ToAlign: Task Oriented Alignment for Unsupervised Domain Adaptation

Presented at NeurIPS 2021

Shrusti
CS22MTECH11017
IIT Hyderabad
cs22mtech11017@iith.ac.in

Abstract

ToAlign is a method designed to address the problem of unsupervised domain adaptation, where a machine learning model trained on a source domain is expected to generalize well to a different target domain without any labeled data from the target domain. The method aims to learn a shared feature representation between a source domain and a target domain. The approach involves aligning the feature distributions of both domains while preserving their task-specific information i.e; we decompose the source features into task-relevant features that should be aligned and task-irrelevant features that should be avoided. Experiment was conducted on the benchmark Office-Home dataset and was able to reproduce the results with an average accuracy of 69.25%.

1. Introduction

Trained models usually perform well on the testing data when they share the same data distribution, but here as the data of the source domain and target domain come from different data distributions, the performance of the model will be affected. To alleviate the adverse effect of domain shift, many approaches align the source and target domains in the feature space.
Fine-tuning on labeled target data is a direct solution but is costly due to the requirement of target sample annotations. However, a feature is usually taken as a whole for alignment without explicitly making domain alignment proactively serve the classification task, leading to suboptimal solution.

Hence an effective Task-oriented Alignment (ToAlign) for unsupervised domain adaptation (UDA) was proposed where we study what features should be aligned across domains.

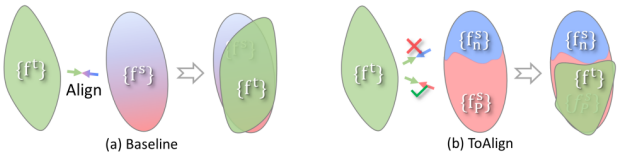


Figure 1. (a) Baseline considers all the features of source (b) ToAlign considers only the task relevant (positive) source features. We can observe that the alignment in better in ToAlign than in Baseline.

2. Related Work

Ben et al theoretically proved that learning domain-specific features makes the image classifier trained on source domain applicable to target domain.

Maximum Mean Discrepancy (MMD) is a popular domain adaptation method that measures the difference in means between two domains. However, MMD only considers means and may not capture other important features for the task, potentially affecting generalization performance.

Multi-adversarial domain adaptation (MADA) is a domain adaptation method that adapts a model to a target domain using multiple source domains. While MADA has been shown to be effective in improving the model's performance on the target domain, it requires many source domains, which can increase training time and require additional computational resources.

Centerness aware alignment is a domain adaptation method that aligns the center part of objects between the

source and target domains, while excluding background distraction and noise.But features in the object center position could be task-irrelevant and not suited for alignment, which could affect the model's generalization performance.

Different from these methods here we investigate what features to be aligned by decomposing source features in task-relevant and task-irrelevant features.

3. Methodology

To align the features of source and target domains, we use adversarial learning whose goal is to bring the source and target domains closer.

3.1. Domain Adversarial UDA

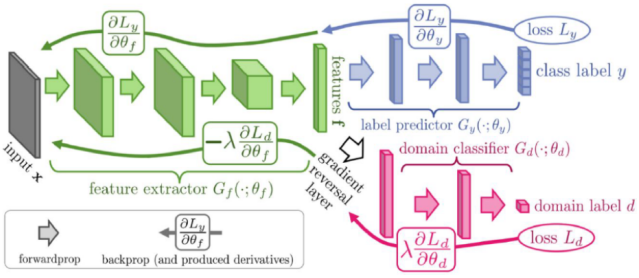


Figure 2. Architecture of the domain adversarial neural network.

DANN's architecture involves the following components : 1) Label predictor : This is a neural network that will learn to perform classification on the transformed source distribution. Since, source domain is labelled.

2) Domain classifier : This is a neural network that predicts whether output from feature extractor is from source domain or target domain.

In order to identify which domain a sample belongs to, domain adversarial learning-based UDAs typically train a domain discriminator D, and then adversarially train a feature extractor G to fool the discriminator D in order to learn domain-invariant feature representations. Particularly,

D is optimised to minimise domain classification lossL_D and G is optimised to maximise domain classification loss L_D and minimise image classification loss L_{cls}.

$$\underset{D}{\operatorname{argmin}}\, L_D$$

$$\underset{G}{\operatorname{argmin}}\, L_{\text{cls}} - L_D$$

L_D is usually defined using BCE loss [6] as the discriminator just has two choices of deciding whether it is a source domain or a target domain.

$$\mathrm{L}_D(X_s; X_t) = \mathbb{E}_{x_s \sim X_s}[\log(D(G(x_s)))] - \mathbb{E}_{x_t \sim X_t}[\log(1 - D(G(x_t)))].$$

Gradient Reversal Layer (GRL) is placed between feature extractor and domain classifier, it helps to maximise the domain classification loss . It basically acts as an identity function(output is same as input) during forward propagation and during backpropagation it multiplies input by -1 leading to the opposite of gradient descent.

The main idea here is instead of using all the features of the source, just use task-relevant features.
ToAlign is generic and can be applied to different adversarial learning based UDAs.

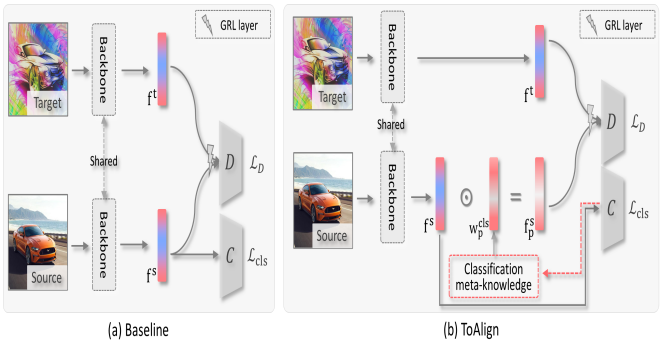


Figure 3. Illustration of adversarial learning based (a) Baseline and (b) ToAlign.

3.2. Task Oriented Feature Decomposition

We can observe that aligining the target features with the source-irrelevant features drastically reduces the classification accuarcy of the model.

Reweight the feature vector f_s to get f_p^s and f_n^s, this is done using Grad CAM. Grad CAM identifies the relevant features to recognise the image correctly. It generate a heatmap that highlights the most important regions in the image for the class prediction.

From the final convolution block of DANN's feature extractor we obtain a feature vector $F \in \mathbb{R}^{H \times W \times M}$, then apply global average pooling to obtain a feature vector $f = \text{pool}(F) \in \mathbb{R}^M$.
The labels of the class are predicted via classifier C, based on response obtained from the classifier we derive the gradient w^{cls} of y^k w.r.t. f.We are doing this because gradient captures the influence of each dimension of the feature on prediction class k and these gradients highlight the regions of input that are most important for the prediction of class

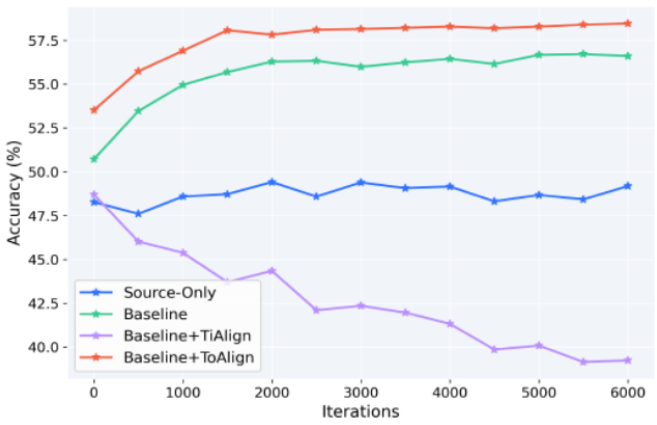


Figure 4. Classification accuracies of RealWorld $\Longrightarrow$ ClipArt on using various techniques.

k and as stated in [5] w^{cls} tells how the information of feature f is used for classifying sample into its groundtruth class k.

$$w^{cls} = \frac{\partial y^k}{\partial f}$$

where yk is the predicted score corresponding to the ground-truth class k.

Grad CAM uses w^{cls} to obtain task-relevant i.e; positive features as

$$f_p = w^{cls} \odot f = s w^{cls} \odot f$$

where $s \in \mathbb{R}^+$ is a non-negative parameter called as 'attention score' which gives higher weights to the elements that are important for the task and lower weights to the elements that are less important. This helps in identifying task-relevant features and suppresses the task-irrelevant features.

$$s = \sqrt{\frac{\|f\|_2^2}{\|w_{cls} \odot f\|_2^2}} = \sqrt{\frac{\sum_{m=1}^{M} f_m^2}{\sum_{m=1}^{M} (w_m^{cls} f_m)_2^2}}$$

The attention score s is computed as the ratio of the squared L2 norm (because it measures the strength of the feature vector) of the feature map to the squared L2 norm of the weighted feature map, which measures the degree of contribution of each channel to the weighted feature map.

In the below figure, we can see that foreground is the positive information and background is the negative information.

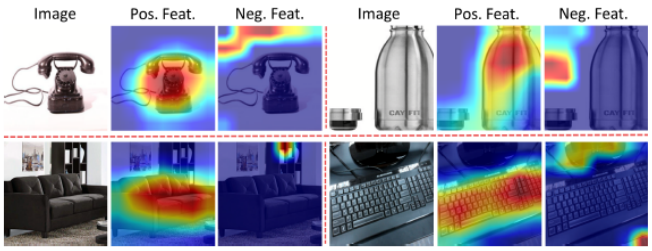


Figure 5. Visualisation of task relevant and task irrelevant features as positive information and negative information respectively.

3.3. Task-oriented Domain Alignment

Now, the input source feature f^s is replaced by f_p^s. Thus the domain classification loss is defined with the small modification as follows

$L_D(X_s; X_t) = -E_{x_s \sim X_s}[\log(D(G^p(x_s)))] - E_{x_t \sim X_t}[\log(1 - D(G(x_t)))]$

where, $G_p(x_s) = f_p^s$ denotes the positive feature of source x_s.

4. Implementation and Results

4.1. Dataset

The commonly used benchmark dataset *Office Home* [4]. It consists of images from four different domains namely Art (Ar), Clipart (Cl), Product (Pr),and Real-World (Rw). Each domain contains 65 object categories in office and home environments.

4.2. Implementation Details

To evaluate the effectiveness of ToAlign we conducted an experiment under SUDA (Single source Unsupervised Domain Adaptation). SUDA means training a model on a single source domain, and then adapting it to perform well on a target domain, without the need for labeled data in the target domain.

Baselines used are DANNP which is an improved version of DANN where only privileged information (positive source features) are used and the other baseline used is HDA which identifies task relevant features through heuristic search.

Resnet-50 architecture is used as a starting point(backbone) of the implementation. One fully connected layer is used in the label classifier and the discriminator D consists of three fully connected layers with inserted dropout and ReLU layers.

The learning rate is set as follows

$$\eta_t = \frac{\eta_0}{(1+\gamma p)^\tau}$$

where p indicates the progress of training that increases linearly from 0 to 1, it controls how quickly or slowly the learning rate should change. γ indicates the decay rate (i.e.; gradually reducing the learning rate over time training process helping the model to converge towards better optimum), it is a regularization technique that helps prevent overfitting. τ controls the strength of the regularization applied.

Initial values of γ = 5e-4, τ = 0.75, η_0 = 1e - 3 for Office-home dataset. The batch size used is 36 and the optimizer used is SGD with a momentum of 0.9.

4.3. Results

The results given in the paper are as follows

Method	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
Source-Only [22]	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
MCD(CVPR'18) [53]	48.9	68.3	74.6	61.3	67.6	68.8	57.0	47.1	75.1	69.1	52.2	79.6	64.1
CDAN(NeurIPS'18) [41]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
ALDA(AAAI'20) [11]	53.7	70.1	76.4	60.2	72.6	71.5	56.8	51.9	77.1	70.2	56.3	82.1	66.6
SymNet(NeurIPS'18) [74]	47.7	72.9	78.5	64.2	71.3	74.2	63.6	47.6	79.4	73.8	50.8	82.6	67.2
TADA(AAAI'19) [66]	53.1	72.3	77.2	59.1	71.2	72.1	59.7	53.1	78.4	72.4	60.0	82.9	67.6
MDD(ICML'19) [73]	54.9	73.7	77.8	60.0	71.4	71.8	61.2	53.6	78.1	72.5	60.2	82.3	68.1
BNM(CVPR'20) [14]	56.2	73.7	79.0	63.1	73.6	74.0	62.4	54.8	80.7	72.4	58.9	83.5	69.4
GSDA(CVPR'20) [25]	**61.3**	76.1	79.4	65.4	73.3	74.3	65.0	53.2	80.0	72.2	60.6	83.1	70.3
GVB(CVPR'20) [15]	57.0	74.7	79.8	64.6	74.1	74.6	65.2	55.1	81.0	74.6	59.7	84.3	70.4
E-Mix(AAAI'21) [77]	57.7	76.6	79.8	63.6	74.1	75.0	63.4	56.4	79.7	72.8	**62.4**	**85.5**	70.6
MetaAlign(CVPR'21) [68]	59.3	76.0	80.2	65.7	74.7	75.1	65.7	56.5	81.6	74.1	61.1	85.2	71.3
DANNP [68]	54.2	70.0	77.6	62.3	72.4	73.1	61.3	52.7	80.0	72.0	56.8	83.1	67.9
DANNP+ToAlign	56.8↑	74.8↑	79.9↑	64.0↑	73.9↑	75.3↑	63.8↑	53.7↑	81.1↑	73.1↑	58.2↑	84.0↑	69.9↑
HDA(NeurIPS'20) [13]	56.8	75.2	79.8	65.1	73.9	75.2	66.3	56.7	81.8	**75.4**	59.7	84.7	70.9
HDA+ToAlign	57.9↑	**76.9**↑	**80.8**↑	**66.7**↑	**75.6**↑	**77.0**↑	**67.8**↑	**57.0**↑	**82.5**↑	75.1↓	60.0↑	84.9↑	**72.0**↑

Figure 6. The table presents the accuracy percentage of various UDA methods on the Office-Home dataset, using ResNet-50 as the backbone. The highest accuracy values are indicated in bold.

Method		Acc.
DANNP		67.9
DANNP+ToAlign	s =1	59.7
	s =8	68.8
	s =16	69.7
	s =64	70.0
	s =128	69.8
	Adaptive s	69.9

Figure 7. Ablation study on the influence of s

Comparison between the results given in the paper and the reproduced results are as follows

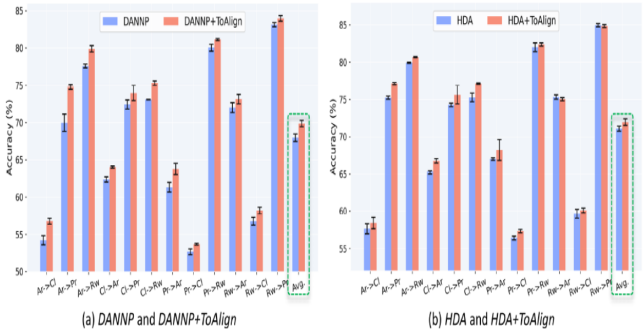
As we saw that paper implementation uses a batch size of 36 but while reproducing results the batch size used was 15 due to computational constraints and also randomness of data the reproduced accuracies are slightly lower than the published accuracies.

Method	Time/ms	GPU mem./MB	Acc./%
DANNP	550	6,660	67.9
DANNP+ MetaAlign[68]	1,000	10,004	69.5
DANNP+ ToAlign	590	6,668	69.9

Figure 8. Computational time of one iteration and GPU memory for a mini-batch with batch size 32

	Paper's Accuracies in %	Reproduced Accuracies in %
Art -> ClipArt	57.9	57.2
Art -> Product	76.9	76.3
Art -> Real World	80.8	77.9
ClipArt -> Art	66.7	59.7
ClipArt -> Product	75.6	73.5
ClipArt -> Real World	77.0	71.3
Product -> Art	67.8	62.4
Product -> ClipArt	57.0	56.5
Product -> Real World	82.5	79.8
Real World -> Art	75.1	74.2
Real World -> ClipArt	60.0	59.2
Real World -> Product	84.9	84

Figure 9. Comparison of paper's accuracies and reproduced accuracies



(a) *DANNP and DANNP+ToAlign* (b) *HDA and HDA+ToAlign*

Figure 10. Error bars of ToAlign on top of DANNP and HDA on Office-Home dataset

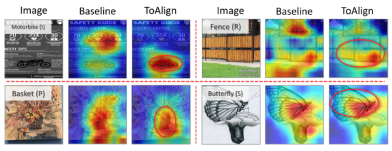


Figure 11. Visualization of the feature response maps on target test images

5. Novelty Idea

5.1. Limitations and Implications of Domain Adaptation Models

- A task-oriented alignment is a data-driven approach that relies on the similarity between the feature representations of the source and target domains. As such,

it can be challenging to interpret the alignment process and understand how the adapted model makes predictions.

- Also, ToAlign assumes that the domain shift between the source and target domains is limited and can be captured by a linear transformation. However, in practice, domain shifts can be much more complex.

- The lack of understanding of the knowledge transfer in DA models might raise doubts about their real-life applications especially when the stakes are high.

Some of the ways in which we can overcome this limitations is as follows:

5.2. Proposed Approach 1

Instead of using Grad CAM, we can try Grad CAM ++. [2]

Grad CAM is not able to identify multiple instances of objects. It suppresses activation maps that have a lesser spatial footprint. When computing weights of activation maps Grad CAM considers all pixel weights equally.

Whereas Grad CAM ++ uses 'pixel-wise weight' i.e; α .This pixel-wise weight tells how they must contribute to the saliency maps.

$$w_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial Y_c}{\partial A_{ij}^k}$$

where Z is a constant (number of pixels in the activation map), Y^c is a differentiable function of the activation maps A^k.

$$\alpha_{ij}^{kc} = \frac{\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2}}{2\left(\frac{\partial^2 Y^c}{(\partial A_{ij}^k)^2} + \sum_a \sum_b A_{ab}^k \left\{\frac{\partial^3 Y^c}{(\partial A_{ij}^k)^3}\right\}\right)}$$

The value depends on the second and third partial derivatives of the loss function with respect to the activation map, as well as the values of it at all other pixel locations. It can be calculated using these values during training and is used to adjust the gradient computed during backpropagation so that the model parameters are updated more effectively.

From the above sample images we can observe that Grad CAM++ performs better than Grad CAM.

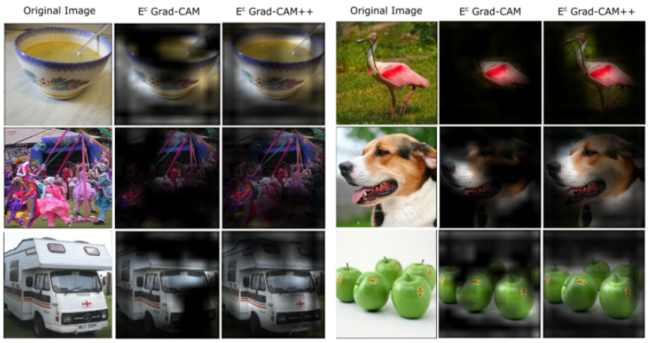


Figure 12. Sample visual explanations on ImageNet generated by Grad-CAM and Grad-CAM++

5.3. Proposed Approach 2

As there is a lack of understanding of the knowledge transfer we can visualize it by following the below approach. [3]

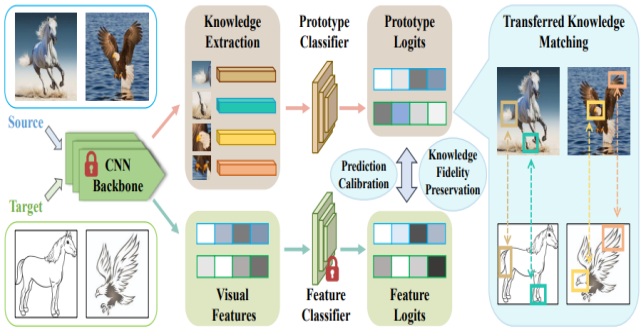


Figure 13. Framework of the proposed transferred knowledge visualization in unsupervised domain adaptation

We must be able to say "This target image part looks like that source sample part since they share the same semantics."
Overall, the proposed model aims to provide a visual interpretation of the transferred knowledge in unsupervised domain adaptation.
Consists of 3 models in it :

The Knowledge extraction model extracts the features learned from the pre-trained model.It helps to identify the specific feature which is used to distinguish it from others.

The Prediction calibration model learns to predict based on how similar the new data is to the extracted features. Considers the original model's predictions on the same data because it has already learned to recognize patterns that are relevant to the task.

The Knowledge fidelity preservation model ensures that the extracted knowledge is accurate and faithfully represents the original information learned by the pre-trained model. It does this by comparing the output probabilities of the new model and the original pre-trained model on the same data, and then making sure that the distribution of predicted probabilities is aligned between the two models. Essentially, this model checks whether the new model is truly utilizing the same knowledge as the original model, rather than just memorizing the training data.

6. Observations

ToAlign + HDA outperforms all previous method and achieves state of art performance, i.e; it outperforms HDA by 0.9%.

We can see that ToAlign occupies almost the same GPU memory as the baseline without adding any extra overhead as shown in the figure 7.

Baseline sometimes focuses on the background features which are useless to the image classification task, since it aligns the holistic features without considering the discriminativeness of different channels/sub-features.

Our proposed ToAlign emphasizes that domain alignment task should assist/serve classification task, where we perform alignment under the guidance of the meta-knowledge induced from classification task.

References

[1] https://arxiv.org/pdf/2106.10812.pdf Guoqiang Wei1, Cuiling Lan2,Wenjun Zeng, Zhizheng Zhang, Zhibo Chen1 (2021): ToAlign: Task-oriented Alignment for Unsupervised Domain Adaptation

[2] https://arxiv.org/pdf/1710.11063.pdf Aditya Chattopadhyay ,Anirban Sarkar ,Prantik Howlader, and Vineeth N Balasubramanian (2018):Grad-CAM++: Improved Visual Explanations for Deep Convolutional Networks. 5

[3] https://arxiv.org/pdf/2303.02302v1.pdf Wenxiao Xiao, Zhengming Ding, Hongfu Liu (2023): Visualizing transferred knowledge : An interpretive model of unsupervised domain adaptation. 5

[4] H. Venkateswara, J. Eusebio, S. Chakraborty, and S. Panchanathan. Deep hashing network for unsupervised domain adaptation. In CVPR, pages 5018–5027, 2017. 3

[5] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In ICCV, pages 618–626, 2017. 3

[6] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(1):2096–2030, 2016. 2