


Article

# C<sup>2</sup>DAN: An Improved Deep Adaptation Network with Domain Confusion and Classifier Adaptation

Han Sun <sup>1,2,\*</sup> , Xinyi Chen <sup>1,2</sup>, Ling Wang <sup>1,2</sup>, Dong Liang <sup>1,2</sup>, Ningzhong Liu <sup>1,2</sup> and Huiyu Zhou <sup>3</sup>

<sup>1</sup> College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China; 17817356323@163.com (X.C.); wl941017@163.com (L.W.); liangdong@nuaa.edu.cn (D.L.); liunz@163.com (N.L.)

<sup>2</sup> MIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing 211106, China

<sup>3</sup> School of Informatics, University of Leicester, Leicester LE1 7RH, UK; hz143@leicester.ac.uk

\* Correspondence: sunhan@nuaa.edu.cn

Received: 3 June 2020; Accepted: 23 June 2020; Published: 26 June 2020



**Abstract:** Deep neural networks have been successfully applied in domain adaptation which uses the labeled data of source domain to supplement useful information for target domain. Deep Adaptation Network (DAN) is one of these efficient frameworks, it utilizes Multi-Kernel Maximum Mean Discrepancy (MK-MMD) to align the feature distribution in a reproducing kernel Hilbert space. However, DAN does not perform very well in feature level transfer, and the assumption that source and target domain share classifiers is too strict in different adaptation scenarios. In this paper, we further improve the adaptability of DAN by incorporating Domain Confusion (DC) and Classifier Adaptation (CA). To achieve this, we propose a novel domain adaptation method named C<sup>2</sup>DAN. Our approach first enables Domain Confusion (DC) by using a domain discriminator for adversarial training. For Classifier Adaptation (CA), a residual block is added to the source domain classifier in order to learn the difference between source classifier and target classifier. Beyond validating our framework on the standard domain adaptation dataset office-31, we also introduce and evaluate on the Comprehensive Cars (CompCars) dataset, and the experiment results demonstrate the effectiveness of the proposed framework C<sup>2</sup>DAN.

**Keywords:** transfer learning; domain adaptation; MK-MMD; domain confusion; classifier adaptation; vehicle classification

## 1. Introduction

In recent years, deep learning has made great achievements in a large number of computer vision tasks, such as image recognition [1–3], object detection [4,5], fine-grained classification [6,7], semantic segmentation [8,9] and so on. For such satisfactory results, training with huge labeled datasets is essential, but in real applications, data labeling is too expensive, and sometimes impracticable. For example, in video surveillance, images can be affected seriously by camera position and illumination variance, and the application scenarios are also ever-changing, so these factors make the task of labeling the entire dataset impossible. In order to solve such problems, Domain Adaptation (DA) [10] is a very efficient method, which aims to extend the prediction model learned from the source domain with abundant labeled data to the target domain which only has the unlabeled or a small number of labeled data. Here the data between the source and target domain are different but related.

The main idea of domain adaptation is to reduce the difference between domains and to learn a prediction model. Two types of mechanism are used most to address domain shift: discrepancy-based and adversarial-based. The representative discrepancy-based methods [11–20]

minimize the domain discrepancy by using a distance metric such as Maximum Mean Discrepancy (MMD) [17], CORAL [18,19] and Kullback-Leibler divergence [20], among which MMD-based methods are widely used. In this sort of methods, the difference between source and target domains is usually reduced by optimizing the MMD in Reproducing Kernel Hilbert Space (RKHS), and the feature representation with domain invariance needs to be learned. Adversarial-based methods [21–30] use a discriminator to distinguish whether the data comes from the source domain or the target domain, so as to increase the domain confusion and minimize the distance between the source and the target domain distribution.

However, these methods are not without problems. The typical discrepancy-based Deep Adaptation Network (DAN) framework [12] which is superior to other similar methods still faces performance degradation problems in feature level transfer. Since adversarial-based methods show efficiency in DA, we investigate herein if combining a discrepancy-based method with adversarial learning will be helpful to improve the performance. In addition, most previous works are based on the assumption that the source domain and target domain share the classifier, but the assumption is too strict to maintain stable effects in different adaptation scenarios, so it is necessary to relax the assumption and reduce the differences between source classifier and target classifier.

In this paper, inspired by the adversarial-based method [21] and Domain Adaptation with Residual Transfer Networks (RTN) [14], we propose our framework called  $C^2$ DAN which improves the DAN framework by combining it with Domain Confusion (DC) and Classifier Adaptation (CA). These changes make the extracted feature representation more adaptive to the target domain. At the same time, aiming at solving the problem of different adaptation effects of MK-MMD in different scenarios, this paper explores the suitable weight selection of MK-MMD. In addition, we pursue the best combination of MK-MMD, DC and CA to further enhance the efficiency of domain adaptation. Experiments on the standard domain adaptation dataset Office-31 [31] and the CompCars [32] dataset which is used for vehicle classification task demonstrate the effectiveness of our proposed method  $C^2$ DAN.

The contributions of this paper are as follows:

- (1) We combine Domain Confusion (DC) with MK-MMD in DAN for both feature alignment and domain alignment, which makes the model more generalized in the target domain.
- (2) The model is extended by adding Classifier Adaptation (CA) to minimize the difference of source classifier and target classifier, the accuracy of the proposed method is further improved.
- (3) The best combination of MK-MMD, DC and CA in different scenarios is obtained through experiments on office-31 and CompCars dataset, the experimental results show that our improved method  $C^2$ DAN surpass the performance of DAN.

The structure of this paper is as follows: Section 1 briefly introduces the research objectives and contributions of this paper. Section 2 gives a review of related works about deep domain adaptation. In Section 3, the principle and implementation of the deep fusion method between MK-MMD, DC and CA are proposed. In Section 4 the experimental results on dataset Office-31 are demonstrated and analyzed. The suitable weight selection of MK-MMD in different scenarios is explored. The best combination of MK-MMD, DC and CA is also discussed in Section 4. The details and results of our vehicle classification experiments are shown in Section 5. The conclusions of this paper are summarized in Section 6.

## 2. Related Work

For the deep domain adaptation methods, the basic criterion is to add the adaptation metric after selecting the adaptive layer, and then to fine tune the network. At the PRICAL meeting in 2014, the Domain Adaptive Neural Network (DaNN) concept [11] was put forward. This network has only two layers of neurons, which include a feature layer and a classifier layer. The characteristic of this network lies in the MMD adaptive layer after feature layer, which calculates the distance in Reproducing kernel Hilbert space (RKHS) between the source domain and target domain, and then optimizes its loss, but

because of the poor ability of feature representation based on the shallow network, it is also difficult to solve practical problems. After this, many researchers combined this idea into deep networks. These works adopt AlexNet trained on ImageNet for domain adaptation. The main idea is to fix the first seven layers and add MMD metrics before the classifier layer to implement domain adaptation. However, the effect of such single adaptive layer is limited, so the DAN network was proposed in [12]. It uses MK-MMD, which has stronger representation ability by combing the multi-kernel concept. At the same time, three adaptive layers are added into the network, which achieves better classification effect without increasing the training time. At the ICML conference in 2017, JAN [13] was proposed to extend the data adaptive method to the category adaptation, which used Joint MMD (JMMD) metrics. Inspired by the deep residual network (ResNet) [2] framework, RTN [14] learns adaptive classifiers and transferable features from labeled source domains and unlabeled target domains by embedding the adaptation process of classifiers and features into a unified deep network architecture.

Adversarial-based models use a generator to align the source and target domain data in feature space by working against the discriminator. Generative Adversarial Network (GAN) [33] has been transferred into different application scenarios since it was first presented in 2014. Isola et al. proposed [34] and tried to do image translation with conditional GAN (CGAN) [35], allowing the network to learn image to image mapping functions without having to customize features manually. Coupled Generative Adversarial Networks (CoGAN) [36] learns the joint distribution of source domain and target domain data by imposing a constraint on both networks to share parameters, the ability of the network is limited so that the data generated from noise is about the same for both networks. CoGAN consists of two GAN, which is the same as CycleGAN [37], DualGAN [38] and DiscoGAN [39]. Shen et al. proposed the WGDRL [40] metric which is based on WGAN [41] to measure the distance between source domain and target domain.

In this work, the DAN [12] method is further improved by domain alignment (DC) and classifier alignment (CA). For domain alignment, DC is used to further utilize the source domain information, which makes the extracted features more powerful in the target domain representation. Then CA is added to improve the domain adaptation effect by aligning the classifiers of source domain and target domain. Meanwhile, we explore the applicable weights of MK-MMD under different scenarios and the best combination of the three aspects. We prove the effectiveness of the proposed method both on standard dataset and vehicle classification task.

### 3. C<sup>2</sup>DAN: Improved Deep Adaptive Network

The network structure proposed in this paper is based on DAN [12] which adopts multi-layer and multi-kernel MMD. We add a domain classification layer  $f_{DC}$  to perform domain confusion. Domain alignment will be completed when the domain classifier cannot distinguish whether the input is from the source domain or from the target domain. And in order to learn the difference between source classifier and target classifier, a residual block which connects the classifiers of two domain is added to the source classifier. The joint optimization of the four loss functions which include the loss of multi-layer MK-MMD, the loss of  $f_{DC}$  layer, the loss of CA and the final classification loss of the overall network, is used to achieve good unsupervised domain adaptation effect. The network structure of C<sup>2</sup>DAN is shown in Figure 1.

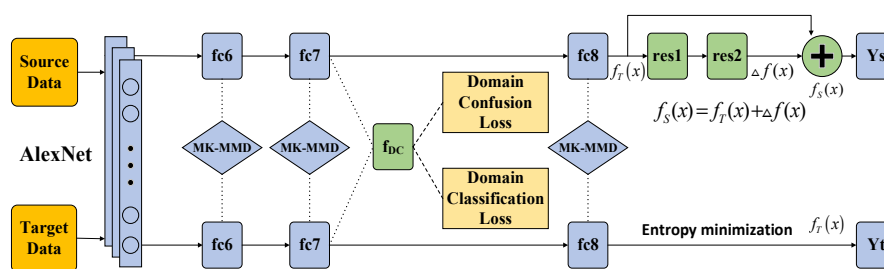


Figure 1. The structure of proposed network C<sup>2</sup>DAN.

### 3.1. MK-MMD

Multi-kernel Maximum Mean Discrepancy (MK-MMD) is an extension of MMD. MMD is one of the most commonly used non-parametric methods to measure the distribution difference between two domain datasets. The detailed operation is to map the feature representation of source domain and target domain to the reproducing kernel Hilbert space (RKHS), and then calculate the mean distance between the two datasets in RKHS. Given the data distribution  $s$  and  $t$  of the two domains, and by using the function  $\phi(\cdot)$ , the MMD between  $s$  and  $t$  is calculated as follows:

$$\text{MMD}^2(s, t) = \sup_{\|\phi\|_{\mathcal{H}} \leq 1} \left\| \mathbb{E}_{x^s \sim s} [\phi(x^s)] - \mathbb{E}_{x^t \sim t} [\phi(x^t)] \right\|_{\mathcal{H}}^2 \quad (1)$$

where  $\mathbb{E}_{x^s \sim s}[\cdot]$  represents the mathematical expectation of the source domain's distribution. And  $\|\phi\| \leq 1$  represents a series of functions in the unit ball of reproducing kernel Hilbert space  $\mathcal{H}$ . Let  $D_s = \{X_i^s\}_{i=1}^M$  represent sample sets of distribution  $s$ , then an empirical estimate of MMD can be expressed as follows:

$$\text{MMD}^2(D_s, D_t) = \left\| \frac{1}{M} \sum_{i=1}^M \phi(x_i^s) - \frac{1}{N} \sum_{j=1}^N \phi(x_j^t) \right\|_{\mathcal{H}}^2 \quad (2)$$

where  $\phi(\cdot)$  represents the feature mapping about  $k(x^s, x^t) = \langle \phi(x^s), \phi(x^t) \rangle$ , and  $k(x^s, x^t)$  is usually defined as a convex combination of  $L$  basis kernels  $k_l(x^s, x^t)$ , like the following:

$$k(x^s, x^t) = \sum_{l=1}^L \beta_l k_l(x^s, x^t), \text{ s.t. } \beta_l \geq 0, \sum_{l=1}^L \beta_l = 1 \quad (3)$$

The MMD method is based on single kernel transformation. The multi-kernel MMD (MK-MMD) assumes that the optimal kernel can be obtained by linear combination of multiple kernels. One of the most successful ways to use MK-MMD is DAN. As shown in [12], let  $H_k$  denote the RKHS with the characteristic kernel  $k$ , and let the mean value of distribution  $p$  in  $H_k$  be an independent element  $\mu_k(p)$ , then we can get  $\mathbb{E}_{x \sim p} f(x) = \langle f(x), \mu_k(p) \rangle_{H_k}$ , where  $f \in (H_k)$ . The distance of mean value between probability distribution  $p$  and  $q$  is expressed as  $d_k(p, q)$ , and its square formula is as the following:

$$d_k^2(p, q) \triangleq \left\| \mathbb{E}_p[\phi(x^s)] - \mathbb{E}_q[\phi(x^t)] \right\|_{\mathcal{H}_k}^2 \quad (4)$$

Similar to MMD, feature mapping  $\phi$  is related to characteristic kernel, so  $k(x^s, x^t) = \langle \phi(x^s), \phi(x^t) \rangle$ . Here  $k(x^s, x^t)$  is defined as a convex combination of  $m$  PSD kernels  $\{k_u\}$ . Multi-kernel  $k$  can use different kernels to enhance the effect of MK-MMD and achieve an optimal and reasonable kernel selection:

$$K \triangleq \left\{ k = \sum_{u=1}^m \beta_u k_u : \sum_{u=1}^m \beta_u = 1, \beta_u \geq 0, \forall u \right\} \quad (5)$$

$k$  is weighted by different kernel and the coefficients  $\{\beta_u\}$  is the weight to ensure that the generated multi-kernel  $k$  is characteristic.

### 3.2. Domain Confusion

In order to reduce the difference of marginal distribution between two domains, maximizing domain confusion (DC) is an effective method. The schematic diagram of domain confusion is shown in Figure 2. For maximizing domain confusion, a domain classification layer  $f_{DC}$  is considered in this paper. The main function of this layer is to judge whether the sample belongs to the source domain or target domain by using the feature representation gained from trained samples. From an

intuitive point of view, the more domain-specific the extracted features belong to, the better the effect of domain classification is. Meanwhile the more common the extracted features are, the better the effect of domain confusion is. When the classifier trained by a specific feature representation cannot distinguish the samples of source domain or target domain, we can call this feature representation is domain invariant. In this paper, we use a domain confusion loss, which is optimized to obtain a domain invariant feature representation.

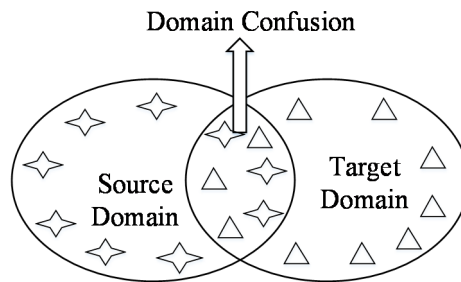


Figure 2. The definition of domain confusion.

According to [12], for a feature representation  $\phi_{repr}$ , we measure its domain invariance by learning the best domain classifier based on this representation.  $\phi_D$  is the parameter of the domain classifier to be learned. By optimizing the following loss function (6), the best domain classifier can be learned. Besides, we introduce the loss of domain confusion for a domain classifier, which calculates cross entropy between the predicted output domain label and a uniform distribution of domain labels, so as to maximize the confusion and reduce the distribution differences between the two domains. The uniform distribution of domain labels means that the possibility of feature representation belonging to source or target domain is  $\frac{1}{D}$ , in which case the domain label is most difficult to judge.  $D$  is the number of domains and here  $D$  is 2. And the loss function of domain confusion is shown in Equation (7):

$$L_D \geq (\phi_{repr}; \theta_D) = -\sum_d 1[y_D = d] \log q_d \quad (6)$$

$$L_{conf}(\theta_D; \theta_{repr}) = -\sum_d \frac{1}{D} \log q_d \quad (7)$$

where  $y_D$  denotes the domain to which the sample belongs, and  $q$  represents the softmax value of the domain classifier,  $q = \text{softmax}(\theta_D f(x; \theta_{repr}))$ .

The optimization of the domain confusion loss above is to find a domain invariant feature representation, in which the best domain classifier cannot achieve good results. Ideally, we want to optimize these two losses simultaneously in the training process. But there are two contradictions between domain classification and domain confusion. Learning a good domain classifier means that the effect of domain confusion is very poor, and also reaching a good domain confusion effect means that the effect of domain classification is very poor. Therefore, the two losses of layer  $f_{DC}$  need to be optimized jointly to achieve a compromise and reasonable optimization results.

### 3.3. Classifier Adaptation

In order to reduce the difference of classifiers between two domains, adding classifier adaptation (CA) is also an effective method. CA is motivated by the deep residual learning [2] which is shown in Figure 3 and Domain Adaptation with Residual Transfer Networks [14]. Based on the formula  $H(x) = F(x) + x$  in [2], we can deduce our classifier adaptation (CA) function:

$$f_s(x) = \Delta f(x) + f_t(x) \quad (8)$$

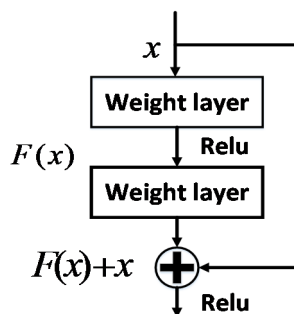


Figure 3. Residual learning: a building block.

$f_s(x)$  is the output of source classifier and  $f_t(x)$  is the output of target classifier,  $\Delta f(x)$  is a perturbation of classifiers between two domains. Let  $x \triangleq f_t(x)$  and  $H(x) = f_s(x)$ . There is an assumption that when target domain and source domain are connected, perturbation function  $\Delta f(x)$  can be learned from the labeled data in the source domain and the unlabeled data in the target domain. The source label classifier's loss function can be computed as:

$$\min_{f_s(x)=\Delta f(x)+f_t(x)} \frac{1}{n_s} \sum_{i=1}^{n_s} J(\theta_{repr}(x_i^s), y_i^s) \quad (9)$$

where  $J$  is the cross-entropy loss function, and  $\theta_{repr}(x_i^s)$  denotes the conditional probability that the network attaches label  $y_i^s$  to the sample  $x_i^s$ .

Although classifier adaptation (CA) has reduced the difference of source classifier and target classifier, the output of target classifier  $f_t(x)$  cannot be guaranteed to fit the target domain very well. Therefore, the entropy minimization principle is used to optimize the parameters and minimize the entropy of conditional distribution of each class in order to encourages the low-density separation among classes of target domain. The target classifier's loss function can be computed as:

$$\min_{f_t} \frac{1}{n_t} \sum_{i=1}^{n_t} H(f_t(x_i^t)) \quad (10)$$

where  $H$  is the entropy loss function,  $H(f_t(x_i^t)) = -\sum_{j=1}^c f_j^t(x_i^t) \log f_j^t(x_i^t)$ ,  $c$  is the number of classes and  $f_j^t(x_i^t)$  denotes the probability that the label is  $j$  for  $x_i^t$ .

### 3.4. Loss Function

According to DAN's network, this paper uses the classic five convolutional layers and three full connection layers. Each full connection layer  $L$  learns a nonlinear mapping  $h_i^l = f^l(W^l h_i^{l-1} + b^l)$ , where  $W^l$  and  $b^l$  are the weights and offsets of the  $L$ th layer,  $h_i^l$  is the feature representation for  $x_i$  in the  $L$ th layer, and  $f^l$  is the activation function. Let  $\theta_{repr} = \{W^l, b^l\}_{l=1}^L$  represents the parameter set of the convolutional neural networks, the empirical loss function of the whole network is as follows:

$$\min_{\theta_{repr}} \frac{1}{N} \sum_{i=1}^N J(\theta_{repr}(x_i^t), y_i) \quad (11)$$

In this paper, we add classifier adaptation (CA) by adding two fully connected layers as residual layers to ensure that  $f_t(x)$  does not deviate too far from  $f_s(x)$ , so Equation (11) is rewritten into Equation (9).

In the standard convolutional neural network, the deep feature representation becomes from generalization to specialization with the layer from low to high. The higher the network layer is, the



greater domain's difference is, which cannot be reduced by fine-tune alone. Therefore, it is necessary to make adaptation on the full connection layers instead of convolutional layers. In this paper, we propose two ways to do domain adaptation on the full connection layers and reduce the difference of domain distribution. One way is to add MK-MMD based multi-layer adaptation, and the other way is to add a domain classification layer  $f_{DC}$ . We further extend our method by adding classifier adaptation (CA). Then the three factors are considered into the corresponding loss function, and the loss function of the whole network is as follows:

$$L = \min_{f_s(x)=\Delta f(x)+f_t(x)} \frac{1}{n_s} \sum_{i=1}^{n_s} J(\theta_{repr}(x_i^s), y_i^s) + \lambda \sum_{l=1}^{l_2} d_k^2(D_s^l, D_t^l) + \gamma \left\{ -\sum_d 1[y_D = d] \log q_d - \sum_d \frac{1}{D} \log q_d \right\} + \frac{\beta}{n_t} \sum_{i=1}^{n_t} H(f_t(x_i^t)) \quad (12)$$

where  $\lambda$ ,  $\gamma$  and  $\beta$  are the balanced weights of MK-MMD, DC and low-density separation between classes of target domain respectively.

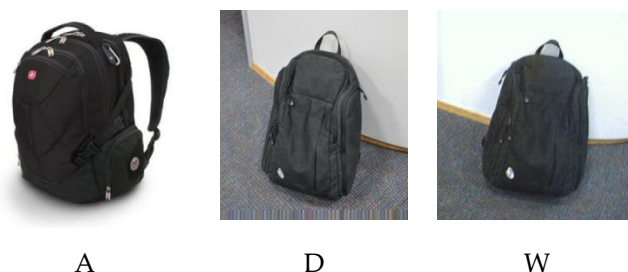
To train a convolutional neural network from scratch requires a large amount of labeled data, which is also impossible for domain adaptation problems, so in this paper the initial network adopts AlexNet model trained on ImageNet 2012 as the pre-trained model. Then we freeze the convolution layer from layer 1 to layer 3 and fine-tunes the 4th and 5th convolution layer, and other parameters are calculated by optimizing the loss function through training.

For different domain adaptation scenarios, the weight of loss function could be different. When we combine MK-MMD, domain confusion and classifier adaptation, we also explore the contribution's difference with different domain adaptation metric to different scenarios. We also consider the reasonable combination of the two methods. This part is explained in Section 4.

## 4. Experiment Results and Analysis

### 4.1. Data Set

Office-31 [31] is a standard data set in the research field of domain adaptation. The Office-31 dataset contains 31 categories of 4,652 images collected from three separate domains, which are (1) Amazon (A, downloaded from amazon.com) with 2817 images, (2) Webcam (W captured from webcams) with 795 images, and (3) DSLR (D captured from digital SLR cameras) with 498 images. The example images are shown in Figure 4. There are six transfer scenarios in these three domains, which are A  $\rightarrow$  W, D  $\rightarrow$  W, W  $\rightarrow$  D, A  $\rightarrow$  D, D  $\rightarrow$  A and W  $\rightarrow$  A.



**Figure 4.** Examples of the Office-31 dataset. (A) (Amazon, downloaded from amazon.com), (D) (DSLR, captured from digital SLR cameras), (W) (Webcam, captured from webcams).

Office-10+Caltech10 dataset contains 10 categories which are selected from Office-31 and Caltech-256. There are four domains in total, three domains (A, W, D) separated from Office-31 and one domain (C) from Caltech-256. Then six transfer tasks can be built: A  $\rightarrow$  C, W  $\rightarrow$  C, D  $\rightarrow$  C,

C  $\rightarrow$  A, C  $\rightarrow$  W and C  $\rightarrow$  D. While Office-10+Caltech has less categories, it will be easier to make adaptation than Office-31 dataset. In this paper, experiments and comparative analysis will be conducted on the two datasets.

In addition to the improved deep adaptation network which combines DC and MK-MMD proposed in this paper, the experiment will be compared with the traditional CNN method, the DAN method only using MK-MMD and the Residual Transfer Networks (RTN) method using classifier adaptation (CA). At the same time, in order to explore the application value and conditions of domain confusion in this network, the experiments of FC6 and FC7 based on DAN and Combination of RTN and DC also will be conducted and compared. Finally, we do the experiments which combining MK-MMD, DC with CA.

#### 4.2. Experiment Procedure

In this paper, we use Caffe to develop the proposed deep neural network, which contains the classic five convolutional layers and three full connection layers. Similar to the method of DAN, MK-MMD layers are attached to the three full connection layers to calculate the multi kernel maximum mean discrepancy of feature representation between source domain and target domain in Hilbert space. At the same time, the domain classification layer  $f_{DC}$  (called DC layer), which is derived from the idea of domain confusion, is added after the seventh full connection layer. Besides, two residual layers are added after FC8 to reduce the difference between the source classifier and target classifier. Then the classification loss, domain confusion loss and entropy loss of target classed are also calculated. In order to implement the function of the layer  $f_{DC}$ , it is necessary to add the corresponding domain labels 0 and 1 to the data of source and target domains in the model file. And the input data of layer  $f_{DC}$  include the feature representation obtained by the seventh full connection layer and the labels of the two domains.

Here we use fine-tuning method to train this model. Considering the problem of the amount of data in the training set, we get the parameters of the first three convolutional layers from the pre-training model and freeze them. Then we fine-tune the subsequent convolutional layers and the full connection layers by the way of back-propagation. The random gradient descent method is used in the experiment, which parameter is 0.9. And the learning rate adopts 'inv' method, which parameter is 0.75, and the initial learning rate is 0.001. We set the batch size to 64 for all methods, and optimize the learning rate for each model independently. Since all the comparative methods use mmd as test statistic, Gaussian kernel is used to median pairwise squared distances on training data. We perform cross-valuation on labeled source data to select candidate parameters, the weights are selected following the strategy: (1) Try to get the best results by adjusting the mmd weight when we apply the DAN method. Then the mmd weight is fixed, which means we use the same mmd weight in all other methods. (2) Try to get the best results by adjusting the entropy weight when we realize the RTN method and fix the entropy weight. (3) Realize our method  $C^2$ DAN based on the mmd weight and entropy weight. The detail to choose weights is introduced in Section 4.4.

#### 4.3. Experiment Results and Analysis

This experiment implements the unsupervised domain adaptation of the improved deep adaptation network  $C^2$ DAN combining DC, CA and MK-MMD, which means DAN+DC+CA, on the Office-31 dataset and Office-10+Caltech-10 dataset. And comparing with the ordinary convolution neural network CNN (Baseline), the classical deep adaptation network DAN [12] and RTN [14], the experimental results are shown in Tables 1 and 2.



**Table 1.** Result of unsupervised domain adaptation experiment on Office-31 dataset.

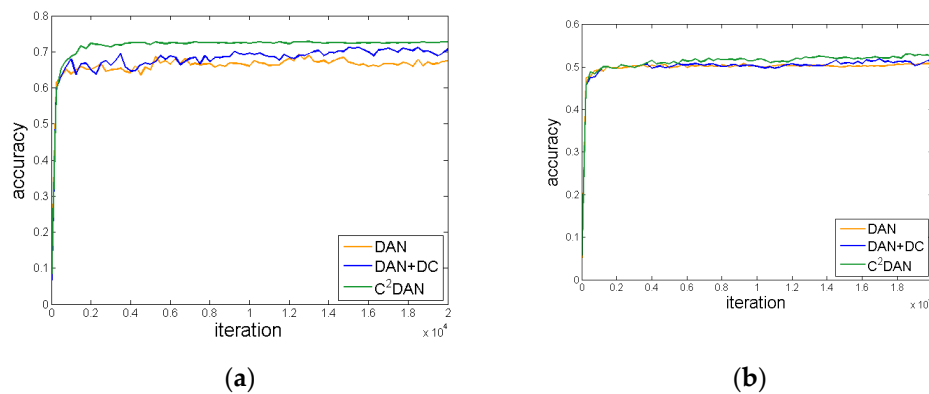
	A-W	D-W	W-D	A-D	D-A	W-A	Average
Baseline	60.6(~0.6)	95.0(~0.5)	99.1(~0.2)	59.0(~0.7)	49.7(~0.3)	46.2(~0.5)	68.2
DAN [12]	66.9(~0.6)	96.3(~0.4)	99.3(~0.2)	66.3(~0.5)	52.2(~0.3)	49.4(~0.4)	71.6
RTN [14]	70.0(~0.4)	96.8(~0.2)	99.6(~0.1)	69.8(~0.2)	50.2(~0.4)	50.0(~0.6)	72.7
DAN+DC (fc6)	67.3(~0.6)	96.0(~0.3)	99.1(~0.2)	66.0(~0.7)	51.5(~0.3)	49.6(~0.5)	71.5
DAN+DC (fc7)	69.0(~0.7)	96.2(~0.4)	99.5(~0.2)	67.0(~0.6)	52.5(~0.5)	50.2(~0.5)	72.5
RTN+DC	73.0(~0.7)	97.3(~0.5)	99.6(~0.1)	70.8(~0.2)	50.4(~0.4)	51.8(~0.6)	73.8
C <sup>2</sup> DAN	74.0(~0.6)	96.6(~0.7)	99.6(~0.1)	71.5(~0.3)	53.0(~0.6)	52.2(~0.4)	74.4

**Table 2.** Result of unsupervised domain adaptation experiment on Office-10+Caltech10 dataset.

	A-C	W-C	D-C	C-A	C-W	C-D	Average
Baseline	82.6(~0.3)	75.8(~0.3)	77.1(~0.5)	90.5(0.1)	79.6(0.2)	83.5(0.5)	81.5
DAN [12]	86.0(~0.5)	81.5(~0.2)	81.8(~0.3)	92.0(~0.5)	90.6(~0.5)	90.2(~0.3)	87.0
RTN [14]	88.1(~0.2)	85.6(~0.1)	84.1(~0.2)	93.0(~0.1)	96.3(~0.3)	94.2(~0.2)	90.2
DAN+DC (fc6)	85.0(~0.1)	80.4(~0.3)	80.0(~0.3)	91.7(~0.3)	85.6(~0.2)	88.6(~0.2)	85.2
DAN+DC (fc7)	86.4(~0.2)	82.2(~0.5)	82.5(~0.1)	92.8(~0.3)	92.3(~0.5)	91.3(~0.5)	87.9
RTN+DC	88.4(~0.4)	86.5(~0.2)	85.3(~0.3)	93.7(~0.3)	96.3(~0.1)	95.0(~0.2)	90.8
C <sup>2</sup> DAN	88.7(~0.3)	86.3(~0.5)	85.0(~0.5)	93.5(~0.2)	97.0(~0.3)	95.6(~0.1)	91.0

From the experimental results in Tables 1 and 2, we can make the following observations: (1) For different domain adaptation scenarios, the domain adaptation method has different effects. For the adaptation between domain D and W, the effect is not obvious because the similarity between the two domains is very high. Then for the adaptation of domain A to D or W, the result is obviously better. (2) Comparing to the DAN and RTN, the average accuracy of the six adaptation scenarios is improved by about 1% with the use of DC. Combining DC with MMD makes the features obtained from source domain more capable of representing the target domain samples, and further reduces the difference of domain distribution. (3) The average accuracy of the method RTN combined with DC is higher than the DAN combined with DC. The main reason is that RTN has classifier adaptation module which can enhance the accuracy of classification. From the data point of view, the use of classifier adaptation can further improve the effect of domain adaptation.

To go deeper into different modules of C<sup>2</sup>DAN, we show the results of each module of C<sup>2</sup>DAN in Tables 1 and 2. (1) The average accuracy of the method combined with DC6 is not as good as the DAN method using only MK-MMD or the method combining with DC7. Theoretically, with the increase of the number of convolutional layers, the features extracted from each convolutional layer change from generalization to specialization. Therefore, DC7 is selected to make the best domain classifier achieve the maximum domain confusion. (2) Seen from different adaptation scenarios, the adaptation effect of domain A to W is the best. And the comparison of accuracy can be seen in Figure 5a. For the adaptation from domain W to A, the experimental results are not improved obviously, which are shown in Figure 5b. Due to the small amount of data in the source domain W and the poor ability of feature representation for real-world images, the improvement is not obvious when the adapting from the real-world images in domain W to the information-rich website images in domain A. (3) Comparing RTN+DC with C<sup>2</sup>DAN, the average accuracy of C<sup>2</sup>DAN is 73.8% on the Office-31 dataset, which is 0.6% higher than RTN+DC, but the accuracy of C<sup>2</sup>DAN is similar to RTN+DC in Office-10+Caltech-10, which seems little improvement. The difference between the two datasets is Office-10+Caltech-10 has less categories which make it easier to be transferred. Both of these two methods have combined domain confusion with classifier adaptation, the difference is C<sup>2</sup>DAN conducts feature alignment by MK-MMD and RTN by tensor MMD. From the data point of view, the combination of MK-MMD, DC and CA is more effective in more difficult tasks.



**Figure 5.** Accuracy of different iterations using different method, (a) A-W; (b) W-A.

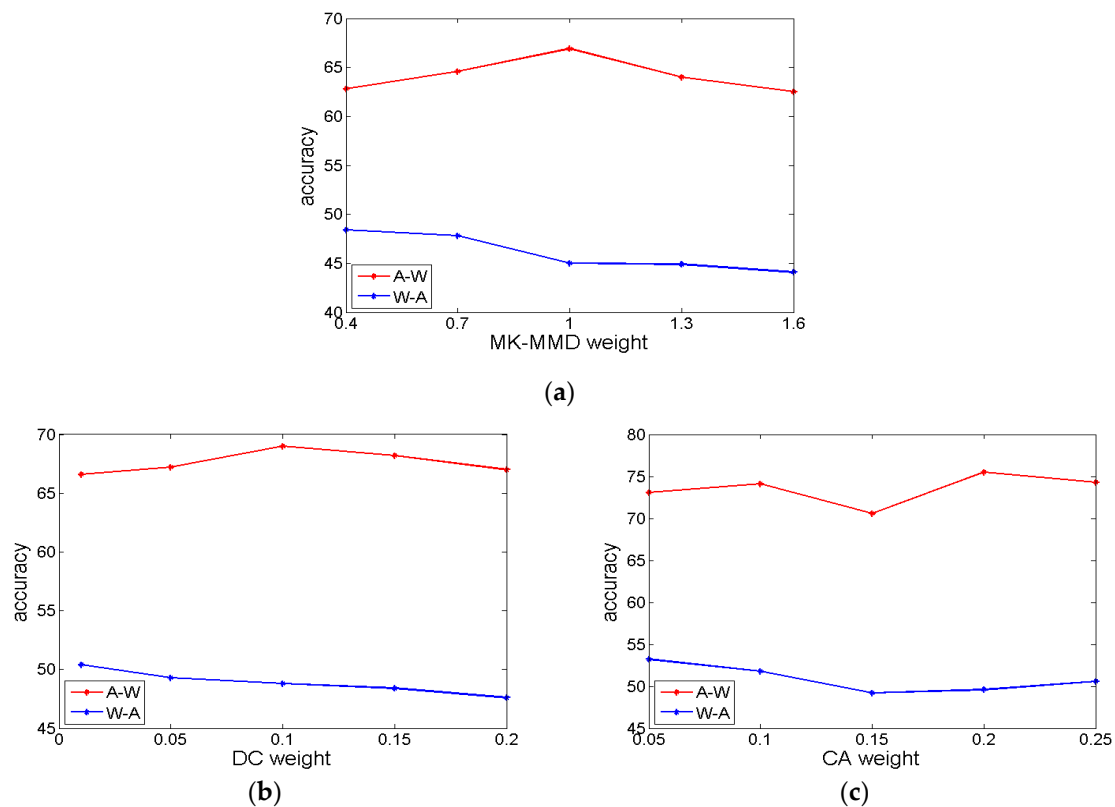
#### 4.4. Analysis of Weights

In this paper, we explore and analyze the suitable weights of three different domain adaptation loss. The weight parameter of MK-MMD ranges from {0.1, 0.4, 0.7, 1, 1.4, 1.7, 2}, the weight parameter of domain confusion ranges from {0.01, 0.05, 0.1, 0.15, 0.2} and the weight parameter of classifier adaptation ranges from {0.05, 0.1, 0.15, 0.2, 0.25}. It's necessary to select appropriate weights and best combination in different scenarios.

For the weight parameter of MK-MMD, as shown in [12], the adaptation effect of domain A (website images with rich information) to domain D (complex images captured by digital SLR camera) is best when  $\lambda$  is 1, as shown in Figure 6a. However, this conclusion is not suitable when the source domain is W or D. From the experimental result, we can see that the effect of domain adaptation becomes worse with the increase of weight and the result is best when  $\lambda$  is 0.1. Therefore, in the actual application, the weight of MK-MMD  $\lambda$  should be 1 in the case of abundant information in the source domain, such as website images. Then, the weight  $\lambda$  should be 0.1 in the case of less information and complex environment in the source domain, such as images captured by web camera or digital SLR camera.

For the weight parameter  $\gamma$  of  $f_{DC}$  layer, we should consider the optimal combination of the two weight parameters. In this paper, we conduct experiments and analyze to get the best fit  $\gamma$  under the optimal  $\lambda$  for every adaptation scenario. When the source domain is A and  $\lambda$  is 1, the best result can be obtained if  $\gamma$  equals 0.1. The comparison of different weights  $\gamma$  from domain A to domain W is shown in Figure 6b. Meanwhile, when the source domain is W or D and  $\lambda$  is 0.1, the best result can be obtained if  $\gamma$  equals 0.01. The comparison of different weights  $\gamma$  from domain W to domain A is shown in Figure 6b.

For the weight parameter  $\beta$  of entropy loss, we should consider the optimal combination of the three weight parameters. In this paper, we conduct experiments and analyze to get the best fit  $\beta$  under the optimal  $\lambda$  and  $\gamma$  for every adaptation scenario. When the source domain is A and  $\lambda$  is 1 and  $\gamma$  equals 0.1, the best result can be obtained if  $\beta$  equals 0.2. The comparison of different weights  $\beta$  from domain A to domain W is shown in Figure 6c. Meanwhile, when the source domain is W or D and  $\lambda$  is 0.1 and  $\gamma$  is 0.01, the best result can be obtained if  $\beta$  equals 0.05. The comparison of different weights  $\beta$  from domain W to domain A is shown in Figure 6c.



**Figure 6.** Accuracy comparison in different weights, (a) weights of MK-MMD; (b) weights of DC; (c) weights of CA.

From the experimental results, as a domain adaptation method, the weight of domain confusion loss should be one tenth of MK-MMD loss. In theory, domain confusion aims to make the most distinguishable feature representation get the worst classification effect. If the weight of  $f_{DC}$  is too high, the training will tend to domain confusion at the beginning, in which case, feature representation with rich and effective information cannot be learned. Therefore, the weight of domain confusion should be relatively small. At the same time, the fundamental purpose of domain adaptation is to obtain domain invariant feature representation. From this point of view, MK-MMD narrows the distribution difference in the mapping space by narrowing the distance of feature representation to obtain domain invariant feature, but domain confusion only judges the degree of domain confusion and improves the domain invariance of feature representation by optimizing its loss. It can be said that MK-MMD acquire domain invariance actively, but domain confusion is a passive verification. MK-MMD plays a stronger role and domain confusion further improve the effect of domain adaptation at this level. As a result, the proportion of domain confusion loss to the total loss function is less than that of MK-MMD loss.

In summary, when domain adaptation is carried out from the domain with rich obvious information (e.g. network images) to the domain with less complex information (e.g. real word camera images), the weight of MK-MMD is 1, the weight of domain confusion is 0.1 and the weight of classifier adaptation is 0.2. For reverse domain adaptation scenario, the weight of MK-MMD is 0.1, the weight of domain confusion is 0.01 and the weight of classifier adaptation is 0.05. The weights above make the combination of MK-MMD, domain confusion and classifier adaptation more reasonable and lead to better result.

## 5. Application on Vehicle Classification

### 5.1. The Introduction of the Dataset

This paper uses Comprehensive Cars (CompCars) [32] dataset to carry out unsupervised domain adaptation vehicle classification experiments. CompCars dataset is one of the largest datasets for fine-grained vehicle recognition. This dataset consists of two parts, vehicle image dataset from website (data) and vehicle image dataset from road surveillance (sv\_data).

Each section contains 163 types of vehicles, each of which is subdivided into smaller categories according to specific models and years. The model of the experiment is based on AlexNet. The model level is not deep enough and the capability is limited. Therefore, 20 kinds of vehicles are selected for unsupervised domain adaptation experiments, instead of using the whole dataset. The examples of website vehicle image dataset in CompCars are shown in Figure 7.



**Figure 7.** website vehicle images, (a) Zhonghua, (b) Mitsubishi, (c) Besturn

From the above images, we can see that the shooting angle of the vehicle in the website dataset is not the same. The front, side and back angles are all included. But in the surveillance images, the vehicles are taken from the front with only slight angle change. The angle of view of side and back has changed too much and the distribution is totally different, so this kind of images is not suitable for domain adaptation training. Therefore, we need to filter the website images and only select the pictures taken from the front and the side view which include the vehicle head information.

The examples are shown in Figure 8. The examples of the surveillance vehicle dataset sv\_data are shown in Figure 9.



**Figure 8.** filtered website vehicle images.

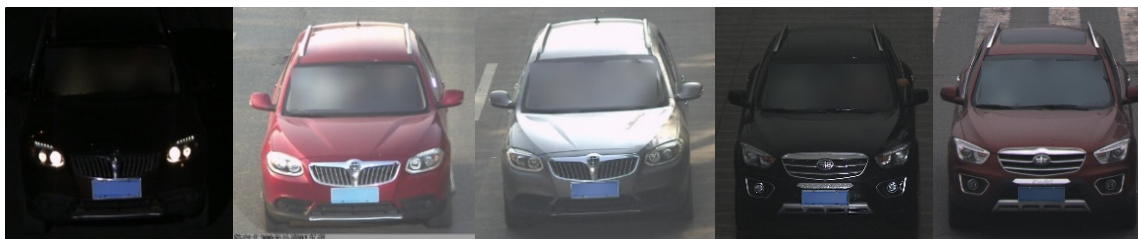


Figure 9. Examples of sv\_data.

From Figure 9, we can see that the surveillance pictures are taken from the front angle but under different weather and illumination conditions. In this experiment, the selected website images are used as labeled source domain and the surveillance vehicle images belong to unlabeled target domain. Although the website images are taken approximate frontally, they are collected under good lighting conditions and the angle is biased, which results in the distribution differences between the source domain and the target domain due to the angle of view, illumination and other factors. The vehicle classification experiments using these two parts of data can prove the validity of the model in solving the problem of domain shift.

Details of the dataset used in this experiment are as follows. The website dataset(data) contains 20 categories, totaling 6425 pictures, and the surveillance dataset(sv\_data) contains 20 categories, totaling 6960 pictures. The categories and their quantities are shown in the Table 3 below.

Table 3. The categories and quantities.

	Acura	Benz	Besturn	BYD	Changan
data	157	570	72	356	405
sv_data	370	155	68	395	465
	Dongfengfengdu	Geely	Haima	Honda	Hyundai
data	46	426	69	360	645
sv_data	92	576	203	380	572
	Jeep	Lexus	MAZDA	Mitsubishi	Nissan
data	200	283	314	275	431
sv_data	304	188	371	281	462
	Shuanglong	Toyota	Volkswagen	Volvo	Zhonghua
data	190	511	553	370	193
sv_data	264	572	533	598	111

## 5.2. Experiments Details and the Result

We use four methods to conduct comparative experiments: convolutional neural network CNN (baseline), DAN, the proposed deep adaptation network combining domain confusion with MK-MMD and the combination of DC, CA and MK-MMD. According to the analysis result, the experiment adapts from the domain with rich information network images to the domain with surveillance images including less rich information. Therefore, in our experiment, the loss weight of MK-MMD is 1, the loss weight of domain confusion is 0.1, and the loss weight of classifier adaptation is 0.1.

The experiment is carried out under the framework of caffe. We need to add MK-MMD layer, domain confusion layer and residual layers, and then recompile caffe. Twenty types of vehicles are labeled with "0" ~ "19". During the training process, there is no label information in the vehicle image of the target domain. The accuracy results of vehicle classification experiments are shown in Table 4.



**Table 4.** Comparison of vehicle classification accuracy.

Method	Accuracy
CNN (Baseline)	0.351
DAN [12]	0.449
RTN [14]	0.443
DAN+DC	0.476
RTN+DC	0.456
C <sup>2</sup> DAN (DAN+DC+CA)	0.507

First of all, we need to point out that although the CompCars dataset is used for fine-grained classification, the purpose of our experiment is to verify the ability of unsupervised domain adaptation method to solve the model degradation problem when the source domain training model is applied in the target domain (without any labels). The emphasis is on the improvement of the domain adaptation effect, so it is inappropriate to compare with advanced fine-grained classification model.

Meanwhile, it can be seen from the examples of website images and surveillance images that the source and target images used in this experiment have various changes in background, illumination, angle of view, etc. The domain shift is large, so the overall accuracy of this experiment is not high. However, even in this complex case, the method in this paper has achieved better results. In the follow-up study, assuming that the available source and target domain data are purer or more task-specific, the domain adaptation effect of this method will be better. The experimental results are analyzed as below.

From the experimental results, it can be seen that the accuracy of unsupervised domain adaptation in vehicle classification task from the labeled source domain to the unlabeled target domain is 35.1% without any domain adaptation components. The DAN method based on the MK-MMD component has a classification accuracy of 44.9%, which is 9.8% higher than that of baseline. This proves that the domain adaptation can overcome inter-domain distribution differences caused by the variation of illumination and angle of view. The classification accuracy of DAN+DC and RTN+DC are both higher than the original methods DAN and RTN, which can prove the effectiveness of DC. The accuracy of our method C<sup>2</sup>DAN is 50.7%, 5.8% higher than DAN and 6.4% higher than RTN. It can be inferred that the classification accuracy is greatly enhanced because the huge difference between source domain and target domain has been reduced by CA and DC. The results demonstrate that the combination of MK-MMD, DC and CA is reasonable and efficient. Moreover, comparing with the standard domain adaptation datasets Office-31 and Office-10+Caltech-10, the improvement of C<sup>2</sup>DAN on CompCars is more significant, which means C<sup>2</sup>DAN has powerful adaptation ability on challenging tasks.

### 5.3. Accuracy and Analysis of Various Categories

Here, we compare and analyze the classification accuracy of each vehicle type. The accuracy results of each vehicle type are shown in the following Table 5.

From the results above, the following conclusions can be drawn:

- (1) It can be seen that the classification accuracy of 17 types of vehicles has been improved after using the proposed DAN+DC or C<sup>2</sup>DAN methods. Compared with that of the DAN method, only one type has decreased, the data in the table show that the difference is little and the decline is not serious. The accuracy of 85% vehicle types have been improved, which proves that the proposed method is reasonable and effective.
- (2) For vehicle types with less data, such as the Besturn and Dongfengfengdu cars, in which type the number of samples in source domain and target domain are both less than 2% of the total number of datasets, the proposed method improves the accuracy of vehicle classification by 28.5% and 5.5% respectively compared with the DAN method, and improves by 38.7% and 13.1% compared with the method using only CNN. It proves that the proposed method's superiority is obvious. The feature extracted by the model in a limited number of samples greatly improves the



representation ability in the target domain. By enhancing the feature invariance, the distribution difference between the source domain and the target domain is further reduced.

- (3) For the performance degradation of some classes, the main reason is that MK-MMD is an active acquisition while domain confusion is a passive verification for domain invariant feature. Besides, when the domain invariant property of the feature extracted by MK-MMD has reached the optimum level, the improvable space is very limited and the balance training of two kinds of loss may sacrifice the ability of domain adaptation in some categories. However, the sacrifice degree of this part is not too large. It is within the acceptable range and the accuracy of most categories has been improved.

**Table 5.** Comparison of classification accuracy for 20 vehicle types.

	CNN (Baseline)	DAN	DAN + DC	C <sup>2</sup> DAN
Acura	0.511	0.600	0.614	0.608
Benz	0.265	0.696	0.587	0.781
Besturn	0.118	0.220	0.505	0.206
BYD	0.083	0.387	0.332	0.504
Changan	0.606	0.326	0.328	0.338
Dongfengfengdu	0.054	0.130	0.185	0.054
Geely	0.474	0.534	0.520	0.641
Haima	0.000	0.039	0.060	0.014
Honda	0.431	0.281	0.389	0.409
Hyundai	0.271	0.470	0.472	0.530
Jeep	0.740	0.815	0.803	0.869
Lexus	0.617	0.399	0.479	0.724
MAZDA	0.218	0.498	0.496	0.517
Mitsubishi	0.238	0.476	0.605	0.514
Nissan	0.530	0.510	0.574	0.500
Shuanglong	0.273	0.401	0.409	0.391
Toyota	0.196	0.222	0.234	0.195
Volkswagen	0.580	0.656	0.658	0.714
Volvo	0.582	0.698	0.652	0.679
Zhonghua	0.234	0.612	0.622	0.712

Generally speaking, the proposed method indeed improves the effect of domain adaptation in the whole vehicle classification dataset.

## 6. Conclusions

This paper proposes an improved deep adaptive network C<sup>2</sup>DAN, which combines MK-MMD with Domain Confusion (DC) and Classifier Adaptation (CA). DC is added to learn domain invariant features by using a domain classification layer to perform adversarial training. We add residual blocks in source classification layer to preform CA which can reduce the discrepancy between source classifier and target classifier.

Experiments on standard domain adaptation datasets Office-31, Office-10+Caltech-10 and vehicle classification dataset CompCars show that: (1) DC and CA are both efficient to reduce the domain shift, which means the proposed method can improve the adaptation ability when knowledges are transferred from labeled source domain to unlabeled target domain. (2) The most suitable weights of MK-MMD and the best weight combination of MK-MMD, DC and CA in different transfer scenarios are explored from theory and experiments. (3) When used in challenging transfer tasks, C<sup>2</sup>DAN shows great advantages to reduce the huge differences between two domains. It can be inferred that DC is helpful to align the feature distribution with the use of MK-MMD. CA is also necessary especially in difficult transfer scenarios, we can't use shared classifier to get good performance because the classification standards of source domain and target domain could be quite different.

In the future work, domain adaptation is going to face challenges that are harder than ever, DC and CA are both illuminating ideas which can be added to new domain adaptation methods. Furthermore, our work is expected to be applied in other computer vision tasks such as fine-grained classification and object detection.

**Author Contributions:** Conceptualization, H.S.; Data curation, H.S., X.C., L.W. and D.L.; Formal analysis, H.S., X.C. and L.W.; Funding acquisition, H.S.; Investigation, X.C. and L.W.; Project administration, D.L. and N.L.; Resources, H.Z.; Supervision, H.S. and H.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work is supported by the Fundamental Research Funds for the Central Universities of China under Grant NS2016091. H. Zhou was supported by UK EPSRC under Grant EP/N011074/1, Royal Society-Newton Advanced Fellowship under Grant NA160342, and European Union's Horizon 2020 research and innovation program under the Marie-Sklodowska-Curie grant agreement No 720325.

**Acknowledgments:** The authors would like to thank the handling associate editor and all the anonymous reviewers for their constructive comments.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.S.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [[CrossRef](#)]
2. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778. [[CrossRef](#)]
3. Cheng, X.; Ren, Y.; Cheng, K.; Cao, J.; Hao, Q. Method for Training Convolutional Neural Networks for In Situ Plankton Image Recognition and Classification Based on the Mechanisms of the Human Eye. *Sensors* **2020**, *20*, 2592. [[CrossRef](#)] [[PubMed](#)]
4. Everingham, M.; Eslami, S.M.A.; Gool, L.V.; Williams, C.K.I.; Winn, J.M.; Zisserman, A. The Pascal Visual Object Classes Challenge: A Retrospective. *Int. J. Comput. Vis.* **2015**, *111*, 98–136. [[CrossRef](#)]
5. Khaki, S.; Pham, H.; Han, Y.; Kuhl, A.; Kent, W.; Wang, L. Convolutional Neural Networks for Image-Based Corn Kernel Detection and Counting. *Sensors* **2020**, *20*, 2721. [[CrossRef](#)] [[PubMed](#)]
6. Krause, J.; Stark, M.; Deng, J.; Li, F. 3D Object Representations for Fine-Grained Categorization. In Proceedings of the 2013 IEEE International Conference on Computer Vision Workshops, Sydney, Australia, 1–8 December 2013; pp. 554–561. [[CrossRef](#)]
7. Gebu, T.; Hoffman, J.; Li, F. Fine-Grained Recognition in the Wild: A Multi-task Domain Adaptation Approach. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 1358–1367. [[CrossRef](#)]
8. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 3213–3223. [[CrossRef](#)]
9. Ansari, R.A.; Malhotra, R.; Buddhiraju, K.M. Identifying Informal Settlements Using Contourlet Assisted Deep Learning. *Sensors* **2020**, *20*, 2733. [[CrossRef](#)] [[PubMed](#)]
10. Patel, V.M.; Gopalan, R.; Li, R.; Chellappa, R. Visual Domain Adaptation: A survey of recent advances. *IEEE Signal Process. Mag.* **2015**, *32*, 53–69. [[CrossRef](#)]
11. Ghifary, M.; Kleijn, W.B.; Zhang, M. Domain Adaptive Neural Networks for Object Recognition. In *PRICAI 2014: Trends in Artificial Intelligence, Proceedings of the 13th Pacific Rim International Conference on Artificial Intelligence, Gold Coast, QLD, Australia, 1–5 December 2014*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 898–904. [[CrossRef](#)]
12. Long, M.; Cao, Y.; Wang, J.; Jordan, M.I. Learning Transferable Features with Deep Adaptation Networks. In Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6–11 July 2015; pp. 97–105.

13. Long, M.; Zhu, H.; Wang, J.; Jordan, M.I. Deep Transfer Learning with Joint Adaptation Networks. In Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017; pp. 2208–2217.
14. Long, M.; Zhu, H.; Wang, J.; Jordan, M.I. Unsupervised Domain Adaptation with Residual Transfer Networks. In *Advances in Neural Information Processing Systems 29, Proceedings of the 30th International Conference on Neural Information Processing, Barcelona, Spain, 2016*; Lee, D.D., Sugiyama, M., Eds.; Curran Associates, Inc.: New York, NY, USA, 2016; pp. 136–144.
15. Zhang, X.; Yu, F.X.; Chang, S.; Wang, S. Deep Transfer Network: Unsupervised Domain Adaptation. *arXiv* **2015**, arXiv:1503.00591.
16. Yan, H.; Ding, Y.; Li, P.; Wang, Q.; Xu, Y.; Zuo, W. Mind the Class Weight Bias: Weighted Maximum Mean Discrepancy for Unsupervised Domain Adaptation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 22–25 July 2017; pp. 945–954. [[CrossRef](#)]
17. Borgwardt, K.M.; Gretton, A.; Rasch, M.J.; Kriegel, H.; Schölkopf, B.; Smola, A.J. Integrating structured biological data by Kernel Maximum Mean Discrepancy. *ISMB. Bioinformatics* **2006**, *22*, 49–57. [[CrossRef](#)] [[PubMed](#)]
18. Sun, B.; Feng, J.; Saenko, K. Return of Frustratingly Easy Domain Adaptation. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, Arizona, USA, 12–17 February 2016; pp. 2058–2065.
19. Sun, B.; Saenko, K. Deep CORAL: Correlation Alignment for Deep Domain Adaptation. In *Computer Vision—ECCV 2016 Workshops, Proceedings of the 14th European Conference on Computer Vision, Amsterdam, The Netherlands, 8–16 October 2016*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 9915, pp. 443–450. [[CrossRef](#)]
20. Zhuang, F.; Cheng, X.; Luo, P.; Pan, S.J.; He, Q. Supervised Representation Learning: Transfer Learning with Deep Autoencoders. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015), Buenos Aires, Argentina, 25–31 July 2015; pp. 4119–4125.
21. Hoffman, J.; Tzeng, E.; Darrell, T.; Saenko, K. Simultaneous Deep Transfer Across Domains and Tasks. In *Domain Adaptation in Computer Vision Applications*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 173–187. [[CrossRef](#)]
22. Ganin, Y.; Lempitsky, V.S. Unsupervised Domain Adaptation by Backpropagation. In *JMLR Workshop and Conference Proceedings, Proceedings of The 31st International Conference on Machine Learning, Beijing, China, on 21–26 June 2014*; Microtome Publishing: Brookline, MA, USA; pp. 1180–1189.
23. Ganin, Y.; Ustinova, E.; Ajakan, H.; Germain, P.; Larochelle, H.; Laviolette, F.; Marchand, M.; Lempitsky, V.S. Domain-Adversarial Training of Neural Networks. *J. Mach. Learn. Res.* **2016**, *17*, 1–35.
24. Tzeng, E.; Devin, C.; Hoffman, J.; Finn, C.; Abbeel, P.; Levine, S.; Saenko, K.; Darrell, T. *Adapting Deep Visuomotor Representations with Weak Pairwise Constraints. WAFR. Springer Proceedings in Advanced Robotics*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 13, pp. 688–703. [[CrossRef](#)]
25. Tzeng, E.; Hoffman, J.; Saenko, K.; Darrell, T. Adversarial Discriminative Domain Adaptation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 22–25 July 2017; pp. 2962–2971. [[CrossRef](#)]
26. Wang, L.; Sindagi, V.; Patel, V.M. High-Quality Facial Photo-Sketch Synthesis Using Multi-Adversarial Networks. In Proceedings of the 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), Xi'an, China, 15–19 May 2018; pp. 83–90. [[CrossRef](#)]
27. Cao, Z.; Long, M.; Wang, J.; Jordan, M.I. Partial Transfer Learning With Selective Adversarial Networks. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 2724–2732. [[CrossRef](#)]
28. Zhang, J.; Ding, Z.; Li, W.; Ogunbona, P. Importance Weighted Adversarial Nets for Partial Domain Adaptation. In Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018; pp. 8156–8164. [[CrossRef](#)]
29. Bousmalis, K.; Silberman, N.; Dohan, D.; Erhan, D.; Krishnan, D. Unsupervised Pixel-Level Domain Adaptation with Generative Adversarial Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 22–25 July 2017; pp. 95–104. [[CrossRef](#)]

30. Chadha, A.; Andreopoulos, Y. Improved Techniques for Adversarial Discriminative Domain Adaptation. *IEEE Trans. Image Process.* **2020**, *29*, 2622–2637. [[CrossRef](#)] [[PubMed](#)]
31. Saenko, K.; Kulis, B.; Fritz, M.; Darrell, T. Adapting Visual Category Models to New Domains. In *Computer Vision-ECCV 2010, Proceedings of the 11th European Conference On Computer Vision, Heraklion, Crete, Greece, 5–11 September 2010*; Springer: Berlin/Heidelberg, Germany, 2010; Volume 6314, pp. 213–226. [[CrossRef](#)]
32. Yang, L.; Luo, P.; Loy, C.C.; Tang, X. A large-scale car dataset for fine-grained categorization and verification. In *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, MA, USA, 7–12 June 2015; pp. 3973–3981. [[CrossRef](#)]
33. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; WardeFarley, D.; Ozair, S.; Courville, A.C.; Bengio, Y. Generative Adversarial Nets. In *Proceedings of the 28th Annual Conference on Neural Information Processing Systems*, Montreal, QC, Canada, 8–13 December 2014; pp. 2672–2680.
34. Isola, P.; Zhu, J.; Zhou, T.; Efros, A.A. Image-to-Image Translation with Conditional Adversarial Networks. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, 22–25 July 2017; pp. 5967–5976. [[CrossRef](#)]
35. Mirza, M.; Osindero, S. Conditional Generative Adversarial Nets. *arXiv* **2014**, arXiv:1411.1784v1.
36. Liu, M.; Tuzel, O. Coupled Generative Adversarial Networks. In *Advances in Neural Information Processing Systems 29, Proceedings of the 30th International Conference on Neural Information Processing, Barcelona, Spain, 2016*; Lee, D.D., Sugiyama, M., Eds.; Curran Associates, Inc.: New York, NY, USA, 2016; pp. 469–477.
37. Zhu, J.; Park, T.; Isola, P.; Efros, A.A. Unpaired Image-to-Image Translation Using CycleConsistent Adversarial Networks. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 22–29 October 2017; pp. 2242–2251. [[CrossRef](#)]
38. Yi, Z.; Zhang, H.R.; Tan, P.; Gong, M. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. In *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 22–29 October 2017; pp. 2868–2876. [[CrossRef](#)]
39. Kim, T.; Cha, M.; Kim, H.; Lee, J.K.; Kim, J. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, Sydney, NSW, Australia, 6–11 August 2017; pp. 1857–1865.
40. Shen, J.; Qu, Y.; Zhang, W.; Yu, Y. Wasserstein Distance Guided Representation Learning for Domain Adaptation. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018)*, New Orleans, LA, USA, 2–7 February 2018; pp. 4058–4065.
41. Arjovsky, M.; Chintala, S.; Bottou, L. Wasserstein Generative Adversarial Networks. Available online: <http://proceedings.mlr.press/v70/arjovsky17a.html> (accessed on 25 June 2020).



© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<http://creativecommons.org/licenses/by/4.0/>).