Adapting Object Detectors with Conditional Domain Normalization

Peng Su[1,2], Kun Wang[2], Xingyu Zeng[2], Shixiang Tang[2]
Dapeng Chen[2], Di Qiu[2], and Xiaogang Wang[1]

[1] The Chinese University of Hong Kong
[2] SenseTime Research
{psu,xgwang}@ee.cuhk.edu.hk

1 Interpret the Domain Embedding

Conditional domain normalization (CDN) disentangles the domain-specific attribute out of the semantic features from one domain via a learning a domain embedding to characterize the domain attribute information. In this section, we interpret the learned domain embedding via reconstructing the RGB images from the features. As shown in Fig. 1, we first built a decoder network $Decoder(\cdot;\theta_{dec})$ upon the backbone network $G(\cdot;\theta_g^*)$ of fixed weights. The parameters of the backbone network are obtained in the adaptation training (see Eq.1). The decoder network mostly mirrors the backbone network, with all pooling layers replaced by nearest up-sampling and all normalization layers removed. The decoder network is trained to reconstruct the RGB images from the features extracted by the backbone,

$$\arg\min_{\theta_{dec}} \mathcal{L} = ||Decoder(G(x;\theta_g^*);\theta_{dec}) - x||_2. \tag{1}$$

For contrast analysis, only single domain images are used to train the decoder network, i.e. the decoder for Cityscapes experiment is trained on Foggy Cityscapes images, the decoder for SIM10K experiment is trained on SIM10K images. After we got a trained decoder network, we use it to reconstruct the RGB image from features encoded with the domain embedding.

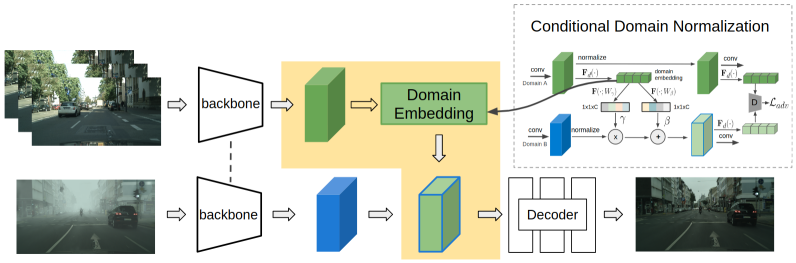

Fig. 1: Interpreting the learned domain embedding with a decoder network.

Fig. 2 shows the effect of domain embedding learned in Cityscapes to Foggy Cityscapes adaptation experiments (Section 5.1). The top row shows the inputs of Foggy Cityscapes; the middle row shows the reconstructed results from features of Foggy Cityscapes inputs; the bottom row is reconstructed results from Foggy Cityscapes features encoded with the domain embedding learned on Cityscapes. With the help of the domain embedding learned on Cityscapes, the reconstructed results from Foggy Cityscapes features no longer exhibit foggy characteristics, suggesting that both Cityscapes and Foggy Cityscapes inputs are embedded into a shared latent space, where their features carry the same domain attribute. Given the domain gap bridged, the object detector supervised trained on Cityscapes also works on Foggy Cityscapes.

Fig. 2: Top row: Original inputs of Foggy Cityscapes; Middle row: Reconstructed results from features of original inputs; Bottom row: Reconstructed results from features encoded with the domain embedding of Cityscapes.

Fig. 3 and 4 show the reconstructed results from synthetic data's features encoded with domain embedding of real data (BDD100K), which are learned in SIM10K-to-BDD100K and Synscapes-to-BDD10K adaptation experiments, respectively (see Section 5.4). Without the domain embedding of real data, the reconstructed images (middle row of Fig 3 and 4) still exhibit characteristic of CG (computer graphic), that look identical to the original images. When the same features of original inputs are encoded with the domain embedding of real data, the reconstructed images (bottom row of Fig 3 and 4) obviously becomes more realistic. For example, the color of the sky, the texture of the road and objects in the reconstructed images look similar to the real images. It

proves that the learned domain embedding well captures the domain attribute information of real data, and it can be used to effectively translate the synthetic images towards real images.

Fig. 3: Top row: Original inputs of SIM10K; Middle row: Reconstructed results from features of original inputs; Bottom row: Reconstructed results from features encoded with the domain embedding of BDD100K.

Fig. 4: Top row: Original inputs of Synscapes; Middle row: Reconstructed results from features of original inputs; Bottom row: Reconstructed results from features encoded with the domain embedding of BDD100K.

2 Qualitative Results

Fig. 5 shows example results on the Foggy Cityscapes dataset. Our method can detect distant objects in fog, suggesting strong generalization capability under foggy weather conditions.

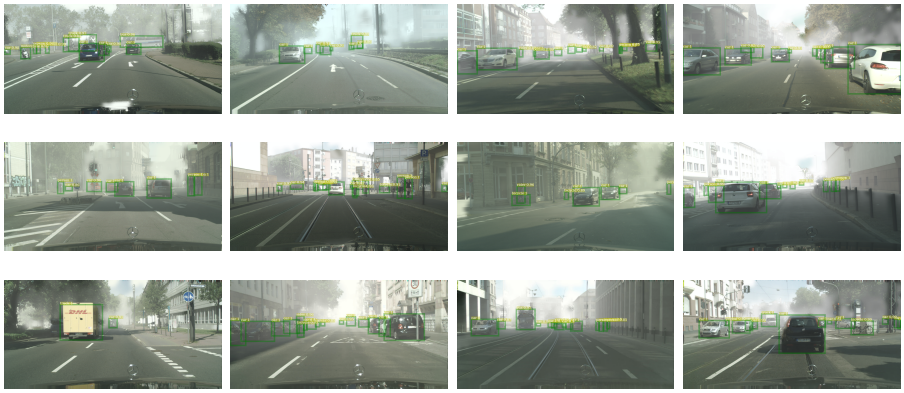

Fig. 5: Detection results on *Foggy Cityscapes* test. The class and score predictions are at the top left corner of the bounding box. The results are produced by a Faster RCNN model incorporated with the Conditional Domain Normalization.

Fig. 6 shows example results on the KITTI dataset.

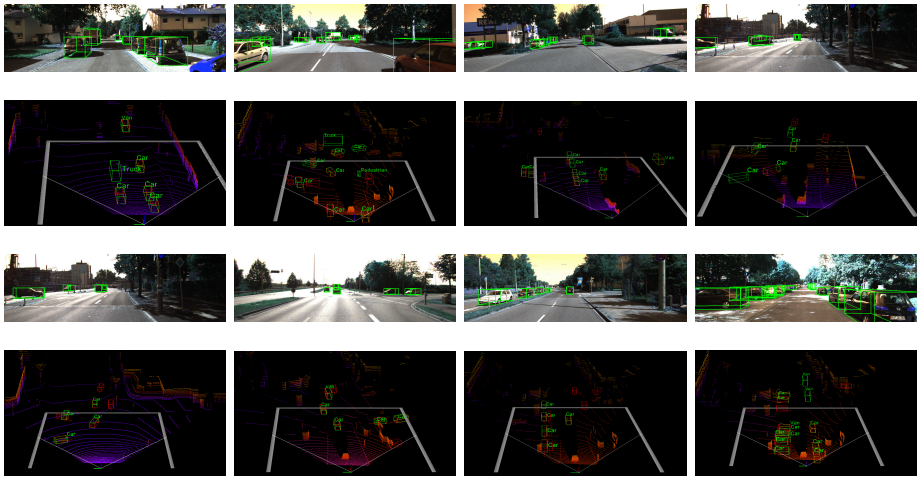

Fig. 6: Example results of PointRCNN model with Conditional Domain Normalization. For each example, the upper part is the image and the lower part is the corresponding point cloud. The detected objects are shown with red 3D bounding boxes. The green bounding boxes represent the ground truth.

Fig. 7 shows example results on the Synthetic and Real dataset. Our method produces accurate bounding boxes on the real dataset (BDD100K).

Fig. 7: Example results on *Virtual KITTI* (1st row), *Synscapes* (2nd row), *SIM10K* (3rd row) and *BDD100K* (4-6th row). The class and score predictions are at the top left corner of the bounding box. Zoom in to visualize the details. The results are produced by a Faster RCNN model incorporated with the Conditional Domain Normalization layer.