MILA: Memory-Based Instance-Level Adaptation for Cross-Domain Object Detection

Onkar Krishna[1]
onkar.krishna.vb@hitachi.com
Hiroki Ohashi[1]
hiroki.ohashi.uo@hitachi.com
Saptarshi Sinha[2]
saptarshi.sinha@bristol.ac.uk

[1] Hitachi Ltd.
Tokyo, Japan

[2] University of Bristol
Bristol, UK

Abstract

Cross-domain object detection is challenging, and it involves aligning labeled source and unlabeled target domains. Previous approaches have used adversarial training to align features at both image-level and instance-level. At the instance level, finding a suitable source sample that aligns with a target sample is crucial. A source sample is considered suitable if it differs from the target sample only in domain, without differences in unimportant characteristics such as orientation and color, which can hinder the model's focus on aligning the domain difference. However, existing instance-level feature alignment methods struggle to find suitable source instances because their search scope is limited to mini-batches. The insufficient diversity of mini-batches becomes problematic particularly when the target instances have high intra-class variance. To address this issue, we propose a memory-based instance-level domain adaptation framework. Our method aligns a target instance with the most similar source instance of the same category retrieved from a memory storage. Specifically, we introduce a memory module that dynamically stores the pooled features of all labeled source instances, categorized by their labels. Additionally, we introduce a simple yet effective memory retrieval module that retrieves a set of matching memory slots for target instances. Our experiments on various domain shift scenarios demonstrate that our approach outperforms existing non-memory-based methods significantly. Code is available at https://github.com/hitachi-rd-cv/MILA

1 Introduction

Although recent object detection models have achieved success on public datasets [5, 7, 18], they often suffer from a drop in performance when deployed in real-world use cases due to their inability to automatically generalize to unseen target environments. Retraining models on the target is a possible solution, but it is not viable due to the high cost of annotations.

To address this issue, Unsupervised Domain Adaptation (UDA) leverages knowledge transfer from a labeled source domain to an unlabeled target domain. The previous UDA approaches for object detection have focused on aligning the instance-level features extracted

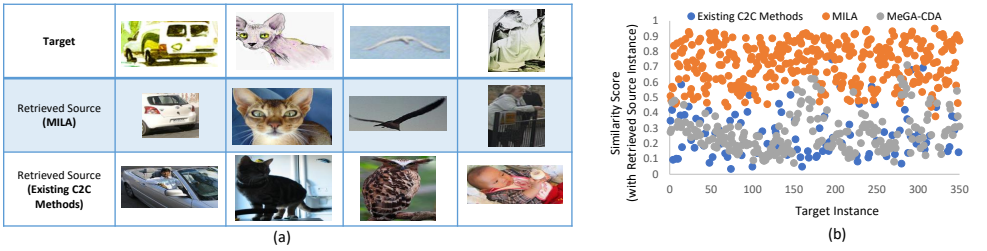

Figure 1: (a) Examples of retrieved source instances by MILA and the C2C method for alignment. Our approach retrieves source instances with similar visual characteristics compared to existing C2C methods. (b) Comparison of similarity scores among selected cross-domain pairs for alignment using different methods reveals that MILA aligns a target instance only with a source instance that has a very high similarity score.

from source and target object proposals, which are intermediately generated by the detector model. However, most previous works [8, 21] ignore the category of the instances during the alignment, leading to negative knowledge transfer. To solve this, some recent works [29, 34, 36, 38] proposed category-to-category (C2C) alignment methods. They employ various techniques to identify instances with matching categories in a sampled mini-batch of source and target images, and then align these instances using either adversarial [21, 30] or contrastive learning [34, 36, 38].

Even though C2C methods achieve better performance than vanilla instance alignment methods, we argue that aligning a target instance in a category to an arbitrary source instance in the same category is sub-optimal. We claim that it is important to align a target instance in a category to the source instance in the same category that is similar in terms of the *non-defining* visual characteristics. We assume that the characteristics of an image in the view of the object detection task are divided into three groups: *defining characteristics*, which are indispensable for defining a category (*e.g.*, shape), *non-defining characteristics*, which can be different and diverse within a category (*e.g.*, orientation, color), and *domain specific characteristics*, which can be different within a category but shared within a domain (*e.g.*, style). In the domain adaptation process, it is important to focus on aligning domain specific features and aligning non-defining features are not necessary. By finding a source instance that has similar non-defining characteristics with a target instance, a model is not disturbed by the unimportant differences and can focus solely on the difference in domains. The existing C2C methods often struggle to find a suitable source instance because they search a matching instance only within a mini-batch, which does not necessarily contain a suitable source instance since mini-batches are usually small in size and thus tend to lack diversity of the samples (see Fig. 1).

To address this issue, we propose memory-based instance-level adaptation (MILA). We design MILA in such a way that it can learn alignment from the *'reliable'* matching pairs. A pair of source and target instance is regarded as *reliable* when *(i)* their features are expected to well represent the categories and *(ii)* these features are similar enough so that a model can focus on domain differences. To increase the chance of finding reliable matching pairs, MILA has four unique characteristics. (1) MILA has a memory module for storing source features, which is much larger than mini-batch and thus greatly increases chance to find a suitable source instance for the alignment (contribution to *(ii)*). (2) MILA stores source features only when the model can correctly predict the category of the source instance using

the features. This is to guarantee the quality of the stored features (contribution to *(i)*). (3) MILA uses a target instance for the alignment only when the category prediction confidence is sufficiently high. This is to guarantee the quality of the target features (contribution to *(i)*). (4) MILA assigns different weights to different matching pairs according to their similarities to emphasize more reliable pairs (contribution to *(ii)*).

Our main contributions are summarized into 3 points. (1) We are the first, to the best of our knowledge, to argue the importance of finding 'reliable' pairs for the domain-adaptive object detection (DAOD) task, and provide with the empirical evidence for it (Fig. 1). (2) We propose a dedicated design based on the memory module for increasing the chance of finding 'reliable' pairs. (3) We verify the effectiveness of the proposed method and its four characteristics by extensive experiments. It achieved state-of-the-art results on five cross-domain object detection tasks, with significant relative improvements of 5.5%, 4.1%, and 4.0% on the *Sim10k*, *Comic2k* and *Watercolor2k* benchmark datasets, respectively.

2 Related Works

Domain Adaptive Object Detection (DAOD). DA-Faster [3] proposed an early DAOD method that performs feature alignment at both image-level and instance-level. MAF [8] and [32] extended this idea to multi-layer feature adaptation of backbone network. SWDA [23] suggests that aligning local features strongly is more effective than aligning global features strongly. CRDA [33] and MCAR [37] introduce a multi-label classifier upon backbone network to regularize the features. Recent methods [9, 16, 27, 30, 34, 36, 39] align instance-level features from object proposals using category-aware manner (C2C). They derive prototype representation of each category by aggregating multiple instances before inducing alignment. However, this causes loss of intra-class variance information and leads to suboptimal prototypes for alignment.

Memory Networks. The memory network [28, 31] is a type of neural network module that utilizes an external memory to store and retrieve relevant information. It has been widely utilized in vision-related tasks, such as video object segmentation [20, 22], movie understanding [19], and visual tracking [35] due to its ability to retain diverse knowledge types.

Memory networks have also been used in domain adaptation [14], and DAOD [30]. The closest to our work is MeGA-CDA [30], which utilizes memory modules for storing class-prototypes and use them to create category-specific attention maps for better C2C alignment between source and target instances. Although MeGA-CDA is similar to MILA in a sense that both use memory module, their motivations differ significantly. MeGA-CDA is the method dedicated for C2C alignment and it aims to find source regions that correspond to a particular category within a limited search scope of each mini-batch, while MILA is designed to find most 'reliable' source instance of a category for the alignment and exploit more from reliable pairs. In other words, MeGA-CDA cares only categories, while MILA also cares specific instances in addition to categories. MILA has many unique characteristics to increase the chance of finding reliable matching pairs, as mentioned in the introduction. In fact, sometimes MeGA-CDA fails to find the appropriate source regions to match with a given target sample because there is no guarantee that a mini-batch always contains an object of a particular category. In contrast, MILA can store wide variety of source instances of all the categories in the memory and therefore can always find a source instance of the same category as a given target instance.

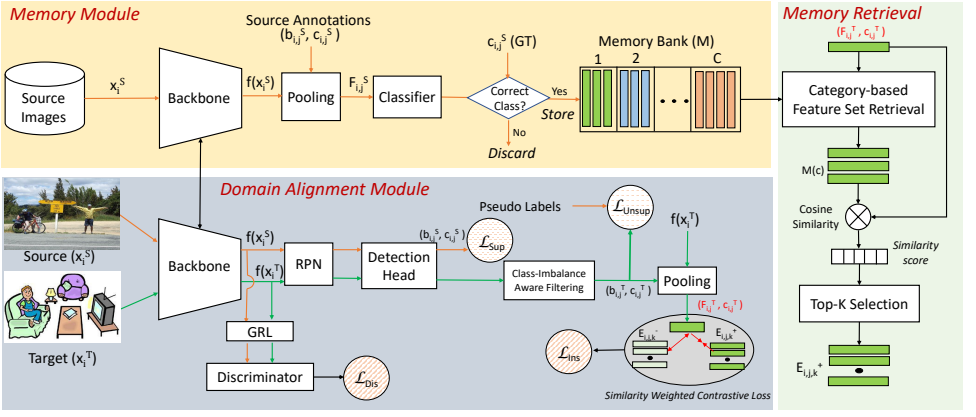

Figure 2: Network Overview: Mainly consist of memory module, similarity-based memory retrieval module, and instance-level domain adaptation module

3 Method

3.1 Problem Formulation

Given a labeled source dataset $D_S = \{x_i^S, y_i^S\}_{i=1}^{N_S}$ and an unlabeled target dataset $D_T = \{x_i^T\}_{i=1}^{N_T}$, the task of UDA is to transfer knowledge from D_S to D_T and predict accurate labels for D_T. Even though D_S and D_T have the same label space, they come from very different data distributions, which makes UDA a very challenging task. Since we focus on UDA for object detection, $y_i^S = \{b_{i,j}^S, c_{i,j}^S\}_{j=1}^{K_i^S}$ contains bounding box labels $b_{i,j}^S \in [0,1]^4$ and category labels $c_{i,j}^S \in \{1,\ldots,C\}$ for K_i^S objects in image x_i^S, where there are C different categories of objects.

3.2 Overview

To overcome the limitations of existing C2C alignment methods discussed in Sec. 1, we present MILA. Fig. 2 illustrates our proposed approach, which is built on Faster R-CNN, following prior works [17, 23, 34]. MILA includes three main modules: (1) a *memory module* that stores instance-level features and category information from previously-seen source images, (2) a *memory retrieval module* that retrieve most 'reliable' source instances of the same category as a given target instance feature, and (3) a *similarity-weighted alignment* that scales target instance alignment based on similarity to retrieved source instances. Detailed descriptions of these modules are provided in the following sections. We consider the two-stage Faster-RCNN to be made up of a feature extractor $f(;\theta)$ followed by a region-proposal network $rpn(;\phi)$ and a detection head $det(;\psi)$, where θ, ϕ and ψ are learnable parameters.

3.3 Memory Module

The memory module is used to save the instance-level features of source images extracted from the Faster R-CNN backbone in a memory bank. Specifically, for a source image x_i^S, f computes the feature map as $f(x_i^S;\theta)$. The features for j^{th} instance in the image can be extracted from $f(x_i^S;\theta)$ by pooling corresponding instance bounding box $b_{i,j}^S$ available in y_i^S.

Therefore,

$$\hat{F}^S_{i,j} = roipool\left(f(x^S_i;\theta), b^S_{i,j}\right) \ , \tag{1}$$

where *roipool* is the conventional region-of-interest (ROI) pooling function used in Faster R-CNN. The extracted instance features $\hat{F}^S_{i,j}$ is then filtered based on the classification accuracy as follows:

$$F^S_{i,j} = \begin{cases} \hat{F}^S_{i,j} & \text{if } \hat{c}^S_{i,j} = c^S_{i,j} \\ discard & \text{otherwise} \end{cases} \tag{2}$$

where $\hat{c}^S_{i,j}$ is the predicted class of the source proposal $\hat{F}^S_{i,j}$ of original class label $c^S_{i,j}$. The filtered instance features $F^S_{i,j}$ are category-wise stored in the memory bank, which is created as follows:

$$M(c) = \left\{ \left\{ \mathbf{I}\left(c^S_{i,j} = c\right) F^S_{i,j} \right\}_{j=1}^{K^S_i} \right\}_{i=1}^{N_S} \ , \tag{3}$$

where $\mathbf{I}$ denotes the indicator function and $M(c)$ denotes all the stored instance features of category c, where $1 \leq c \leq C$.

During training, as θ gets updated and f extracts more domain-aligned features, we propose dynamically updating the memory to store high-quality representations. Note that the memory M is only created during training.

3.4 Similarity-Based Memory Retrieval Module

This module's objective is to retrieve 'reliable' source instances that have similar features to the target instance, allowing the model to focus on domain differences. The retrieval process is structured as follows: (1) Predicting the bounding boxes and their categories in the target image, and filtering out the inaccurate predictions. (2) Extracting the instance-level features from the predicted bounding boxes in the target images. (3) Retrieving the top-K similar source instance features from memory M that correspond to each target instance feature.

For i^{th} target image x^T_i, the feature map is generated as $f(x^T_i;\theta)$. This feature map is then used by $rpn(;\phi)$ and $det(;\psi)$ to predict $\hat{y}^T_i$, which has K^T_i instances with bounding box, category, and confidence score, i.e.,

$$\hat{y}^T_i = \{b^T_{i,j}, c^T_{i,j}, s^T_{i,j}\}_{j=1}^{K^T_i} = det(rpn(f(x^T_i;\theta);\phi);\psi) \tag{4}$$

To ensure MILA's effectiveness, it's crucial to filter out noisy predictions as they can hamper performance. Existing methods [17] use a fixed threshold δ for all classes to remove predictions with confidence scores ($s^T_{i,j}$) below δ. However, due to class imbalance in the training data, lower confidence scores are often produced for underrepresented classes, making a fixed threshold impractical. To address this challenge, we assign category-specific thresholds δ_c based on the detection accuracy of each category in the source dataset inspired by [26]. Note before filtering, we perform non-maximum suppression (NMS) σ to remove duplicate bounding boxes, i.e.,

$$\tilde{y}^T_i = \left\{ \sigma\left(b^T_{i,j}, c^T_{i,j}, s^T_{i,j}\right) \middle| s^T_{i,j} \geq \delta_c \right\}_{j=1}^{j=K^T_i} \ , \tag{5}$$

Once we have the filtered bounding boxes in $\tilde{y}^T_i$, we can extract instance-level features from $f(x^T_i;\theta)$ by pooling from the predicted bounding boxes $b^T_{i,j}$ (j^{th} box for i^{th} target image) as

$$F^T_{i,j} = roipool(f(x^T_i;\theta), b^T_{i,j}) \ , \tag{6}$$

From memory M, we select the source features to be used for the alignment with $F^T_{i,j}$ from the features that has the same category as the predicted category $c^T_{i,j}$. The cosine similarity scores of $F^T_{i,j}$ with each source feature of the same category is computed as

$$S(F^T_{i,j}, M(c^T_{i,j})_z) = \frac{F^T_{i,j}.M(c^T_{i,j})_z}{\| F^T_{i,j} \| \| M(c^T_{i,j})_z \|}, \tag{7}$$

where $M(c)_z$ is the z^{th} source instance feature stored in $M(c)$. Now, we can easily retrieve top-K most similar source instance features of the same category as the target instance $c^T_{i,j}$. We call them positive samples and denote by $E^+_{i,j,k}$, where $1 \leq k \leq K$ represents the index. Similarly, negative samples $E^-_{i,j,k}$ are obtained by randomly selecting one sample from each of the categories that are different from the category of the positive pairs.

3.5 Similarity-weighted Instance-level Domain Adaptation

Once we retrieve positive and negative set of instances from source memory for a given target instance, we align them by applying a specially designed max-margin contrastive losses. For a target instance feature $F^T_{i,j}$ and k-th positive sample $E^+_{i,j,k}$, contrastive loss enforces them to come closer in latent space, whereas pushes the features apart for negative samples $E^-_{i,j,k}$.

$$\mathcal{L}^+_{i,j} = \frac{1}{K}\sum_{k=1}^{K} S(F^T_{i,j}, E^+_{i,j,k}) d_{pos}, \quad \mathcal{L}^-_{i,j} = \frac{1}{C-1}\sum_{k=1}^{C-1} max(0, m - d_{neg}),$$
$$\mathcal{L}_{Ins} = \frac{1}{N^T}\sum_{i=1}^{N^T}\left(\frac{1}{K^T_i}\sum_{j=1}^{K^T_i}\mathcal{L}^+_{i,j} + \mathcal{L}^-_{i,j}\right). \tag{8}$$

Where d_{pos} and d_{neg} are the Euclidean distances of $F^T_{i,j}$ with $E^+_{i,j,k}$ and $E^-_{i,j,k}$, respectively. Recall that cardinality of the negative sample sets is $C-1$, i.e., $|E^-_{i,j,k}| = C-1$.

3.6 Overall Objective

In addition to the instance-level alignment loss discussed earlier, we use supervised loss $\mathcal{L}_{Sup}$, unsupervised loss $\mathcal{L}_{Unsup}$, and discriminator loss $\mathcal{L}_{Dis}$. $\mathcal{L}_{Sup}$ is the object detection loss optimized using labeled data from the source domain. $\mathcal{L}_{Unsup}$ is the object detection loss computed on target domain images with pseudo labels generated using a similar approach as in [17]. $\mathcal{L}_{Dis}$ is the loss of an image-level binary domain discriminator used in [8, 32]. $\mathcal{L}_{Dis}$ is computed by a domain discriminator D whose aim is to discriminate where the backbone feature is from ($d = 0$) or target ($d = 1$). $\mathcal{L}_{Dis}$ is formulated as follows:

$$\mathcal{L}_{Dis} = \frac{1}{(N^T + N^S)}\sum_{i=1}^{N^T+N^S} -d\log D(f(x_i;\theta)) - (1-d)\log(1 - D(f(x_i;\theta))),$$

Note that the gradients obtained from this loss term are used to update not only the parameters of the discriminator D, but also reversed by the gradient reversal layer (GRL) during

Method	bicycle	bird	car	cat	dog	person	mAP
Source	32.5	12.0	21.1	10.4	12.4	29.9	19.7
DA-Faster [8]	31.1	10.3	15.5	12.4	19.3	39.0	21.2
SWADA [23]	36.4	21.8	29.8	15.1	23.5	49.6	29.4
MCAR [35]	49.7	20.5	37.4	20.6	24.5	53.6	33.5
D-Adapt [23]	52.4	25.4	42.3	**43.7**	25.7	53.5	40.5
Ours	**59.1**	**28.5**	**49.8**	28.3	**35.7**	**66.3**	**44.6**
Oracle	44.2	35.3	31.9	46.2	40.9	70.9	44.6

Table 1: Results on the Comic2k test set for **Pascal VOC→Comic2k** adaptation (ResNet-101).

Method	bicycle	bird	car	cat	dog	person	mAP
Source	84.2	44.5	53.0	24.9	18.8	56.3	46.9
SCL [35]	82.2	55.1	51.8	39.6	38.4	64.0	55.2
SWADA [23]	82.3	55.9	46.5	32.7	35.5	66.7	53.3
UMT [8]	88.2	55.3	51.7	39.8	43.6	69.9	58.1
AT [35]	94.3	57.2	57.2	34.2	36.9	78.5	59.7
Ours	**97.4**	**59.0**	**58.3**	**40.6**	**47.8**	**79.3**	**63.7**
Oracle	51.8	49.7	42.5	38.7	52.1	68.6	50.6

Table 2: Results on the Watercolor2k test set for **Pascal VOC→Watercolor2k** adaptation (ResNet-101).

backpropagation to update θ. This helps $f(;\theta)$ to learn discriminative features that can confuse D. Combining altogether, the overall objective function is as follows:

$$\mathcal{L} = \mathcal{L}_{Sup} + \lambda_1 \mathcal{L}_{Unsup} + \lambda_2 \mathcal{L}_{Dis} + \lambda_3 \mathcal{L}_{Ins}, \tag{9}$$

where λ_1, λ_2 and λ_3 are the hyperparameters to control the weight of each loss.

4 Experiments

4.1 Datasets

We performed extensive experiments on seven publicly available datasets covering five scenarios of domain shift. The datasets are Pascal VOC [7], Clipart1k [11], Watercolor2k [11], Comic2k [11], Sim10k [13], Cityscapes [4], and FoggyCityscapes [24]. Pascal VOC comprises 16,551 images of 20 categories of common objects from the real world. Clipart1k contains 1k comical images with the same 20 categories as Pascal VOC. Watercolor2k includes 1k training and 1k testing watercolor-style images, sharing six categories with Pascal VOC. Similarly, Comic2k consists of 1k training and 1k test images, sharing six categories with Pascal VOC. Sim10k contains 10,000 images with 58,701 bounding boxes of car categories, while both Cityscapes and FoggyCityscapes comprise 2,975 training images and 500 validation images with eight object categories. We evaluate the domain adaptation performance of different methods using the standard setting [17, 23] on the following five domain adaptation tasks: (i) Pascal VOC→Comic2k, (ii) Pascal VOC→Watercolor2k, (iii) Pascal VOC→Clipart1k, (iv) Sim10k→Cityscapes, and (v) Cityscapes→FoggyCityscapes. Due to space constraints, we provide results for Pascal VOC→Clipart1k in supplementary materials.

4.2 Implementation Details

We adopt the Faster R-CNN model with either ResNet-101 or VGG16 architectures and implement it using Detectron2, following the approach in [17, 23, 29, 34]. We fine-tune the parameters of ResNet-101 from the model pre-trained on ImageNet [5], while VGG16 parameters are learned from scratch. The images are scaled by resizing the shorter side to 600 pixels while maintaining the aspect ratios, following the common practice [8]. We apply a set of strong and weak data augmentations as described in [17]. Our evaluation metric is the average precision (AP) for each class and the mean AP (mAP) over all classes. For the hyperparameters, we set λ_1 to 1.0, λ_2 to 0.1, and λ_3 to 0.1 unless otherwise stated. Margin m in eq. (8) is set to 1.0. We update the memory after every $1/3$ of an epoch and use

Method	bus	bicycle	car	mcycle	person	rider	train	truck	mAP
Source (F-RCNN)	20.1	31.9	39.6	16.9	29.0	37.2	5.2	8.1	23.5
SCL [25]	41.8	36.2	44.8	33.6	31.6	44.0	40.7	30.4	37.9
DA-Faster [3]	35.3	27.1	40.5	20.0	25.0	31.0	20.2	22.1	27.6
SCDA [39]	39.0	33.6	48.5	28.0	33.5	38.0	23.3	26.5	33.8
SWDA [23]	36.2	35.3	43.5	30.0	29.9	42.3	32.6	24.5	34.3
DM [15]	38.4	32.2	44.3	28.4	30.8	40.5	34.5	27.2	34.6
MTOR [1]	38.6	35.6	44.0	28.3	30.6	41.4	40.6	21.9	35.1
MAF [8]	39.9	33.9	43.9	29.2	28.2	39.5	33.3	23.8	34.0
iFAN [24]	45.5	33.0	48.5	22.8	32.6	40.0	31.7	27.9	35.3
CRDA [33]	45.1	34.6	49.2	30.3	32.9	43.8	36.4	27.2	37.4
HTCN [2]	47.4	37.1	47.9	32.3	33.2	47.5	40.9	31.6	39.8
UMT [6]	56.5	37.3	48.6	30.4	33.0	46.7	46.8	34.1	41.7
AT [17]	60.0	49.0	63.6	38.8	45.0	**53.9**	45.1	33.9	49.0
Ours	**61.4**	**51.5**	**64.8**	**39.7**	**45.6**	**52.8**	**54.1**	**34.7**	**50.6**
Oracle (F-RCNN)	50.3	40.7	61.3	32.5	43.1	49.8	35.1	28.6	42.7

Table 3: Results on the FoggyCityscapes test set for **Cityscapes** → **Foggy Cityscapes** adaptation (VGG-16).

Method	Backbone	AP on Car
DA-Faster [3]	VGG-16	38.9
BDC-Faster [23]		31.8
SWADA [23]		40.1
MAF [8]		41.1
SCDA [39]		43.0
CDN [16]		49.3
MeGA-CDA [30]		44.8
UMT [6]		43.1
Source	ResNet-101	41.8
CADA [10]		51.2
D-adapt [12]		51.9
Ours		**57.4**
Oracle		70.4

Table 4: Results on the Cityscapes test set for **Sim10k** → **Cityscapes** adaptation.

stochastic gradient descent (SGD) with momentum 0.9 and a learning rate of 0.04 throughout the training stage, without any learning rate decay. We build on the code provided by the authors of [17] and follow their hyperparameter settings. The experiments are conducted on 4 Nvidia GPU V100 with a batch size of 8 or 4 depending on the dataset, using PyTorch.

4.3 Comparison with state-of-the-arts

We compare the proposed MILA with the state-of-the-art DAOD methods, including SCL [25], SWADA [23], DM [15], CRDA [33], HTCN [2], DA-Faster [3], MCAR [37], D-Adapt [12], MAF [8], SCDA [39], CDN [16], MeGA-CDA [30], CADA [10], UMT [6], and Adaptive Teacher (AT) [17]. Our implementation, as described in Sec. 4.2, builds upon the official code of [17]. To ensure fairness, we compared our method with the best-reproduced results of AT, under the same conditions. The original results from [17] are in the supplementary material. 'Source' is the baseline model without domain adaptation, while 'Oracle' is trained and tested on the target domain.

Adaptation between dissimilar domains. In our first set of experiments, we use the Pascal VOC as the source domain and Comic2k as the target domain, which has a very different style from Pascal VOC and contains many small objects. Table 1 shows that MILA outperforms all the previous methods and improves mAP by 4.1 compared with the state-of-the-art.

In the next set of experiments, we evaluate MILA on Watercolor2k, another target domain with a unique style. As shown in Table 2, MILA achieves the highest average precision for all object categories, surpassing all previous methods by a significant margin. Additionally, MILA surpasses the oracle model on the Watercolor2k dataset by a considerable margin of 13.1%. These results consistently validate the effectiveness of aligning the most similar instances across domains in reducing the domain gap between different scenarios.

Adaptation between similar domains. The results of this setting are presented in Table 3. MILA achieves the highest mAP in majority of the categories. Upon closer inspection, we observed that MILA achieved the largest performance gain of +9.0% in 'train' class, which has the least number of instances, with only 504 training samples. This finding suggests that classes with fewer training instances specially benefit from the proposed memory module. We hypothesize that this is because it is generally more challenging to align less populated classes due to the difficulty of finding an appropriate alignment target. MILA helps to overcome this challenge by storing all the alignment targets in the memory.

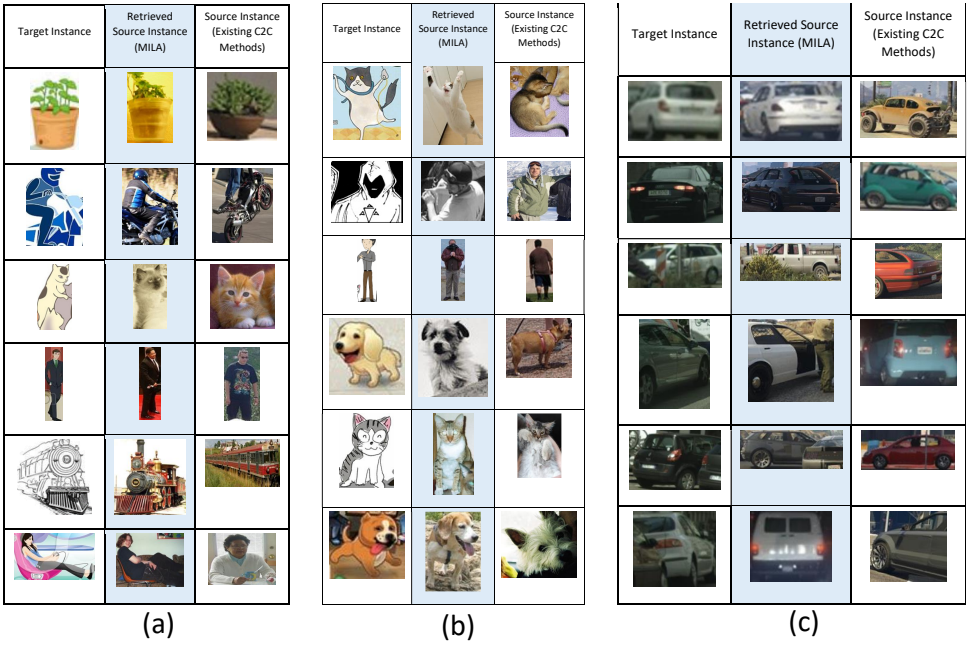

Figure 3: Visualization of instance pairs (a) Pascal VOC→Clipart1k (b) Pascal VOC→Comic2k (c) Sim10k→Cityscapes

Adaptation from synthetic to real images. We use Sim10k as the source domain and Cityscapes as the target domain. Following [12], we evaluate on the validation split of the Cityscapes and report the mAP on car. Table 4 shows that MILA scores the new state-of-the-art adaptation performance, achieving gain of 5.5 points on mAP.

4.4 Visualization of Instance Pairs

To visualize reliable matching pairs used in alignment, we present target instances retrieved by MILA in fig. 3. In the second example of fig. 3 (a), MILA retrieves a biker instance with a matching color scheme for the helmet and bike as the target instance. Similarly, in the fourth example (fig. 3 (a)), MILA successfully identifies a person wearing a matching dress, showcasing its ability to capture subtle visual details essential for effective domain adaptation.

On other datasets, MILA consistently demonstrates its capability to retrieve instances with remarkable visual similarities. For example, in fig. 3 (b), the first and fifth example demonstrate MILA's ability to retrieve instances of the cat category that not only share similar color and orientation but also exhibit an overall appearance that closely matches the target instances. In fig. 3 (c), MILA retrieve car instances with matching color and orientation, as evident in examples 1, 2, and 6 for rear-facing target cars and examples 3, 4, and 5 for side-facing cars. In contrast, existing C2C methods retrieve source instances of car that display significant differences in color and orientation when compared to the target instances.

These visual examples highlight MILA's exceptional capability in identifying source instances with similar non-defining visual characteristics as the target compared to existing

C2C methods. By focusing solely on domain differences and disregarding unimportant dissimilarities, our model achieves superior accuracy.

4.5 Ablation Study

mAP	*m*	*f*	*s*	*t*	*u*
44.6	✓	✓	✓	✓	✗
40.3	✗	✗	✗	✗	✓
42.6	✓	✗	✓	✓	✗
42.2	✓	✓	✗	✓	✗
43.9	✓	✓	✓	✗	✓

Table 5: Ablation study of different components.

In this section, we assess effectiveness of different components in our approach on the Pascal VOC → Comic2k dataset (see Tab. 5). We used our full model with all four characteristics discussed in Sec. 1 and turned off each component one by one to evaluate their impact on the detection accuracy of the model.

First, we evaluate the **memory module's effectiveness (*m*)** by comparing the performance of our model with and without the memory module. We used a non-memory based instance alignment scheme [34] instead and used the same contrastive loss function $\mathcal{L}_{Ins}$, but sampled positive and negative instances from the mini-batch instead of the memory module. Our results showed that MILA improves object detection accuracy by 4.3% compared to the model without the memory module (2^{nd} row), indicating the importance of the memory module in enhancing performance.
Secondly, we analyze the **effectiveness of source feature filtering (*f*)** to ensure the quality of the stored features in memory by checking their predicted category. We observed that the performance drops by 2.0 points (3^{rd} row) if we turn off this component and store all source features in memory without judging their quality.
Thirdly, we evaluate the **effectiveness of similarity weighting of contrastive loss (*s*)** (4^{th} row). We observe that if we use plain contrastive loss function, there is a drop in accuracy (2.2%), which suggests that weighting our loss function by similarity helps to mitigate negative knowledge transfer when the aligned features are highly dissimilar. As a result, the detection accuracy improves.
We also analyzed the **effectiveness of category-aware thresholding (*t*) and fixed thresholding (*u*)**. The model performed slightly better with category-imbalance aware thresholding, compared to the fixed thresholding.
In summary, our results demonstrate that all four components are critical for our approach, and turning them off results in decreased detection accuracy. Further analysis on the memory module and hyperparameters are available in the supplementary material.

5 Conclusion

In this paper, we propose a Memory-based Instance-Level Alignment (MILA) framework for cross-domain object detection. The proposed strategy with four unique characteristics enables the model in finding 'reliable' pairs for alignment in the domain-adaptive object detection (DAOD) task. As evident in the results, this can efficiently improve the adaptation of instances by only focusing on important visual characteristics that distinguish the two domains. Extensive experiments demonstrate that MILA achieves state-of-the-art performance for adapting object detectors on several benchmark datasets.

References

[1] Qi Cai, Yingwei Pan, Chong-Wah Ngo, Xinmei Tian, Lingyu Duan, and Ting Yao. Exploring object relation in mean teacher for cross-domain detection. In *CVPR*, 2019.

[2] Chaoqi Chen, Zebiao Zheng, Xinghao Ding, Yue Huang, and Qi Dou. Harmonizing transferability and discriminability for adapting object detectors. In *CVPR*, 2020.

[3] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *CVPR*, 2018.

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[5] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

[6] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *CVPR*, 2021.

[7] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[8] Zhenwei He and Lei Zhang. Multi-adversarial faster-rcnn for unrestricted object detection. In *ICCV*, 2019.

[9] Zhenwei He and Lei Zhang. Domain adaptive object detection via asymmetric tri-way faster-rcnn. In *ECCV*, 2020.

[10] Cheng-Chun Hsu, Yi-Hsuan Tsai, Yen-Yu Lin, and Ming-Hsuan Yang. Every pixel matters: Center-aware feature alignment for domain adaptive object detector. In *ECCV*, 2020.

[11] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *CVPR*, 2018.

[12] Junguang Jiang, Baixu Chen, Jianmin Wang, and Mingsheng Long. Decoupled adaptation for cross-domain object detection. *arXiv preprint arXiv:2110.02578*, 2021.

[13] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? *arXiv preprint arXiv:1610.01983*, 2016.

[14] Tarun Kalluri, Astuti Sharma, and Manmohan Chandraker. MemSAC:Memory Augmented Sample Consistency for Large Scale Domain Adaptation. In *ECCV*, 2022.

[15] Taekyung Kim, Minki Jeong, Seunghyeon Kim, Seokeon Choi, and Changick Kim. Diversify and match: A domain adaptive representation learning paradigm for object detection. In *CVPR*, 2019.

[16] Congcong Li, Dawei Du, Libo Zhang, Longyin Wen, Tiejian Luo, Yanjun Wu, and Pengfei Zhu. Spatial attention pyramid network for unsupervised domain adaptation. In *ECCV*, 2020.

[17] Yu-Jhe Li, Xiaoliang Dai, Chih-Yao Ma, Yen-Cheng Liu, Kan Chen, Bichen Wu, Zijian He, Kris Kitani, and Peter Vajda. Cross-domain adaptive teacher for object detection. In *CVPR*, 2022.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014.

[19] Seil Na, Sangho Lee, Jisung Kim, and Gunhee Kim. A read-write memory network for movie story understanding. In *ICCV*, 2017.

[20] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019.

[21] Farzaneh Rezaeianaran, Rakshith Shetty, Rahaf Aljundi, Daniel Olmeda Reino, Shanshan Zhang, and Bernt Schiele. Seeking similarities over differences: Similarity-based domain alignment for adaptive object detection. In *ICCV*, 2021.

[22] Adrian Lopez Rodriguez and Krystian Mikolajczyk. Domain adaptation for object detection via style consistency. *arXiv preprint arXiv:1911.10033*, 2019.

[23] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Strong-weak distribution alignment for adaptive object detection. In *CVPR*, 2019.

[24] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Semantic foggy scene understanding with synthetic data. *IJCV*, 126(9):973–992, 2018.

[25] Zhiqiang Shen, Harsh Maheshwari, Weichen Yao, and Marios Savvides. Scl: Towards accurate domain adaptive object detection via gradient detach based stacked complementary losses. *arXiv*, 2019.

[26] Saptarshi Sinha, Hiroki Ohashi, and Katsuyuki Nakamura. Class-wise difficulty-balanced loss for solving class-imbalance. In *ACCV*, 2020.

[27] Peng Su, Kun Wang, Xingyu Zeng, Shixiang Tang, Dapeng Chen, Di Qiu, and Xiaogang Wang. Adapting object detectors with conditional domain normalization. In *ECCV*, 2020.

[28] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *NeurIPS*, 28, 2015.

[29] Kun Tian, Chenghao Zhang, Ying Wang, Shiming Xiang, and Chunhong Pan. Knowledge mining and transferring for domain adaptive object detection. In *ICCV*, 2021.

[30] Vibashan Vs, Vikram Gupta, Poojan Oza, Vishwanath A Sindagi, and Vishal M Patel. Mega-cda: Memory guided attention for category-aware unsupervised domain adaptive object detection. In *CVPR*, 2021.

[31] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[32] Rongchang Xie, Fei Yu, Jiachao Wang, Yizhou Wang, and Li Zhang. Multi-level domain adaptive learning for cross-domain detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[33] Chang-Dong Xu, Xing-Ran Zhao, Xin Jin, and Xiu-Shen Wei. Exploring categorical regularization for domain adaptive object detection. In *CVPR*, 2020.

[34] Minghao Xu, Hang Wang, Bingbing Ni, Qi Tian, and Wenjun Zhang. Cross-domain detection via graph-induced prototype alignment. In *CVPR*, 2020.

[35] Tianyu Yang and Antoni B Chan. Learning dynamic memory networks for object tracking. In *ECCV*, 2018.

[36] Yixin Zhang, Zilei Wang, and Yushi Mao. Rpn prototype alignment for domain adaptive object detector. In *CVPR*, 2021.

[37] Zhen Zhao, Yuhong Guo, Haifeng Shen, and Jieping Ye. Adaptive object detection with dual multi-label prediction. In *ECCV*, 2020.

[38] Yangtao Zheng, Di Huang, Songtao Liu, and Yunhong Wang. Cross-domain object detection through coarse-to-fine feature adaptation. In *CVPR*, 2020.

[39] Xinge Zhu, Jiangmiao Pang, Ceyuan Yang, Jianping Shi, and Dahua Lin. Adapting object detectors via selective cross-domain alignment. In *CVPR*, 2019.

[40] Chenfan Zhuang, Xintong Han, Weilin Huang, and Matthew Scott. ifan: Image-instance full alignment networks for adaptive object detection. In *AAAI*, 2020.