Supplimentary Materials: Learning Object Placement by Inpainting for Compositional Data Augmentation

Lingzhi Zhang[1], Tarmily Wen[1], Jie Min[1],
Jiancong Wang[1], David Han[2], and Jianbo Shi[1]

[1] University of Pennsylvania
[2] Army Research Laboratory
{zlz,went,minjie,jshi}@seas.upenn.edu,
jiancong.wang@pennmedicine.upenn.edu, ctmkhan@gmail.com

In this section, we discuss the implementation details of data acquisition module, object placement newtork, and compositional data augmentation pipeline in our paper.

1 Implementation Details of Image Inpainting

To implement the image inpainting module in our data acquisition system, we utilize a state-of-the-art image inpainting network [3]. We used the unlabeled Cityscape images [1], which has a total number of 184,700. To generate the ocludded images, we crop out the pixel regions at several random locations using the object masks, which contain shapes of car, person, bicycle, and etc. Then, we train the inpainting network to fill the holes of these cropped regions to learn the image prior.

2 Implementation Details of PlaceNet

In the training of object placement network, the foreground and background images have dimensions of 128x128 and 128x256 respectively, and the output placements are parameterized as normalized center point (x,y), width and height in range of 0 and 1.

We employ the following abbreviation: N = Number of filters, K = Kernel size, S = Stride, P = Padding. "Conv" and "FC", "BN", "SN" denote convolutional layer, fully-connected layer, batch normalization, and spectral normalization respectively. The network architecture are shown as follows.

In the encoding space, the foreground image is encoded as a 128d feature vector and the background image is encoded as a 256d feature vector. The dimension of random variable is 2. In the latent space, the foreground code, background code and the random variable are concatenated and then decoded to a predicted placement. Therefore, the input dimension to the decoder is 128+256+2 = 386.

The inputs to the discriminator network are the predicted/ground-truth placement plus the foreground and background images. The foreground and

background are encoded into the feature space using the same architecture as foreground and background encoders but with spectral normalization instead of batch normalization. After that, the foreground and background code are concatenated with the predicted/ground-truth placement and are fed into a few more layers of fully-connected layers to output the real or fake probability. The input dimension for the last few fully-connected layers is 128+256+4 = 388.

During training, we set the relaxation hyper-parameter α to be 0.8 in the diversity loss, and the weight ratio between diversity loss and adversarial loss is 1:1. To enforce the normalized pairwise diversity loss, we sample four random variables at each iteration. We use Adam optimizer [2] with learning rate of 2e-4, beta 1 of 0.5, and beta 2 of 0.999, and use batch size of 16. The maximum iteration is set to be 200K.

Layer	Hyper-parameters
1	Conv(N32-K4-S2-P1) + BN + ReLU
2	Conv(N64-K4-S2-P1) + BN + ReLU
3	Conv(N128-K4-S2-P1) + BN + ReLU
4	Conv(N256-K4-S2-P1) + BN + ReLU
5	Conv(N512-K4-S2-P1) + BN + ReLU
6	Conv(N1024-K4-S1-P0) + ReLU
7	Conv(N128-K1-S1-P0)

Table 1: Foreground Encoder Network.

Layer	Hyper-parameters
1	Conv(N32-K4-S2-P1) + BN + ReLU
2	Conv(N64-K4-S2-P1) + BN + ReLU
3	Conv(N128-K4-S2-P1) + BN + ReLU
4	Conv(N256-K4-S2-P1) + BN + ReLU
5	Conv(N512-K4-S2-P1) + BN + ReLU
6	Conv(N1024-K4-S1-P0) + ReLU
7	Conv(N256-K1-S1-P0)

Table 2: Background encoder network.

Layer	Hyper-parameters
1	FC(I386-O256) + BN + ReLU
2	FC(I256-O128) + BN + ReLU
3	FC(I128-O64) + BN + ReLU
4	FC(I64-O4) + Sigmoid

Table 3: Placement decoder network.

Layer	Hyper-parameters
1	FC(I388-O256) + SN + ReLU
2	FC(I256-O128) + SN + ReLU
3	FC(I128-O64) + SN + ReLU
4	FC(I64-O1) + Sigmoid

Table 4: Last few layers of the discriminator. The foreground and background encoders in the discriminator are the same as table 1 and 2, except for using spectral normalization instead of batch normalization.

References

1. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3213–3223 (2016) 1
2. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014) 2
3. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Generative image inpainting with contextual attention. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5505–5514 (2018) 1