Unsupervised Deep Domain Adaptation on People Detection

Anonymous ECCV submission

Paper ID ***

Abstract. This paper addresses the problem of unsupervised domain adaptation on the task of people detection in crowded scenes. That is, given a well-trained deep detection model on source domain, we aim to adapt it into the target domain for which no annotations are needed. Firstly, we utilize iterative algorithm to auto-annotate samples on target domain. While auto-annotated samples include noise, the rate of false positive tends to explored. Thus, we mixed into training examples from source domain to suppress the data noise. Furthermore, we transform the inner product layers of our network into two separate layers and proposed a weights regularizer to avoid overfitting. Compared to generic model without any fine tuning on samples on target domain, the proposed method increased recall by $xx\%$ with precision drop from $xx\%$ to $xx\%$ on the target scenes. Also, we perform our algorithm on traditional domain adaptation dataset Office Dataset and our results are state of art.

Keywords: Unsupervised Domain Adaptation, Deep Neural Network, People Detection

1 Introduction

Deep neural network has outstanding result on computer vision tasks, however, it require large labelled dataset. In famous challenges like PASCAL VOC and MS COCO, over millions of images with labels are needed for training. In surveillance applications like people detection, the annotation process is even human resource consuming. Till today, there are millions of cameras deployed for surveillance. However, surveillance situations varies in background, lights, viewpoints and so on in real world, which make the labelling work for all surveillance applications ever more unpractice. In the task of people detection, tens of thousands of annotated images are needed to train a deep neural network.

When labelled data are few and even lack of in target domain, domain adaptation help to reduce the amount of labelled data needed when given abundant labelled data in source domain. Most traditional works [1–5] try to learn a shared representation between source and target domain, or project features into a common subspace. Recently, works are also [6–8] proposed to learn scene-specific detector by deep architectures. In our task of people detection, we try to shift deep detection network well-trained on the source scene to target scenes on which no annotation are needed.

In this paper, we proposed a new approach of unsupervised deep domain adaptation on people detection. Using generic model as initialization, we firstly use iterative algorithm to auto-annotate samples on target domain as fake ground truth. During every iteration, target model are updated and utilized to auto-annotate samples for next iteration. However, the data noise in auto-annotated dataset, like lack of true negative instances and false positive instances, will leads to exploration of false positive rate, as well as impeding further increase of recall. In order to eliminate the effect of data noise, two methods are proposed to regularize the training of the deep network. On the one hand, we mixed into auto-annotated dataset with annotated samples from source domain to compensate the lack of true negative instances. On the other hand, we separate the operation of last inner product layer of our network into two sub-operations – element-wise multiply and sum operations, resulting new element-wise multiply layer and sum layer. Thus, a weights regularizer on element-wise layer can be added into the deep network as unsupervised loss to avoid exploration of false positive rate and boost recall.

Also, we further evaluate our method on traditional semi-supervised domain adaptation dataset (Office dataset). (difference compared to people detection task). (Results)

The contributions of our work are n folds.

- We provided a feasible scheme to shift deep detection network well-trained on the source scene to target scenes on which no annotated data are needed. This makes easier the widely deploy of deep neural network on various surveillance applications.
- As most algorithms focus on the last feature vector (also last inner product layer), we have one step further and transform the last inner product layer into element-wise multiply layer and sum layer. A weight regularizer can, thus, be added on element-wise layer to have better effect on suppressing exploration of false positive rate and increasing recall.
- Experiments on traditional domain adaptation dataset also demonstrate the effectiveness of our approach on other deep domain adaptation tasks.

2 Relate Work

In many detection works, generic model trained by large amount of samples on source domain are directly utilized to detect on target domain. They assume that samples on target domain are subsets of source domain. However, when the distribution of data on target and source domain varies largely, the performance will drop significantly. Domain adaptation aims to reduce the amount of data needed for target domain.

Many domain adaptation works tried to learn a common representation space shared between source and target domain. Saenko et al. [1, 2] proposed a both linear-transform-based technique and kernel-transform-based technique to minimize domain changes. Gopalan et al. [3] projected features into Grassmann manifold instead of operating on features of raw data. Alternatively, Mesnil et al.

[9] used transfer learning to obtain good representations. However, these methods are limited because scene-specific features are not learned to boost accuracy. The regularizer of our method are inspired these works.

Another group of works [4, 5, 10] on domain adaptation is to make the distribution of source and target domain more similar. Among these works, Maximum Mean Discrepancy (MMD) is used to as a metric to reselect samples from source domain in order to have similar distribution as target samples. In [11], MMD is incorporated as regularization to reduce distribution mismatch.

There are also works on deep adaptation to construct scene-specific detector. Wang et al.[6] explored context cues to compute confidence, and [7] learns distributions of target samples and proposed a cluster layer for scene-specific visual patterns. These works reweighted auto-annotated samples for their final object function and additional context cues are needed for reliable performance. Alternately, Hattori el al. [8] learned scene-specific detector by generating a spatially-varying pedestrian appearance model. And Pishchulin et al. [12] used 3D shape models to generate training data. However, Synthesis for domain adaptation are also costly.

3 Our Approach

In this section, we introduce our unsupervised deep domain adaptation approach on the task of people detection. We denote the training images of source domain as $\mathbf{X}^S = \{x_i^S\}_{i=1}^{N^S}$ and that of target domain as $\mathbf{X}^T = \{x_j^T\}_{j=1}^{N^T}$. For each image in source domain, we have corresponding annotations denoted as $\mathbf{B}_i^S = \{b_{i,k}^S\}_{k=1}^{N_i^S}$ with $b_{i,k}^S = (x, y, w, h) \in R^4$, however, the annotations for images in target domain are auto-labelled which we denote as $\tilde{\mathbf{B}}_j^T = \{\tilde{b}_{j,k}^T\}_{k=1}^{N_j^T}$ with $\tilde{b}_{j,k}^T = (\tilde{x}, \tilde{y}, \tilde{w}, \tilde{h}) \in R^4$. The annotations for target domain changes for every iteration during the adaptation process. Human annotated images on target domain are used only for evaluation.

The adaptation architecture of our approach consists of two streams – source stream M^S and target stream M^T, as shown in Fig x(!). Source stream takes samples from source domain as input, and target stream operates on samples from target domain. These two streams can utilize any end to end deep detection network as their model. Here we use the below mentioned network in Sec 3.1 in our experiment. In initialization stage, we firstly use abundant annotated samples from source domain to train the model of source stream under a supervised loss function to regress bounding box. After its convergence, the weights of the model of source stream are used to initialize target stream. In adaptation stage, iteration algorithm is used as training method. Target model in target stream is trained and upgraded in this process while source stream stays static.

Both supervised loss and unsupervised regularizer are designed as loss function to train the target stream for learning scene-specific detector as well as avoid overfitting. For supervised loss, the auto-annotated data can be used as training labels. While for unsupervised regularizer, we take the source model in source

stream as a reference for feature vector distribution. We defined the combination of supervised loss and unsupervised loss as our loss function for adaptation:

$$L(\theta^T|\mathbf{X}^S, \mathbf{B}^S, \mathbf{X}^T, \tilde{\mathbf{B}}^T, \theta^S) = L_S + \alpha * L_U \quad (1)$$

$$L_S = \sum_{j=1}^{N^T} \sum_{k=1}^{N_J^T} (r(\theta^T|x_j^T, \tilde{b}_{j,k}^T) + c(\theta^T|x_j^T, \tilde{b}_{j,k}^T)) \quad (2)$$

$$L_U = L_{MMD}(\theta^T|\mathbf{X}^S, \mathbf{X}^T, \theta^S) \quad (3)$$

where L_S is supervised loss to learn the scene-specific detector and L_U is the unsupervised regularizer part. $r(\cdot)$ is a regression loss for bounding box location, like norm-1 loss, and $c(\cdot)$ is a classification loss for bounding box confidence, such as cross-entropy loss. And $L_{MMD}(\cdot)$ is the MMD-based loss for unsupervised regularization. Coefficient α balance the effect of supervised and unsupervised loss. We set $\alpha = 10$ in our experiments.

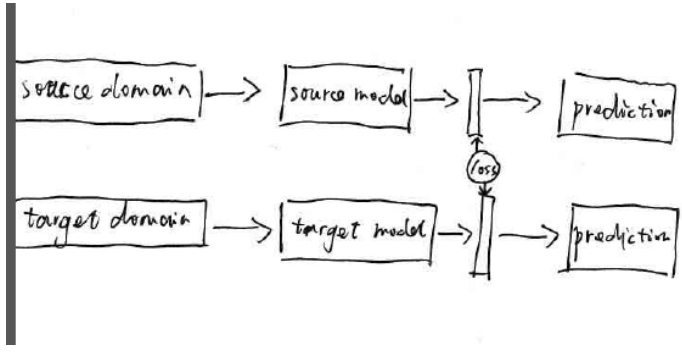

Fig. 1. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s. This shows a figure consisting of different types of lines. Elements of the figure described in the caption should be set in italics, in parentheses, as shown in this sample caption. The last sentence of a figure caption should generally end without a full stop

3.1 Detection Network

The generic model [1] used in our adaptation architecture for source and target stream is an end to end detection network without any precomputed region proposals needed. It consists of a GoogLeNet [13] for feature extraction and a RNN-based decoder for output of bounding box and corresponding confidence, as shown in Fig x(!). Firstly, the GoogLeNet model encode the image into a

[1] Proposed by Russel et al.

feature map (15x20x1024) of which each 1024 dimension vector are data representation of its receptive field corresponding to a subregion of the image. Then the RNN-based layers with batch size 300 will decode the data representation and sequentially predict 5 possible bounding boxes by the order of its corresponding confidence. Finally, all outputs are summarized to give final detection results. When trained with abundant samples from source domain, the obtained model has a high precision on target domain, however, its recall is low. Also, different from other detection network [14, 15], which need precomputed proposals for classification and fine regression, this generic model directly predict bounding boxes with high confidence. Thus, negative instances may contain people and non-people predictions, and cannot be employed in adaptation training.

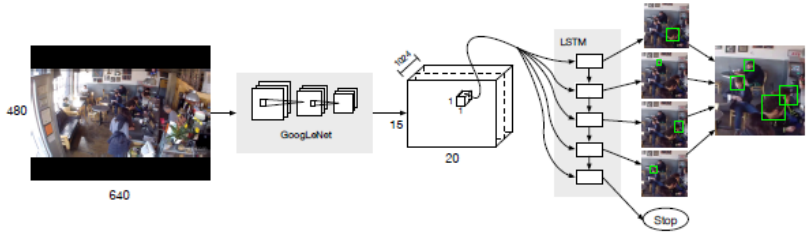

Fig. 2. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s. This shows a figure consisting of different types of lines. Elements of the figure described in the caption should be set in italics, in parentheses, as shown in this sample caption. The last sentence of a figure caption should generally end without a full stop

3.2 Iterative Algorithm

In this section we introduce the iteration algorithm as training method at adaptation stage. To generate the auto-annotated data for the first iteration, we utilize the generic model well-trained on source domain, which results in training set on the target domain. As mentioned in Sec 3.1, the training set on target domain auto-annotated by the generic model in our experiment have low recall and high precision. At subsequent iterations, training set on target domain are auto-annotated by upgraded model resulting from training of last iteration. Among every iteration, these auto-annotated data are used as training data to upgrade the deep network.

As the training set on target domain auto-annotated for the training of first iteration have low recall rate and people and non-people regions mixed in negative instances, we ignore back propagation of bounding boxes with low confidence

during training. That is, we encourage the network to have more confidence on positive instances and stay conservative toward negative instances. This policy will no doubt resulting predictions of many non-people instance and impeding the further training when many non-people instances are regarded as people instances by the target model. To compensate for lack of true negative instances, we add annotated samples from source domain into the training data, which are human annotated and can thus provide true negative samples. In our experiment, when training on additional samples from source domain, we only do back propagation of bounding boxes with low confidence. At the same time, the unsupervised loss will also regularize the network. The complete adaptation process is illustrated in Algorithm 1. After a predetermined iteration limit N^I is reached, we obtain our final detection model on the target domain.

Algorithm 1 Deep domain adaptation algorithm (to be completed)

procedure DEEP DOMAIN ADAPTATION
Train source model on source stream with abundant annotated data
Use well-trained source model on source stream to initialize model on target stream as M_0
  for i = 0:N^I **do**
    M_i generate "fake ground truth" Gi of target domain
    balbal
    balbabla
    Gi = Gi + random samples from source domain
    Take Gi as training data to upgrade M_i into M_{i+1}
  end for
M_{N^I}: final model.
end procedure

3.3 Unsupervised weights regularizer on Element-wise Multiply Layer

Element-wise Multiply Layer In deep neural network, the last feature vector layer are taken as an important data representation of input images. However, in this paper, we take one step further to focus on the last full connected layer which serves as an decoder to decode rich information contained in the last feature vector into final outputs. As source model are trained with abundant labelled data on source domain, the parameters of the last full connected layer are also well converged. We assume that a regularizer on the last full connected layer will achieve better results compared with the last feature vector layer. Firstly, denote the last feature vector, parameters of the last full connected layer and final outputs as $\mathbf{F}_{(N^B \times N^D)}$, $\mathbf{C}_{(N^D \times N^O)}$ and $\mathbf{P}_{(N^B \times N^O)}$, respectively. The

operation of full connected layer can be thus formulated as matrix multiply:

$$\mathbf{P} = \mathbf{F} * \mathbf{C} \tag{4}$$

$$P_{b,o} = \sum_{d} F_{b,d} * K_{d,o} \tag{5}$$

Inspired by this form, we separate the above formula into two sub-operations – element-wise multiply and sum, which can be formulated as:

$$M_{b,o,d} = F_{b,d} * C_{d,o} \tag{6}$$

$$P_{b,o} = \sum_{d} M_{b,o,d} \tag{7}$$

where $\mathbf{M}_{(N^B \times N^O \times N^D)} = [\boldsymbol{m}_{b,o}]$ is the parameter tensor of element-wise multiply operations. $\boldsymbol{m}_{b,o}$ is a vector with N^D dimensions, which will be the basis of unsupervised regularizer. Finally, we can equivalent-transform the last full connected layer between the last feature vector layer and final outputs layer into element-wise multiply layer and sum layer. The transformed element-wise layer is thus the last layer with parameters before output layers. Fig x (!) illustrates the transform.

Fig. 3. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s.

Unsupervised weights regularizer on Element-wise Multiply Layer In works [decaf][][], the last feature vector layer are regarded as the final representation of images. (how to introduce beyond sharing weights) In domain adaptation tasks, when generic deep model are trained with abundant data from source domain, the last element-wise layer mentioned in Sec 3.3 also includes rich information leading to the final output. For the nodes in the output layer, inputs from element-wise layers that will contribute to its value are not randomly decided. In this paper, we assume that the distribution of $\boldsymbol{m}_{b,o}$ of the last

element-wise layer on both source and target domain should be similar. In the last element-wise multiply layer, every dimension of $\boldsymbol{m}_{b,o}$ may contribute to the final output. However, as the old model on source domain are trained with abundant images, the distribution of $\boldsymbol{m}^T_{b,o}$ should be consistent compared with $\boldsymbol{m}^S_{b,o}$ of source domain when adapting deep network on target domain. We utilize MMD (maximum mean discrepancy) to encode the similarity of element-wise multiply layer of source domain and target domain:

$$L_{MMD}(\theta^T|\mathbf{X}^S,\mathbf{X}^T,\theta^S)=\frac{1}{N^O}\sum_{o=1}^{N^O}\|\frac{1}{N^B}\sum_{b=1}^{N^B}\boldsymbol{m}^T_{b,o}-\frac{1}{N^B}\sum_{b=1}^{N^B}\boldsymbol{m}^S_{b,o}\|^2 \quad (8)$$

which can also interpreted as the Euclidean distance between the center of $\boldsymbol{m}^T_{b,o}$ and $\boldsymbol{m}^S_{b,o}$ across all output nodes. It's unpractical to get the distribution of the whole training set, while too few images cannot obtain a stable center for regularization. In our experiments, the $L_{MMD}(\cdot)$ loss is calculated for every batch. An example comparison of centers of $\boldsymbol{m}^S_{b_i,o}$ and $\boldsymbol{m}^S_{b_j,o}$ are shown in Fig x(1).

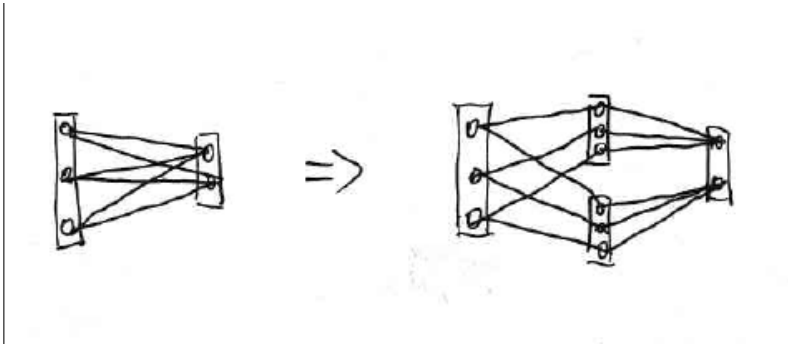

Fig. 4. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s.

4 Experiment Results

In this section, we introduce experiment results on both surveillance applications and standard domain adaptation dataset. Our motivation for unsupervised domain adaptation method is for easier deployment of We firstly evaluate our approach on video surveillance. Then we employ our approach to standard domain adaptation dataset to demonstrate the effectiveness of our method.

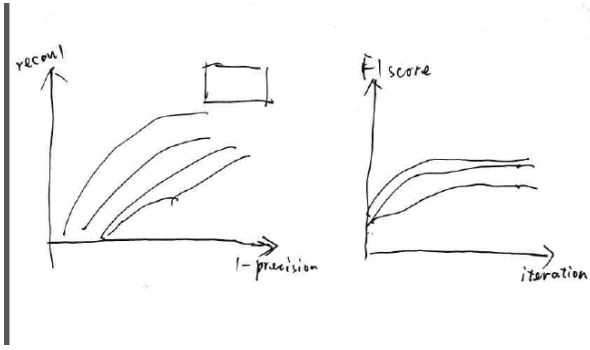

Fig. 5. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s. This shows a figure consisting of different types of lines. Elements of the figure described in the caption should be set in italics, in parentheses, as shown in this sample caption. The last sentence of a figure caption should generally end without a full stop

4.1 Domain Adaptation on Crowd Dataset

Dataset and evaluation metrics To show the effectiveness of our domain adaptation approach on people detection, we collected a dataset consisting of 3 target scenes for target domain. These three scenes contain 1308, 1213 and 331 un-annotated images with 0000, 0000, 0000 people instances respectively. For each scene, 100 images are annotated for evaluation. Instead of labelling the whole body of a person, we labels the head of a person as bounding box during training. The motivation for labelling only people heads comes from detection of indoor people or in crowded scenes, where the body of a person may be invisible. The dataset for source domain are Brainwash Dataset released on http://d2.mpi-inf.mpg.de/datasets. Brainwash Dataset consists of over 11917 images from 3 crowded scenes. Examples of images from source and target domain are shown in Fig x(!).

Our evaluation metrics for detection uses the protocol defined in PASCAL VOC [16]. To judge a predicted bounding box whether correctly matches a ground truth bounding box, their intersection over their union must exceed 50%. And Multiple detections of the same ground truth bounding box are regarded as one correct prediction. We plot the precision-recall curve in Fig x(!). Also, the F1 score $F1 = 2 * precision * recall/(precision + recall)$ during the adaptation process are also shown in Fig x(!).

Experimental settings We use deep learning framework Caffe [17] as the adaptation architecture of our approach. During the adaptation, we set learning rate as 0.01 and momentum as 0.5. At initialization stage, GoogLeNet weights are firstly used to initialize source model of source stream, while parameters in RNN layers are randomly initialized from a uniform distribution. For each iteration, 100 auto-annotated images from target domain and 1000 annotated images from

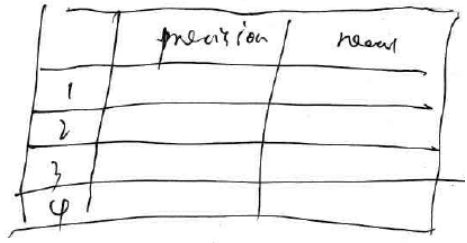

Fig. 6. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s. This shows a figure consisting of different types of lines. Elements of the figure described in the caption should be set in italics, in parentheses, as shown in this sample caption. The last sentence of a figure caption should generally end without a full stop

source domain are alternatively used for training. The outputs of our detection network contains bounding box locations and corresponding confidence, thus there are two full connected layers between the last feature vector layer and the final outputs. Experiments of unsupervised regularizer on element-wise layer for bounding box regression have no additional improvement on the performance when regularizer on element-wise layer for box confidence classification are added already. Our approach on domain adaptation are executed separately in the 3 target scenes.

Comparison with baseline methods To demonstrate the effectiveness of our approach, 4 methods are compared with method 4 as our final approach:

1. Only auto-labeled samples on target domain are used for training, and without any unsupervised regularizer.
2. Only auto-labeled samples on target domain are used for training, with MMD regularizer on last element-wise multiply layer as unsupervised weights regularizer.
3. Both auto-labeled images from target domain and labeled images from source domain are alternately sampled for training, with MMD regularizer on last feature vector as unsupervised weights regularizer.
4. Both auto-labeled images from target domain and labeled images from source domain are alternately sampled for training, with MMD regularizer on last element-wise multiply layer as unsupervised weights regularizer.

Fig x(!) plots the precision-recall curve of the above comparison methods in target scene 1). Also, the F1 score changes of every iteration during adaptation process are also depicted in Fig x(!). Table x(!) gives concrete precision and recall value of the 4 comparison methods on three target scenes when the F1 scores are at their highest. Examples of adaptation results are shown in Fig x(!).

Performance evaluation Table x shows the results of above 4 methods on 5 target scenes. The overwhelming performance of method 4) on both 5 target scenes shows the effectiveness of our approach.

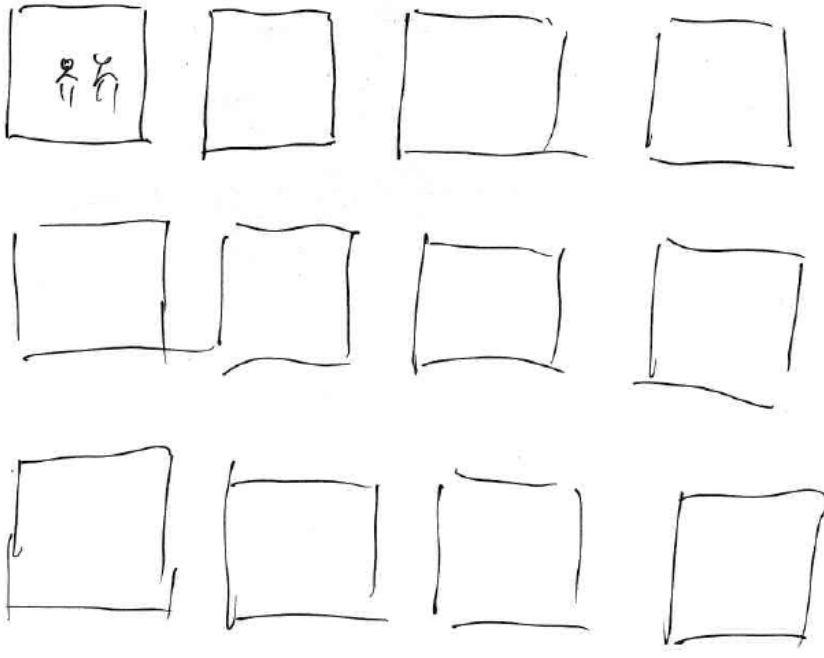

Fig. 7. One kernel at x_s (*dotted kernel*) or two kernels at x_i and x_j (*left and right*) lead to the same summed estimate at x_s. This shows a figure consisting of different types of lines. Elements of the figure described in the caption should be set in italics, in parentheses, as shown in this sample caption. The last sentence of a figure caption should generally end without a full stop

4.2 Domain Adaptation on Office Dataset

balabala...

5 Conclusions

The paper ends with a conclusion.

References

1. Saenko, K., Kulis, B., Fritz, M., Darrell, T.: Adapting visual category models to new domains. In: Computer Vision–ECCV 2010. Springer (2010) 213–226
2. Kulis, B., Saenko, K., Darrell, T.: What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE (2011) 1785–1792
3. Gopalan, R., Li, R., Chellappa, R.: Domain adaptation for object recognition: An unsupervised approach. In: Computer Vision (ICCV), 2011 IEEE International Conference on, IEEE (2011) 999–1006
4. Huang, J., Gretton, A., Borgwardt, K.M., Schölkopf, B., Smola, A.J.: Correcting sample selection bias by unlabeled data. In: Advances in neural information processing systems. (2006) 601–608
5. Gretton, A., Smola, A., Huang, J., Schmittfull, M., Borgwardt, K., Schölkopf, B.: Covariate shift by kernel mean matching. Dataset shift in machine learning **3**(4) (2009) 5
6. Wang, X., Wang, M., Li, W.: Scene-specific pedestrian detection for static video surveillance. Pattern Analysis and Machine Intelligence, IEEE Transactions on **36**(2) (2014) 361–374
7. Zeng, X., Ouyang, W., Wang, M., Wang, X.: Deep learning of scene-specific classifier for pedestrian detection. In: Computer Vision–ECCV 2014. Springer (2014) 472–487
8. Hattori, H., Naresh Boddeti, V., Kitani, K.M., Kanade, T.: Learning scene-specific pedestrian detectors without real data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 3819–3827
9. Mesnil, G., Dauphin, Y., Glorot, X., Rifai, S., Bengio, Y., Goodfellow, I.J., Lavoie, E., Muller, X., Desjardins, G., Warde-Farley, D., et al.: Unsupervised and transfer learning challenge: a deep learning approach. ICML Unsupervised and Transfer Learning **27** (2012) 97–110
10. Gong, B., Grauman, K., Sha, F.: Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In: Proceedings of The 30th International Conference on Machine Learning. (2013) 222–230
11. Ghifary, M., Kleijn, W.B., Zhang, M.: Domain adaptive neural networks for object recognition. In: PRICAI 2014: Trends in Artificial Intelligence. Springer (2014) 898–904
12. Pishchulin, L., Jain, A., Wojek, C., Andriluka, M., Thormählen, T., Schiele, B.: Learning people detection models from few training samples. In: Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on, IEEE (2011) 1473–1480
13. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 1–9
14. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 1440–1448
15. Vu, T.H., Osokin, A., Laptev, I.: Context-aware cnns for person head detection. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 2893–2901
16. Everingham, M., Eslami, S.A., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision **111**(1) (2015) 98–136

17. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. arXiv preprint arXiv:1408.5093 (2014)