Object detection on haze image with domain adaptive Mask R-CNN

Boyuan Gong
Texas A&M University
boyuangong@tamu.edu

Yang Li
Texas A&M University
li13157@tamu.edu

Abstract

Nowadays, deep structure has achieved great success in computer vision society. Object detection is one of the most important application. Even though the deep neural network detection models have high mAP(Mean Average Precision) on training datasets, it still has a big performance drop when tested on cross-domain datasets. The reason behind this is the distribution gap between the training and testing datasets. In this report, we used domain adaptation method to tackle this problem. Our method focus on the feature map level domain robustness. Our model is trained on MSCOCO dataset and test on the RESIDE dataset(haze images), the best mAP we got is 63.36% *and the domain adaptation method increase the mAP by* 0.65%.

1. Introduction

Deep learning has powered many aspect of the engineering area, especially in the computer vision in the past decade[6]. Equipped with deep neural network technology, several algorithms have been developed to improve object detection and semantic segmentation [1][2][3][5]. The state-of-art algorithm Mask-R-CNN [3] even allows us to segment the objects in a pixel level.

However, many computer vision algorithms that used for detection can only work well with the scene radiance that is haze-free. The quality of images taken outdoors maybe severely influenced under the haze environment. The poor quality images sometimes fail to be recognized by the well trained detection models. Thus, remove haze and improve the detection precision becomes extremely important. Because of the distribution difference between the dehazing images and the clean images, although, many excellent dehazing method has been developed recent years [7] [8] [9][10], the detection precision still has a huge space for improvement.

In this project, our purpose is to tackle this cross-domain object detection problem. Our method can be used in the scenario when we have large well annotated clean image dataset(source domain) and small unannotated haze image dataset(target domain). The main contribution of our work can be summarized as follows:

1. We designed domain adaptation classifier to minimize the domain discrepancy at feature map level.
2. We integrated classifier to state-of-art Mask R-CNN model to build an end-to-end trainable domain adaptive Mask R-CNN.

2. Related Work

In this section we will give a brief introduction to the techniques related to our work which includes object detection, dehazing and domain adaptation methods.

2.1. Object detection method

2.1.1 Fast/Faster R-CNN

R-CNN:
The Region-based CNN (R-CNN) approach[19] to bounding-box object detection is to attend to a manageable number of candidate object regions and evaluate convolutional networks independently on each RoI. Based on that R-CNN was extended to allow attending to RoIs on feature maps using RoIPool, leading to fast speed and better accuracy[3].

Fast R-CNN[1]:

Fast R-CNN is Fast Region-based Convolutional Network. A Fast R-CNN network takes as input an entire image and a set of object proposals. The network first processes the whole image with several convolutional (conv) and max pooling layers to produce a conv feature map. Then, for each object proposal a region of interest (RoI) pooling layer extracts a fixed-length feature vector from the feature map. Each feature vector is fed into a sequence of fully connected (fc) layers that finally branch into two sibling output layers: one that produces softmax probability estimates over K object classes plus a catch-all background class and another layer that outputs four real-valued numbers for each of the K object classes. Each set of 4 values encodes refined bounding-box positions for

one of the K classes.

Faster R-CNN:

Faster R-CNN consists of two stages. The first stage, called a Region Proposal Network (RPN), proposes candidate object bounding boxes. The second stage, which is in essence Fast R-CNN[1], extracts features using RoIPool from each candidate box and performs classification and bounding-box regression. The features used by both stages can be shared for faster inference.

2.1.2 Mask R-CNN

Mask R-CNN is a powerful a structure as it can detect object with high MAP while do instance segmentation in a pixel level. The framework of Mask R-CNN shown as Figure 1. It extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition and classification.

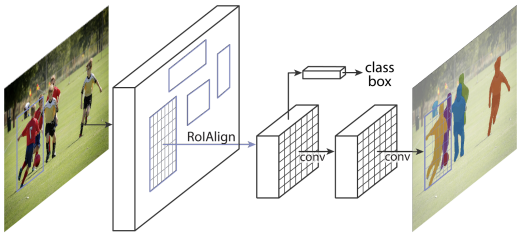


Figure 1. Mask R-CNN framework for instance segmentation [3]

2.2. Dehazing method

2.2.1 MSCNN-Net

The MSCNN-Net stands for Multi-scale CNN, is a image dehazing networks. This dehazing method is based on the physical model called *atmospheric scattering model* [13], which can be described as:

$$I(x) = J(x)t(x) + A(1 - t(x)) \tag{1}$$

where $I(x)$ is the observed hazy image and $J(x)$ is the clean image to be recovered. $A$is a matrix denote to global atmospheric light and t is the transmission matrix defined as:

$$t(x) = e^{-\beta d(x)} \tag{2}$$

where β is the scattering coefficient of the atmosphere and $d(x)$ is the distance between the object and camera.

Given a single hazy input, it aims to recover the latent clean image by estimating the scene transmission map. To estimate transmission map this dehazing method consists of a coarse-scale net which predicts a holistic transmission map based on the entire image, and a fine-scale net which refines results locally. The network is shown in figure2

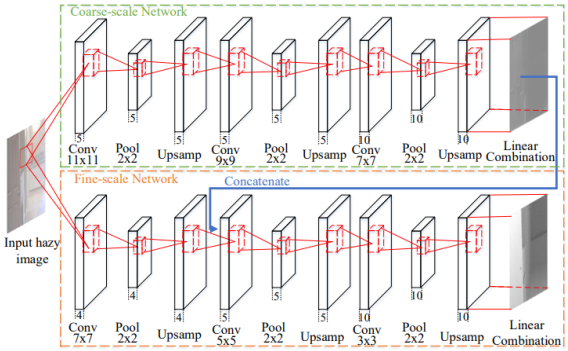
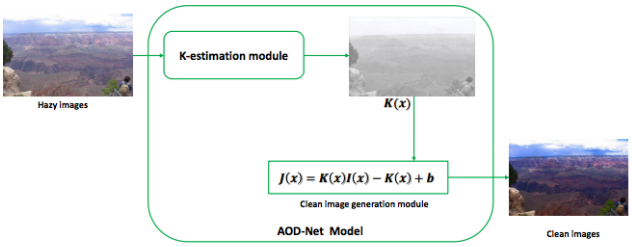


Figure 2. MSCNN [10]



(a) The diagram of AOD-Net

Figure 3. AOD-net [7]

After get scene transmission map $t(x)$ we need to estimate the atmospheric light A in order to recover the clear image. It estimates the atmosphere light A by selecting 0.1% darkest pixels in a transmission map $t(x)$. Among these pixels, the one with the highest intensity in the corresponding hazy image I is selected as the atmospheric light[10].

After get $t(x)$ and A we can use this Haze removal formulation to dehaze image:[10]

$$J(x) = \frac{I(x) - A}{max\{0.1, t(x)\}} + A \tag{3}$$

2.2.2 AOD-Net-Net

The AOD model is shown in Figure3. It builds based on the physical model called *atmospheric scattering model* [13], The equation is the same as $equation(1)$. The meaning of all shamble's are also same. From $equation(1)$, we can rewrite the equation to switch $J(x)$ to the RHS, so we can get:

$$J(x) = K(x)I(x) - K(x) + b \tag{4}$$

$$K(x) = \frac{\frac{1}{t(x)}(I(x) - A) + (A - b)}{I(x) - 1} \tag{5}$$

From universal approximation theorem[14], the matrix $K(X)$ can be approximated by a multiple layer neural net-

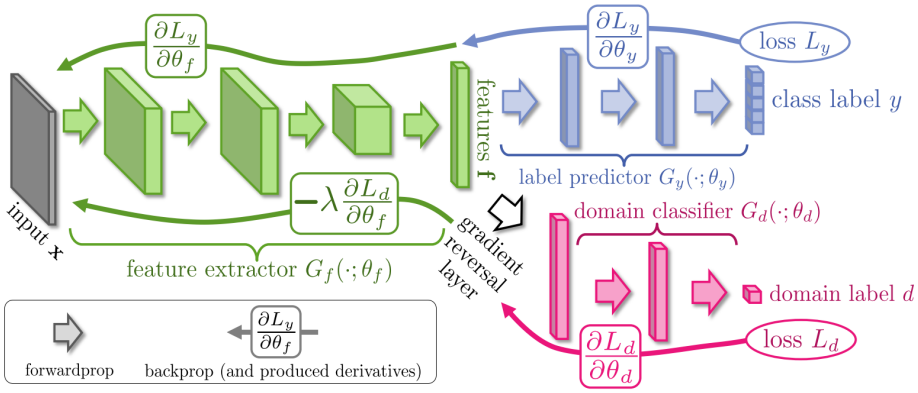


Figure 4. An domain adaptation implement example [11]

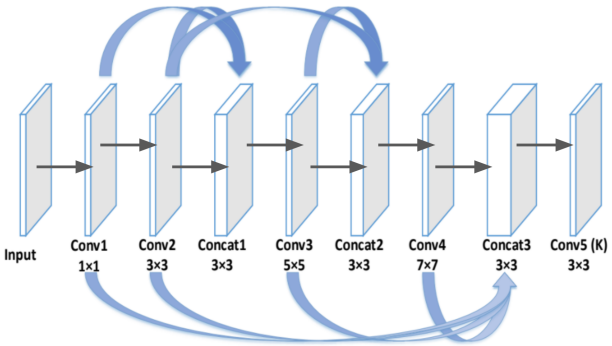


Figure 5. AOD-net Structure [7]

works. AOD-net [7], which shows in Figure5, is proposed based on this theorem and achieved a state-of-art performance to approximate the $K(x)$.

After we get $K(x)$ we can use the $equation(4)$ to rebuild the dehazed image, just like the Figure3

2.3. Domain adaptation method

Domain adaptation is proposed when people wish to train a model on the source data distribution while has a well performance on a different(but related) target data distribution. Domain adaptation allows us to do trained on large amount of labeled data from the source domain and large amount of unlabeled data from the target domain. This helps model get more the domain-invariant features such that model can predict labels given the input from the target distribution. After be proposed in 2015 by Ganin [11], it has been well developed by the community [15] [17] [18] [16].

The domain adaptation model is trained with the input x both from the source domain and the target domain(unlabeled). Use $S(x, y)$ and $T(x, y)$ be the two data and label distribution on domain $X \otimes Y$ represents source and data distribution respectively. And θ to be the weights of the correspond part. Showed in an example of domain adaptation in Figure 4: $f = G_f(x;\theta_f)$ is the feature vector. It will then be mapped to label space by $y = G_y(f;\theta_y)$ as well as the domain classifier by $d = G_d(f;\theta_d)$. The domain space is used to distinguish whether the input from the source domain or target domain.

During the training phase, we wish to minimize the loss of the label predictor L_y. Meanwhile, we wish to make the features f domain-invariant. Formally, we need to make $S(f) = G_f(x;\theta_f)|x \sim S(x)$ and $T(f) = G_f(x;\theta_f)|x \sim T(x)$ have small distance. However, there's no directly way to measure the difference of this two feature vector in a high dimension. The method here is to choose the weights θ_f which can fool the domain classifier(make the loss of domain classifier L_d) as larger as possible. Of course, we still need to make the domain classifier to be accurate, which means the loss of the domain classifier L_d should be as small as possible. Thus the error during the training can be written as:

$$\begin{aligned} L(\theta_f, \theta_y, \theta_d) = L_y(G_y|\theta_f, \theta_y, x \in \text{source domain}) \\ - \lambda L_d(G_d|\theta_f, \theta_d, x \in \text{all dataset}) \end{aligned} \quad (6)$$

The $-\lambda$ is used to make features similar when doing the gradient descent to θ_f rather than make it dissimilar.

Further more, for Mask R-CNN domain adaptation, we assume the posterior distribution $P(C, B|I)$ is the same for different domain images. where I is the feature map distribution, B is the bounding-box of an object and C is the class of object. Use Bayes formula:

$$P(C, B, I) = P(C, B|I)P(I)$$

We know that if the distribution of the feature map $p(I)$ is domain irrelevant, then the joint distribution(output of the MaskR-CNN) is domain irrelevant.

3. General framework

Our model which can be referenced as DMaskR-CNN is shown in Figure6.

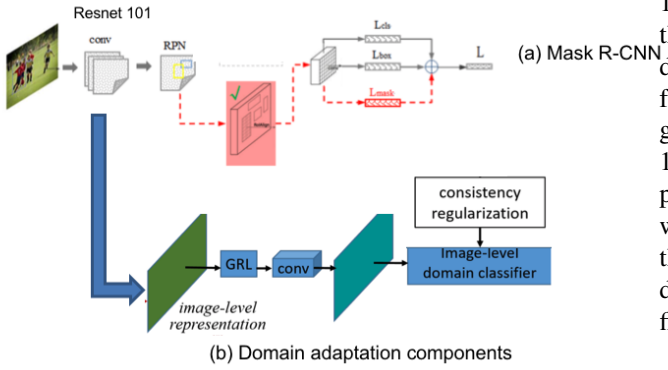


Figure 6. DMask RCNN-net Structure

Based on the section 2 Mask R-CNN's framework, the Domain adaptive Mask R-CNN's structure is relatively easy to understand. It has domain adaptive components after the base feature extract convolution layers of Mask R-CNN. The purpose of Domain adaptive Mask R-CNN is to mask the features generated by feature extraction network are as domain invariant as possible.

3.1. Mask R-CNN

Mask R-CNN combined ResNet[20] and FPN[21]. That combination is proved to be powerful in doing feature extraction. In the RPN, using ROIAlign instead of ROIPooling. Then comes the classification, bounding boxes regerssion, mask generate parts.

3.2. Domain classifier

As discussed in section 2.2, our goal is to train the feature map generated in the Resnet part to be domain irrelevant. Thus we build a domain classifier which takes the input of the four feature maps used for the head(bounding box and mask generation) part of the Mask R-CNN.The general frame work of the domain classifier is shown in figure7. Because the feature map is generated corresponding to each image, thus the domain classifier actually is predicting the domain label for each image.

The benefits of using the feature map as the domain classifier input has two reasons: First it greatly reduces the global difference of each image in the same domain. As the feature maps will extract high level features of each image, it will reduce the difference caused by the image style, object position, illumination or other unavoidable difference between each image. It will force the domain classifier focus on the distribution difference of the image. Second is that the feature maps have more information than a single image, which can make the domain classifier more accurate.

The detailed parameters for the domain classifier is as follows:

The inputs are four feature maps: 256*256 (F1), 128*128(F2), 64*64(F3), 32*32(F4). In order to make all those features to be same size and concatenate together to do domain classification(Binary classification) we let those features go through following convolution networks. F1 goes through four 2*2 conv2D with maxpolling and one 1*1 conv. F2 goes through three 2*2 conv2D with maxpolling and two 1*1 conv. F3 goes through two 2*2 conv2D with maxpolling and three 1*1 conv.Then we concatenate the output of these four convolutional layers and use one dense layer and one softmax layer to form a binary classifier.

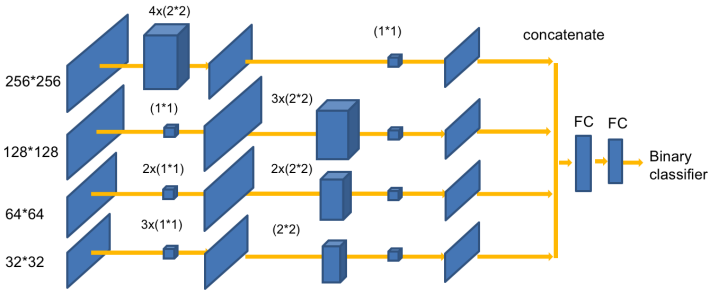


Figure 7. Convolution layers in domain classifier

3.3. Optimization details

The loss of the domain classifier is a binary cross entropy loss, which can be written as:

$$-\sum_i (y_i log(p_i) + (1-y_i) log(1-p_i)) \tag{7}$$

Where y_i is the domain label of the i_{th} image and p_i is the prediction probability from the domain classifier.

During the training phase, we set the total loss of the DMask R-CNN is:

$$\begin{aligned} L(\theta_{res}, \theta_{head}, \theta_{domain}) = \\ L_{C,B}(C, B|\theta_{res}, \theta_{head}, x \in \text{source domain}) \\ - \lambda L_d(G_d|\theta_{res}, x \in \text{source \& target}) \\ + \lambda L_d(G_d|\theta_{domain}, x \in \text{source \& target}) \end{aligned} \tag{8}$$

When we use gradient descent to update the model parameter, we have:

$$\theta_{head} = \theta_{head} - \partial L_{C,B}/\partial\theta_{head} \tag{9}$$

$$\theta_{domain} = \theta_{domain} - \partial L_d/\partial\theta_{domain} \tag{10}$$

$$\theta_{res} = \theta_{res} - \partial L_{C,B}/\partial\theta_{res} \tag{11}$$

$$\theta_{res} = \theta_{res} + \partial L_d/\partial\theta_{res} \tag{12}$$

Because to train domain adaptation model, we need to update the weights using the above four equation at the same time to make sure the convergence. Thus, we need to reverse the gradient from the domain classifier to the feature map extraction part, which is the weights of the Resnet.

Although this idea cannot be accomplished by simply using any optimization method in machine learning technique such as SGD, we can solve this by introducing a GRL(gradient reverse layer)[11] right after the feature maps(the input of the domain classifier). The definition of the GRL as follows: this layer has no parameters except for a hyper-parameter λ. During the forward propagation, it acts only as an identity transform. However, during the back propagation. It takes the gradient from the upper level and multiplies it by $-\lambda$ and pass it to the preceding layer.

Mathematically, we can refer the GRL

After integrated the GRL to our model, now we can rewrite the loss function of the model as the following, where $||w||_2$ is the weights regularization.

$$\begin{aligned} L(\theta_{res}, \theta_{head}, \theta_{domain}) = \\ L_{C,B}(C, B|\theta_{res}, \theta_{head}, x \in \text{source domain}) \\ - \lambda L_d(G_d|\theta_{res}, x \in \text{source \& target}) \\ + \lambda L_d(G_d|\theta_{domain}, x \in \text{source \& target}) \\ + ||w||_2 \end{aligned} \quad (13)$$

Then we can simply update the weights as follows:

$$\theta_{head} = \theta_{head} - \partial L_{C,B}/\partial\theta_{head} \quad (14)$$

$$\theta_{domain} = \theta_{domain} - \partial L_d/\partial\theta_{domain} \quad (15)$$

$$\theta_{res} = \theta_{res} - \partial L(\theta_{res}, \theta_{head}, \theta_{domain})/\partial\theta_{res} \quad (16)$$

To train the head parameters, we need to split the feature map into two parts: one is the feature map generated from the images in source domain. These images have ground truth annotation, so it can use for calculate the $L_{C,B}$. However, the other part which is generated from the images in target domain have no ground truth to calculate $L_{C,B}$. Thus, when calculate the $L_{C,B}$, we should only use half of the feature maps with respect to the batch dimension.

3.4. Relation to $\mathcal{H}$-divergence

$\mathcal{H}$-divergence [22] is used to measure the difference between to distribution by certain classifier. The distance between two distribution $\mathcal{T}, \mathcal{S}$ can be represent as follows:

$$d_{\mathcal{H}}(\mathcal{S}, \mathcal{T}) = 2(1 - min_{h \in \mathcal{H}}(err_{\mathcal{S}}(h(X)) + err_{\mathcal{T}}(h(X))) \quad (17)$$

Where X can be regard as the feature map. The $\mathcal{H}$ is a symmetric hypothesis class (one where for every $h \in \mathcal{H}$, the inverse hypothesis $1 - h$ is also in $\mathcal{H}$), which is suitable in the binary classification case.

Thus, to minimize the distance between two distributions, we need to find a saddle point where:

$$min_X d_{\mathcal{H}}(\mathcal{S}, \mathcal{T}) \leftrightarrow max_X min_{\theta_d}(err_{\mathcal{S}}(h(X)) + err_{\mathcal{T}}(h(X)) \quad (18)$$

To find the saddle point, the optimization method is exactly the optimization process introduced in the above section.

4. Experiment setup and result analysis

4.1. Experiment setup

We use a pretrained Mask R-CNN model on MSCOCO dataset for the Mask R-CNN part. We train two types of DMask R-CNN. Both are trained with one source dataset and one target dataset for 50k iteration with learning rate 0.001 and 20k iteration with learning rate 0.0001. The batch size is 2 which includes one random image selected from domain dataset and one random selected from target dataset.Then we measure the mAP on the RTTS (Real-world Task-Driven Testing Set) dataset from RESIDE.

4.2. Result analysis

For both DMask R-CNN model, we choose MSCOCO as our source dataset. We choose the original unannotated realistic haze image as one target dataset, and use the MSCNN dehazed unannotated realistic haze image for another. We refer the first DMask R-CNN model as DMask R-CNN1 and the second as DMask R-CNN2.

The mAP result shows as follows, we use dehaze method + model means the image from the dataset preprocessed with dehaze method before feed into the detection model. And the Mask R-CNN model is the baseline model which is the pretrained model.

Table 1. mAP results for different model

Framework	mAP(%)
Mask R-CNN	61.01
DMask R-CNN1	61.21
DMask R-CNN2	61.72
AOD-Net + DMask R-CNN1	60.21
AOD-Net + DMask R-CNN2	60.47
MSCNN + Mask R-CNN	62.72
MSCNN + DMask R-CNN1	62.71
MSCNN + DMask R-CNN2	63.36

Compare the mAP results show in above table, we achieved our best result is about 2.35% compare to using pre-trained model which is 63.36%. By combining domain adaptation to the Mask R-CNN, it boosts the result around 0.71%. Also from the difference between MSCNN + DMask R-CNN1 and MSCNN + DMask R-CNN2 we can

see that the domain adaptation works well in the trained target domain but sometimes may even has negative effect for other target domain. Different dehaze methods may have different effect on object detection. Although the PSNR and SSIM result for AOD-Net is above most of the current dehaze techniques, the detection precision may even drop when use it as image pre-processing. Thus, this lead us to an interesting result: PSNR and SSIM have weak relationship with the precision to some extent.

5. Conclusion

From the result, we show that domain adaptation has improvement on the object detection model. We also shows that some dehaze preprocessing methods of haze image also improve the object detection precision.

Further improve of our model may can focus on the domain irrelevant on the head part of Mask R-CNN. In the future we will further explore this problem. Hope we can get better result.

6. Individual contributions

Boyuan Gong:

- Paper review.
- Implement baseline Mask R-CNN.
- Implement Dmask R-CNN.
- Training, testing model and do result analyzation.

Yang Li:

- Paper review.
- Implement dehaze methods.
- Implement Dmask R-CNN.
- Training, testing Model and do result analyzation.

References

[1] Girshick, Ross. "Fast r-cnn." arXiv preprint arXiv:1504.08083 (2015).

[2] Ren, Shaoqing, et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." Advances in neural information processing systems. 2015.

[3] He, Kaiming, et al. "Mask r-cnn." Computer Vision (ICCV), 2017 IEEE International Conference on. IEEE, 2017.

[4] Redmon, Joseph, et al. "You only look once: Unified, real-time object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[5] Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[6] LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015): 436.

[7] Li, Boyi, et al. "Aod-net: All-in-one dehazing network." Proceedings of the IEEE International Conference on Computer Vision. 2017.

[8] Cai, Bolun, et al. "Dehazenet: An end-to-end system for single image haze removal." IEEE Transactions on Image Processing 25.11 (2016): 5187-5198.

[9] Tan, Robby T. "Visibility in bad weather from a single image." Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on. IEEE, 2008.

[10] Ren, Wenqi, et al. "Single image dehazing via multi-scale convolutional neural networks." European conference on computer vision. Springer, Cham, 2016.

[11] Ganin, Yaroslav, and Victor Lempitsky. "Unsupervised domain adaptation by backpropagation." International Conference on Machine Learning. 2015.

[12] He, Kaiming, et al. "Mask r-cnn." Computer Vision (ICCV), 2017 IEEE International Conference on. IEEE, 2017.

[13] Meng, Gaofeng, et al. "Efficient image dehazing with boundary constraint and contextual regularization." Computer Vision (ICCV), 2013 IEEE International Conference on. IEEE, 2013.

[14] Hornik, Kurt, Maxwell Stinchcombe, and Halbert White. "Multilayer feedforward networks are universal approximators." Neural networks 2.5 (1989): 359-366.

[15] Tzeng, Eric, et al. "Adversarial discriminative domain adaptation." Computer Vision and Pattern Recognition (CVPR). Vol. 1. No. 2. 2017.

[16] Chen, Yuhua, et al. "Domain Adaptive Faster R-CNN for Object Detection in the Wild." arXiv preprint arXiv:1803.03243 (2018).

[17] Liu, Ming-Yu, and Oncel Tuzel. "Coupled generative adversarial networks." Advances in neural information processing systems. 2016.

[18] Sankaranarayanan, Swami, Yogesh Balaji, and Carlos D. Castillo Rama Chellappa. "Generate To Adapt: Unsupervised Domain Adaptation using Generative Adversarial Networks."

[19] Girshick, Ross, et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[20] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[21] Lin, Tsung-Yi, et al. "Feature pyramid networks for object detection." CVPR. Vol. 1. No. 2. 2017.

[22] Ben-David, Shai, et al. "A theory of learning from different domains." Machine learning 79.1-2 (2010): 151-175.