Cross Domain Object Detection through Domain Adaptation

Prashant Kumar Ray
Politecnico di Torino
Torino, Italy
s271098@studenti.polito.it

Musaab Awouda
Politecnico di Torino
Torino, Italy
s261304@studenti.polito.it

Willmar Rengifo
Politecnico di Torino
Torino, Italy
s206856@studenti.polito.it

Abstract

Despite recent advances in object detection using deep learning neural networks, these neural networks still struggle to identify objects in art images such as paintings and drawings. This challenge is known as the cross depiction problem and it stems in part from the tendency of neural networks to prioritize identification of an object's texture over its shape.
To address this issue, this paper aims to study the object detection task, and how we can transfer the learned knowledge from a source domain to a target domain.
We have started with a fully supervised object detector that is trained on natural images for which instance level annotations are available. Then, we have proposed two different domain adaptation strategies to transfer the domain of the images from source domain into target one. Later, we have fine-tuned the detector with the domain transferred images obtained by applying the domain transfer strategy on natural images of the source domain. At the end, we test our model to achieve an improvement of approximately 9 to 14 percentage in terms of mean average precision(mAP) when compared to the baseline model, and approximately 14 percentage behind the ideal case. Code is available at `https://github.com/pkr076/AIML_ExamProject.git`

1. Introduction

The Object detection is an essential task that handle the identification and localization of that objects in a image. It is a fundamental task and has advanced rapidly due to the development of convolutional neural networks (CNNs). [8, 21, 24, 5, 23, 20] are some of the Best-performing models. The fully supervised detectors usually have best performance and results, however they require a huge amount of images with instance-level annotations for training. An instance-level annotation is composed of a label (i.e., the object class of an instance) and a bounding box (i.e., the location of the instance).

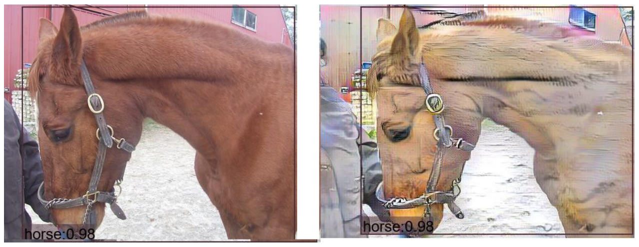


Figure 1: Example of domain transfer used in our framework. Instance-level annotation is available only for the source domain.

Despite the performance of this kind of models in natural image domain, it is unrealistic to prepare the annotations for large number of images of different domains. The possible reasons can be the lack of image sources, copyright issues, and the cost of annotation.
We have instance-level annotations and images in a source domain, and only images-level annotations in the target domain. The classes to be detected in the target domain are same or subset of the classes in the source domain. The goal is to accurately detect objects in the target domain under these conditions.
The straightforward solution would be to train a fully supervised object detector model in the source domain, and then we fine-tune the FSD in the target domain. However, there are no instance-level annotations available in the target domain. Therefore, we present two different domain transfer (DT) methods to generate images with instance-level annotations artificially and automatically, and fine-tune the FSD on them. Domain transfer (DT), is used to generate images that look like those in the target domain from images in the source domain having instance-level annotations. In this work, our aim is to tackle the cross-domain object detection problem through domain adaptation. The goal of domain adaptation is to be able to apply an algorithm trained in one or more source domains, and performs well on a different but related target domain [12, 27].
Most of recent domain adaptation works focus on how to

minimize the domain divergence. To do that most model try to align source and target domains by creating a domain-invariant feature representation [2] [1] [3], typically in the form of a feature extractor neural network. A feature representation is domain-invariant assuming the features have the same distribution regardless of whether the input data are from the source or target domain [28]. We can achieve alignment by trying to minimize divergence [10], performing reconstruction [7], and some employ adversarial training commonly using generative adversarial networks (GANs). However, these methods assume that such a feature representation exists and the marginal label distributions do not differ significantly.
An alternative to creating a domain-invariant feature representation is use image to image translation, or translating the image from source domain to target domain, so the two images can be very similar from the model point of view. This mapping is created at the pixel level. An example of image to image translation methods from unpaired examples is Cycle-Consistent Adversarial Networks or Cycle-GANs [30]. Another model for pixel-level domain adaptation strategy is Adaptive Instance Normalization or AdaIN [13] which is a method that allows to perform arbitrary style transfer in real-time.

2. Related Work

2.1. Object Detection

Object detection [9, 25, 8] as a fundamental problem in computer vision has achieved great progress since 2012 with the development of deep neural networks. Based on Alexnet, many different Convolutional Neural Networks (CNN), such as VGGnet, GoogLeNet, ResNet, DenseNet, etc., were proposed to learn more powerful deep features from data. Object detection algorithms also benefit from these architectures since better features are also helpful for other vision tasks. Apart from the different network architectures, recent CNN-based object detectors can be mainly divided into two categories, single-stage detectors, and two-stage detectors. Single-stage detectors like YOLO [22, 17] and SSD [21, 6], while two-stage detectors like Faster R-CNN [24] and RFCN [5].
Single stage detectors directly predict object labels and bounding box coordinates within one image based on the default anchor boxes, which is fast but not accurate enough. On the other hand, two stage detectors firstly generate large number of region proposals based on the CNN features, and then recognize the proposals with heavy head. Therefore, it has better performance than single stage detectors, but the detection speed is slower. Recently, many researchers also attempt to train the detector with only weak labels [18, 26] or a few ground-truth labels [19] to deal with more practical application scenarios. [21, 6] is simple relative to methods that require object proposals because it completely eliminates proposal generation and subsequent pixel or feature re-sampling stages and encapsulates all computation in a single network, this also means that it has real-time detection ability. This makes SSD easy to train and straightforward to integrate into systems that require a detection component. Original SSD architecture is based on a VGG16 backbone and an extra feature layer box-head. Such models are much faster than two-stage object detectors. For these reasons, In this paper, we decided to use SSD as baseline detector.

2.2. Unsupervised Domain Adaptation

Unsupervised domain adaptation (UDA) is a task used for learning domain invariant models, where pairs of image and instance-annotation are available in the source domain while only images are available in the target domain. Previous works for UDA in image classification is mostly distribution-matching-based, in which features extracted from two domains are made to closely resemble each other using the maximum mean discrepancy (MMD) [11] or a domain classifier network. Although current distribution-matching-based methods are applicable, it is primarily challenging to fully align the distribution for tasks that require structured outputs, such as object detection. This is because it is essential to keep the spatial information in the feature map.
Here we adopted style transfer and fine-tuning the model to avoid this problem.

2.3. Image-to-Image Translation

Image to image translation can be done at pixel-level using pixel to pixel method [16], Cycle-GAN [29] is actually based on the same pixel to pixel technique.
This approach is capable of translating an image from source domain to target domain while input-output examples are unpaired. However, the generation of new images can't be achieved in real-time, during the fine-tuning process. It requires a CycleGAN training phase, where the model learns how to generate effective translated images. Training the model is a heavy process and can be very time consuming, even with power full GPUs.

2.4. Style Transfer

The Style transfer strategy aims at altering the low-level visual style within an image while preserving its high-level semantic content.
In [4] they proposed a similar idea to combine content loss and style loss based on the pre-trained neural networks on ImageNet. Based on this pioneering work, Huang et al proposed the AdaIN [14] to match the mean and variance statistics of the latent embedding of the content image and style image, then embedded the normalized feature into a styl-

ized image. We used the latter technique to achieve domain translate.

3. Datasets

The first objective is replicate the same result shown in paper [15] for detecting objects in a target domain with SSD trained on source domain. For this reason , the datasets we used to train the base model are the same datasets used in the reference project, the VOC2007 and VOC2012 containing 20 object classes. For the target domain we used Clipart1k dataset. The object classes in the Clipart1k dataset are the same as in the source domain. Clipart1k dataset contains 20 classes, 1000 images and 3165 instance-level annotations.

4. Method

In this section, we describe the methodology and steps for this work. Section 4.1 describes the SSD [21] model for object detection and is used as baseline for performance. Section 4.2 describes CycleGAN [29], which is is a technique for training unsupervised image translation models using unpaired collections of images from two different domains. Section 4.3 explains the implementation of our domain adaptation choice which is AdaIN [14].

4.1. SSD

The SSD [21] approach is based on a feed-forward convolution network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detection.

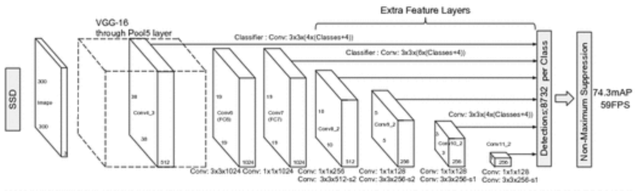

Extra Feature Layers
VGG-16 through Pool5 layer
SSD
Classifier : Conv: 3x3x(4x(Classes+4))
Classifier : Conv: 3x3x(6x(Classes+4))
Conv: 3x3x(4x(Classes+4))
Detections:8732 per Class
Non-Maximum Suppression
74.3mAP 59FPS
Conv: 3x3x1024
Conv: 1x1x1024
Conv: 1x1x256 Conv: 3x3x512-s2
Conv: 1x1x128 Conv: 3x3x256-s2
Conv: 1x1x128 Conv: 3x3x256-s1
Conv: 1x1x128 Conv: 3x3x256-s1

Figure 2: SSD with a 300x300 input size

4.2. CycleGAN

CycleGAN model uses pixel level domain adaptation. It is a technique that involves the automatic training of image-to-image translation models without paired examples. The models are trained in an unsupervised manner using a collection of images from the source and target domain. It's objective is to learn the mapping functions in the domains X and Y.
The goal is to learn a mapping G : X $\rightarrow$ Y such that the distribution of images from G(X) is indistinguishable from the distribution Y using an adversarial loss. This is coupled with an inverse mapping F : Y $\rightarrow$ X and cycle consistency loss.

4.3. AdaIN

AdaIN [14] is a method that allows to perform arbitrary style transfer in real-time, the model contain an adaptive instance normalization layer that aligns the content of the source images with the style of target images. This layer effectively combines the content of input x and the style of input y by transferring feature statistics, specifically the channel wise mean and variance, so that the values of x and y match.

$$AdaIN(x,y) = \sigma(y)\frac{x-\mu(x)}{\sigma(x)} + \mu(y) \tag{1}$$

where $\mu(x)$ and $\sigma(x)$ are the mean and standard deviation of the input images. The normalized content input is scaled with $\sigma(x)$ and shifted with $\mu(y)$.
The final image is passed through decoder then generates the final stylized image by inverting the output back to the image space.

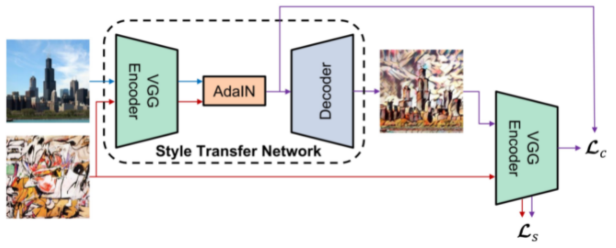


Figure 3: An overview of style transfer algorithm. AdaIN uses a VGG-19 network to encode the content and style images. An AdaIN layer is used to perform style transfer in the feature space. A decoder is then used to invert the output to the content image space.

5. Experiments

In Sec. 5.1, we explain the details of the implementations, Dataset Arrangement and the evaluation metrics. In Sec. 5.2, we describe the training details of the baseline and its performance on the test dataset. In Sec. 5.3, we describe the training details of the CycleGAN [29] which is used to style the source domain images to look like target domain images. In Sec. 5.4, we describe the details for fine-tuning the baseline model with the augmented source domain images through CycleGAN. In Sec. 5.5 we describe the details for fine-tuning the baseline model with the augmented

Table 1: Experiment results of the baseline training with different configurations

Exp. Name	**Learning Rate**	**LR steps**	**Max Iterations**	**mAP**
Exp1	1e-3	[60k,80k]	80k	26.32
Exp2	1e-3	[80k,100k]	120k	26.25
Exp3	1e-3	[50k,60k]	60k	26.16
Exp4	2e-4	[50k,60k]	60k	23.18

source domain images through AdaIN [14]. In Sec. 5.6, we describe the ideal case.

5.1. Implementation and Evaluation Metrics

We have used the SSD implementation provided at `https://github.com/lufficc/SSD`. We evaluated our methods using average precision (AP) and its mean, i.e., mAP.

Dataset Arrangement for Training and Evaluation : VOC2007-trainval and VOC2012-trainval are used as images in the source domain (i.e., natural image). We have used the Clipart1k dataset as the target domain. For the target-domain images, the ones with instance-level annotations were used when discussing the performance gap between our experiments and the ideal case quantitatively.

5.2. SSD Baseline

Initially, we trained the SSD300 model with batch size equal to 16, learning rate $lr = 10^{-3}$ for 60k iterations, then $lr = 10^{-4}$ for next 20k iterations with the images having instance-level annotations in the source domain as mentioned in SSD . We followed the original paper on hyperparameters of SSD300 [21] unless specified. The input images were resized to 300 * 300 in SSD300. The IoU threshold for NMS (0.45) and the confidence threshold for discarding low confidence detections (0.01) were employed. We have performed other experiments by training the SSD [21] with different values for the learning rate and number of iterations. The experiments and their performance is summarized in table 1.

5.3. CycleGAN

For training the CycleGAN [29], we have used the images of source domain without annotation and the images of target domain. We trained CycleGAN with a learning rate of $1.0 * 10^{-5}$ for the first ten epochs and a linear decaying rate to zero over the next ten epochs. We followed the original paper on the other hyper-parameters of CycleGAN. Examples of source domain images and its domain transferred version is shown in Fig. 4.

5.4. SSD Model Finetuned with Augmented Images through CycleGAN : DT1

We fine tune the SSD baseline model which we obtained in experiment 5.2 and train it with the augmented source domain images that we got from CycleGAN [29]. We have referred this experiment from here onwards as DT1. The fine-tuning was conducted for 10000 iterations with the learning rate of $1.0 * 10^{-5}$. Note that no layers of the FSD are replaced to preserve the original network's detection ability. we test our model on the target domain images and the result is showed in Table 2. If we compare this with the baseline model, we can notice that the fine-tuned model performs better in terms of mAP.

5.5. SSD Model Finetuned with Augmented Images through AdaIN : DT2

We fine tune the SSD baseline model which we obtained in experiment 5.2 and train it with the augmented source domain images that we obtained from AdaIN [14] with style factor α. We have referred this experiment from here onwards as DT2. Examples of source domain images and its domain transferred version is shown in Fig. 5.
The fine-tuning was conducted for 10000 iterations with the learning rate of $1.0 * 10^{-5}$. No layers of the FSD are replaced to preserve the original network's detection ability. We have repeated this experiment multiple times by varying the style factor α[1]. We observed that the SSD performance increases if we finetune it with augmented images having higher value of α. The results of the SSD models finetuned on augmented source domain images with different α is shown in Table 3. we test our model on the target domain and the result is showed in Table 2. If we compare this with the baseline model and DT1, we can notice that this fine-tuned model performs better than both in terms of mAP.

5.6. Ideal Case

In this case, we had access to the instance-level annotations for the training split of the target domain dataset. We simply fine-tune the baseline model of SSD [21] using these instance level annotations. This experiment was to confirm the weak upper-bound performance of the model. We test our model on the testing split of the target domain dataset and the result is showed in Table 2.

[1]Further details about style factor α can be found in [14]

Table 2: Comparison of all the methods in terms of AP using SSD300 as the baseline FSD in Clipart1k

Object Class	**Baseline Ref.**	**Baseline**	**DT1**	**DT2** ($\alpha = 1$)	**Ideal Case**
aeroplane	19.8	23.72	34.37	34.94	46.21
bicycle	49.5	60.98	66.49	62.39	76.09
bird	20.1	18.97	18.86	23.27	38.40
boat	23.0	17.81	30.97	33.55	49.83
bottle	11.3	16.47	22.14	26.81	30.59
bus	38.6	29.57	39.92	41.32	60.85
car	34.2	34.39	45.77	54.51	60.74
cat	2.5	3.48	2.64	4.84	9.11
chair	39.1	44.87	46.84	50.23	56.46
cow	21.6	12.87	44.97	50.52	68.27
diningtable	27.3	29.35	32.12	43.05	60.41
dog	10.8	8.34	8.43	15.15	32.22
horse	32.5	26.94	23.93	39.59	51.45
motorbike	54.1	45.26	56.14	72.84	85.66
person	45.3	37.65	57.03	62.68	74.74
pottedplant	31.2	26.67	40.85	36.96	52.11
sheep	19.0	5.59	12.72	12.0	36.45
sofa	19.5	25.81	45.61	45.52	49.64
train	19.1	29.83	38.46	49.94	72.09
tvmonitor	17.9	27.89	40.98	46.46	73.37
mAP	**26.8**	**26.32**	**35.46**	**40.33**	**54.23**

Table 3: Experiment results of DT2 for different values of α

α	0	0.25	0.50	0.75	1.0
mAP	26.32	30.17	33.98	37.48	40.33

6. Quantitative Results on Clipart1k

Table 2 shows the comparison of AP for each class and mAP among our methods against the baseline SSD model and its fine-tuned versions. We observe that SSD300 baseline model doesn't perform so well on clipart images. We could reach upto 26.32 percentage points in terms of mAP which is comparable to the result obtained in paper [15]. DT1 provides an improvement of 9.14 percentage points improvement from the baseline SSD300 in terms of mAP. In case of DT2 we observed that as we increase the degree of stylization factor α, we get improvement in terms of mAP. For $\alpha = 1$, DT2 provides an improvement of 14.01 percentage points improvement from the baseline SSD300

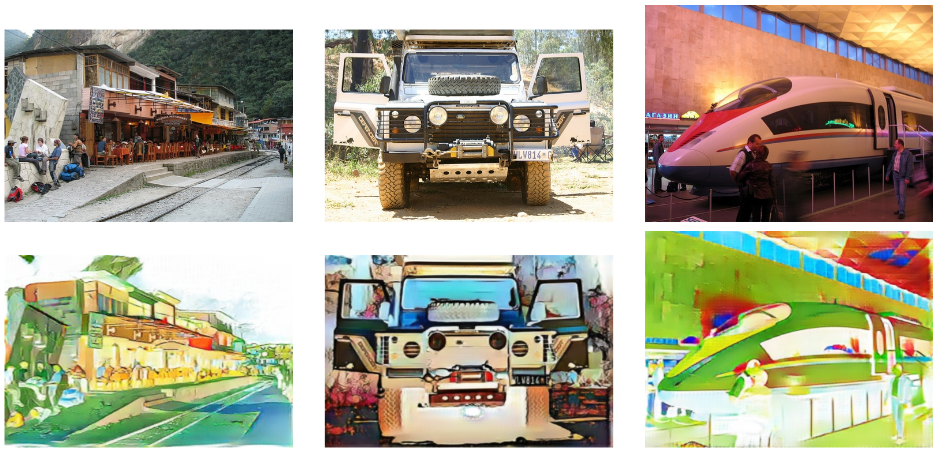
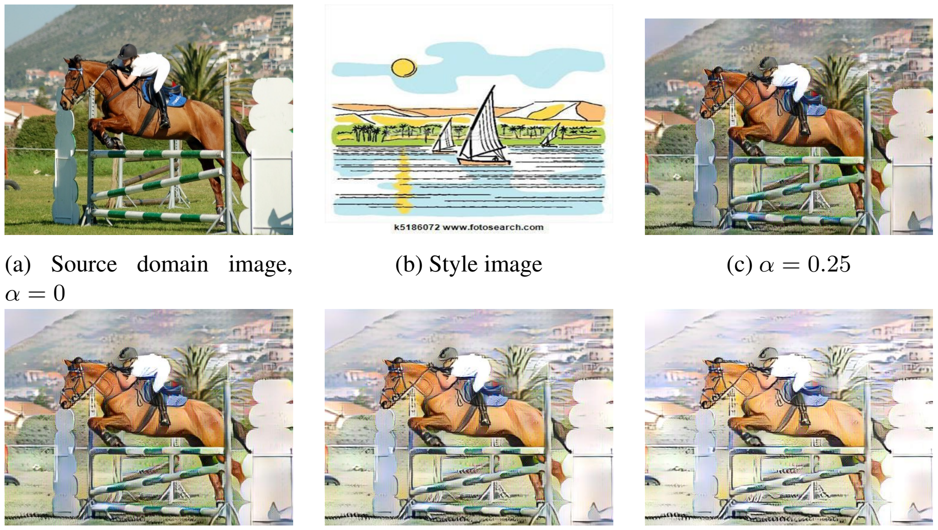


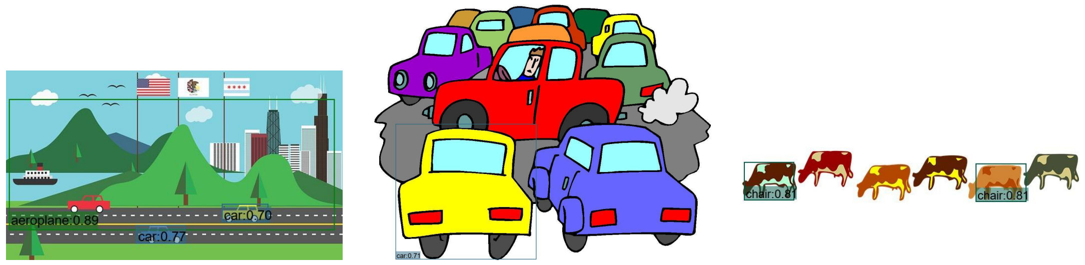
Figure 4: Examples of source domain image and its domain transferred version obtained through CycleGAN



(a) Source domain image, $\alpha = 0$ (b) Style image (c) $\alpha = 0.25$

(d) $\alpha = 0.50$ (e) $\alpha = 0.75$ (f) $\alpha = 1.0$

Figure 5: Examples of source domain image and its domain transferred version obtained through AdaIN



(a) (b) (c)

Figure 6: Object detection performance of the baseline SSD in the target domain

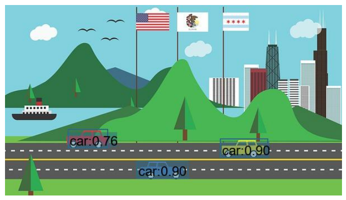
in terms of mAP. If we compare DT2 result with the result obtained for domain transfer(DT) case in paper [15], DT2 outperforms it by 2.33 percentage points.

7. Discussion

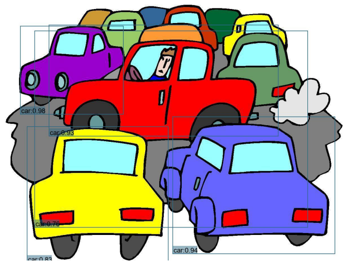
The performance of the baseline FSD , DT1 and DT2 methods on clipart images are shown in Fig. 6, 7, 8 respectively. In Fig. 6(a), we can observe that the model is not able to detect all the cars and also it makes a false detection of aeroplane. In Fig. 6(b), it could detect only one car. In Fig. 6(c), it could detect only one 2 cows out of 6. So, we can say that the baseline model does not perform well when we use it for object detection in images of different domain. The performance of the model is improved in case of DT1 as we can see that it is able to detect all the cars in Fig. 7(a) and also does not makes any false detection. In Fig. 7(b), it is now able to detect five cars. In Fig. 7(c), it is able to detect 4 cows out of 6. Similarly, DT2 model also performed well in comparison to the baseline model. When compared with the DT1 model, it is able to detect all the cows but, at the same time it could detect less number of cars than DT1 as in Fig. 8(b).

8. Conclusion

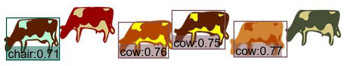
In this paper, we have shown that the SSD300 does not perform well in case of domain shift between source images and target ones when instance-level annotations available only in the source dataset. We can improve its performance by fine-tuning it with the stylized source domain images which looks like images of target domain. For styling the source domain images, we have investigated two domain transfer techniques - CycleGAN and AdaIN. SSD300 performed better when AdaIN is used for domain transfer of the source images.



(a)

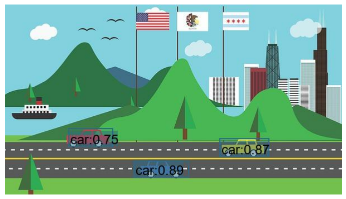


(b)

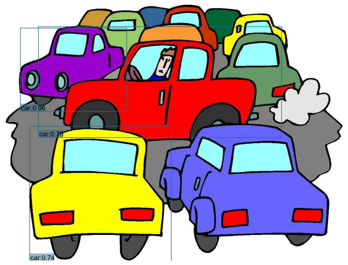


(c)

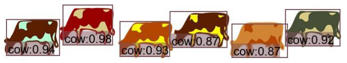
Figure 7: Object detection performance of the DT1 in the target domain



(a)



(b)



(c)

Figure 8: Object detection performance of the DT2 in the target domain

References

[1] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. Vaughan. A theory of learning from different domains. machine learning, 2010. 2

[2] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, S. L. K. Konolige, and V. Vanhoucke. Using simulation and domain adaptation to improve efficiency of deep robotic grasping, 2017. 2

[3] Z. Cao, M. Long, J. Wang, and M. I. Jordan. Partial transfer learning with selective adversarial networks. in 2018 ieee/cvf conference on computer vision and pattern recognition, 2018. 2

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding, 2016. 2

[5] J. Dai, Y. Li, K. He, and J. Sun. R-fcn: Object detection via region-based fully convolutional networks, in: Advances in neural information processing systems, 2016. 1, 2

[6] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg. Dssd: Deconvolutional single shot detector, 2017. 2

[7] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation, 2016. 2

[8] R. Girshick. Fast r-cnn, in: Proceedings of the ieee international conference on computer vision, 2015. 1, 2

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, in: Proceedings of the ieee conference on computer vision and pattern recognition, 2014. 2

[10] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Scholkopf, and Smola. A kernel two-sample test. journal of machine learning research, 2012. 2

[11] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. in: Proceedings of the IEEE conference on computer vision and pattern recognition. 2

[12] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. in eccv, 2012. 1

[13] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization, 2017. 2

[14] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization, 2017. 2, 3, 4

[15] N. Inoue, R. Furuta, T. Yamasaki, and K. Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 3, 5, 6

[16] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. mage-to-image translation with conditional adversarial networks, 2018. 2

[17] A. F. J. Redmon. Yolov3: An incremental improvement, arxiv preprint arxiv:1804.02767, 2018. 2

[18] Z. Jie, Y. Wei, X. Jin, J. Feng, and W. Liu. Deep self-taught learning for weakly supervised object localizationn, in: Proceedings of the ieee con8 ference on computer vision and pattern recognition, 2017. 2

[19] B. Kang, Z. Liu, X. Wang, F. Yu, J. Feng, and T. Darrell. Few-shot object detection via feature reweighting, 2019. 2

[20] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. in iccv. in: Proceedings of the IEEE conference on computer vision and pattern recognition. 1

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector, in: European conference on computer vision, springer, 2016. 1, 2, 3, 4

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, in: Proceedings of the ieee conference on computer vision and pattern recognition, 2016. 2

[23] J. Redmon and A. Farhadi. Better, faster, stronger. in cvpr, 2017. 1

[24] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, in: Advances in neural information processing systems, 2015. 1, 2

[25] Y. Shan, W. F. Lu, and C. M. Chew. Pixel and feature level based domain adaptation for object detection in autonomous driving, 2019. 2

[26] P. Tang, X. Wang, A. Wang, Y. Yan, W. Liu, J. Huang, and A. Yuille. Weakly supervised region proposal network and object detection, in: Proceedings of the european conference on computer vision (eccv, 2018. 2

[27] J. R. Uijlings, K. E. V. D. Sande, T. Gevers, and A. W. Smeulders. Selective search for object recognition. ijcv, 2013. 1

[28] H. Zhao, R. T. des Combes, K. Zhang, and G. J. Gordon. On learning invariant representation for domain adaptation,, 2019. 2

[29] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. in proceedings of the ieee international conference on computer vision, 2017. 2, 3, 4

[30] P. I. J.-Y. Zhu, T. Park, and A. A. Efros. Unpaired imageto-image translation using cycle-consistent adversarial networks. in 2017 ieee international conference on computer vision (iccv), 2017. 2