the patch pixels and embed them into D dimensions with a trainable linear projection, which is the input of Transformer. Besides, position embeddings are added to the patch embeddings to retain positional information. In order to perform classification task, they use the stantard approach of adding an extra learnable "classification token" to the sequence. Then all the input sequences pass through a standard Transformer Encoder, which consists of alternating layers of multiheaded self-attention and MLP blocks. Layernorm is applied before every block, and residual connections after every block. The process of Encoder is as follows:

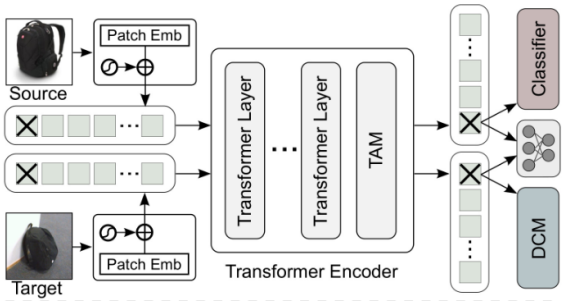


Figure 2. Framework of TVT.

$$
\begin{aligned}
z_0 &= [x_{class}; x_p^1 E; x_p^2 E; \ldots; x_p^N E] + E_{pos} \\
z_l^{'} &= MSA(LN(z_{l-1})) + z_{l-1} \\
z_l &= MLP(LN(z_l^{'})) + z_l^{'} \\
y &= LN(z_L^0)
\end{aligned}
$$

After getting the hidden vector which corresponding to the "classification token", they input it into a MLP head to get the classification result.

Following this idea, Yang *et al*. proposed TVT (Transferable Vision Transformer) [22], the framework is shown in Fig.2. The main difference is using a adversarial loss to decrease the discrepancy between source images and target images. There are some other tricks to let it perform better, but because they are incremental works and are not my

emphasis of work, I won't elaborate on these tricks. I implement the method I proposed based on the open-source code of this article.

3.3. Balancing MMD Loss and Adversarial Loss

In this subsection, I will introduce MMD loss specifically, and then I will show how I combine two types of losses together.

3.3.1 MMD Loss

MMD(Maximum Mean Discrepancy) loss is widely used in transfer learning. It is mainly used to measure the distance of two different but relative distributions. The distance is defined as below:

$$\begin{aligned} MMD(X,Y) &= \|\frac{1}{n}\sum_{i=1}^{n}\phi(x_i) - \frac{1}{m}\sum_{j=1}^{m}\phi(y_j)\|_H^2 \\ &= \|\frac{1}{n^2}\sum_{i}^{n}\sum_{i'}^{n}\phi(x_i)\phi(x_{i'}) \\ &- \frac{2}{nm}\sum_{i}^{n}\sum_{j}^{m}\phi(x_i)\phi(y_j) \\ &+ \frac{1}{m^2}\sum_{j}^{m}\sum_{j'}^{m}\phi(y_j)\phi(y_{j'})\| \end{aligned}$$

Subscript H represents the distance is measured in the Reproducing Hilbert Kernel Space. Representing it using kernel function, which is similar to the kernel in SVM:

$$\begin{aligned} MMD(X,Y) &= \|\frac{1}{n^2}\sum_{i}^{n}\sum_{i'}^{n}k(x_i,x_{i'}) \\ &- \frac{2}{nm}\sum_{i}^{n}\sum_{j}^{m}k(x_i,y_j) \\ &+ \frac{1}{m^2}\sum_{j}^{m}\sum_{j'}^{m}k(y_j,y_{j'})\| \end{aligned}$$

In most cases, previous works use gaussian kernel function $k(u,v) = e^{-\frac{\|u-v\|^2}{\sigma}}$ to project the original representations into Hilbert space.

3.3.2 Automatic weighting mechanism

Inspired by [10], I borrow their idea that using uncertainty to balance the different proportion of the losses. It's worth noting that MMD loss and other distance measures can be seen as a regression problem. This is intuitively because, in most cases, minimizing the distance between two vectors is equivalent to the regression task targeting at zero. In addition, discriminator is a two-category classification task in essence. Now the balancing problem are converted to how to combine a regression task and a classification task. Thanks to the theory in [10], the regression task and classification task are balanced as follows:

To begin with, let me introduce what is task uncertainty. Task uncertainty captures the relative confidence between tasks, reflecting the uncertainty inherent to the regression or classification task. It depends on the task's representation or unit of measure.

For the regression task: Let $f^W(x)$ be the output of a neural network with parameters W on input x. Then we can define the likelihood as a Gaussian with mean given by the model output.

$$p(y|f^W(x)) = \mathcal{N}(f^W(x),\sigma^2)$$

with a noise scalar σ. The σ also demonstrates the weight of loss, and I will show it later. For the classification task: we usually utilize Softmax function to get the probability vector.

$$p(y|f^W(x)) = Softmax(\frac{1}{\sigma^2}f^W(x))$$

Here, we use a scaled version of Softmax.

For multiple model outputs, due to the tasks are independent, the likelihood can be written as:

$$p(y_1,...,y_K|f^W(x)) = p(y_1|f^W(x))...p(y_K|f^W(x))$$

In maximum likelihood inference, we want to maximize the log likelihood of the model. For regression task, we have

$$\log p(y|f^W(x)) \propto -\frac{1}{2\sigma^2}\|y - f^W(x)\|^2 - \log\sigma$$

For classification task, we have

$$\log p(y=c|f^W(x),\sigma) = \frac{1}{\sigma^2}f_c^W(x) - \log\sum_{c'}\exp(\frac{1}{\sigma^2}f_{c'}^W(x))$$

Subscript c represents the c'th element of the vector $f^W(x)$.

Then, the joint loss $\mathcal{L}(W,\sigma_1,\sigma_2)$ is given as:

$$\begin{aligned} &= -\log p(y_1,y_2=c|f^W(x)) \\ &= -log\mathcal{N}(y_1;f^W(x),\sigma_1^2)Softmax(y_2=c|f^W(x),\sigma_2) \\ &= \frac{1}{2\sigma_1^2}\|y_1 - f^W(x)\|^2 + \log\sigma_1 - \log p(y_2=c|f^W(x),\sigma_2) \\ &= \frac{1}{2\sigma_1^2}\|y_1 - f^W(x)\|^2 - \frac{1}{\sigma_2^2}\log Softmax(y_2,f^W(x)) \\ &+ \log\sigma_1 + \log\frac{\sum_{c'}\exp(\frac{1}{\sigma_2^2}f_{c'}^W(x))}{\sum_{c'}\exp(f_{c'}^W(x))^{\frac{1}{\sigma_2^2}}} \\ &\approx \frac{1}{2\sigma_1^2}\mathcal{L}_1(W) + \frac{1}{\sigma_2^2}\mathcal{L}_2(W) + \log\sigma_1\sigma_2 \end{aligned}$$

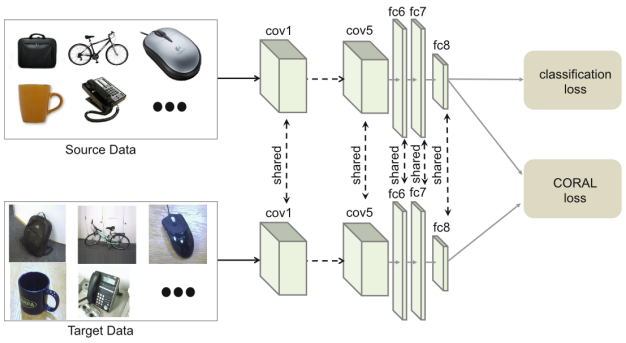


Figure 3. Framework of discrepancy based method.

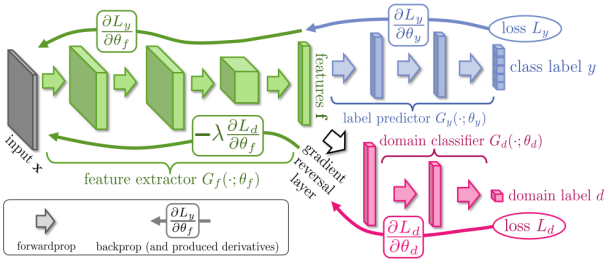


Figure 4. Framework of adversarial based method.

where $\mathcal{L}_1(W) = \|y_1 - f^W(x)\|^2$ and $\mathcal{L}_2(W) = -\log Softmax(y_2, f^W(x)) = -\log p_{x,Y(x)}$. Noting that MMD loss can be written as $MMD(X,Y) = \|\frac{1}{n}\sum_{i=1}^{n}\phi(x_i) - \frac{1}{m}\sum_{j=1}^{m}\phi(y_j)\|_H^2$, hence it follows the same paradigm with square distance. Consequently, MMD can be converted into regression in this problem.

$p_{x,Y(x)}$ represents the probability of the ground truth label of current data x.

Noting that the last step in $\mathcal{L}(W,\sigma_1,\sigma_2)$ is because the assumtion

$$\frac{1}{\sigma_2}\sum_{c'}\exp(\frac{1}{\sigma_2^2}f_{c'}^W(x)) \approx \sum_{c'}\exp(f_{c'}^W(x))^{\frac{1}{\sigma_2^2}}$$

The $\mathcal{L}_1(W)$ represents a distance between two vectors, which is consistent with MMD loss. Because the discriminator's last layer is Sigmoid function, which aims to convert the logits into probability in two-category classification task, the loss function is almost the same as $\mathcal{L}_2(W)$ after binary cross-entropy. Hence, we can use this way to balance the proportion of MMD loss and the adversarial loss.

As for how to design a structure which can adopt both discrepancy measure and adversarial loss, please pay attention to the location of two losses in the framework of discrepancy based method (Fig. 3) and the counterpart of adversarial based method (Fig. 4). Both methods operate on the feature out of the last layer. Hence we don't need extra structure to utilize both losses in one network.

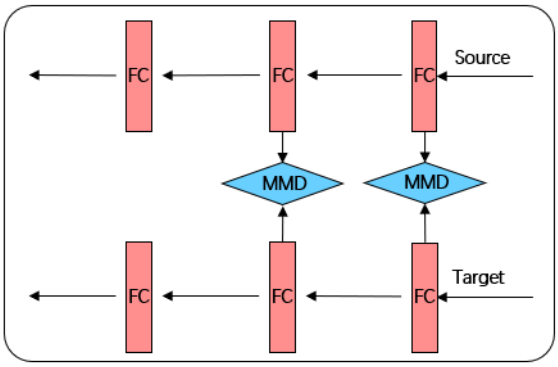


Figure 5. Little change need to make on the MLP Head.

3.3.3 Multi-layer MMD Loss

Some little changes need to be done on the MLP Head to get multiple MMD loss, as shown in Fig. 5. I construct several fully connected layers to calculate MMD loss in different layers. In this task, different layers means different representations and different transferabilities. Hence it is of significance to adapt multiple layer instead of only one layer. In other words, adapting a single layer sometimes cannot undo the data bias and domain shift between the source and the target images. Now that the network has variant representations, why don't we use all of them to align the features? Obviously, the weights of the layers can be significantly different. In the same time, tuning the hyperparameters of the weights can be a time-consuming job and sometimes cannot achieve the best performance. Therefore, I utilize the automatic weighting mechanism in multi-layer MMD loss. This time, two objective functions are both regression task.

It's pretty similar to the former method. The joint loss $\mathcal{L}(W,\sigma_1,\sigma_2)$ is given as:

$$\begin{aligned}
&= -\log p(y_1, y_2 | f^W(x)) \\
&\approx \frac{1}{2\sigma_1^2}\|y_1 - f^W(X)\|^2 + \frac{1}{2\sigma_2^2}\|y_2 - f^W(X)\|^2 + \log\sigma_1\sigma_2 \\
&= \frac{1}{2\sigma_1^2}\mathcal{L}_1(W) + \frac{1}{2\sigma_2^2}\mathcal{L}_2(W) + \log\sigma_1\sigma_2
\end{aligned}$$

Still, $\mathcal{L}(W) = \|y - f^W(X)\|^2$ can represent MMD loss due to the similar form.

3.4. SHOT

SHOT [11] was recognized as the best UDA model before Transferable Vision Transformer. No one has tested its effectiveness on Vision Transformer, hence it is worth trying. Now I will introduce the main method of SHOT. It aims to cluster the representations to generate pseudo-labels for target images. The detailed procedure is provided in the following:

1) Attaining representation centroids for each class through weighted k-means clustering.

$$c_k^{(0)} = \frac{\sum_{x \in X_t} \delta_k(\hat{f}_t(x)) \hat{g}_t(x)}{\sum_{x \in X_t} \delta_k(\hat{f}_t(x))}$$

where $\delta_k()$ represents the softmax layer, $\hat{f}(x)$ represents the result of classifier layer and $\hat{g}(x)$ represents the hidden vector. The perceptual intuition of this method is using the probability out of Softmax to weight each feature and then calculate the centroids of each class.

2) Then we need to obtain pseudo labels accoding to the nearest centroid:

$$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)})$$

It means that we need to assign a label based on the feature representation centroids to each sample image.

3) Consequently, we can easily compute the target centroids:

$$c^{(1)} = \frac{\sum_{x \in X_t} \mathbb{I}(\hat{y}_t = k) \hat{g}_t(x)}{\sum_{x \in X_t} \mathbb{I}(\hat{y}_t = k)}$$

$$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(1)})$$

These two formulas are the same as the formulas in step 1) and 2). Repeating this loop until convergence provide accurate pseudo labels. According to the original paper, we only need to do this loop once to get pretty precise labels. In my experiments, it do generate the same pseudo labels even if I do the loop for many times. So the conclusion is SHOT can get pseudo labels efficiently.

The next step is using these pseudo labels to supervise the target images. Noting that this process doesn't need the source images at all, hence we need to change the training procedure a little. In the first step, we train a model on the source images until convergence. In the second step, we discard the source images and use target images to generate pseudo labels. Finally, we can use these pseudo labels to finetune our model.

3.5. BNM

BNM [4] (Batch Nuclear-norm Maximization) aims to constrain both prediction discriminability and diversity of the unlabeled data simultaneously instead of combining two objective functions together. Discriminability means how discriminative a prediction vector can be. The vector achieves the highest discriminability when it is an one-hot vector. Diversity means how diverse the predicted labels in a batch. Usually, diversity can be measured via the rank of classification matrix. To comprehend this, supposing that all the images in a batch are classified into one class, then the rank of the classification matrix is 1. On the contrary, if the images are classfied into many classes, the rank can be higher.

Figure 6. Some examples of the dataset.

Frobenius-norm(F-norm) can measure the discriminability.

$$\|A\|_F = \sqrt{\sum_{i=1}^{B} \sum_{j=1}^{C} |A_{i,j}|^2}$$

The Shannon Entropy is as follows:

$$H(A) = -\frac{1}{B} \sum_{i=1}^{B} \sum_{j=1}^{C} A_{i,j} \log(A_{i,j})$$

This paper proved that H(A) and $\|A\|_F$ have strict opposite monotonicity and the prediction discriminability could also be enhanced by maximizing$\|A\|_F$.

Intuitively, the rank of the prediction matrix represents the diversity of prediction. However, optimizing the rank directly is NP-hard. Thus, they resort to optimizing the Nuclear-norm $\|A\|_*$, which is the convex approximation of the rank of a matrix.

Furthermore, the relationship of the range between $\|A\|_*$ and $\|A\|_F$ can be expressed as follows:

$$\frac{1}{\sqrt{D}} \|A\|_* \leq \|A\|_F \leq \|A\|_* \leq \sqrt{D} \|A\|_F$$

where $D = min(B, C)$

Thus, $\|A\|_*$ and $\|A\|_F$ can bound each other.

$$\|A\|_* <\leq \sqrt{D} \|A\|_F \leq \sqrt{DB}$$

Maximizing $\|A\|_*$ is equivalent to maximizing $\|A\|_F$. And $\|A\|_*$ can be calculated as the sum of the singular values.

4. Experiments

4.1. Office-home Dataset

Office-home [19] contains 15,588 images in 65 categories across four domain: Art, Clip Art, Product and Real World. To be specific, Art denotes artistic depictions for objects; Clip Art is picture collections of clipart; Product are the object images with clear background, and Real World describes object images collected with a regular camera. Here are some examples from this dataset in Fig. 6. Interestingly, the precision of classification seems to have an

Method	Art→RealWorld	Clip Art→RealWorld	Product→RealWorld	Avg
TVT	89.47	88.27	90.13	89.29
1 MMD	89.53	87.99	90.81	89.44
2 MMD	89.58	86.57	90.82	88.99
AW(2 MMD)	89.88	88.41	90.75	89.68(↑ 0.39)
adversarial+MMD	89.46	88.08	89.94	89.16
AW(adversarial+MMD)	90.30	88.98	90.59	89.96(↑ 0.67)
4 MMD	86.85	86.21	88.11	87.09
TVT+SHOT	90.08	88.54	90.44	89.69(↑ 0.40)
TVT+BNM	90.22	88.93	90.31	89.82(↑ 0.53)

Table 1. Results on Office-home dataset.

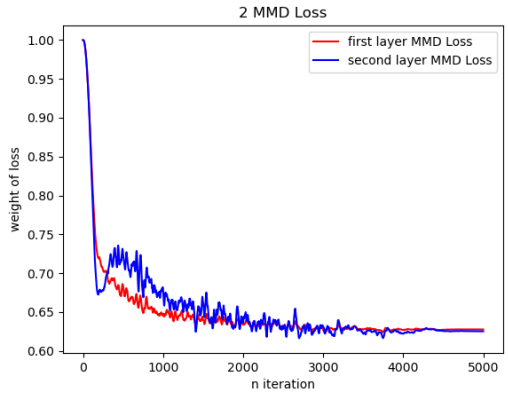


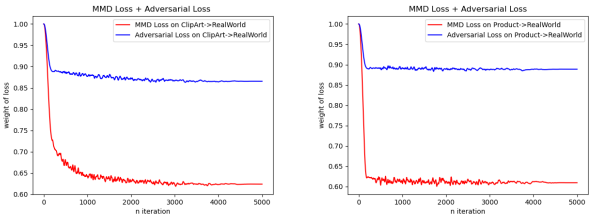
Figure 7. Comparison on the weights of two MMD losses.

upper bound, because if I use the target image-label pairs to train the network, I only get approximately 94% accuracy. Therefore, it is pretty challenging to improve the performance if ViT's accuracy has achieved 90% in average.

The experiments on Office-home are shown in the Table. 1. I will illustrate each line in detail in the following paragraphs.

4.2. Baseline

The first line is the results of Transferable Vision Transformer [22]. Because the random seed has been fixed in the open-source code, I reproduce the same result with the original paper. Then the second line is the result after replacing the adversarial loss in TVT with one layer MMD loss. Appearantly, there is no explicit difference between two types of losses. Noting that in the Clip Art→Real World case, adversarial loss outperforms MMD loss. While the product→Real World case is just the opposite! This observation gives me a lot of insight to design a mechanism of combining adversarial and MMD loss together, even though no one has tried this method before, to the best of my knowledge.



(a) Clip Art→Real World. (b) Product→Real World.

Figure 8. Weights of adversarial loss and MMD loss.

4.3. Combining Two MMD Loss

When it comes to the third line of the table, the performance decreases a little even though I deepen the architecture of neural network. The implicit reason will be explained later. However, if I automatically tune the coefficients of two losses using the method I proposed, it achieves a higher performance than SOTA! The weights of two MMD losses is shown in Fig.7. As the result you can see, the initial weight of the loss is two low for this task (Actually, the curve is plotted on the parameter σ, which is inversely proportional to the weight).

4.4. Combining Adversarial Loss and MMD Loss

Still, pretty amazing, automatic weighting mechanism works well when combining two types of losses. Despite adding them directly will lead to bad performance, automatic weighting is powerful to balance the two types of losses. To demonstrate the effectiveness of the insight mentioned in Section 4.2, I plot the weights of losses in different scenes in Fig. 8a and Fig. 8b. We can easily identify the patterns and the proportions are different.

4.5. Analysis on Side-effect of Too Much MMD Loss

But when it comes to the 7-th line, the performance drops pretty quickly even though the network is much deeper than the other cases. This is because forcing the features' distribution of two domains to be as close as possible will lead

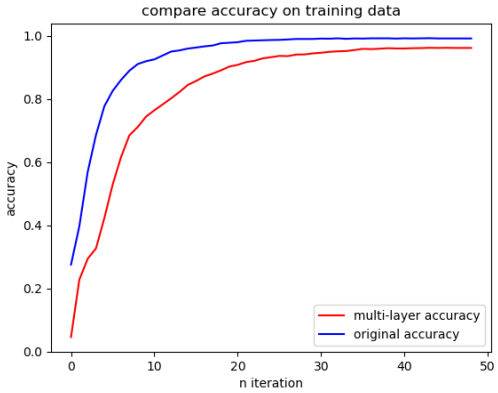


Figure 9. Compare accuracy on training set.

to the decrease in the classification task. Even though many previous works argued that the goal of transfer learning is to minimizing the discrepancy between two domains, they didn't experiment or even consider the side-effect of narrowing the gap between them——lower the accuracy on classification task. Hence, I conducted a simple yet convincing experiment on training set. As shown in Fig. 9, when applying multiple MMD losses on the network, even though the representations are much closer, the capability of classification task drops apparently.

4.6. Combining Other Methods

As mentioned before, I implement some other methods to verify the effectiveness of my method. This is because the paradigm of Vision Transformer is brand-new while other methods are all implemented on ResNet. Another reason is that the pre-trained models (ViT, ResNet) have totally variant number of parameters. ViT-base has 86 million parameters while ResNet only has 11 million parameters. The ViT backbone can outperform any ResNet-based method easily without any trick. Thus the comparison is unfair except for transplanting the tricks into ViT. When implementing SHOT [11], because it need relative accuracy classification result, the whole process of generating pseudo labels has to be done after training. So I change the policy of training as follows: 1) using source images to train a model. 2) discarding source images and saving the model. 3) generating pseudo labels for target images and training the target images based on these pseudo labels. Through this procedure, I can combine the key part of SHOT with ViT. As for BNM [4], I use SVD to get the classification matrix's singular values. Then the loss is calculated by the mean of its singular values.

Analysis. There are some parameters to adjust manually when implementing SHOT and BNM, but I think in this project, there is no need to find the best parameters. Thus I didn't spend too much time on adjusting parameters. However, we can see that automatic weighting mechanism outperforms these two methods, which were seen as SOTA previously. At least it proves that combining two kinds of losses or aligning two discrepancy losses is feasible and quite effective. At the same time, there are mathematical theories backup the method I proposed. Thus it's reasonable without loss of simplicity.

5. Conclusion

In this project, I propose a simple yet effective way to combine and balance different losses. Specifically speaking, the theory in this paper supports two cases: 1)a discrepancy loss plus an adversarial loss and 2)two discrepancy losses. Even though I only tested on a sample discrepancy loss, there is no denying that it can fit many other losses which have similar form. Last but not least, I demonstrate some convincing results of experiments and analysis detailedly. Even though there exists some wrong attempts, I still rigorously confirm that the proposed method can be extended to any losses which have similar forms.

As for my gains in this project, I think there are three points need to be emphasized: 1) the speed of paper reading has been built up. I find that in most cases, I only need to read the abstract, contributions, methodology and conclusion carefully, scan the introduction and experiments, ignore the related works. But there are some exceptions that I need to read related works carefully because I'm unfamiliar with some concepts. By doing this, I can save a lot of time to get the main ideas in a paper. 2) In the future, I need to focus on math basics. This is because I consume a lot of time when understanding or deriving the equations I write in this paper. If I'm proficient in these mathematical concepts, I can save a lot of time in understanding the meanings in the equations. 3) To set about a new topic or project like transfer learning, which I have never reached before, I'd better start with a review or survey of the whole area. It is such a pity that I read the papers for specific methods when I start to do this project. And the disadvantage of this behavior is that sometimes I don't know if previous works have completed the ideas that I thought would be work. For instance, I first thought aligning the features in convolutional layers would be work. But when I checked and searched this idea on Google Scholar, I found some works had done this before yet had bad performance. If I had done a survey before the whole project, I won't waste a lot of time thinking about how to design a structure for the feature maps out of convolutional layers, which has been proven not work at all. In a nutshell, I reaped a lot in this project, especially in paper reading and how to embark on a new project.

References

[1] Irwan Bello, Barret Zoph, Ashish Vaswani, Jonathon Shlens, and Quoc V Le. Attention augmented convolutional networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3286–3295, 2019. 2

[2] Karsten M Borgwardt, Arthur Gretton, Malte J Rasch, Hans-Peter Kriegel, Bernhard Schölkopf, and Alex J Smola. Integrating structured biological data by kernel maximum mean discrepancy. *Bioinformatics*, 22(14):e49–e57, 2006. 2

[3] Zhangjie Cao, Lijia Ma, Mingsheng Long, and Jianmin Wang. Partial adversarial domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 135–150, 2018. 1

[4] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Liang Li, Qingming Huang, and Qi Tian. Towards discriminability and diversity: Batch nuclear-norm maximization under label insufficient situations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3941–3950, 2020. 2, 6, 8

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2

[6] Bent Fuglede and Flemming Topsoe. Jensen-shannon divergence and hilbert space embedding. In *International Symposium onInformation Theory, 2004. ISIT 2004. Proceedings.*, page 31. IEEE, 2004. 1

[7] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. 1, 2

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 2

[9] Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *International conference on machine learning*, pages 1558–1567. PMLR, 2017. 2

[10] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7482–7491, 2018. 2, 4

[11] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039. PMLR, 2020. 2, 5, 8

[12] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. *Advances in neural information processing systems*, 29:469–477, 2016. 1

[13] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International conference on machine learning*, pages 97–105. PMLR, 2015. 1, 2

[14] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 2988–2997. PMLR, 2017. 2

[15] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *European conference on computer vision*, pages 443–450. Springer, 2016. 1

[16] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 2

[17] Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014. 1

[18] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 2

[19] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5018–5027, 2017. 6

[20] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. *arXiv preprint arXiv:2102.12122*, 2021. 2

[21] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7794–7803, 2018. 2

[22] Jinyu Yang, Jingjing Liu, Ning Xu, and Junzhou Huang. Tvt: Transferable vision transformer for unsupervised domain adaptation. *arXiv preprint arXiv:2108.05988*, 2021. 2, 3, 7

[23] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014. 2

[24] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6881–6890, 2021. 2