భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

Task-oriented Alignment for Unsupervised Domain Adaptation

Presented at **NeurIPS 2021**

CS6450 : Visual Computing

Presentation-1

Supervisor:

Prof. C. Krishna Mohan

Presenter :

Shrusti

CS22MTECH11017

Teaching Assistant :
Aveen Dayal

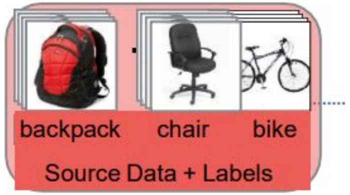
Contents

Motivation

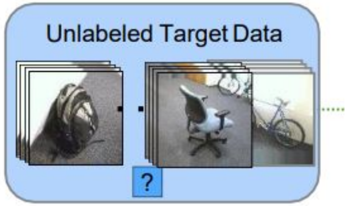
- The trained models typically perform well when applied to testing data, which has same data distribution to the training data.

- However, in practice we observe performance degradation due to **Domain shift.**

Myth : You cant do deep learning unless you have million labelled examples for your problem.

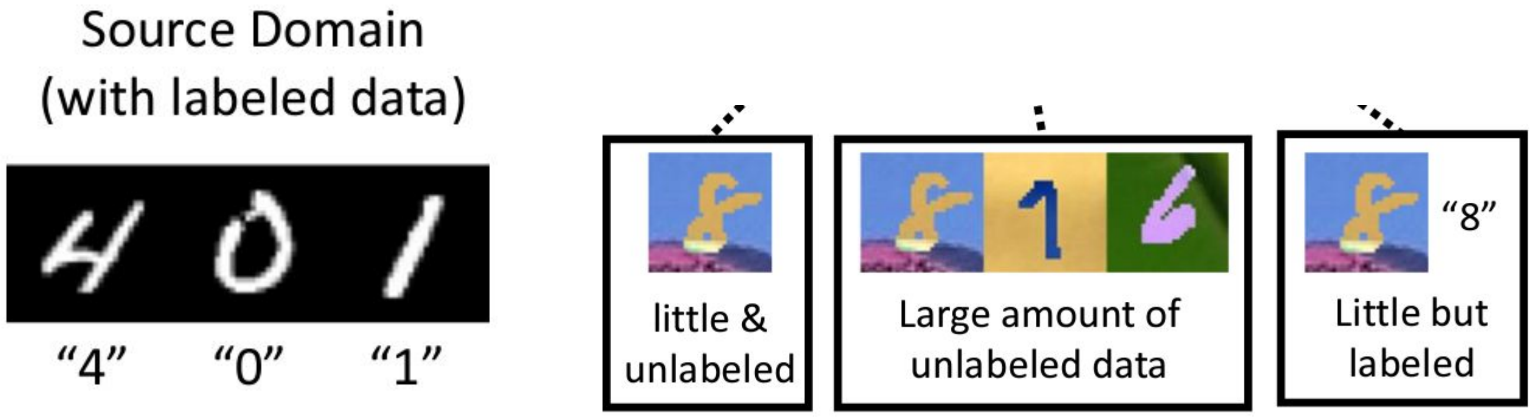


Reality :
1) You can learn useful representations from unlabelled data
2) You can transfer learned representations from a related task



Labelled data is available sometimes in cases of image recognition, speech recognition, recommendation, etc
But difficult to collect sometimes in Robotics, Disaster, Medical diagnosis, etc

Types of data on Target domain



Challenges

There are several challenges in task-oriented alignment for unsupervised domain adaptation:

Distribution shift: The distribution of the data in the source and target domains may be different, which can make it difficult to align the feature spaces in a way that is effective for the specific task.

Fine-tuning (updating weights) on labeled target data is a direct solution but is costly due to the requirement of target sample's label.

Lack of labeled data in the target domain:

Solution : Use unsupervised domain adaptation (UDA) requires only the labeled source data and unlabeled target data to enhance the model's performance on the target domain.

Task-specificity: Aligning the feature spaces in a way that is specific to the task can be challenging, as it requires a good understanding of the task and the relationship between the source and target domains.

Problem Statement

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.

In this paper, we propose an effective **Task-oriented Alignment (ToAlign)** for unsupervised domain adaptation (UDA).

- We study what features should be aligned across domains and propose to make the domain alignment proactively serve classification by performing feature decomposition and alignment under the guidance of the prior knowledge induced from the classification task itself.

Related Work

Method	**Explanation**	**Limitation**
Maximum Mean Discrepancy (MMD)	Difference in the means (averages) of the features between two domains.	Considers only mean and ignores other features
Multi-adversarial domain adaptation(MADA)	Adapting a model to a target domain using multiple source domains	● Require many source domains ● Can increase training time and computational resources are required
Centerness aware alignment	aligning the center part of the objects to exclude the background distraction/noise	Feature in object center position could be task-irrelevant and thus is not suited for alignment
HDA (Heuristic Domain Adaptation)	leverages domain-specific representations as heuristics to obtain domain-invariant representations from a heuristic search perspective	Search space is limited

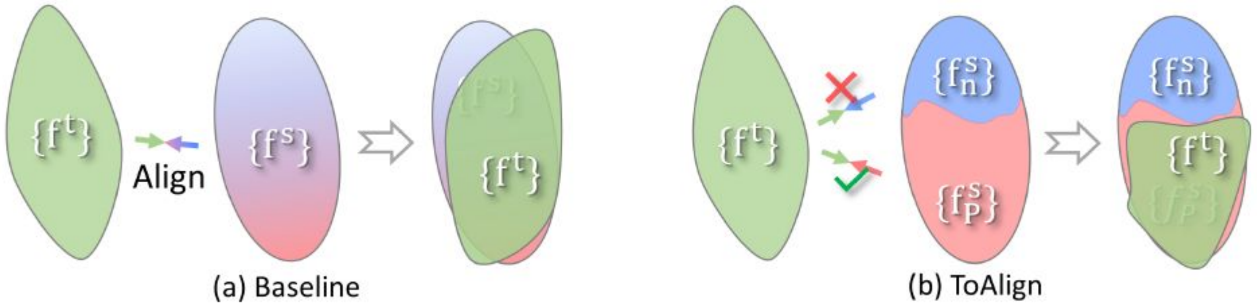
Paper’s Method

- The authors suggest a new method for making the process of aligning features across domains more effective for the image classification task.

- This new method focuses on using information about the classification task (known as "**meta-knowledge**") to choose which specific features to align, instead of just aligning all features at once. This approach is expected to provide clearer results and be better at helping the classification task.

- The authors are the first to perform **task-oriented alignment** by decomposing source feature into **task-discriminative and task-irrelevant** feature, and explicitly guides the network what sub-features should be aligned.

Task Oriented Alignment



- We decompose a source sample feature into a task-discriminative one that should be aligned, and a task-irrelevant one that should be ignored under the guidance of classification-meta knowledge for performing **classification-oriented alignment**, which explicitly guides the network what features should be aligned.

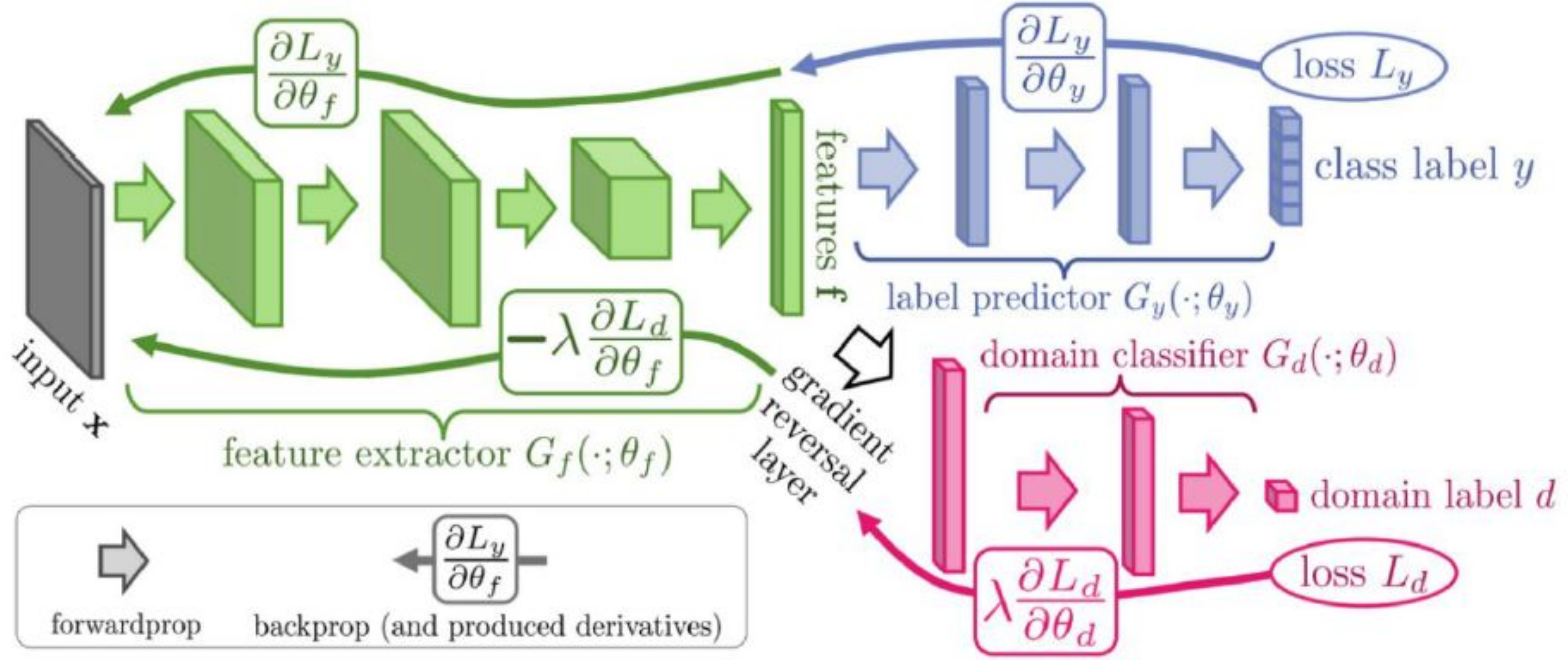
- Then, we perform alignment between the target features and the positive source features, which is consistent with the essence of the classification task.

- **Domain invariant features** - are those features that are consistent between the source and the target domain.

➢ And our goal is to learn domain invariant features.

Domain Adversarial UDA

Goal is to bring the distributions of source and target closer.



$\frac{\partial L_y}{\partial \theta_f}$
$\frac{\partial L_y}{\partial \theta_y}$
loss L_y
features $\mathbf{f}$
class label y
label predictor $G_y(\cdot;\theta_y)$
$-\lambda \frac{\partial L_d}{\partial \theta_f}$
input $\mathbf{x}$
feature extractor $G_f(\cdot;\theta_f)$
gradient reversal layer
domain classifier $G_d(\cdot;\theta_d)$
domain label d
$\lambda \frac{\partial L_d}{\partial \theta_d}$
loss L_d
forwardprop
backprop (and produced derivatives)

Domain Adversarial UDA

Label predictor : This is a neural network that will learn to **perform classification** on the transformed source distribution. Since, source domain is labelled.

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.

Intuition : Feature extractor will try to perform some transformation on source and target instances such that the transformed instances appear as it is coming from same domain.

- Particularly, D is optimised to **minimise domain classification** loss L_D and G is optimised to **maximise domain classification loss** L_D and **minimise image classification loss** L_{cls}.

$$\underset{D}{\operatorname{argmin}} \mathcal{L}_D,$$
$$\underset{G}{\operatorname{argmin}} \mathcal{L}_{cls} - \mathcal{L}_D,$$

- L_D is typically defined as (BCE loss)

$$\mathcal{L}_D(\mathbf{X}_s, \mathbf{X}_t) = -\mathbb{E}_{\mathbf{x}_s \sim \mathbf{X}_s}[\log(D(G(\mathbf{x}_s)))] - \mathbb{E}_{\mathbf{x}_t \sim \mathbf{X}_t}[\log(1 - D(G(\mathbf{x}_t)))]$$

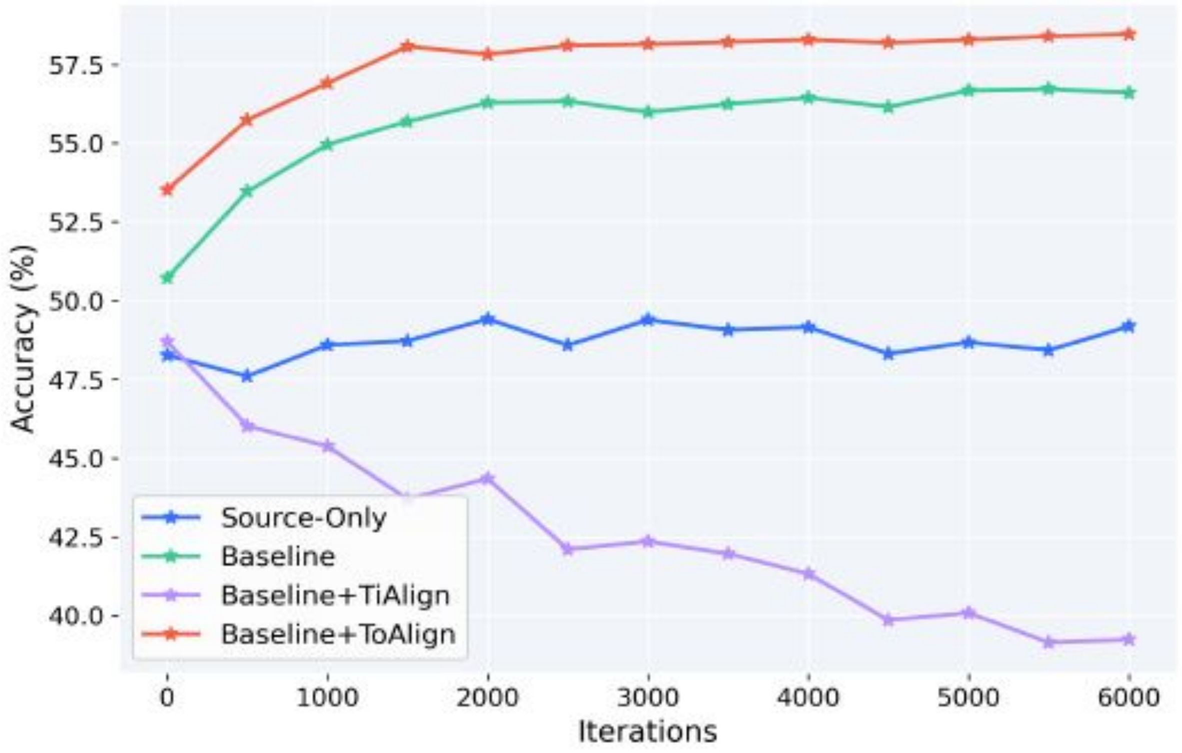
And L_{cls} is the Cross Entropy loss.

- **<u>Gradient Reversal Layer</u>** : 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.

Task oriented feature decomposition

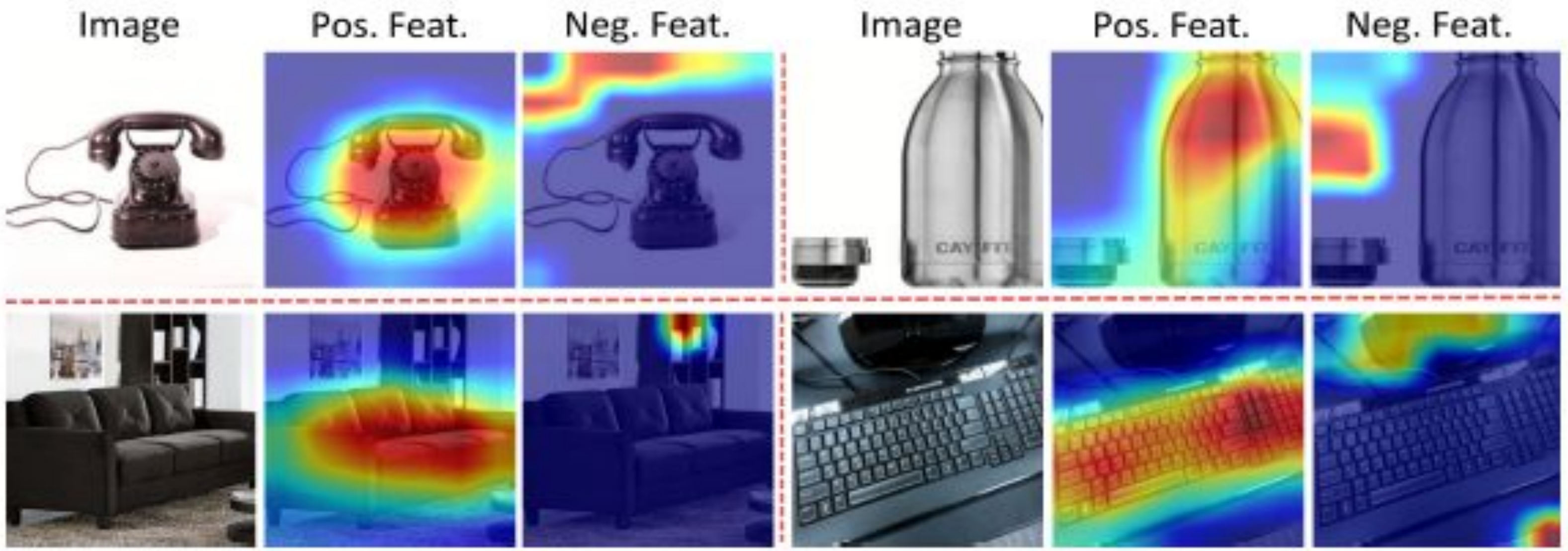
Mistakenly aligning the target features with the source task-irrelevant features would hurt the discrimination power of the target features.

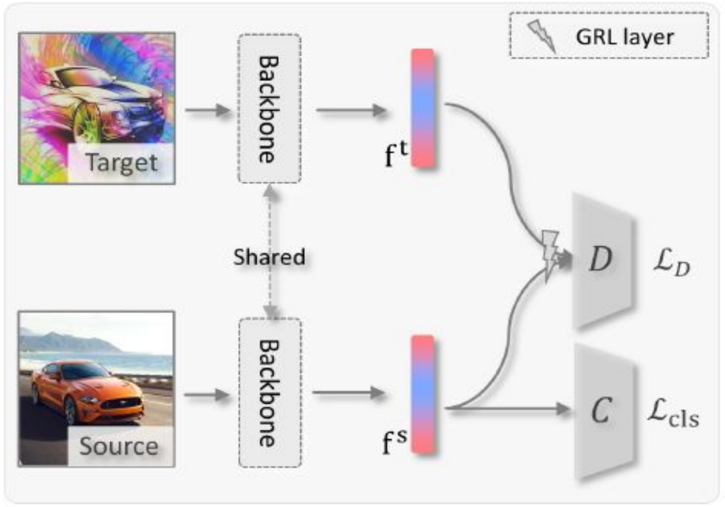


Solution : Reweight the feature vector f{s} to get f{s,p} and f{s, n}, this is done using Grad CAM.

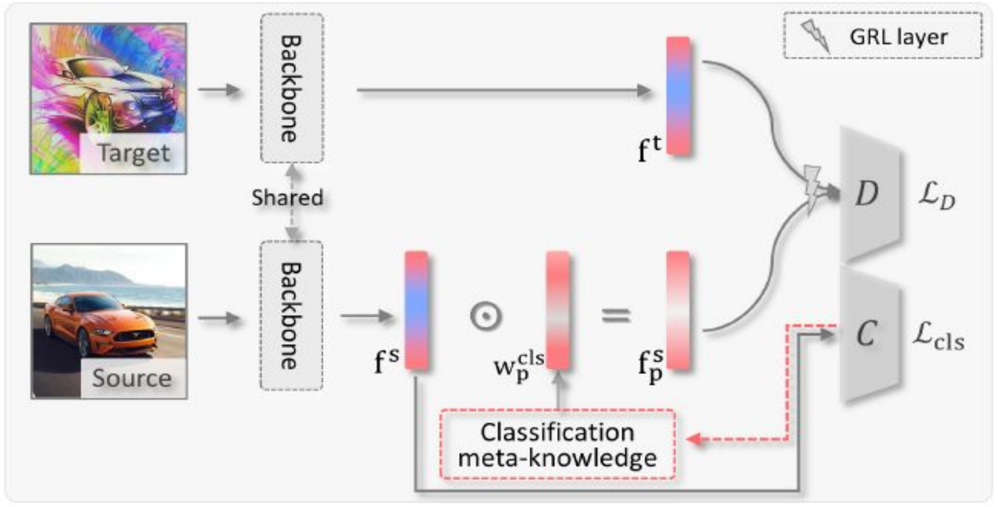
Grad CAM identifies the relevant features to recognise the image correctly.

It generate a heatmap that highlights the most important regions in the image for the class prediction.





(a) Baseline



(b) ToAlign

We obtain a feature map F with height H, width W, and M channels from the final convolutional block of the feature extractor. After spatial-wise global average pooling (GAP), we have a feature vector

f = pool(F) belongs to R^M

GAP is a type of pooling that computes **average of each channel** of feature map over height and width.

- The labels of the class are predicted via classifier C(.), based on response of classifier we derive the gradient w_{cls} of y^k w.r.t. f
- Gradient captures the influence of each dimension of the feature on prediction class k and these gradients highlights the regions of input that are most important for prediction of class k.

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

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

- Grad CAM uses w^{cls} to obtain discriminative features(i.e positive features) as :

$$\mathbf{f}_p = \mathbf{w}_p^{cls} \odot \mathbf{f} = s\mathbf{w}^{cls} \odot \mathbf{f},$$

Where $s \in R_+$ 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.

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

- 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.**

Task oriented Domain Alignment

Now, the input source feature f^s is replaced by f{s,p}.

Thus the domain classification loss is defined with the small modification as :

$$\mathcal{L}_D(\mathbf{X}_s, \mathbf{X}_t) = -\mathbb{E}_{\mathbf{x}_s \sim \mathbf{X}_s}\left[\log(D(G^p(\mathbf{x}_s)))\right] - \mathbb{E}_{\mathbf{x}_t \sim \mathbf{X}_t}\left[\log(1 - D(G(\mathbf{x}_t)))\right].$$

where $G^p(\mathbf{x}_s) = \mathbf{f}_p^s$ denotes the positive feature of source $\mathbf{x}_s$.

Understanding from Meta knowledge perspective

- **Meta learning** - learning about what features to learn.
- In a domain adversarial UDA framework, there are two tasks:
  - the image classification task → meta training task
  - domain alignment task → meta testing task

- ToAlign uses the meta-knowledge learned from the meta-training task to i**dentify relevant features** in the feature space and guide the optimization of the meta-testing task.

- The meta-knowledge is learned through the **gradients of the parameters** w.r.t. the loss function of the meta-training task.
- The learned meta-knowledge is then used to guide the optimization of the meta-test task and improve its performance.

Experiments

To evaluate effectiveness of ToAlign we conducted experiment under **SUDA** (Single source Unsupervised Domain Adaptation).

SUDA - 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

1. **DANNP** is an improved version of DANN where only **privileged information** (positive source features) are used
2. **HDA**

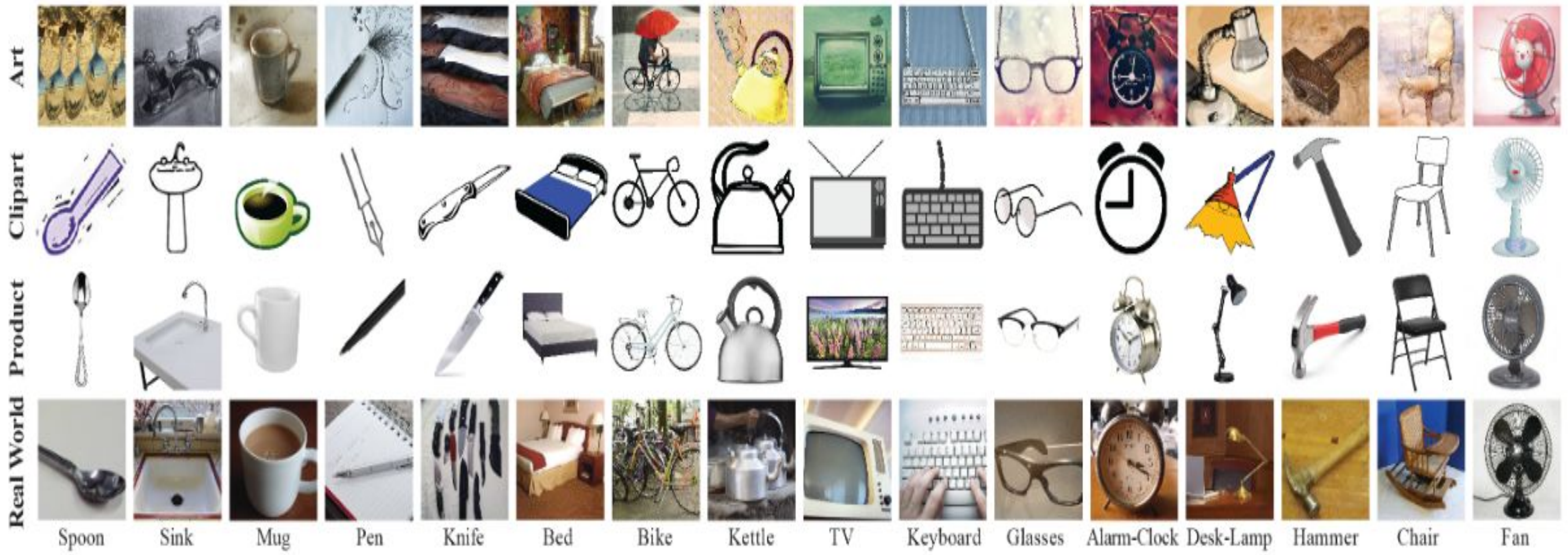
Datasets :

Office - Home

a) Consists of images from four different domains: **Art (Ar), Clipart (Cl), Product (Pr),and Real-World (Rw).**
b) Each domain contains **65** object categories in office and home environments.

Office - Home dataset

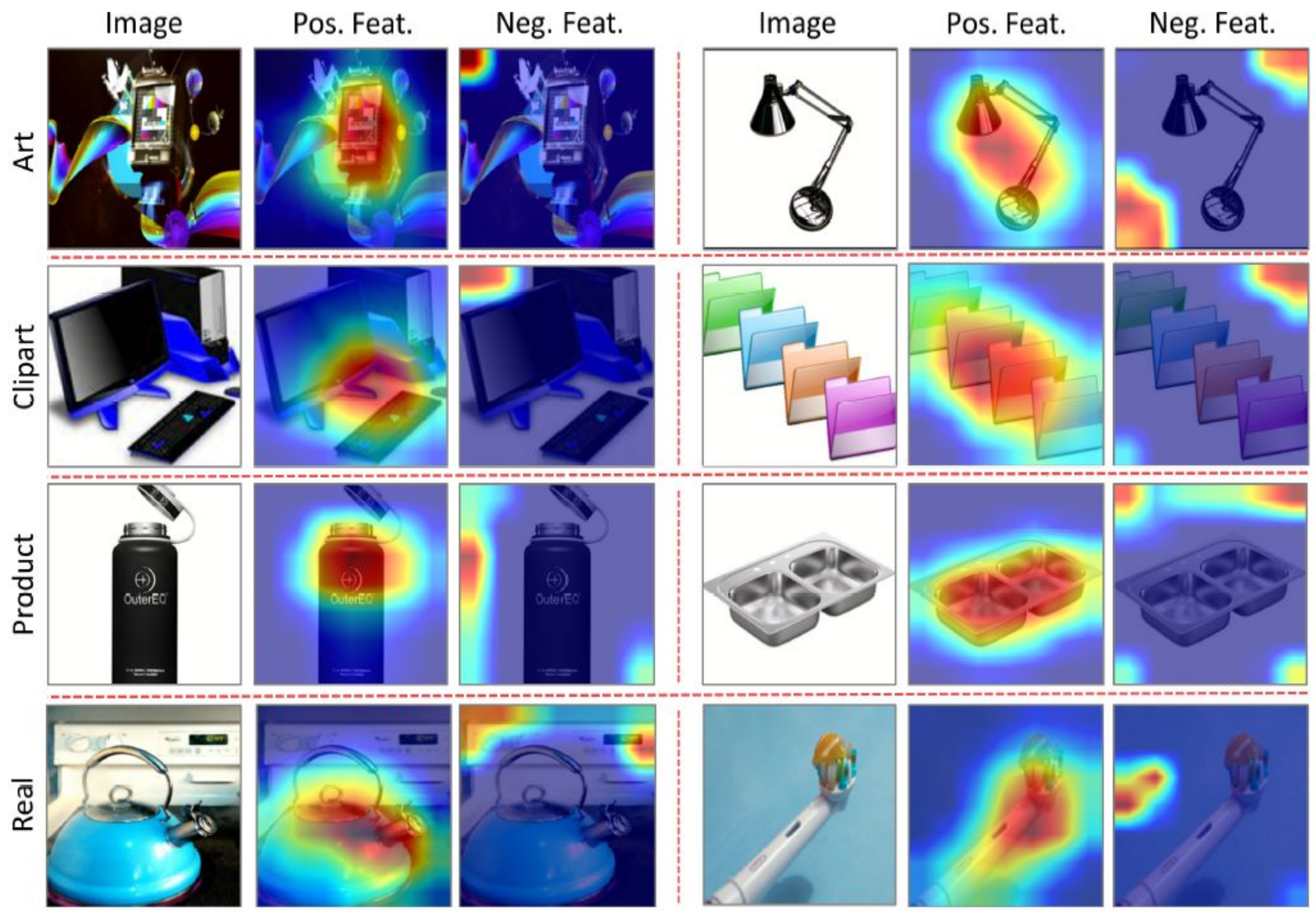
భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad



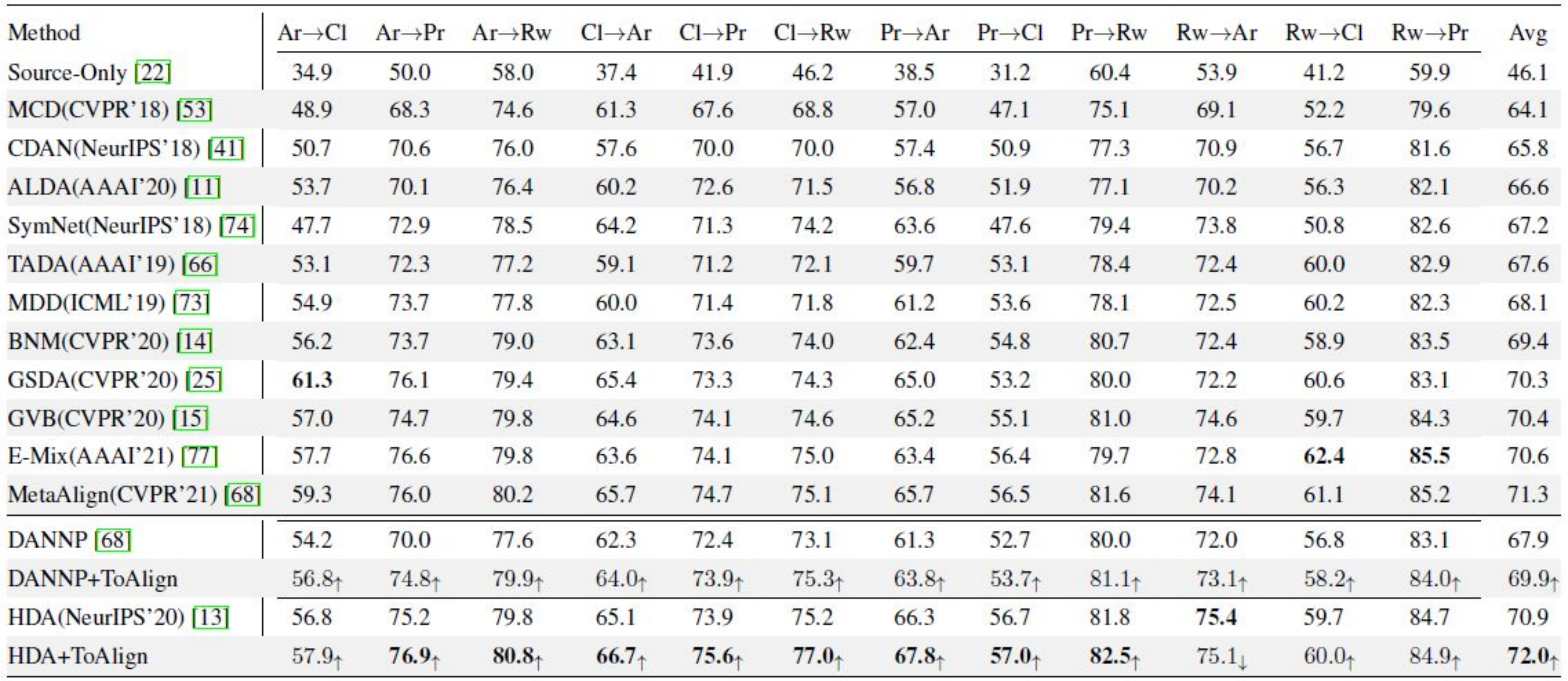
*Sample images from the **Office-Home** dataset. The dataset consists of images of everyday objects organized into 4 domains; **Art**: paintings, sketches and*or artistic depictions, **Clipart**: clipart images, **Product**: images without background and **Real-World**: regular images captured with a camera.

Visualization of Decomposed Features

భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

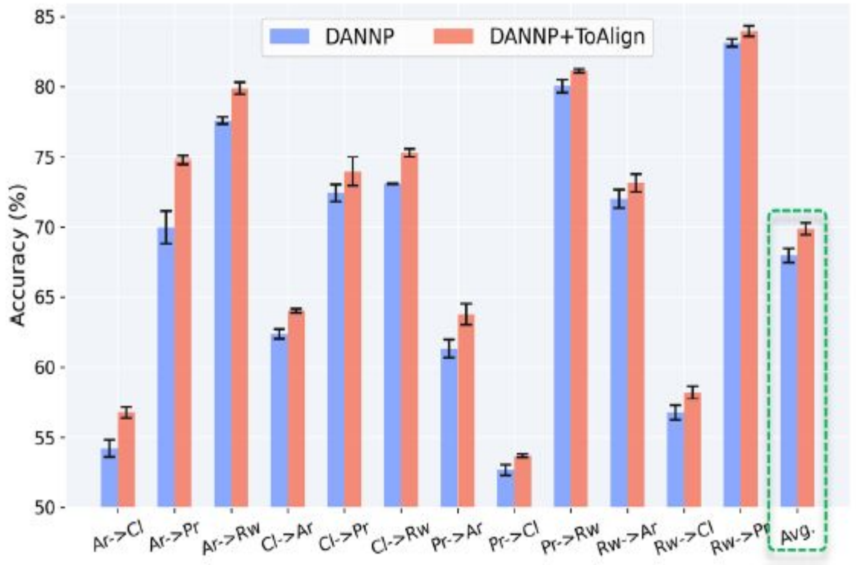


Results

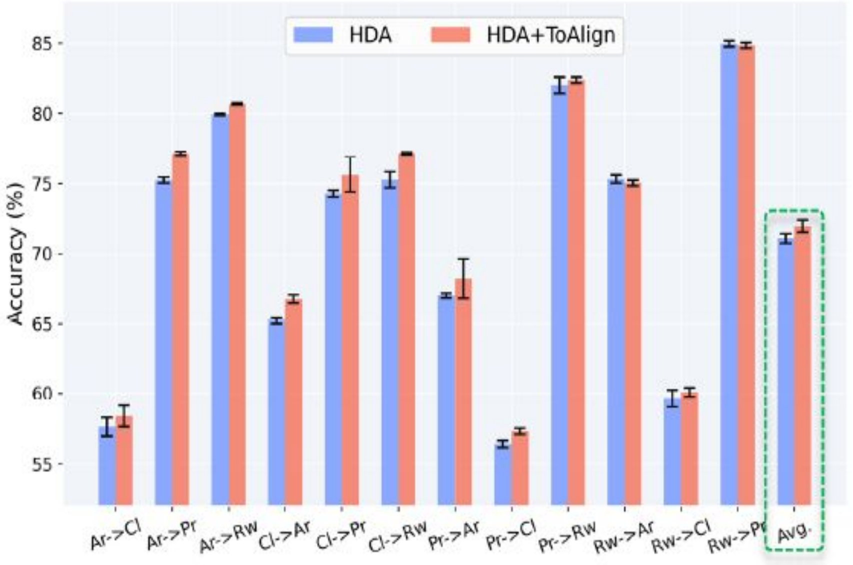


Method	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	Avg
Source-Only [22]	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
MCD(CVPR'18) [53]	48.9	68.3	74.6	61.3	67.6	68.8	57.0	47.1	75.1	69.1	52.2	79.6	64.1
CDAN(NeurIPS'18) [41]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	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$_\uparrow$	74.8$_\uparrow$	79.9$_\uparrow$	64.0$_\uparrow$	73.9$_\uparrow$	75.3$_\uparrow$	63.8$_\uparrow$	53.7$_\uparrow$	81.1$_\uparrow$	73.1$_\uparrow$	58.2$_\uparrow$	84.0$_\uparrow$	69.9$_\uparrow$
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$_\uparrow$	**76.9**$_\uparrow$	**80.8**$_\uparrow$	**66.7**$_\uparrow$	**75.6**$_\uparrow$	**77.0**$_\uparrow$	**67.8**$_\uparrow$	**57.0**$_\uparrow$	**82.5**$_\uparrow$	75.1$_\downarrow$	60.0$_\uparrow$	84.9$_\uparrow$	**72.0**$_\uparrow$

Table 1: Accuracy (%) of different UDAs on Office-Home with ResNet-50 as backbone. Best in bold.

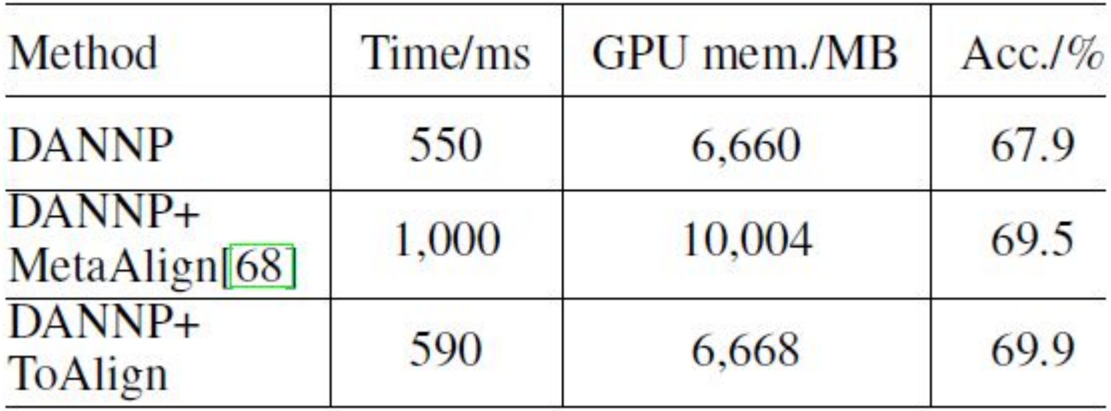


(a) *DANNP* and *DANNP+ToAlign*



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

Error bars of ToAlign on top of DANNP and HDA on Office-Home



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

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

Observations

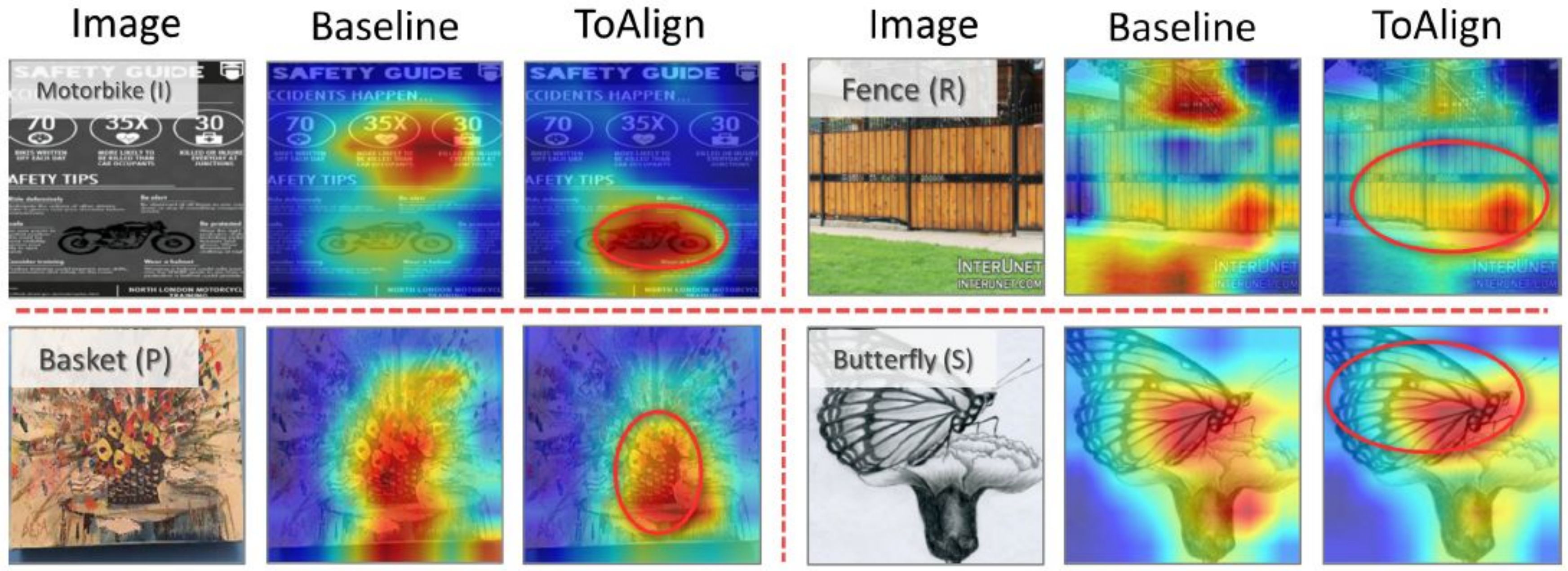
- ToAlign + HDA outperforms all previous method and achieves state of art performance, i.e; it outperforms HDA by **0.9%.**

- Occupies almost the same GPU memory as the baseline, which is much smaller than that of DANNP+MetaAlign.

- 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.

Visualization on target test images



Visualization of the feature response maps on target test images

References

భారతీయ సాంకేతిక విజ్ఞాన సంస్థ హైదరాబాద్
भारतीय प्रौद्योगिकी संस्थान हैदराबाद
Indian Institute of Technology Hyderabad

1. Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In ICML, pages 1180–1189, 2015.
2. Z. Pei, Z. Cao, M. Long, and J. Wang. Multi-adversarial domain adaptation. In AAAI, volume 32, 2018.
3. S. Cui, X. Jin, S. Wang, Y. He, and Q. Huang. Heuristic domain adaptation. In NeurIPS, 2020.
4. G. Wei, C. Lan, W. Zeng, and Z. Chen. Metaalign: Coordinating domain alignment and classification for unsupervised domain adaptation. In CVPR, 2021.
5. Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky.
6. Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(1):2096–2030, 2016.

THANK YOU