Domain Disentanglement through Natural Language

YuFei Ma
Politecnico di Torino
s300570@studenti.polito.it

Maria Akl
Politecnico di Torino
s293829@studenti.polito.it

Jiaqi Wu
Politecnico di Torino
s296187@studenti.polito.it

Abstract

Unsupervised domain adaptation is a subfield of machine learning that aims to address the challenge of domain shift. In order to overcome this problem, we train a model to learn domain-invariant features that are robust to the differences between the source and target domains, while still preserving the relevant information for the target task. With this knowledge, we utilized the Domain2Vec model as a reference to construct our domain disentanglement model. We achieved the domain adaptation objective through the cross-disentanglement method, which separates features into category-specific and domain-specific features. We evaluated our model on the PACS dataset using a ResNet-18 pre-trained model. Finally, we introduced the CLIP model to enhance the performance of our domain disentangler.

1. Introduction

In the field of computer vision, the inability to train a model on all possible domains and the limitations of a well-trained model in maintaining satisfactory performance have led to the emergence of domain adaptation as a prevalent and significant area of investigation. Domain adaptation involves adapting models trained on a source domain to a related but distinct target domain. This technique is commonly employed to enhance model performance on the target domain,in situations where labeled data is limited or unavailable.

The motivation behind domain adaptation is that the distribution of data in the source and target domains is often different, which can result in a model trained on the source domain to have poor performance on the target domain. Domain adaptation addresses this issue by finding a mapping between the source and target domains that aligns the distributions and reduces the mismatch between the two domains. So we want to achieve a model that can do a perfect classification task, regardless of the change of domain or in other terms change of data distribution.

There are two main approaches to domain adaptation: supervised and unsupervised domain adaptation. In supervised domain adaptation, the model is trained on labeled data from the source domain and fine-tuned on a small amount of labeled data from the target domain. In unsupervised domain adaptation, the model is trained on labeled data from the source domain and fine-tuned on unlabeled data from the target domain.

In this report, we consider the Domain2Vec model for unsupervised domain adaptation. The model employs a combination of adversarial loss and classification loss to achieve domain adaptation. The adversarial loss is used to reduce the discrepancy between the source and target domains, while the classification loss ensures that class label information is preserved.

Therefore, we focus on combining (i) discriminativeness and (ii) domain-invariant features by using a special cross-disentanglement operation. This operation involves (i) separating the features related to domain or category into separate classifiers and (ii) freezing each classifier separately to retrain the category encoders, so that it can extract domain-features that the domain classifier cannot distinguish. As a result, the object classifier can perform the classification task accurately.

We propose a domain-disentanglement model that separates images into domain-specific and category-specific features. Additionally, we incorporate the CLIP model to aid in model tuning. CLIP can also be used as a pre-trained model, fine-tuned for improved accuracy.[1]

2. Related work

2.1. Unsupervised domain adaptation

Unsupervised Domain Adaptation aims to modify a model that has been trained on one dataset (known as the source domain) to perform effectively on another related dataset (known as the target domain) without using any labeled data from the target domain [1, 2].

This issue is tackled by various methods such as adversary-based approaches that solve the adaptation prob-

[1] All the relevant source code can be found at https://github.com/MurphyAAA/Vision-Language-AML

lem from a single source to a single target [3]. Multi-source domain adaptation (MSDA) is another approach (used in multiple fields) that deals with the situation where the training data comes from multiple sources [4]. The Domain2Vec model is introduced in this report. This model uses adversarial learning to disentangle features into two parts by training two encoders simultaneously. The feature extractor is used to obtain the features and the well-trained domain classifier separates the domain-related features from the images. The encoders are then trained to only extract either category-related information or domain-related information, thereby improving the performance of the classification. [5]

2.2. Feature Disentanglement

Feature disentanglement is the decomposition of the representation learned by a machine learning model into distinct and interpretable features that correspond to various underlying factors of variation in the data. The goal of feature disentanglement is to learn representations that capture the key aspects of the data, leading to improved generalization, interpretability, and transferability of the learned features to other tasks.

A large body of work has focused on feature disentanglement, including the use of generative adversarial networks (GANs) [6] and variational autoencoders (VAEs) [7] to learn representations. More recently, Lee et al. [8] proposed disentangling the features into domain-invariant content space and domain-specific attribute space. Our Domain2Vec method adopts this approach and separates the deep features into two distinct components: domain-specific features and category-specific features [5].

2.3. Contrastive Language-Image Pretraining

The CLIP model has demonstrated remarkable results across a wide range of cross-modal (language-image) tasks. CLIP learns to encode the association between language and images by predicting the missing information in a given pair of text and image. In our scenario, we utilize CLIP to train the domain encoder, enabling it to extract more domain-related features.

3. Method

This paper will provide a brief overview of the architecture of each model implemented in this project. We first consider the baseline model, whose performance serves as the starting point. Next, the domain disentanglement model will be presented, along with its optimization function. The third section introduces the updated model incorporating CLIP as a secondary component to aid in the domain disentanglement task. Finally, the CLIP model is fine-tuned to perform the downstream classification task.

3.1. Baseline Model

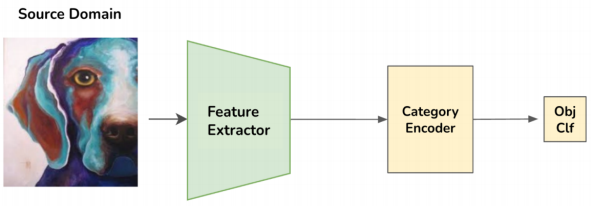


Figure 1. Baseline architecture

The baseline architecture is defined as follows: a feature extractor is used to obtain basic features f_G from images. The next component is the category encoder, which is composed of several fully connected layers, and is trained to capture category-related information. The final component is the classifier, which minimizes the cross-entropy loss calculated on the source domain.

3.2. Domain Disentanglement Model

In contrast to a typical single-domain classification task, when multiple domains are involved in a task, domain information can be viewed as noise that affects the performance. To mitigate the impact of domain shift, it is beneficial to extract domain-specific features, thereby making the model more robust to changes in the domain. To accomplish this, this model employs two encoders, each of which is responsible for extracting domain-specific and category-specific features, respectively. This helps improve the accuracy of the classification.

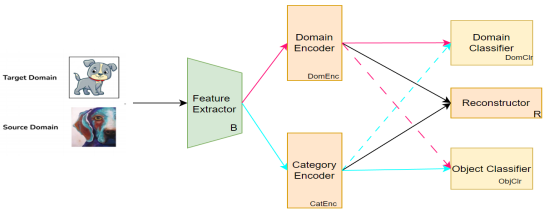


Figure 2. Domain Disentanglement architecture: A feature extractor is used to obtain features, and adversarial learning is employed to simultaneously train encoders, ensuring that the encoders only extract information related to either categories or domains.

This model introduces cross-disentanglement schema to separate the deep features into domain-specific and category-specific features. Our training architecture (Figure 2) is comprised of the following components: **(1)** A backbone *B* (ResNet-18), this is used to extract features which contains domain and category features; **(2)** A domain encoder *DomEnc* to disentangle the domain specific features from the latent representation, it is combined by several

linear layers and activation layers; **(3)** A category encoder *CatEnc* aims to disentangle the object features; **(4)** A category classifier *ObjClr*, which is used to implement category classification task, we use only one fully connected layer and output dimension is the number of category; **(5)** A domain classifier *DomClr*, which is used to implement domain classification task; **(6)** A reconstructor *R*, since deeper information could be missing in feature disentanglement process, this block recovers the original features with disentangled domain-specific features *fd* and category-specific features *fc*.

Feature extractor In our model, we use ResNet-18 as our backbone to extract the features which contains domain-specific and category-specific features. In order to achieve better performance, we tune the extractor as well when we train our model.

Category Disentanglement This procedure is realized by a two-step adversarial training. First of all, we train a category encoder *CatEnc* and category classifier *objClr* to predict correctly the class labels, this performance is managed by cross-entropy loss:

$$L_{ce}^{class} = -\sum_{i=1}^{N} t_{i,c} \log(p_i) \tag{1}$$

where N is the number of classes, t is a binary indicator (0 or 1) if class label c is the correct classification for observation i, and p is a predicted probability of observation i if it belongs to class c.

The aim of second step is to remove the domain-specific information from *fcs*. In order to achieve this, we freeze the parameters used in category classifier and retrain encoder to generate new *fcs* to fool the domain classifier *domClr*. This performance can be achieved by minimizing negative entropy :

$$L_{ent}^{class} = -\sum_{i=1}^{N} p_{i,c} \log(p_{i,c}) \tag{2}$$

where p is the probability when it belongs to class c.

This adversarial training process corresponds to the blue straight line and dashed line respectively in Figure 2.

Domain Disentanglement This is the opposite from the previous one, this domain disentanglement aims to extract domain features *fds* from the latent representations. We also used two-step adversarial training, however this time we first train the extractor *B* and domain encoder *domEnc* to obtain domain-specific features *fds*. This is supervised by cross-entropy loss:

$$L_{ce}^{domain} = -\sum_{i=1}^{M} t_{i,d} \log(p_i) \tag{3}$$

where M is the number of domains, t is the true label if the observation belongs to the correct domain.

Second, we focus on moving the category-specific information from *fds*. We freeze the parameters in category classifier *objClr* and train domain encoder to generate *fds* to fool category classifier *objClr*. Similarly, we can minimize the negative loss of the predicted class distribution:

$$L_{ent}^{class} = -\sum_{c=1}^{M} p_{i,d} \log(p_{i,d}) \tag{4}$$

This process corresponds to the red straight line and dotted line in Figure 2, respectively.

Reconstructor According to [9], deep information could be missing in the feature disentanglement process, especially in the situation where the encoders are composed of several fully connected and RELU layers. Hence, a feature reconstructor *R* is used to rebuild the original feature f_G which is combined with disentangled domain-specific and category-specific features. We use only one fully connected layer. As for the loss function, only MSE loss is used:

$$L_{res} = \sum_{i=1}^{N} ||\hat{f_G} - f_G||^2 \tag{5}$$

where $\hat{f_G}$ is the reconstructed feature.

For the optimizer, we adopt Adam [10]. Finally, the overall optimization object is:

$$L = w_1 L_{class} + w_2 L_{domain} + w_3 L_{rec} \tag{6}$$

Where w_1,w_2,w_3 are hyperparameters. Furthermore, we add α_1 and α_2 that denote the category disentanglement loss and domain disentanglement loss respectively. In the end, the whole optimization function becomes:

$$L = w_1(L_{ce}^{class} + \alpha_1 L_{ent}^{class}) + w_2(L_{ce}^{domain} + \alpha_2 L_{ent}^{domain}) + w_3 L_{rec} \tag{7}$$

3.3. CLIP using labeled PACS

The CLIP model addresses the challenge of improving classification when there is a shortage of images, the literal description of a given image can help model perform a better classification task. To meet the requirements of CLIP, images are first labeled with domain-related characteristics such as color saturation, edges, and background. These labeled data are fed into the model. Initially, the CLIP model is used without pre-training, meaning that the Text encoder of CLIP is frozen. The performance is evaluated based on the mean squared error (MSE) loss between the CLIP model's output and the output from the domain encoder f_{ds}. Hence, the total optimization function becomes:

$$L = w_1 L_{class} + w_2 L_{domain} + w_3 L_{rec} + w_4 L_{clip} \tag{8}$$

We can see that there is one more hyperparameter, w_4, that needs to be considered in this updated version of model.

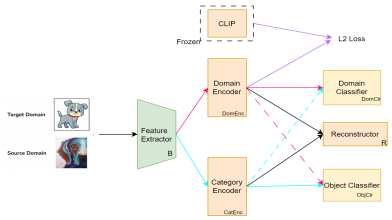


Figure 3. domain disentanglement loss function

3.4. Fine tune CLIP model

In this phase, the CLIP model parameters are not frozen, meaning that the CLIP model will be pre-trained but its parameters can be updated. The output from the pre-trained CLIP model will then be fed into the main model and both will be trained together.

4. Dataset

To evaluate the performance of our model, PACS (Photo-Artistic Cross-domain Simulation) benchmark is used. This data-set is used for evaluating the performance of cross-domain adaptation methods. It contains four domains: art paintings (2,048 images), cartoon (2344 images), photo (1,670 images), and sketch (3929 images). Each domain has 7 different categories of objects.

The PACS dataset is widely used due to its high degree of variability across domains, making it challenging for models to generalize from one domain to another. This variability makes the PACS dataset an effective test for unsupervised domain adaptation methods, as it demands that models learn domain-invariant features that can be generalized to new domains.

In our model, we only use art painting as our source domain (which has 2048 images), this data is divided into training and validation sets: 80% of images are used for training, and 20% are used for validation. In the baseline phase, training only uses source domain data. In domain disentanglement, we also feed the target domain's domain features into the model in order to train domain disentangle, category disentangle cannot be trained as we don't have the category's label.

With CLIP, the description of images already in the database was added, resulting in a total of 2540 images for training. The split criteria remained the same as in the domain disentanglement approach.

5. Experiments on PACS

5.1. Hyperparameter setting

To improve training speed, when selecting suitable hyperparameters, a maximum iteration of 2000 is used to train the data, reducing training time. The use of 2000 iterations is a trade-off between the computation capability of the GPU and the performance of the loss functions.

The weights in the optimization function are tested on a logarithmic scale of $(0.1, 1, 10)$ due to computational efficiency. After testing various combinations, a weight of 1 was found to be the most suitable. Additionally, since the focus was on the category classification task, the weight w_1 was modified to 2, while others remained at 1. For the parameter α, which is responsible for extracting domain information from images, a higher value was assigned to α_1 than α_2 due to its importance in the domain adaptation task. After several iterations, $\alpha_1 = 1.2$ and $\alpha_2 = 0.5$ were chosen. The results of other combinations can be seen in Table 1. Based on the final feedback, the combination with the highest average accuracy was selected.

5.2. Domain Disentanglement model

With the selected hyperparameters, we achieved an accuracy of 66.63% on the sketch target domain, 67.62% on the cartoon target domain, and 94.85% on the photo target domain. The best checkpoint was selected as the baseline, leading to improved accuracy performance. The loss function and accuracy of each module in the model were plotted, corresponding to the three tasks of domain disentanglement, category disentanglement, and reconstruction, to evaluate their performance individually. Since the results for the three target domains were found to be similar, the sketch target domain was chosen to be shown and explained, while the results for the cartoon and photo domains can be found in the supplementary material.

The adversarial network requires the encoder to generate data that the classifier cannot differentiate, which is represented by maximizing the entropy of the encoder. As a result, one of the loss functions decreases over time like in classical machine learning methods. Meanwhile, the other one shows the opposite trend, increasing during training. The graphs for the loss functions of category disentanglement and domain disentanglement are shown below. The slight decline in the class entropy loss at the start of the training indicates that the model has just begun the training phase and the loss is still decreasing. The opposite trends in the classification and encoder's loss function indicate that the encoder is effectively extracting domain-specific features from the images and deceiving the domain classifier during the classification task.

This trend can be observed to be consistent across the different categories. The domain classifier attempts to

Dropout rate	w_1	w_2	w_3	α_1	α_2	cartoon	sketch	Average
0.3	2	1	1	0.48	0.028	61.59		
0.3	2	1	1	1.2	0.5	67.62	66.63	67.12
0	2	1	1	1.2	0.5	49		
0.3	2	1	1	0.48	0.048	68.45	56.32	62.52

Table 1. These are the results of several combinations when the target domain is cartoon. Due to the limitations of the equipment, cartoon was considered as the starting point, and the best result was selected. These results were then applied to the sketch and photo domains.

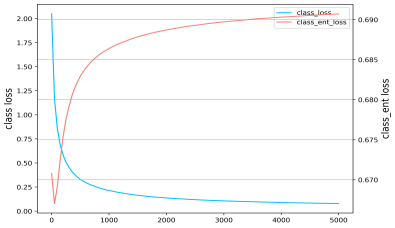


Figure 4. The loss function of class disentanglement.

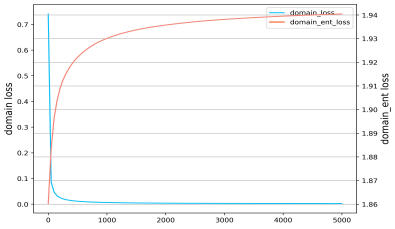


Figure 5. domain disentanglement loss function

differentiate between images while the encoder is trained to extract category-specific features, tricking the category classifier.

In domain adaptation, instead of evaluating overfitting or underfitting between the training and validation sets (which are in the same domain), we focus on whether the loss function converges to a stable situation. This helps us determine if the model has learned the necessary features for the target domain. The convergence of the loss function is a crucial aspect in evaluating the performance of the model.

We can see that after 3000 iterations, our model's loss functions are reaching a plateau, although the class one still shows a little trend to increase but its increase rate is not quite high if compared with the beginning. It means the model is entering the convergence. The convergence of the reconstruction loss suggests that the model has learned the relationship between the extracted features and the original image, allowing it to generate high-quality images from the extracted features.

Then, let us move to the reconstruction loss, the trend should be decrease during the time. We can see the loss value decrease continually, it means our model has good resilience in the situation when we recovery those features

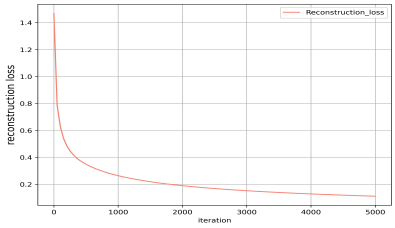


Figure 6. Reconstruction loss

into the original images.

Finally, we can see our total train loss to see our model's general performance during training It can be seen that the

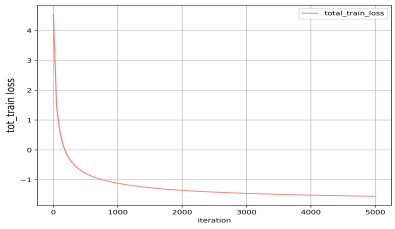


Figure 7. Total train loss

total train loss decreases continually, it means the model is learning and continues to optimize its performance.

During the validation phase, we plot the total validation lose and also its accuracy which can represent its performance It can be observed that the validation losses are de-

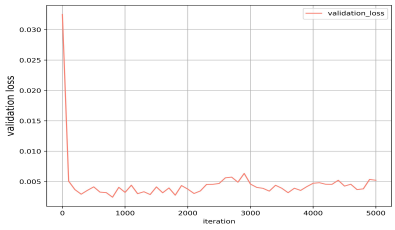


Figure 8. The validation loss

creasing. Though after 2000 iterations, the value rose a little, this could be the overfitting situation, but since we chose model parameters when its validation accuracy is highest (the accuracy graph will show below), we could avoid the

case that model picked parameters in a relatively worse situation.

Finally we focus on the accuracy which we care more about,

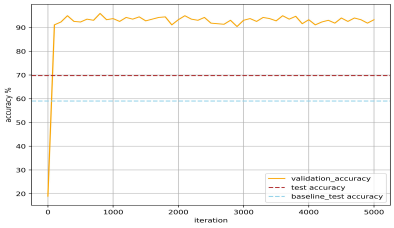


Figure 9. Validation and test accuracy of our model and the baseline's test accuracy

It can be seen that the training accuracy is always above 90%. It means our model still has better performance, and test accuracy is 66.63%. The value is lower than the validation accuracy but higher than the baseline accuracy (blue line), so it is within the acceptable range.

In summary, we began by examining various loss functions to assess our model's performance from different perspectives. Next, we evaluated the overall performance when we tested it on the target domain. Our model showed improved performance in the category classification task, resulting in higher accuracy, allowing it to effectively handle scenarios where the target domain differs from the source domain.

5.3. Domain Disentanglement model with CLIP

Because the labeled images alone are not sufficient to train our model, we decided to feed all images (with and without description) to achieve better training. Following is the result that we put **cartoon** as our target domain:

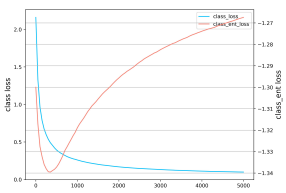


Figure 10. The category disentanglement loss

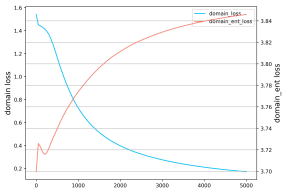
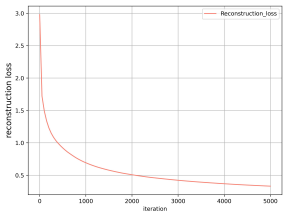


Figure 11. The domain disentanglement loss

It can be observed that the overall trend is similar with the plain domain disentanglement model. Because the CLIP help domain encoder to extract



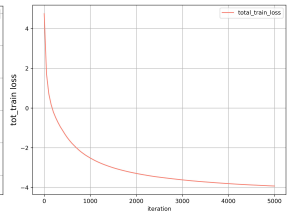
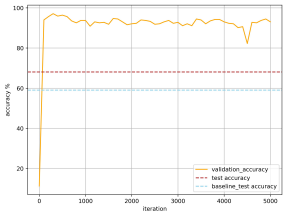


Figure 12. The reconstructionFigure 13. The total train loss loss

Reconstruction loss keeps decreased over time means our model is able to recovery the information even after deep learning.



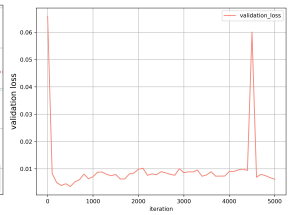


Figure 14. The validation ac-Figure 15. The validation loss curacy loss during training

From Figure 14, it can be seen that the test accuracy is still lower than the validation accuracy, this means the domain shift still affects the performance of model. However, the overall value is still higher than baseline (blue line) meaning our model is able to filter some noise like domain information to learn more useful information.

5.4. Domain Disentanglement model with fine tuned Clip

We set pre-trained iteration to 2000.

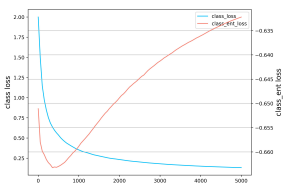


Figure 16. The category disentanglement loss

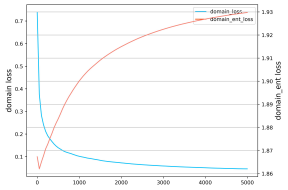


Figure 17. The domain disentanglement loss

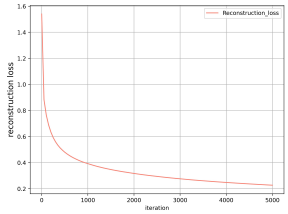


Figure 18. The reconstruction loss

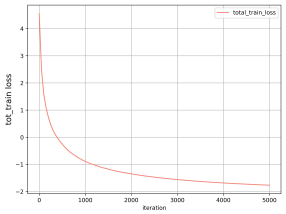


Figure 19. The total train loss

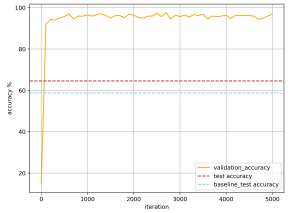


Figure 20. The validation accuracy loss during training

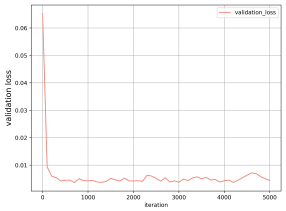


Figure 21. The validation loss

Comparing the three models Finally, we put all of our three models together to check the overall performance. The results are compiled in Table 2.

6. Conclusion and Limitation

In conclusion, this report presented a cross-disentanglement approach for separating features into domain-specific and category-specific ones, and incorporating pre-trained CLIP to improve disentanglement performance. Then, The CLIP model was fine-tuned and trained on images with their descriptions to identify domain information, resulting in improved the accuracy of the classifier.

The study used the PACS dataset, which consists of approximately 2500 images, and achieved satisfactory performance despite its small size. However, when the target domain is "sketch", the results obtained from fine-tuning the CLIP model were not optimal. Further research will be conducted to determine the reasons behind this and efforts will be made to improve the accuracy of the model for each domain, with the goal of minimizing the performance discrepancy across target domains.

A. Supplementary Materials

This is the performance of the rest of domains in different models.

A.1. Domain disentanglement model

A.1.1 Cartoon

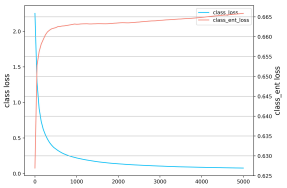


Figure 22. The category disentanglement loss

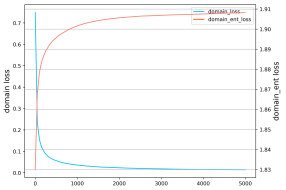


Figure 23. The domain disentanglement loss

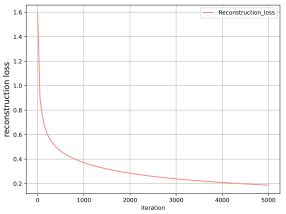


Figure 24. The reconstruction loss

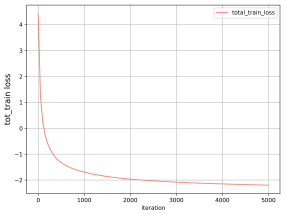


Figure 25. The total train loss

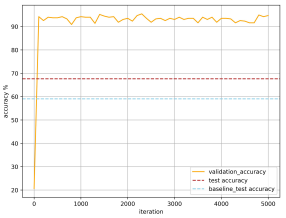


Figure 26. The validation accuracy loss during training

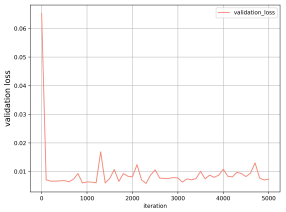


Figure 27. The validation loss

We could observe that the results are similar with the sketch domain except for the validation loss that has a little fluctuation but the overall value is still quite low.

A.1.2 Photo

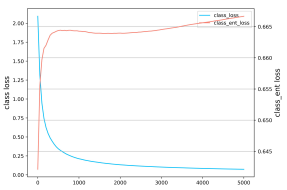


Figure 28. The category disentanglement loss

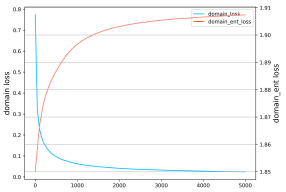


Figure 29. The domain disentanglement loss

Domain	Art Painting → Cartoon	Art Painting → Sketch	Art Painting → Photo	Average
BaseLine	59.04	58.72	94.07	00
Domain Disentanglement	67.62	66.63	94.85	76.36
Clip with labeled PACS	68.09	59.03	95.13	74.08
Fined tuned CLIP model	69.75	64.57	95.23	76.51

Table 2. Accuracy (%) result on PACS. All results are based on ResNet-18 backbone. Baseline uses a linear encoder and classifier. Overall and Avg. indicate the overall test data accuracy and the mean of the per-domain accuracy, respectively. They are different since the test sets of different domains are not of the same size.

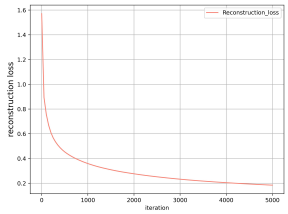


Figure 30. The reconstruction loss

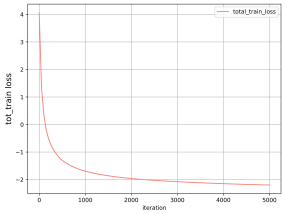


Figure 31. The total train loss

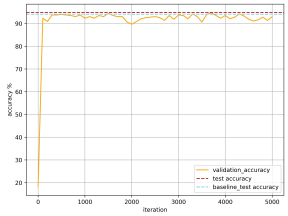


Figure 32. The validation accuracy loss during training

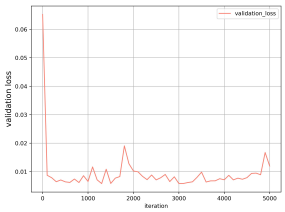


Figure 33. The validation loss

When the target domain is photo, we can see the class disentanglement's performance drop a little by the 2000^{th} iteration, and then it bounces back again until the end of training.

A.2. No Pre–train CLIP

Following are the graphs obtained from the model with the frozen CLIP model.

A.2.1 Sketch

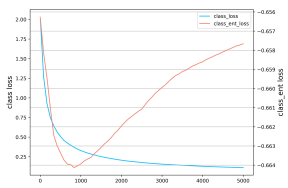


Figure 34. The category disentanglement loss

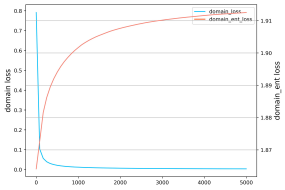


Figure 35. The domain disentanglement loss

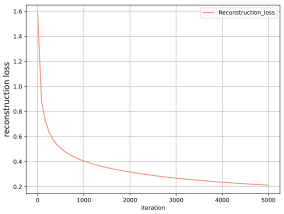


Figure 36. The reconstruction loss

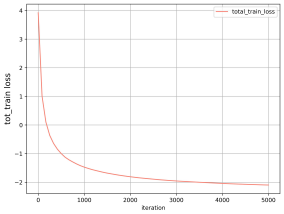


Figure 37. The total train loss

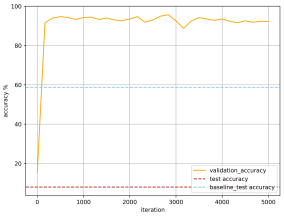


Figure 38. The validation accuracy loss during training

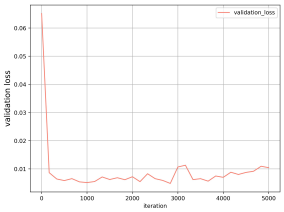


Figure 39. The validation loss

A.3. Fine Tune CLIP

A.3.1 Photo

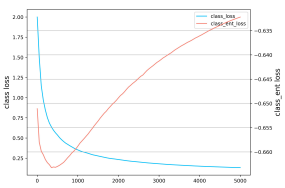


Figure 40. The category disentanglement loss

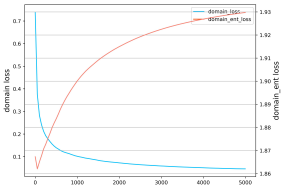


Figure 41. The domain disentanglement loss

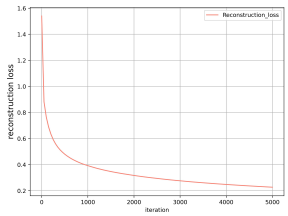


Figure 42. The reconstruction loss

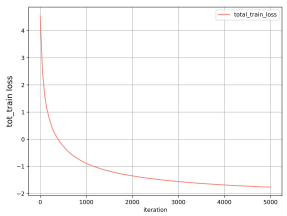


Figure 43. The total train loss

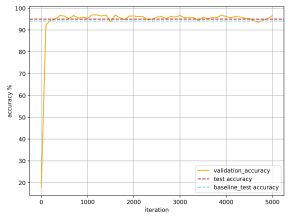


Figure 44. The validation accuracy loss during training

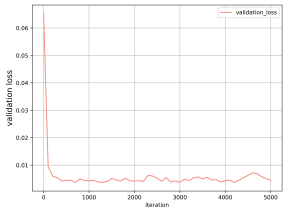


Figure 45. The validation loss

A.3.2 Sketch

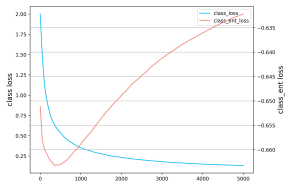


Figure 46. The category disentanglement loss

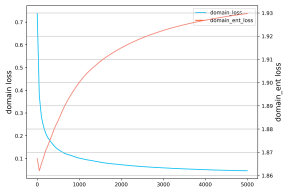


Figure 47. The domain disentanglement loss

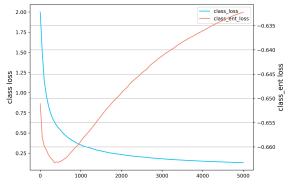


Figure 48. The reconstruction loss

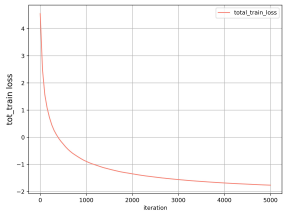


Figure 49. The total train loss

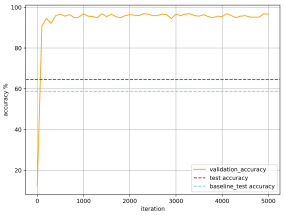


Figure 50. The validation accuracy loss during training

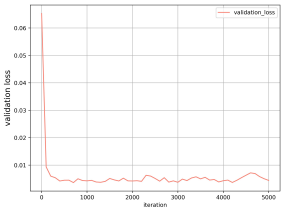


Figure 51. The validation loss

References

[1] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pages 213–226. Springer, 2010. 1

[2] 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

[3] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. *Advances in neural information processing systems*, 29, 2016. 2

[4] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3964–3973, 2018. 2

[5] Xingchao Peng, Yichen Li, and Kate Saenko. Domain2vec: Domain embedding for unsupervised domain adaptation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VI 16*, pages 756–774. Springer, 2020. 2

[6] 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

[7] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 2

[8] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European conference on computer vision (ECCV)*, pages 35–51, 2018. 2

[9] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. In *International Conference on Machine Learning*, pages 5102–5112. PMLR, 2019. 3

[10] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3