One Network for Multi-Domains: Domain Adaptive Hashing with Intersectant Generative Adversarial Networks

Tao He[1] , **Yuan-Fang Li**[1] , **Lianli Gao**[2] , **Dongxiang Zhang**[2] , **Jingkuan Song**[2*]

[1]Facult of Information Technology, Monash University

[2] Center for Future Media and School of Computer Science and Engineering, University of Electronic Science and Technology of China

{tao.he,yuanfang.li}@monash.edu, {lianli.gao, dongxiang.zhang}@uestc.edu.cn, jingkuan.song@gmail.com

Abstract

With the recent explosive increase of digital data, image recognition and retrieval have become a critical practical application. Hashing is an effective solution to this problem, due to its low storage requirement and high query speed. However, most of past works focus on hashing in a single (source) domain. Thus, the learned hash function may not adapt well in a new (target) domain that has a large distributional difference with the source domain. In this paper, we explore an end-to-end domain adaptive learning framework that simultaneously and precisely generates discriminative hash codes and classifies target domain images. Our method encodes images into a semantic common space, followed by two independent generative adversarial networks aiming at crosswise reconstructing two domain's images, reducing domain disparity and improving alignment in the shared space. We evaluate our framework on four public benchmark datasets, all of which show that our method is superior to the other state-of-the-art methods on the tasks of object recognition and image retrieval.

1 Introduction

With the explosive increase of digital data, the efficient retrieval (in terms of time) of images and storage (in terms of space) has become an increasingly important problem. Hashing-based techniques are a perfect approach to address this problem due to its high query speed and low storage cost. The main goal of hashing is to convert high-dimensional features into low-dimensional, discriminative and compact binary codes that preserve semantic information. Many efficient hash methods are based on deep neural networks. Most of them focus on single domain, like ITQ [Gong *et al.*, 2013], BA [Carreira-Perpiñán and Raziperchikolaei, 2015], and BDNN [Do *et al.*, 2016]. As a result, such hash methods only exhibit good performance on datasets that have little distribution difference with the training data.

Addressing this drawback, adaptive or cross-model hash techniques [Li *et al.*, 2018] have been proposed in recent years to deal with the domain shift problem. These methods are trained on a *source* dataset and then applied on a different *target* dataset, which may have large distributional variances with the source domain. In [Venkateswara *et al.*, 2017], the authors first proposed to use deep neural networks to learn representative hash codes, but they did not consider semantic information in the target domain and only applied a cross-entropy loss on the target domain. An end-to-end framework, DeDAHA [Long *et al.*, 2018a], was proposed to learn domain adaptive hash codes by two loss functions that preserve semantic similarity in hash codes. In [Li *et al.*, 2018], the authors leveraged two adversarial networks to maximize the semantic information of the representations between different modalities, but their network uses the label information to bridge the domain gap between two modalities. Although these methods have achieved high performance, they typically operate under a supervised setting, assuming the availability of labeled data in the target domain. However, in the real world, access to labeled data for the target domain may be very limited or entirely unavailable. Moreover, it is usually assumed that the true labels of the target domain are unavailable, making adaptive learning a more challenging problem.

Thus, more and more works focus on unsupervised domain adaption, like [Xie *et al.*, 2018]. The basic idea for domain adaptive learning is to embed the source and target domains into a common space so that both datasets in the latent layer have similar feature distributions. As for unsupervised domain adaption, one strategy is to train (under supervision) a classifier on the labeled source domain and then adapt it to a new domain. Some works [Hu *et al.*, 2018] focus on how to assign high-confidence pseudo labels to the target domain and treat those predicted labels as the ground truth of the target domain to fine-tune the model. In [Zhang *et al.*, 2018], the authors proposed a novel sample selection method aiming to select high-confidence labels. Another strategy is to project the source and target domain into a common space and reduce the domain disparity. Maximum Mean Discrepancy (MMD) has been widely used as a measure of variance between distributions in a reproducing-kernel Hilbert space. Subsequently, many works use MMD to achieve nonlinear alignment of domains. Deep Domain Confusion [Tzeng *et al.*, 2014] leverages domain confusion loss to learn a representation which is semantically meaningful and domain invariant.

Recently, generative Adversarial Networks (GAN) [Good-

*Contact Author

fellow *et al.*, 2014] was explored to generate new data having the same distribution with the input data. Later on, many works started to study how to apply adversarial networks into adaptive learning. In [Ganin and Lempitsky, 2015], the authors adopted an adversarial training mechanism to address the domain shift problem. The key component is the training of a discriminator that judges whether the feature comes from the source domain or the target domain. Specifically, the feature learning component tries to fool the discriminator so that it cannot distinguish the origin of the features. When the discriminator cannot differentiate the origins, it means that the domain disparity has been reduced to a relatively low level. The main drawbacks of those methods are that the discriminators can only judge the overall domain's distribution but cannot distinguish whether their subspaces in the common space is invariant or aligned. Inspired by this, we concentrate on how to leverage the discriminator to distinguish whether those subspaces are aligned.

In this paper, we propose a new, unsupervised method to tackle the above deficiencies of existing cross-domain hash techniques. Our method simultaneously addresses the tasks of recognition and retrieval in a unified network. In summary, our main contributions are threefold.

1. We develop a novel end-to-end transfer learning network which can not only learn the semantic hash codes for the unlabeled target domain but also predict labels of the new domain.
2. We employ two identical, but separate generative adversarial networks (GAN) to reduce source and target domain difference. Each of them independently generates data for both domains from the common space in which label information is preserved.
3. We evaluate our method on four public benchmark datasets. Our results strongly demonstrate that our method outperforms the other state-of-the-art unsupervised methods, Duplex, CDAN-M, I2I, etc, in both object classification and image retrieval.

2 Methods

Let $D_s = \{(x_i^s, y_i^s)\}_{i=1}^{n_s}$ denote the source domain, where $y_i^s \in \{1, \ldots, N\}$ is the *label* of x_i^s. $D_t = \{x_i^t\}_{i=1}^{n_t}$ denotes the target domain. Note that the target domain D_t is unlabeled. Our goal is to train a hash function in the source domain data and then test it in the target domain.

The high-level architecture of our framework is shown in Figure 1. Our model consists of three components: (1) one shared encoder network $u = E(x)$, (2) two independent generators, denoted as G^s for the source domain and G^t for the target domain, and (3) two distinct discriminators D^s and D^t, one for each domain. The shared encoder learns a common feature space for the two domains. Each generator generates images in both domains. Each discriminator judges whether an image comes from the source domain or the target domain.

2.1 Semantic Common Space Learning

As shown in Figure 1, the semantic common space layer bridges the gap between the encoder and the generators. In this section, we will split our framework into three parts and illustrate them in detail.

Supervised Hashing for Source Data

Supervised hashing focuses on how to preserve the label similarity information into compact binary codes. In this work, we choose pairwise loss [Song *et al.*, 2018] as our supervised hashing function:

$$L_h = \min_{W^E}\left(\frac{1}{2}\sum_{s_{ij}\in S^s}\left(\frac{1}{d}{b_i}^T b_j - s_{ij}\right)^2\right) \quad (1)$$

where W^E is the set of parameters of the encoder, b_i, b_j are binary codes, $S^s \in \{-1, 1\}$ is the similarity matrix constructed from ground-truth labels of the source domain D_s, and d is the length of hash code. Specifically, if two points have the same label, their similarity is defined as 1 and otherwise -1. By optimizing Equation 1, the hash function minimizes the feature distance of images with the same category but maximizes the distance across different categories. Unfortunately, hash codes b_i, b_j are discrete and cannot be minimized directly. So we relax the hashing function into a continuous closed interval $[-1, 1]$ and use u_i to approximate the binary code b_i, where u_i is the output of the last layer of network. We use $tanh(\cdot)$ as activation function to compute u_i. The updated loss function is given in Equation 2:

$$\begin{aligned} L_h = &\min_{W^E}\left(\frac{1}{2}\sum_{s_{ij}\in S^s}\left(\frac{1}{d}u_i^T u_j - s_{ij}\right)^2 + \right. \\ &\left. \upsilon\frac{1}{2}\sum\left(u_i - sign(u_i)\right)^2\right) \end{aligned} \quad (2)$$

where the second term is the relaxation term aiming at reducing quantization error. In our experiments, we set a very small number for υ.

Semantic Centroid Alignment

As noted in the introduction, the supervised hash loss learned in the source domain is inapplicable to the target domain in cross-domain hashing. To solve this problem, we proposed a *semantic centroid alignment loss* to handle it. Especially, through our semantic centroid alignment loss, we force the target domain to have a cluster center distribution similar to that of the source domain.

In fact, we can view Equation 2 as a clustering process: if two images have the same class label, they should have a small hash distance (i.e. Hamming distance), otherwise they should have a large hash distance. Consequently, a small hash distance means that their features should belong to the same cluster in the feature space whereas a large hash distance implies the they should belong in different clusters. Thus, if the two domains have similar cluster center distributions in the learned common semantic space, the hash function trained on the source domain should be applicable on the target domain. We adopt the K-means algorithm as the clustering algorithm on the two domains. The formulation is showing as follows:

$$L_s = \min_{W^E}\left(\sum_{i=1}^{N}\varphi\left(\frac{1}{m_i}\sum\nolimits_{y^s=i} x^s, \frac{1}{k_i}\sum\nolimits_{y^{\tilde{t}}=i} x^t\right)\right) \quad (3)$$

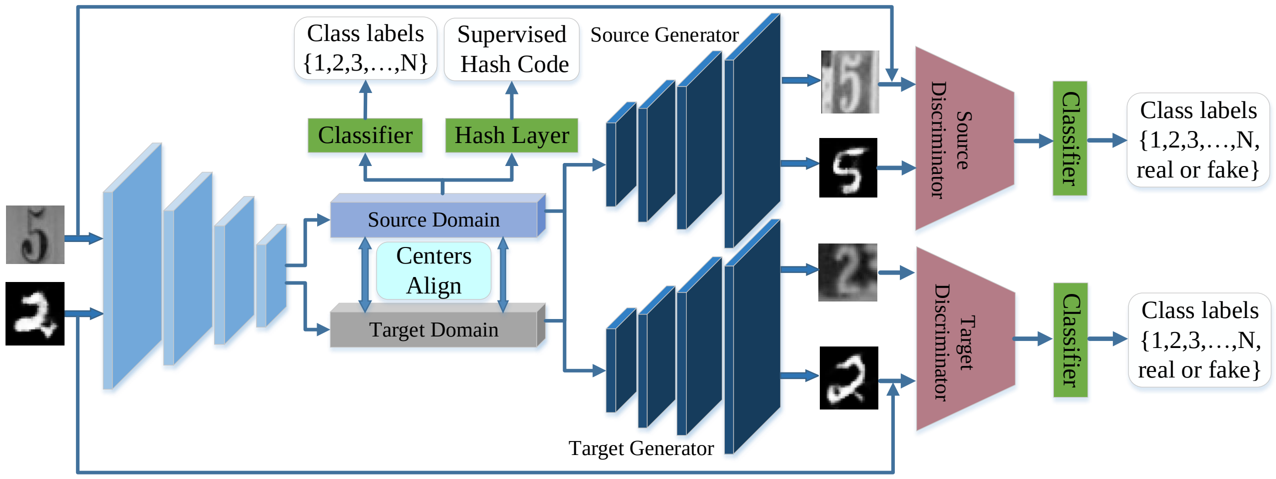

Figure 1: The overview of our framework, which is consisted of five networks: encoder, two independent generators and two distinct discriminators. Two generators are responsible for reconstructing two domain's data and discriminators aim at recogizing their labels.

where N is the number of classes and m_i (resp. k_i) denotes the number of samples in the same cluster in the source (resp. k_i) domain. $\varphi(\cdot,\cdot)$ is the function that measures the distance of different centers. In this work, we leverage Euclidean distance to define the distance between centers, i.e. $\varphi(x_i, x_j) = \|x_i - x_j\|^2$. $y^{\tilde{t}}$ denotes the pseudo label of the target domain. To obtain highly precise pseudo labels, we set a threshold to select the target label, as shown below:

$$y^{\tilde{t}} = \begin{cases} \arg\max(p(x^t)) & \text{if } p(x^t) > T \\ -1 & \text{otherwise} \end{cases} \tag{4}$$

where $p(x^t)$ is the probability that x^t belongs to each category and T is the threshold.

Label Prediction

To predict pseudo labels, a classifier C is learned on the common space layer u with the following cross-entropy loss:

$$L_c = \min_{W^E}(-\sum_{i=1}^{n_s} y_i^s \log(p(x_i^s)) - \varepsilon \sum_{i=1}^{n_t} \tilde{y}_i^t \log(p(x_i^t))) \tag{5}$$

where y_s is the labels of the source domain and $\tilde{y}^t$ is the pseudo label of target domain, obtained via Equation 4. It is unavoidable that a small number of pseudo labels are wrong. Thus, we add a weight parameter ε to balance the impact of pseudo labels.

2.2 Cross-domain Semantic Reconstruction

As showing in Figure 1, our model consists of two independent generators, denoted G^s and G^t, aiming at reducing the domain disparity in the common space. Specifically, G^s is responsible for reconstructing source domain images from the common space u while G^t reconstructs target domain images also from u. The intuitive reconstruction directions are formulated as follows:

$$\begin{aligned} \tilde{x}^s &= G^s(u^s) \\ \tilde{x}^t &= G^t(u^t) \end{aligned} \tag{6}$$

where $u = E(x)$ denotes the common space feature. To reconstruct vivid images, we use the l_1 pixel-wise loss to constrain the original images and reproduced images, as follows:

$$L_1 = \min_{W^E, W^G} \sum_{x^s \in D^s} \|x^s - \tilde{x}^s\| + \sum_{x^t \in D^t} \|x^t - \tilde{x}^t\| \tag{7}$$

where W^G denotes the parameters of the two generators. Note that the two generators do not share parameters, and W^G is a *notational convenience*.

However, the two reconstruction directions alone cannot benefit the domain information transfer, and it is necessary to build some cross-domain relationships between the two domains. Thus, we let each generator generate the other domain's data, as follows:

$$\begin{aligned} \tilde{x}^{ts} &= G^s(u^t) \\ \tilde{x}^{st} &= G^t(u^s) \end{aligned} \tag{8}$$

Intuitively, if the pairs of $\tilde{x}^{st}$ and $\tilde{x}^t$, $\tilde{x}^{ts}$ and $\tilde{x}^s$ are reconstructed well, the learned common space adapts well on both domains. To illustrate this, taking the pair of $\tilde{x}^{st}$ and $\tilde{x}^t$ as an example, when they are indistinguishable, the domain disparity on the common space is small.On the other hand, if we remove the generator G^s, it will render that the target domain space is a subspace of the source domain because of asymmetric training with only one reconstructing path [Ghifary *et al.*, 2016]. In the training stage, both generators are optimized adversarially until they find the best common space suitable for both domains.

Semantic Discriminators

It is worth noting that our model not only reconstructs the opponent domain's images but also ensures that the reproduced image has the same class label with the original input image. Specifically, following previous work [Hu *et al.*, 2018], our discriminators are designed such that it distinguishes the fake and the real, and at the same time predicts the label for real images. Thus, the output of a discriminator has $N+1$ distinct values, of which N values describe the image's labels and the last value defines whether the image is reconstructed or original. As in GAN [Goodfellow *et al.*, 2014], we can treat the GAN training stage as the generators and discriminators playing a minimax game, where the generators try to fool the discriminators by generating realistic data while the discriminators try to distinguish whether the input data is original or reconstructed. The original GAN loss is formulated as:

$$L_a = \min_{W^G} \max_{W^D} (\log(D(x)) + \log(1 - D(\tilde{x}))) \tag{9}$$

where W^D denotes the parameters of the two (separate) discriminators and W^G is the parameters of the two generators.

In our work, the discriminators need to not only differentiate fake data but also recognize the label of the real data. Thus, we augment the adversarial loss as follows:

$$\begin{aligned} L_a = \min_{W^G} \max_{W^D} (y^s \log(D(x^s)) + \tilde{y}^t \log(D(x^t)) + \\ y^f \log(D(\tilde{x}^{st})) + y^f \log(D(\tilde{x}^{ts}))) \end{aligned} \tag{10}$$

where y^f denotes the fake label and W^G/W^D are the parameters of the generators/discriminators.

2.3 Overall Objective Function

In summary, the overall loss function is rewritten as follows:

$$L = \min_{W^E, W^G, W^D} (L_c + L_a + \alpha L_h + \beta L_s + \chi L_1) \tag{11}$$

where α, β and χ are balance weights. We use stochastic gradient descent (SGD) to optimize the parameters. The update rules are formulated as follows, where η is the learning rate:

$$\begin{aligned} W^E \leftarrow W^E - \eta \times (\frac{\partial L_c}{\partial W^E} + \frac{\partial L_a}{\partial W^G} \times \frac{\partial W^G}{\partial W^E} + \alpha \frac{\partial L_s}{\partial W^E} \\ + \beta \frac{\partial L_h}{\partial W^E} + \chi \frac{\partial L_1}{\partial W^G} \times \frac{\partial W^G}{\partial W^E}) \end{aligned} \tag{12}$$

$$W^G \leftarrow W^G - \eta \times (\frac{\partial L_a}{\partial W^G} + \chi \frac{\partial L_1}{\partial W^G}) \tag{13}$$

$$W^D \leftarrow W^D - \eta \times (\frac{\partial L_a}{\partial W^D}) \tag{14}$$

As L_s is calculated by mini-batches, it is reasonable that the larger the mini-batch size is, the more accurate cluster centroids can be obtained. Moreover, the batch size must be larger than the number of classes (N). In our experiment, we set the batch size as 10 times bigger of the number of labels.

2.4 Differences from Previous Works

Difference from Duplex [Hu *et al.*, 2018]. Both Duplex and our framework adapt the generative adversarial networks to learn the common space to reduce domain disparity. However, Duplex uses only one generator to reconstruct cross-domain images, which requires it to add extra condition information to the common space, making it hard to align the two domains. In this work, on the other hand, we employ two separate generators, both are able to reconstruct cross-domain images. Moreover, we do not require any additional information for the common space, which allows the latent representations on the two domains to have consistent distributions.

Difference from MSTN [Xie *et al.*, 2018] and **Co-GAN** [Liu and Tuzel, 2016]. Both of MSTN and Co-GAN use discriminators to judge the origin of domain representations. However, the discriminators do not recognize the label of images. In contrast, our discriminators can predict class labels of images, hence are able to preserve semantic information in the common feature space, which makes the learned space more discriminative.

Difference from GTA [Sankaranarayanan *et al.*, 2018]. Although both GTA and our framework adopts the generative adversarial network to reconstruct images from the common space, GTA only uses one direction to reconstruct images: only from target to source. In contrast, our framework employs the cross-domain reconstruction strategy (both source to target and target to source), which avoids the situation that one subspace became a joint subspace affine to the opposed domain.

3 Experiments

We evaluate our model on the tasks of object recognition and image retrieval. Specifically, the experiments are conducted to answer the following research questions:

RQ1: Is our method superior to the state-of-the-art domain adaptive methods?

RQ2: How does each part of our model affect the performance of object recognition and image retrieval?

RQ3: How well does our method learn the common space representation?

3.1 Datasets

We test our method on three public digits datasets: MNIST (M) [LeCun *et al.*, 1998], SVHN (S) [Netzer *et al.*, 2011], and USPS (U) [Denker *et al.*, 1989]. Specifically, the three datasets have the same 10 categories but have various types. MNIST contains of 60,000 training and 10,000 testing images, and USPS contains 7,291 training and 2,007 testing images. SVHN, obtained from house numbers in Google Street View images, has 73,257 training and 26,032 testing images.

Additionally, we also evaluate our method on the standard benchmark dataset Office-31 [Saenko *et al.*, 2010], which has three types of images: Amazon (A), Webcam (W) and DSLR (D). Office-31 in total has 4,110 images, of which 2,817 in Amazon, 795 in Webcam, and 498 in DSLR.

For the digits datasets, we select three directions of domain shift: SVHN $\rightarrow$ MNIST, MNIST $\rightarrow$ USPS, and USPS $\rightarrow$ MNIST. We choose four directions of domain shifts for the Office-31 dataset: Amazon $\rightarrow$ Webcam, Webcam $\rightarrow$ Amazon, Amazon $\rightarrow$ DSLR, and DSLR $\rightarrow$ Amazon. We discard the directions of Webcam $\rightarrow$ DSLR and DSLR $\rightarrow$ Webcam because the images are highly similar. All the dataset setting are the same as [Hu *et al.*, 2018].

3.2 Implementation Details

The digits datasets contain simple images with the tiny size of 32×32. For SVHN $\rightarrow$ MNIST, we use the network [Hu *et al.*, 2018], which is consisted of four convolutional layers without fully connected layers. Similarly, the generator network is a symmetrical network also consisting of 4 deconvolutional layers [1]. As for USPS $\rightarrow$ MNIST and MNIST $\rightarrow$ USPS, we use the LeNet[LeCun *et al.*, 1998]. The Office-31 dataset contains more complex images that are harder to reconstruct. Thus we choose Alexnet as the encoder network and reconstruct the $fc6$ feature layer instead of reconstructing the original images. In the training stage of Office-31, we use the model pre-trained on ImageNet to initialize our model. As for the digits datasets, we divide the training procedure into several stages. Specifically, we first use the source domain to pre-train the encoder network and then use it to initialize the first stage of the encoder network. After that, every several training epochs we use the last-stage encoder's parameters to initialize the next-stage encoder network, while other components are initialized with random parameters. With regards to training semantic centroid alignment, we do not precisely calculate the centers of all training samples, which is time-consuming. Instead, we calculate the centers of mini-batches to approximate the global centers. We set the batch size as 200 for SVHN $\rightarrow$ MNIST and 100 for MNIST $\rightarrow$ USPS and USPS $\rightarrow$ MNIST. Plus, the length of the hash code is set as 64 bit.

3.3 Comparison with the State-of-the-art Domain Adaptive Hashing Methods (RQ1)

In this section, we will compare our method with state-of-the-art works on both tasks: object recognition and image retrieval.

Object Recognition: we evaluate our method on both datasets: digits and Office-31. For the Office-31, we compare with Duplex [Hu *et al.*, 2018], MSTN[Xie *et al.*, 2018], DRCN [Ghifary *et al.*, 2016], DAN [Long *et al.*, 2015], CDAN [Long *et al.*, 2018b], I2I [Murez *et al.*, 2018], and ADDA [Tzeng *et al.*, 2017]. With regard to digits datasets, we add other two methods: Co-GAN [Liu and Tuzel, 2016] and CyCADA [Hoffman *et al.*, 2018].

Table 1 shows the object recognition results on the digits dataset. As can be seen, our method is superior to all the other methods on average accuracy. For the domain shift of S$\rightarrow$M, our result outperforms the current best method, Duplex, by 3.1%. Similarly, our result on M$\rightarrow$U is also the best, 0.3% higher than CDAN-M. As for U$\rightarrow$M, our result is 1.0% lower than Duplex. Overall, the average performance of our model is the highest and surpasses the current best model Duplex by 0.93%. Table 2 shows the object recognition results of the Office-31 dataset. For A $\rightarrow$ W, our method is lower by 1% than MSTN, but on the other three domain shifts our method outperforms the best method MSTN by 2.7%, 2.2% and 2.0%. Plus, the average performance is also the highest.

Image retrieval. We also test our method on the task of image retrieval. We compare our method with other unsupervised domain adaptive hash methods: ITQ [Gong *et al.*, 2013], BA [Carreira-Perpiñán and Raziperchikolaei, 2015], DAH [Venkateswara *et al.*, 2017], and BDNN [Do *et al.*, 2016]. In addition, we also test our method against the performance upper bound Supervised Hashing [Li *et al.*, 2018], denoted SuH. Table 3 shows the image retrieval results on the Office-31 dataset, in terms of mean average precision (MAP). As stated in Section 3.1, we choose three pairs of domains for image retrieval: AmazontoWebcam , Webcam$\rightarrow$Amazon and Amazon$\rightarrow$Dslr, following [Venkateswara *et al.*, 2017]. As can be seen from Table 3, for every retrieval pair, our method is superior to the other unsupervised hashing methods, with an improvement of about 3.0%, 2.1%, and 2.5% respectively. However, in comparison to SuH, there is still a huge gap between the supervised hash method and the supervised method.

Table 1: Object recognition accuracies (%) on the digits datasets.

Methods	S$\rightarrow$M	M$\rightarrow$U	U$\rightarrow$M	Avg
CyCADA	90.4	95.6	96.5	94.17
Duplex	92.5	96.0	**98.8**	95.80
MSTN	91.7	92.9	-	92.30
CDAN-M	89.2	96.5	97.1	94.30
ADDA	76.0	89.4	90.1	85.17
Co-GAN	-	91.2	89.1	90.15
DRCN	82.0	91.8	73.7	82.50
Ours	**95.6**	**96.8**	97.8	**96.73**

Table 2: Object recognition accuracies (%) on the Office-31 datasets.

Methods	A$\rightarrow$W	A$\rightarrow$D	W$\rightarrow$A	D$\rightarrow$A	Avg
Duplex	73.2	74.1	59.1	61.5	66.96
MSTN	**80.5**	74.5	60.0	62.5	69.38
I2I	75.3	71.1	52.1	50.1	62.15
CDAN-M	78.3	76.3	57.3	57.3	67.30
ADDA	73.5	71.6	53.5	54.6	63.30
Ours	79.5	**77.2**	**62.2**	**64.5**	**70.85**

3.4 Ablation Study (RQ2)

As presented in the previous section, our method consists of several loss functions: label prediction loss L_c, hash loss L_h, semantic centroid alignment loss L_s, adversarial loss L_a, and pixel-wise reconstruction loss L_1. In this section, we will test the effect of each component on the performance.

Table 4 shows the results of object recognition on the digits datasets, where the above loss functions are added one at a

[1]Our code is released at https://github.com/htlsn/igan.

Table 3: MAP (mean average precision) 64 bits(%) on the Office-31 datasets and Office home. Note that SuH is a supervised hashing method that represents the performance upper bound.

Methods	W→A	A→W	A→D	Avg
NoDA	32.4	51.1	51.2	44.90
ITQ	46.5	65.2	64.3	58.67
BDNN	49.1	65.6	66.8	60.50
BA	36.7	48.0	49.7	44.80
DAH	58.2	71.7	72.1	67.33
Ours	**61.2**	**73.8**	**74.6**	**69.87**
SuH	88.1	91.6	92.7	90.80

time. With only the L_c component in our baseline model, the average accuracy is 69.17. Next, we add the centroid alignment loss L_s. Obviously, the centroid alignment loss has a positive effect on the results with about 17.33 improvement. Similarly, adding the hashing loss L_h results in a further improvement of 0.9 over the centroid alignment loss.

With the addition of the adversarial loss L_a produced by our intersectant generators, the performance is improved by 8.47, which demonstrates that the proposed cross-domain reconstruction loss is effective.

Finally, we test the pixel-wise loss L_1. As can be seen from Table 4, L_1 loss further improves the performance by 0.86. The possible reason for L_1 is that it can constrain the reconstructed images to look similar to the original images, which helps improve the judgement of the discriminators.

Briefly, from Table 4, we can obtain the following two conclusions. (1) Each of the loss function contributes positively to recognition performance, and their combination achieves the best results. (2) The adversarial component L_a is the key part of our framework. Specifically, when L_a is added, about 8.47 improvement is obtained compared without it. Thus, the intersectant reconstruction has a significant effect on reducing domain disparity.

Table 4: Ablation study results of object recognition accuracy on the digits datasets.

Methods	S→M	M→U	U→M	Avg
L_c	60.2	85.5	61.8	69.17
L_c+L_s	81.2	89.5	88.8	86.50
L_c+L_s+L_h	82.4	90.3	89.5	87.40
L_c+L_s+L_h+L_a	94.4	96.1	97.2	95.87
L_c+L_s+L_h+L_a+L_1	95.6	96.8	97.8	96.73

3.5 Common Space Feature Visualization (RQ3)

We resort to visualization to assess the quality of the common space representation learned by our method. Figure 2 shows the visualization of the feature representation learned by our method, with feature dimensions reduced by t-SNE from 64 to 2. Figure 2 (a) shows all the 795 points of Webcam, color-coded by the 31 classes, that are generated by only training on the Amazon source domain. Before training by our method, it is easy to see that the points are scattered everywhere, and the points belonging to the same class (same color) do not cluster around a centroid. In contrast, after we add several loss functions 11, the learned features become discriminative. I.e., points of the same class cluster in the same centroid, and class centroids have long distance, as can be seen from Figure 2 (b). The same observation can be made for the Webcam→Amazon direction as well, as shown in Figure 2 (c) and (d).

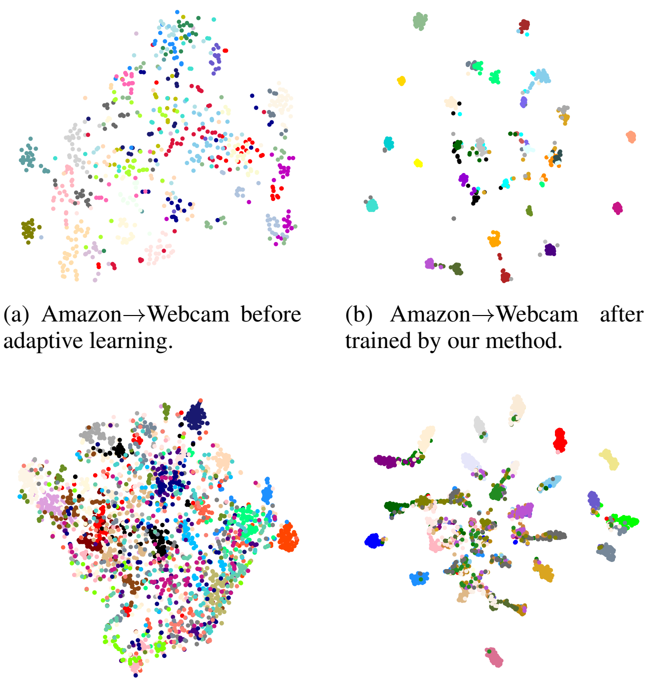
(a) Amazon→Webcam before adaptive learning.
(b) Amazon→Webcam after trained by our method.
(c) Webcam→Amazon before adaptive learning.
(d) Webcam→Amazon after trained by our method.

Figure 2: A visualization of the target domain features learned with and without domain adaption, with feature dimensions reduced by t-SNE from 64 to 2. For Amazon→Webcam, (a) is the feature representation before adaptive learning, and (b) is trained by our method. Similarly, (c) is before adaptive learning and (d) is trained by our method, for Webcam→Amazon.

4 Conclusion

Cross-domain hashing allows hash functions learned on one domain to be applied effectively to a new domain without supervision. In this paper, we propose an end-to-end cross-domain hashing framework based on intersectant generative adversarial networks. Our framework learns representations for both domains in a common space and combines five complementary loss functions: label prediction loss, supervised hash loss, semantic centroid alignment loss, cross-domain reconstruction loss, and pixel-wise loss. All of them help improve learning the common space and reduce the disparity between the two domains. Finally, several comparison experiments show that our method is superior to the other state-of-the-art methods in cross-domain hashing on a number of benchmark datasets.

Acknowlegements

References

[Carreira-Perpiñán and Raziperchikolaei, 2015] Miguel Á. Carreira-Perpiñán and Ramin Raziperchikolaei. Hashing with binary autoencoders. In *CVPR*, pages 557–566, 2015.

[Denker *et al.*, 1989] John S. Denker, W. R. Gardner, Hans Peter Graf, Donnie Henderson, R. E. Howard, W. Hubbard, L. D. Jackel, Henry S. Baird, and Isabelle Guyon. Neural network recognizer for hand-written zip code digits. In D. S. Touretzky, editor, *NIPS*, pages 323–331. Morgan-Kaufmann, 1989.

[Do *et al.*, 2016] Thanh-Toan Do, Anh-Dzung Doan, and Ngai-Man Cheung. Learning to hash with binary deep neural network. In *ECCV*, pages 219–234, 2016.

[Ganin and Lempitsky, 2015] Yaroslav Ganin and Victor S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, pages 1180–1189, 2015.

[Ghifary *et al.*, 2016] Muhammad Ghifary, W. Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. *CoRR*, abs/1607.03516, 2016.

[Gong *et al.*, 2013] Yunchao Gong, Svetlana Lazebnik, Albert Gordo, and Florent Perronnin. Iterative quantization: A procrustean approach to learning binary codes for large-scale image retrieval. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(12):2916–2929, 2013.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, and Sherjil Ozair. Generative adversarial nets. In *NIPS*, pages 2672–2680. Curran Associates, Inc., 2014.

[Hoffman *et al.*, 2018] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *ICML*, pages 1994–2003, 2018.

[Hu *et al.*, 2018] Lanqing Hu, Meina Kan, Shiguang Shan, and Xilin Chen. Duplex generative adversarial network for unsupervised domain adaptation. In *CVPR*, June 2018.

[LeCun *et al.*, 1998] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[Li *et al.*, 2018] Chao Li, Cheng Deng, Ning Li, Wei Liu, Xinbo Gao, and Dacheng Tao. Self-supervised adversarial hashing networks for cross-modal retrieval. In *CVPR*, June 2018.

[Liu and Tuzel, 2016] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *NIPS*, pages 469–477, 2016.

[Long *et al.*, 2015] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, pages 97–105, 2015.

[Long *et al.*, 2018a] Fuchen Long, Ting Yao, Qi Dai, Xinmei Tian, Jiebo Luo, and Tao Mei. Deep domain adaptation hashing with adversarial learning. In *SIGIR*, SIGIR '18, pages 725–734, New York, NY, USA, 2018. ACM.

[Long *et al.*, 2018b] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I. Jordan. Conditional adversarial domain adaptation. In *NIPS*, pages 1647–1657, 2018.

[Murez *et al.*, 2018] Zak Murez, Soheil Kolouri, David J. Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *CVPR*, pages 4500–4509, 2018.

[Netzer *et al.*, 2011] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5, 2011.

[Saenko *et al.*, 2010] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *ECCV*, pages 213–226. Springer, 2010.

[Sankaranarayanan *et al.*, 2018] Swami Sankaranarayanan, Yogesh Balaji, Carlos D. Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *CVPR*, pages 8503–8512, 2018.

[Song *et al.*, 2018] Jingkuan Song, Tao He, Lianli Gao, Xing Xu, Alan Hanjalic, and Heng Tao Shen. Binary generative adversarial networks for image retrieval. In *AAAI*, pages 394–401, 2018.

[Tzeng *et al.*, 2014] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *CoRR*, abs/1412.3474, 2014.

[Tzeng *et al.*, 2017] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, pages 2962–2971, 2017.

[Venkateswara *et al.*, 2017] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, pages 5385–5394, 2017.

[Xie *et al.*, 2018] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. In *ICML*, pages 5419–5428, 2018.

[Zhang *et al.*, 2018] Weichen Zhang, Wanli Ouyang, Wen Li, and Dong Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *CVPR*, June 2018.