Deep Local Descriptors with Domain Adaptation

Shuwen Qiu and Weihong Deng(✉)

School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China
whdeng@bupt.edu.cn

Abstract. Due to the different distributions of training and testing datasets, the performance of the trained model based on the training set can rarely achieve the most optimal. Inspired by the successful application of domain adaptation in the object recognition area, we apply domain adaptation methods to CNN based local feature descriptors based on their own traits. Different from previous domain adaptation methods that focus only on the fully connected layer, we apply maximum mean discrepancy (MMD) criterion to both the fully connected layer and the convolutional layer, which makes the primary local filters of CNN adaptive to the target dataset in an unsupervised manner. Extensive experiments on Photo Tour and HPatches dataset show that domain adaption is effective to local feature descriptors, and, more importantly, the convolutional layer adaption can further improve the performance of traditional domain adaptation.

Keywords: Local feature descriptor · Domain adaptation
Maximum Mean Discrepancy

1 Introduction

One of the most extensively studied problems in computer vision area is to find correspondences between images according to local feature descriptors that embed local features into vectors. Compared with global features, local feature represents only part of the image so that it is more robust to illuminations. Recently, local descriptors based on CNN architectures have been proved to significantly outperform handcrafted local descriptors [4,21,23], meanwhile large datasets are available for training [20,22]. However, due to the distribution discrepancies of different datasets, trained models based on patches from training sets may not generalize the most optimal results in the testing sets, which is mainly caused by potential variations between domains. For example, patches of the training sets are extracted from images of buildings while patches from testing sets are mainly from decorations indoors or natural scenery. Therefore, it is a natural adoption of domain adaptation methods to explore the domain-invariant structure between the source domain (labeled training set) and the target domain (unlabeled testing set).

J.-H. Lai et al. (Eds.): PRCV 2018, LNCS 11257, pp. 344–355, 2018.
https://doi.org/10.1007/978-3-030-03335-4_30

Recent studies have demonstrated that the deep neural networks can learn transferable features to establish knowledge transfer by exploring the invariant factors between different datasets so as to make features robust to noise [13]. However, most of the studies focus on object recognition and a systematic study of the application of domain adaptation in local descriptors is yet to be done. Therefore, in this work, we will investigate the application of domain adaptation in local descriptors. Our contributions include: (1) we investigate the performance of different CNN based local descriptors, combining maximum mean discrepancy (MMD) criterion. Extensive experiments on Photo Tour and HPatches dataset show that domain adaption is effective to local feature descriptors; (2) Different from previous domain adaptation methods that focus only on the fully connected layer, we jointly calculate MMD from both the fully connected layer and the Convolutional layer of the network considering local descriptors' own traits, which can further improve the performance of traditional domain adaptation.

2 Related Work

2.1 Local Descriptors

End-to-end learning local descriptors based on CNN architectures have been investigated in many studies, and the improvement has been shown over state of art descriptors [4,21,23]. In [21], feature layers and metric layers are learnt in the same network. Therefore, the final hinge-based loss can be optimized using the last abstract metric layer of the network. MatchNet [24] also includes both feature extracting layers and metric layers while using entropy loss to update the network.

On the contrary, [4,23] directly use the last feature layer as the feature descriptor of the input patch without training of the metric layer so that it can be judged by traditional evaluation criterion. Based on Siamese network, Deepdesc [4] trains the network using L2 distance meanwhile adopting a mining strategy to select training samples. However, it requires large quantity of samples to guarantee its performance. TFeat [23] uses Triplet network to decrease the distance between matching pairs and increase the distance between non-matching pairs. Based on triplet loss, L2-Net [26] also proposes a progressive sampling method with consideration of the intermediate layers.

Another important observation is that multi-scale network architectures can achieve better results compared with single-scale network architectures.

2.2 Domain Adaptation

Transfer learning [19] aims to build a learning model that can follow different probability distributions according to different domains [3,8,10,16,19]. Recent studies of deep domain adaptation embed an adaptive layer into the deep network to enhance the transfer ability [5–7,13,14,25]. The deep domain confusion

network (DDC) by Tzeng et al. [5] uses two CNNs with shared weights, according to the source and target domain respectively. The network of the source domain is updated by the originally defined loss function when the difference between the two domains is calculated by the MMD metric of the adaptive layer. DDC only adjusts a single layer of the network, which may limit the transfer ability of the multi-layer network. Therefore, Long et al. [13] proposed the deep adaptation network (DAN) combining multi-layer adaptation using multi-kernel MMD metric to match the shift of different domains. In order to avoid mutual influence of layers, a joint adaptation network (JAN) [12] based on a joint maximum mean discrepancy (JMMD) criterion was proposed to align the shift of the joint distribution of multiple layers in the network. Besides, there are several extensions of DAN aimed at aligning the distributions of both the classifier and the feature layer. In this work, we only investigate domain adaptation methods based on feature layers.

3 Model

3.1 Maximum Mean Discrepancy (MMD)

In standard CNN architecture, the features of the last layer tend to transfer from general to specific because it is tailored for the source data at the expense of degraded performance on the target task [13]. Hence, in order to get the most optimal performance, after pre-training on the training set, we require the distributions of the features of the fully connected layer from the source and the target domain to be similar. This can be achieved by adding an MMD metric to the original loss function, which can limit the target error by the source error plus a discrepancy metric between the source and the target [18].

MMD is an efficient metric that can compare the distributions of two datasets using a kernel two-sample test [11]. Given two distributions S and T, MMD is defined as:

$$MMD(X_S, X_T) = \left\| \frac{1}{|X_S|} \sum_{x_s \in X_S} \Phi(x_s) - \frac{1}{|X_T|} \sum_{x_t \in X_T} \Phi(x_t) \right\| \tag{1}$$

where Φ is a kernel function that maps the original data to a reproducing kernel Hilbert space (RKHS) and $\|\Phi\|_H \leq 1$ defines a set of functions in the unit ball of RKHS. This MMD metric considers the distribution of each domain to reduce the mismatch in a latent space. Subsequently, Tzeng et al. [5] and Long et al. [13] extended the MMD metric to a multi-kernel MMD metric. Multi-kernel MMD enhances the two-sample test power meanwhile minimizes the Type II error, i.e., the failure of rejecting a false null hypothesis [13]. Its final result is calculated by a weighted summation of several single kernel tests:

$$K \triangleq \left\{ k = \sum_{u=1}^{m} \beta_u k_u : \sum_{u=1}^{m} \beta_u = 1, \beta_u \geq 0, \forall u \right\} \tag{2}$$

where k_u stands for one single MMD test and the $\{\beta_u\}$ is limited in the way to make each kernel more representative. The multi-kernel MMD improves the testing power of MMD and leads to a more optimal result.

3.2 Adaptative Networks

Based on the idea of domain adaptation, we first combine MMD metric with TFeat [23] to exploit data from both the source and the target domain. Figure 1 gives an illustration of the proposed combined model. TFeat (Fig. 1-left) is a typical CNN based local descriptor. It is comprised of 2 Convolutional layers and one fully connected layer. For each layer, it is followed by an activation $f^l = tanh(x)$. The objective function of TFeat is:

$$\lambda(\delta^+, \delta^-) = max(0, \mu + \delta^+ - \delta^-) \tag{3}$$

where $\delta^+ = \|Net(x^+) - Net(x)\|_2$ is the L2 distance between the matching pairs (x^+, x), and $\delta^- = \|Net(x^-) - Net(x)\|_2$ is the L2 distance between the non-matching pairs (x^-, x), and μ is a constant. The objective function aims to make $\delta^- > \mu + \delta^+$, so the distance between non-matching pairs will be longer and between matching pairs will be shorter.

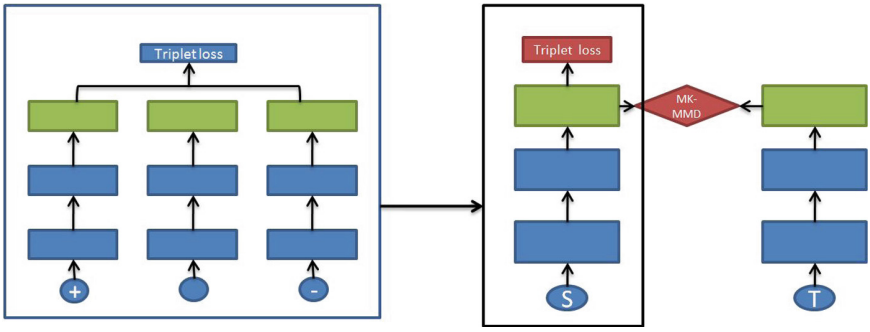


Fig. 1. Left is the original TFeat Network. It is comprised of 2 Convolutional layers(blue) and one fully connected layer(green). Right shows the modified adaptive model. (Color figure online)

In previous studies, deep networks are pre-trained on ImageNet [17], but our network is rather shallow so we only pre-train TFeat on the original training sets. Then we fix the Convolutional layers and update the fully connected layer using the new loss function,

$$L = L_C + \lambda MMD(X_S, X_T) \tag{4}$$

where L_C is the original loss function $\lambda(\delta^+, \delta^-)$. MMD is used for calculating the discrepancy between the training set and the testing set. $\lambda > 0$ is a penalty

parameter that can control the balance between the task specification and the discrepancy between two domains. As pointed out by Gretton et al. [1], kernel choice is important for the testing power of MMD because different kernel will map the probability distribution into different RKHS. Therefore, we choose the performance of multi-kernel MMD on the local descriptor learning.

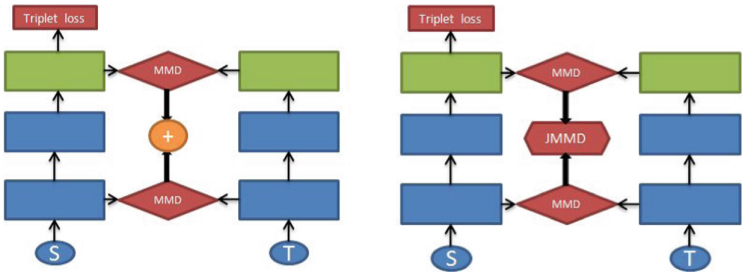
3.3 Joint Adaptation of the Fully Connected Layer and the Convolutional Layer

In [21], it has been pointed out that it is important to jointly use information from the first layer of the network. Therefore, we consider the modification of the MMD loss calculation to fit features from the first layer into the MMD metrics,

$$L = L_C + \lambda\varphi(MMD_{fc}(X_S, X_T), MMD_{cov}(X_S, X_T)) \tag{5}$$

where $\varphi(a, b)$ is a way to combine the MMD loss from both fully connected layer and the first Convolutional layer.

To train the network with multi-layer MMD, there are two ways. On the one hand, we could define $\varphi(a, b) = a + b$, which means we directly add up two MMD results from two separate layers, as Fig. 2(a) illustrates. On the other hand, As [12] points out, separate adaptation of different layers will exert a mutual influence on the conditional distribution of each layer, therefore $\varphi(a, b)$ could be defined as $\varphi(a, b) = a * b$, where $*$ means the joint distribution of the features from the two layers. The modified version is illustrated in Fig. 2(b).



(a) $\varphi(a, b) = a + b$ (b) $\varphi(a, b) = a * b$

Fig. 2. Two architectures to apply MMD loss

3.4 Dimension Reduction

In [1], it is proved that high dimension will decrease the power of MMD to calculate the discrepancies between different distributions. Allowing for the high dimensions of the Convolutional feature maps, we need to reduce the dimension before calculating the MMD metrics. For convenience, we consider simple ways

of average pooling. As the dataset (Fig. 3) shows, location information in patches are less important for domain adaptation since different subsets contains completely different scenes. Therefore, when considering dimension reduction, we could adopt average pooling to get a smoother distribution of pixel tensity in the patch.

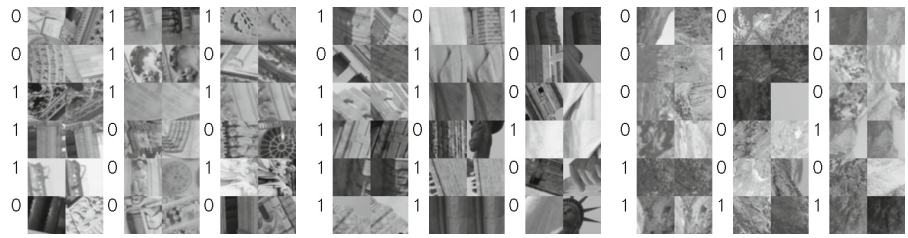
4 Experiment

We combine the CNN based local descriptors with MMD metric, focusing on the improvement of the performance that domain adaptation can offer.

4.1 Photo Tour Dataset

Photo Tour dataset [20] is a standard benchmark for patch training and testing. It consists of around 1M patches from each distinct scene: Notredame(N, grand building), Liberty(L, statue), Yosemite(Y, natural park), which we could think as three subsets. Each subset consists of three components: two patches and their label that shows whether they are matching pair(label = 1) or non-matching pair(label = 0). Figure 3 gives an illustration of the structure of the dataset, which mainly shows pairs of patches and their labels from three different subsets. For each learning task, we take one subset as training set and another as testing set so that there are 6 ways of subset combination. We evaluate the domain adaptation performance on the 6 learning tasks, N → L, N → Y, L → N, L → Y, Y → N, Y → L(training set→testing set).

We use FPR95 to calculate the error rate when the matching accuracy achieves 95%.



(a) Notredame(N) (b) Liberty(L) (c) Yosemite(Y)

Fig. 3. Photo tour dataset examples

4.2 HPatches Dataset

HPatches dataset [22] is a standard benchmark for patch testing. It consists of around 2M patches from 116 scenes. This dataset evaluates the local descriptors on three tasks: patch verification, image matching and patch retrieval. We evaluate the domain adaptation by training the networks on Photo Tour dataset and testing on Hpatches.

4.3 Evaluation Protocol

For evaluation on Photo Tour dataset, we mainly evaluate the performance following the protocol below.

TFeat Network. We first extract 5M triplets from the training set and find the best results in certain epochs as the pre-trained model following the original procedure in [23]. Then we use 5M labeled triplets from the training set and 5M random selected unlabeled triplets from the testing set to update the fully connected layer of the pre-trained network using new loss function and fix the Convolutional layer, and evaluate the descriptors' performance using FPR95. As for joint adaptation of both fully connected layer and Convolutional layer, we update the whole network after pre-training.

Siamese Network. Siamese Network [21] (as Fig. 4 shows) is another typical CNN based local descriptors. It consists of 3 Convolutional layers(blue), two maxpooling layers(red) and two fully connected layers(green) while the output of the last layer is a number representing whether these two patches are matching or not. Compared with TFeat, Siamese Network is trained with matching and non-matching pairs instead of triplets. Its objective function adopts a hinge-based loss. For adaptation, it follows the above protocol.

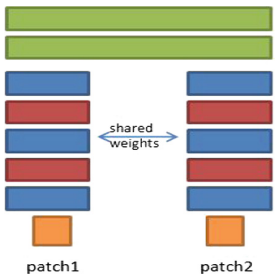


Fig. 4. Siamese network (Color figure online)

4.4 Parameters

When using multi-kernel MMD and considering a family of m Gaussian kernels $\{k_u\}_{u=1}^{m}$, we mainly follow the procedure in [13] to set the varying bandwidth γ_u. We use stochastic gradient descent (SGD) with 0.9 momentum and the learning rate is set to 0.1 at the beginning and is gradually decreased.

5 Results

5.1 Performance Changes on λ Variation

On TFeat Network, we first investigate the effect of the parameter λ. Table 1 illustrates the variation of the error rate with $\lambda \in \{0.005, 0.008, 0.01, 0.02\}$ on tasks N → and the number of MMD kernel is set to 3. We can see from the variation that when λ varies, the error rate first decreases and then increases forming a notching curve. It shows that it is important to find the balance between learning more specific deep features and adapting to target domain.

Table 1. Performance changes on λ variation.

λ	0	0.005	0.008	0.01	0.02
Error rate	7.3	6.14	5.95	5.87	5.88

5.2 Domain Adaptation on Photo Tour Dataset with Fully Connected Layer

For the convenience of implementation, we set λ to 0.01 for all tasks, which means results that Table 2 below shows can be decreased in an effective way even though the performance is not the most optimal. It demonstrates that MMD can effectively transfer features across domains and further boost the performance of our networks.

Table 2. Results of six learning tasks combining local descriptors with domain adaptation. The first row shows the original results and the second row shows results after domain adaptation.

Method	N → L	N → Y	L → Y	L → N	Y → N	Y → L
TFeat	7.30	7.34	8.52	3.10	3.10	9.09
TFeat-fc	5.87	6.56	6.95	2.70	2.79	8.10
Decrease(%)↓	19.60	10.60	18.40	12.90	10.0	10.90
Siamese	13.17	12.07	18.42	6.48	8.22	16.90
Siamese-fc	11.58	11.07	17.24	5.93	7.28	13.89
Decrease(%)↓	12.07	8.29	6.41	8.49	11.44	17.81

5.3 Domain Adaptation on HPatches Dataset with Fully Connected Layer

In [22], experiments show that TFeat Network has achieved higher results. Therefore, we tested TFeat Network on HPatches after domain adaptation. We could see from Fig. 5 that all of the three tasks have gained 2% increase, which proves the effectiveness of domain adaptation.Also, verification tasks between same sequence and matching tasks between illumination changes could gain bigger increase. Also, domain adaptation influences more on tough tasks.

5.4 Multi-layer Adaptation

In previous work, domain adaptation only considers fully connected layers. Given local descriptors' own traits, which implies that the first Convolutional layer contains important information, we add Convolutional layer to layer adaptation. Table 3 shows the comparison of different ways of layer combination. First of all,

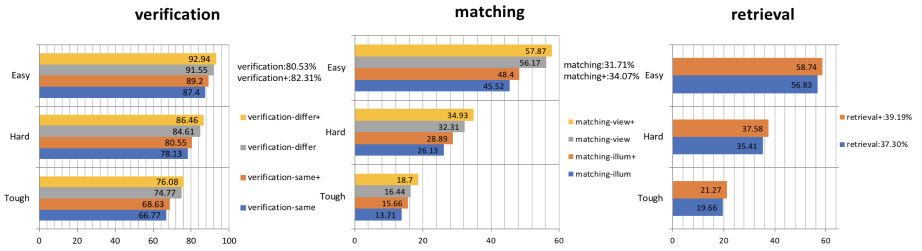


Fig. 5. HPatches evaluation:for each task, '+' represents results after domain adaptation. The mean values of the results before and after domain adaptation are put on the right, which shows that all the mean values get 2% increase. 'differ' and 'same' stand for different and same sequences verification. 'view' and 'illum' show matching under changes of view or illumination.

we adopt the traditional way to only update the fully connected layer. Then, we simply sum the MMD losses from the fully connected layer and the first Convolutional layer respectively. We can see from the results that error rate can be reduced because of the extra information the first layer offers. However, as [12] points out that the update of the former layers will change the distribution of the following layers so the joint MMD losses are also calculated following [12]. We can see from the results that improvement can be further achieved.

Table 3. Error rates with different ways of combining MMD losses from fully connected and Convolutional layers

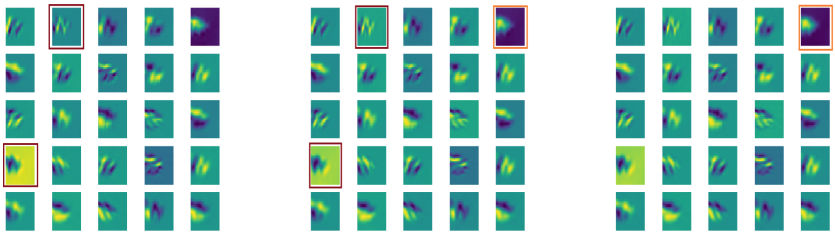
Method	TFeat-fc	TFeat-(fc+cov)	TFeat-(fc*cov)
FPR95(%)	5.87	5.51	**5.45**

5.5 Performance with Respect to Dimension Reduction

First we run several experiments of average pooling to find the variation tendency of the performance with different scale of pooling size. We can see from the figure that the error rate first decreases and then increases which means there is a balance between reducing the dimension and keeping enough feature information (Table 4).

Table 4. Error rates with different average pooling size. The first row shows the final dimension of the first layer after pooling.

Dimension	5408	1152	800	512	128
Error rate	5.76	5.71	5.77	5.8	6.02



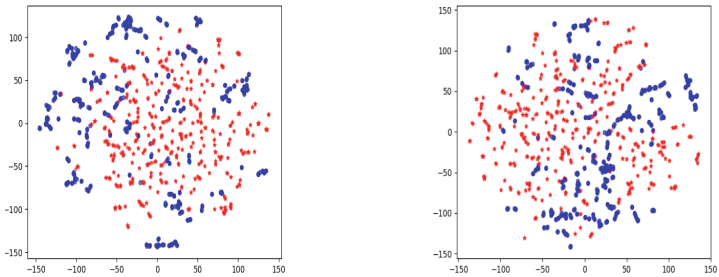
(a) fc layer adaptation (b) fc+cov adaptation (c) fc*cov adaptation

Fig. 6. The changes of feature maps with three ways of adaption

6 Discussion

6.1 First Layer Filter Visualization

For one original patch on the right, Fig. 6 shows the changes of feature maps with three ways of adaptation(TFeat-fc, TFeat-(fc+cov), TFeat-(fc*cov)). From the first layer filter visualization, we could see the color of features from fc adaptation to fc+cov adaptation change more strongly while there are still little changes from fc+cov adaptation to fc*cov adaptation, which shows different ways of domain adaptation indeed influence the features of the first Convolutional layer.



(a) before adaptation (b) after adaptation

Fig. 7. t-SNE visualization of deep features before and after domain adaption

6.2 t-SNE Visualization

Seeing from t-SNE visualization [15] (Fig. 7), features from the source(blue) and the target(red) become more collective and mixed with each other after adaptation while most of the original features of the source lie outside the target features.

7 Conclusion

In this work, we investigate the application of domain adaptation in local descriptors. Experiments results have proven that domain adaptation methods can further enhance the performance of CNN architecture based local descriptors. Besides, the results also demonstrate that it is important to jointly use information to calculate MMD loss from both the first layer and the last layer. It is also interesting to consider how to reduce the high dimension of features from the Convolutional layers so that the joint distribution can be better learnt from domain adaptation. Meanwhile, deeper architecture will further boost the performance.

References

1. Gretton, A., Sejdinovic, D., Strathmann, H., et al.: Optimal kernel choice for large-scale two-sample tests. In: Advances in Neural Information Processing Systems, pp. 1205–1213 (2012)
2. Ramdas, A., Reddi, S.J., Poczos, B., et al.: On the high-dimensional power of linear-time kernel two-sample testing under mean-difference alternatives. arXiv preprint arXiv:1411.6314 (2014)
3. Gong, B., Grauman, K., Sha, F.: Connecting the dots with landmarks: discriminatively learning domain-invariant features for unsupervised domain adaptation. In: International Conference on Machine Learning, pp. 222–230 (2013)
4. Simo-Serra, E., Trulls, E., Ferraz, L., et al.: Discriminative learning of deep convolutional feature point descriptors. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 118–126. IEEE (2015)
5. Tzeng, E., Hoffman, J., Zhang, N., et al.: Deep domain confusion: maximizing for domain invariance. arXiv preprint arXiv:1412.3474 (2014)
6. Tzeng, E., Hoffman, J., Darrell, T., et al.: Simultaneous deep transfer across domains and tasks. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 4068–4076. IEEE (2015)
7. Tzeng, E., Hoffman, J., Saenko, K., Darrell, T.: Adversarial discriminative domain adaptation. arXiv preprint arXiv:1702.05464 (2017)
8. Zhang, K., ScholKopf, B., Muandet, K., Wang, Z.: Domain adaptation under target and conditional shift. In: International Conference on Machine Learning, pp. 819–827 (2013)
9. Duan, L., Tsang, I.W., Xu, D.: Domain transfer multiple kernel learning. IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI) **34**(3), 465–479 (2012)
10. Ghifary, M., Kleijn, W.B., Zhang, M.: Domain adaptive neural networks for object recognition. In: Pham, D.-N., Park, S.-B. (eds.) PRICAI 2014. LNCS (LNAI), vol. 8862, pp. 898–904. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13560-1_76
11. Borgwardt, K.M., Gretton, A., Rasch, M.J., Kriegel, H.P., Smola, A.J.: Integrating structured biological data by kernel maximum mean discrepancy. Bioinformatics **22**(14), e49–e57 (2006)
12. Long, M., Zhu, H., Wang, J., et al.: Deep transfer learning with joint adaptation networks. arXiv preprint arXiv:1605.06636 (2016)
13. Long, M., Cao, Y., Wang, J., et al.: Learning transferable features with deep adaptation networks. arXiv preprint arXiv:1502.02791 (2015)

14. Long, M., Zhu, H., Wang, J., Jordan, M.I.: Unsupervised domain adaptation with residual transfer networks. In: Advances in Neural Information Processing Systems (NIPS), pp. 136–144 (2016)
15. van der Maaten, L.J.P., Hinton, G.E.: Visualizing high-dimensional data using t-SNE. Mach. Learn. Res. **9**, 2579–2605 (2008)
16. Sugiyama, M., Nakajima, S., Kashima, H., et al.: Direct importance estimation with model selection and its application to covariate shift adaptation. In: Advances in Neural Information Processing Systems, pp. 1433–1440 (2008)
17. Russakovsky, O., et al.: ImageNet Large Scale Visual Recognition Challenge. Technical report. arXiv:1409.0575 (2014)
18. Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., Vaughan, J.W.: A theory of learning from different domains. Mach. Learn. **79**(1–2), 151–175 (2010)
19. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 13451359 (2010)
20. Winder, S., Hua, G., Brown, M.: Picking the best daisy. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009, pp. 178–185. IEEE (2009)
21. Zagoruyko, S., Komodakis, N.: Learning to compare image patches via convolutional neural networks. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4353–4361. IEEE (2015)
22. Balntas, V., Lenc, K., Vedaldi, A., et al.: HPatches: a benchmark and evaluation of handcrafted and learned local descriptors. In: Computer Vision and Pattern Recognition (CVPR), vol. 4, no. 5, p. 6 (2017)
23. Balntas, V., Riba, E., Ponsa, D.: Learning local feature descriptors with triplets and shallow convolutional neural networks. BMVC **1**(2), 3 (2016)
24. Han, X., Leung, T., Jia, Y., Sukthankar, R., Berg, A.C.: MatchNet: unifying feature and metric learning for patch-based matching. In: Computer Vision and Pattern Recognition, pp. 3279–3286. IEEE (2015)
25. Ganin, Y., Lempitsky, V.: Unsupervised domain adaptation by backpropagation. arXiv preprint arXiv:1409.7495 (2014)
26. Tian, Y., Fan, B., Wu, F.: L2-Net: deep learning of discriminative patch descriptor in euclidean space. In: Conference on Computer Vision and Pattern Recognition, pp. 6128–6136. IEEE Computer Society (2017)