Supplementary Material for CVPR submission #4406

Ryuhei Hamaguchi, Ken Sakurada, and Ryosuke Nakamura
National Institute of Advanced Industrial Science and Technology (AIST)
{ryuhei.hamaguchi, k.sakurada, r.nakamura}@aist.go.jp

1. Introduction

This document presents the supplementary materials omitted from the main paper due to the space limitation. In Section 2, details about the WDC dataset are presented. In Section 3, the detailed settings for the experiments in the main paper are presented. In Section 4, the additional ablation study is presented.

2. WDC dataset

In this dataset, we consider a more realistic application where we want to find rare events from a large image archives such as satellite images or street view images. To evaluate our method on such applications, we created a new large-scale dataset for detecting newly constructed or destructed buildings in Washington D.C. area from a large archive of aerial images taken in 1995 through 2015.

2.1. Source data and annotations

For the WDC dataset, we acquired the aerial images and the building footprints of Washington D.C. area from open data repository hosted by the Government of District of Columbia [6]. We used the aerial images taken in 1995, 1999, 2002, 2005, 2008, 2010, 2013 and 2015. The images have 16 cm resolution, and covers over $200km^2$ for each year. We automatically annotated changes in buildings by comparing the building footprints produced at different years. Specifically, for each building, we computed IoU between footprints of different years, and annotated the buildings as newly constructed or destructed if the $IoU < 0.01$. We conducted the annotation using the footprint pairs of 1999 and 2005, and of 2010 and 2015. Although the building footprints are provided for other years, we find that the difference between them includes many buildings that are missed in the previous year's survey. Because they are not the true change, we decided not to use the footprints.

2.2. Patch pairs for training and evaluation

First, we paired off with the aerial images that are close to each other in their acquisition year. From the image pairs, we cropped a large amount of noisy negative patch pairs and a small amount of purely negative and purely positive patch pairs. The purely negative and purely positive patch pairs are created from the change annotations. The noisy negative patch pairs are randomly cropped from all the image pairs available regardless of the existence of the change annotation. Because our targets are rare events, such random samples are almost negative except accidentally cropped positives. The noisy negative samples are used to train our representation learning model. Note that the training is fully unsupervised because creating noisy negative samples requires no labels. Then, a part of purely negative and purely positive samples are used for fine-tuning the change classifier, and the rest of samples are used for evaluating the model performance.

2.3. Filtering patches using building footprint

Since the source aerial images include large areas without buildings, randomly cropped patches rarely contain buildings. In order to put more attention to buildings, we used the building footprint of 2015 as side information to filter out patches that are not containing buildings. Specifically, we controlled the ratio of patches containing and not containing buildings as 9:1.

3. Hyperparameter settings

In the experiments, we used two encoder architectures for the proposed method: one is for Augmented MNIST (Enc-MNIST) and the other is for ABCD, PCD, and WDC datasets (Enc-VGG). Table 1 shows the detailed architectures of the encoders. The decoders are set as symmetric to the encoders by replacing convolution with deconvolution and subsampling with upsampling. For fair comparisons in terms of network capacity, we basically used the same encoder architectures for the baseline models so that they have the same capacity in fine-tuning phase (see Table 2). As an exception, we needed to use the specific architecture for the model of mathieu et al. [4], in order to stabilize the GAN training. Specifically, we modified the Enc-MNIST by inserting instance normalization after every convolution. Moreover, for ABCD, PCD and WDC datasets, we used DCGAN-based architecture.

Table 1. Encoder architecture used for Augmented MNIST (Enc-MNIST), and for ABCD, PCD, and WDC dataset (Enc-VGG). The Enc-VGG is based on VGG-16 [5]. Several layers in the encoders have connections to the hidden layers as listed in the 5th column of the table ("Hidden"). For example, "H1" in the column represents connection to the operation "Conv(H1)". The bottom part of the table shows the architectures of hidden layers, where the number of common and specific features are shown in the second column ("#Features")

(a) Enc-MNIST

Operations	#Features	Kernel size	Stride	hidden	Spatial dimensions
Conv-ReLU	20	5×5	1	-	24
Max-pool	20	2×2	2	-	12
Conv-ReLU	50	5×5	1	-	8
Max-pool	50	2×2	2	-	4
Conv-ReLU	500	5×5	1	H1	1
Conv(H1)	c,s=(20, 10)	1×1	1	-	1

(b) Enc-VGG

Operations	#Features	Kernel size	Stride	Hidden	Spatial dimensions
Conv-ReLU	64	3×3	1	-	128
Conv-ReLU	64	3×3	1	H1	128
Max-pool	64	2×2	2	-	64
Conv-ReLU	128	3×3	1	-	64
Conv-ReLU	128	3×3	1	H2	64
Max-pool	128	2×2	2	-	32
Conv-ReLU	256	3×3	1	-	32
Conv-ReLU	256	3×3	1	-	32
Conv-ReLU	256	3×3	1	H3	32
Max-pool	256	2×2	2	-	16
Conv-ReLU	512	3×3	1	-	16
Conv-ReLU	512	3×3	1	-	16
Conv-ReLU	512	3×3	1	H4	16
Max-pool	512	2×2	2	-	8
Conv-ReLU	1024	7×7	1	-	2
Conv-ReLU	1024	1×1	1	H5	2
Conv(H1)	c,s=(0,32)	64×64	1	-	65
Conv(H2)	c,s=(0,32)	32×32	1	-	33
Conv(H3)	c,s=(0,32)	16×16	1	-	17
Conv(H4)	c,s=(16,16)	8×8	1	-	9
Conv(H5)	c,s=(16,16)	1×1	1	-	2

Table 2. Encoder architectures used for each model.

	Aug. MNIST	ABCD, PCD, WDC
Under samp.	Enc-MNIST	Enc-VGG
Over samp.	Enc-MNIST	Enc-VGG
Transfer	Enc-MNIST	Enc-VGG
MLVAE [1]	Enc-MNIST	Enc-VGG, one hidden
Mathieu et al. [4]	Enc-MNIST with instance norm.	DCGAN based
VAE w/o Sim.	Enc-MNIST	Enc-VGG
VAE w/ Sim. (ours)	Enc-MNIST	Enc-VGG

For the experiments on Augmented MNIST, Adam [3] was used for optimization with an initial learning rate of 1.0e-3. A coefficient of weight decay term was set to 5.0e-4. All models were trained for 30 epochs using a batch size of 100. λ_1 and λ_2 were both set to 1, and a sparsity parameter of 0.5 was used.

For the experiments on ABCD, PCD, and WDC datasets, Adam optimizer was used for optimization with an initial learning rate of 1.0e-4. The learning rate decayed linearly with the number of epochs. A coefficient of weight decay term was set to 5.0e-4. All models were trained for 30 epochs using a batch size of 50. λ_1 was set to 10, and λ_2 was set to 1. We also used the KL annealing technique [2], which gradually ascend the coefficient on the KL divergence term in the VAE loss function. The weight was gradually increased during training from 0 to 1.

Table 3. Sensitivity analysis of the kernel size of hidden layers.

Kernel size					
H1	H2	H3	H4	H5	Acc.
64	32	16	8	1	90.01(1.39)
32	32	16	8	1	89.35(1.86)
16	16	16	8	1	89.36(1.36)
8	8	8	8	1	88.79(1.33)
3	3	3	3	1	88.29(2.26)

4. Ablation Study: kernel size for hidden layers

In Table 3, we demonstrated a sensitivity analysis on the kernel size of the hidden layers. The result shows that the performance improves when we use larger kernels at the lower layers. This result can be explained as follows: feature maps in lower layers somewhat retain raw information from the input. If we use a small kernel in lower hidden layers, the spatial information is almost all retained, which enables the model to easily reconstruct input images. When the model can reconstruct inputs by using only lower hidden layers, higher layers cannot receive sufficient error signals and the training will be stacked. On the other hand, if we use large kernels, the convolution operation becomes close to fully connected, which promotes abstraction of the hidden variables.

References

[1] D. Bouchacourt, R. Tomioka, and S. Nowozin. Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations. *arXiv preprint arXiv:1705.08841*, 2017. 2

[2] S. R. Bowman, L. Vilnis, O. Vinyals, A. M. Dai, R. Jozefowicz, and S. Bengio. Generating Sentences from a Continuous Space. *CONLL*, 2016. 2

[3] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 2

[4] M. Mathieu, J. Zhao, P. Sprechmann, A. Ramesh, and Y. LeCun. Disentangling factors of variation in deep representations using adversarial training. *NIPS*, 2016. 1, 2

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014. 2

[6] `http://opendata.dc.gov/pages/dc-from-above`. DC GIS program. 1

(a) Positive samples (b) Negative samples

Figure 1. Examples of positive pairs and negative pairs in ABCD dataset.

Figure 2. Examples of original image pairs in PCD dataset.

(a) Positive samples (b) Negative samples

Figure 3. Examples of positive pairs and negative pairs cropped from image pairs in PCD dataset.

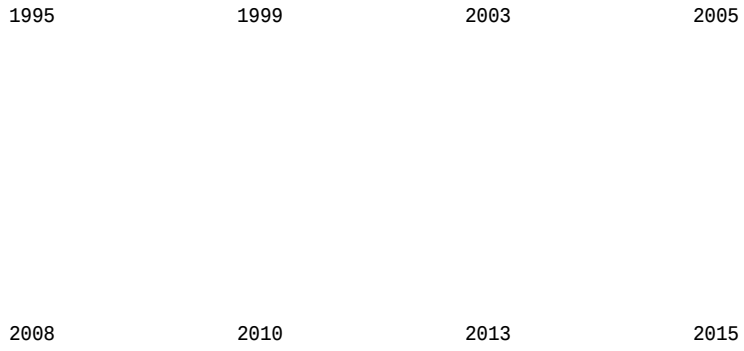

Figure 4. Source aerial images of Washington D.C. area.

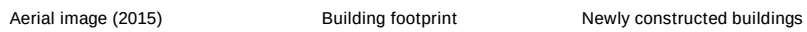

Figure 5. Example of aerial images, building footprints and change labels for WDC dataset. The change label on the right is created by comparing building footprints of 2015 and 2010.

Figure 6. Examples of purely positive pairs and noisy negative pairs in WDC dataset.