Rare Event Detection using Disentangled Representation Learning

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

Abstract

This paper presents a novel method for rare event detection from an image pair with class-imbalanced datasets. A straightforward approach for event detection tasks is to train a detection network from a large-scale dataset in an end-to-end manner. However, in many applications such as building change detection on satellite images, few positive samples are available for the training. Moreover, scene image pairs contain many trivial events, such as in illumination changes or background motions. These many trivial events and the class imbalance problem lead to false alarms for rare event detection. In order to overcome these difficulties, we propose a novel method to learn disentangled representations from only low-cost negative samples. The proposed method disentangles different aspects in a pair of observations: variant and invariant factors that represent trivial events and image contents, respectively. The effectiveness of the proposed approach is verified by the quantitative evaluations on four change detection datasets, and the qualitative analysis shows that the proposed method can acquire the representations that disentangle rare events from trivial ones.

1. Introduction

In the field of computer vision, event detection from an image pair has been comprehensively studied as image similarity estimation. Similarity estimation between images is one of the fundamental problems, which can be applied for many tasks, such as change detection [11, 14, 20, 25], image retrieval and matching [3, 23, 33], identification [26, 31], and stereo matching [9, 34]. Thanks to the recent success of deep features, the image comparison methods have substantially progressed. However, a general draw back is that they require a large amount of dataset to fully utilize the representational power of the deep features.

In the context of image similarity estimation, this paper considers a particular task of detecting rare events from an image pair, such as detecting building changes on a pair of satellite images, or detecting manufacturing defects by comparing images of products. One challenge of the task lies in the difficulty of collecting training samples. Because finding rare samples is labor intensive task, the training dataset often includes few positive samples. Additionally, image pairs often contain many cumbersome events that are not of interest (*e.g.,* illumination changes, registration error of images, shadow changes, background motion, or seasonal changes). These many trivial events and the class imbalance problem lead to false alarms for trivial events, or overlooking the rare events.

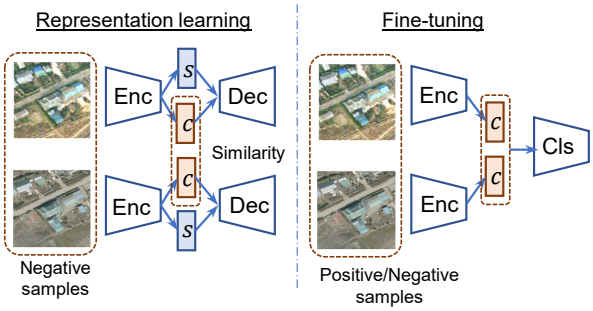

Figure 1. The overall concept of the proposed model. From the negative image pairs, the representation learning model (**left**) learns features that are invariant to trivial events. The rare event detector (**right**) is then trained on the learned invariant features.

In order to overcome these difficulties, we propose a novel network architecture for disentangled representation learning using only low-cost negative image pairs. Figure 1 demonstrates the overall concept of the proposed method. The proposed network is trained to encode each image into the two separated features, *specific* and *common*, by introducing a similarity constraint between the image contents. The common features represent image contents that is invariant to trivial events, and the specific features represent a mixture of information related to trivial events (*e.g.,* illumination, shadows, or background motion). This disentanglement can be learned using only low-cost negative samples because negative samples contain rich information about trivial events. Once we have acquired the common features, we can build rare event detectors on the learned representations using small amount of training samples.

The effectiveness of the proposed method on the class-imbalance scenario is verified by the quantitative evaluations on four change detection datasets, including in-the-wild datasets. In addition, the qualitative analysis shows that the proposed method successfully learns the disentangled representation for both rare events and trivial ones in the image pairs. The contributions of this work are as follows:

- We propose a novel solution to the class imbalance problem in rare event detection tasks, which has not been fully studied in the past literature.
- We propose a novel representation learning method that only requires pairs of observations to learn disentangled representations.
- We create a new large-scale change detection dataset from the open data repository of Washington D.C.

2. Related Work

In change detection tasks, several works have attempted to overcome the difficulties of data collection and cumbersome trivial events as described in the previous section. In order to save the cost of annotation, [13] proposed a weakly supervised method that requires only image-level labels to train their change segmentation models. Although their work saves the pixel-level annotation cost, it still requires image-level labels, which are still difficult to collect for rare change events. To address trivial events, several works on video surveillance tasks [4, 24] utilize background modeling techniques in which foreground changes are detected as outliers. However, these works assume a continuous frame as the input, and their application is limited to change detection in video frames. [12] proposed a semi-supervised method to detect damaged areas from pairs of satellite images. In their method, a bag-of-visual-words vector is extracted for hierarchical shape descriptors and a support vector machine classifier is trained on the extracted features. Since their method is based on the carefully chosen feature descriptors specialized for their task, the method lacks generalizability for application in other domains.

Disentangled representation learning is an actively studied field. [27] proposed a generative adversarial network (GAN) framework to learn disentangled representation for pose and identity of a face using encoder-decoder architecture with auxiliary variables inserted in its latent code. [18] proposed a GAN model that can generate synthetic images conditioned on category labels. [22] proposed a semi-supervised method to learn disentangled representation by introducing graphical model structure between the encoder and decoder of a standard variational auto-encoder (VAE). A drawback of these methods is that during training, they require explicit labels for the target factor of variation. As for an unsupervised approach, [8] proposed a method that learns disentangled representation by maximizing mutual information between a small subset of latent codes and a generated image. However, this method cannot control the disentanglement so that the desired factor of variations is represented in a certain latent code. Some works utilize groups of observations as weak supervision. [16] trains a target latent unit on grouped mini-batches that include only one factor of variation. [5] and [17] proposed a method that effectively disentangles intra-class and inter-class variations using groups of images sharing the same class labels. Our work is similar to the three works mentioned above. The difference is that our work assumes weaker conditions; that is, our method only requires pairs of observations and does not require aligned observations or class labels. Recently, multi-view image generation method that only use a paired observation for feature disentanglement is proposed in [7].

The method that is most related to our work is Domain Separation Network (DSN) [6]. DSN decomposes an image into a common and a specific factors between two different image domains. To learn the disentanglement, DSN penalizes distance between marginals of common features: $\mathcal{D}(p(\boldsymbol{z}_A) \parallel p(\boldsymbol{z}_B))$, where $p(\boldsymbol{z}_A) = \mathbb{E}_{p(\boldsymbol{x}_A)}[p(\boldsymbol{z}_A|\boldsymbol{x}_A)]$ and $p(\boldsymbol{z}_B) = \mathbb{E}_{p(\boldsymbol{x}_B)}[p(\boldsymbol{z}_B|\boldsymbol{x}_B)]$. While the method is effective on domain adaptation tasks, it is not applicable to image comparison tasks such as rare event detection. This is because the image comparison tasks do not assume domain bias across $p(\boldsymbol{x}_A)$ and $p(\boldsymbol{x}_B)$ that is essential for DSN to learn the disentanglement. On the other hand, our method penalizes the distance between the posteriors instead of the marginals; that is, $\mathcal{D}(p(\boldsymbol{z}_A|\boldsymbol{x}_A) \parallel p(\boldsymbol{z}_B|\boldsymbol{x}_B))$. Since the loss does not involve the expectation of $p(\boldsymbol{x}_A)$ and $p(\boldsymbol{x}_B)$, our method is applicable regardless of the existence of domain bias across $p(\boldsymbol{x}_A)$ and $p(\boldsymbol{x}_B)$.

3. Methods

3.1. Overview

Figure 2 shows a schematic of the proposed model. The model consists of two branches of VAEs that share parameters each other. Each VAE extracts two types of feature representations: *common* and *specific*. They represent different aspects of an input image pair, invariant and variant factors, respectively. In the context of rare event detection, the specific features represent trivial events, and the common features represent image contents that are invariant to trivial events. In order to achieve the disentanglement, we introduce a similarity constraint between common features. This constraint promotes common features to lie in a shared latent space of paired images. The key aspect of the common features is that they are invariant to trivial events, which should be helpful to distinguish target events from trivial

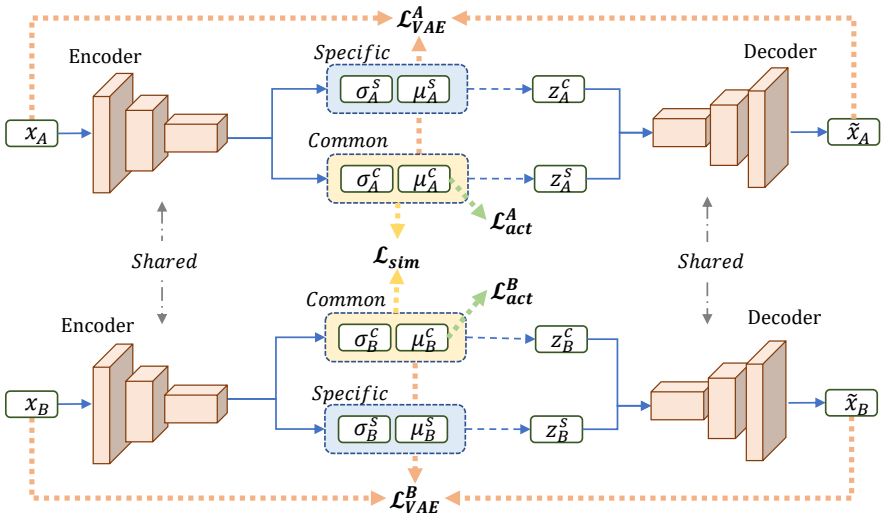

Figure 2. Schematics of the proposed representation learning method. The model takes a pair of images $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$ as input. For each image, the encoder extracts common and specific features, and the decoder reconstructs the input. The key feature of the model is the similarity loss $\mathcal{L}_{sim}$. This loss constrains the common features to extract invariant factors between $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$. Another feature is the activation loss $\mathcal{L}_{act}$. This loss encourages the mean vector of the common features ($\boldsymbol{\mu}^c$) to be activated, which avoids a trivial solution – $(\boldsymbol{\sigma}^c, \boldsymbol{\mu}^c) = (\mathbf{1}, \mathbf{0})$ – for any input.

events. In the successive fine-tuning phase, an event detector is trained on the learned common features using a small number of positive and negative samples.

The contents of this section are as follows. In Section 3.2, we present a brief introduction to VAEs. In Section 3.3, the proposed method of representation learning is explained in detail. Finally, in Section 3.4, the fine-tuning phase of the event detector is explained.

3.2. Variational Auto-encoder

A variational auto-encoder [15, 19] is a kind of deep generative model that defines the joint distribution of an input $\boldsymbol{x} \in \boldsymbol{X}$ and a latent variable $\boldsymbol{z} \in \boldsymbol{Z}$ as $p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})$. $p(\boldsymbol{z})$ is often set to be Gaussian distribution with zero mean and unit variance. The generative distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is modeled by a deep neural network (decoder) with parameters $\boldsymbol{\theta}$, and the model parameters are trained by maximizing the marginal likelihood $p_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z})$. However, in the case that $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})$ is a neural network, the marginal likelihood becomes intractable. Therefore, the following variational lower bound is used instead:

$$\begin{aligned}\mathcal{L}_{VAE} = \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z})] \\ -D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x}) \parallel p(\boldsymbol{z})) \leq \log p_{\boldsymbol{\theta}}(\boldsymbol{x})\end{aligned} \tag{1}$$

In the above equation, $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$ is another deep neural network (encoder) that approximates the posterior distribution $p_{\boldsymbol{\theta}}(\boldsymbol{z}|\boldsymbol{x})$. The first term of Eq. (1) can be seen as the reconstruction error of a classical auto-encoder, and the second term can be seen as the regularization term. In order to make the lower bound differentiable in terms of the encoder parameters, a technique called reparameterization is used:

$$\boldsymbol{z} = \mu_{\boldsymbol{\phi}}(\boldsymbol{x}) + \sigma_{\boldsymbol{\phi}}(\boldsymbol{x}) \odot \boldsymbol{\epsilon} \quad \text{where} \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{1}) \tag{2}$$

Here, $\odot$ represents an element-wise product. In this case, the encoder becomes a deep neural network that outputs mean and variance of the posterior distribution.

3.3. Representation Learning

VAE provides an unsupervised method to learn latent representations. Given input $\boldsymbol{x}$, the latent representation can be inferred using the encoder distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})$. The objective here is to learn the encoder distribution $q_{\boldsymbol{\phi}}(\boldsymbol{z}_c, \boldsymbol{z}_s|\boldsymbol{x})$ in which latent variables are disentangled such that $\boldsymbol{z}_c$ and $\boldsymbol{z}_s$ respectively represent invariant and variant factors in a given image pair. For this, we build a model with two branches of VAEs that share parameters each other. As shown in Figure 2, the input images $\boldsymbol{x}_A, \boldsymbol{x}_B \in \boldsymbol{X}$ are fed into different VAE branches, and the latent variables $\boldsymbol{z}_c$ and $\boldsymbol{z}_s$ are extracted from each branch. The parameters of VAEs are trained using following loss function.

$$\mathcal{L} = \mathcal{L}^A_{VAE} + \mathcal{L}^B_{VAE} + \lambda_1 \mathcal{L}_{sim} + \lambda_2 \mathcal{L}_{act} \tag{3}$$

where $\mathcal{L}^A_{VAE}, \mathcal{L}^B_{VAE}$ are VAE losses for input images $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$, respectively. $\mathcal{L}_{sim}$ is a similarity loss function that constrains common features to represent invariant factors between paired images. $\mathcal{L}_{act}$ is an activation loss function that encourages activation of common features to avoid a trivial solution. λ_1 and λ_2 are the coefficients of similarity

and activation loss, respectively. The following explains the details about each terms of the loss.

Variational auto-encoder loss. The joint distribution of each VAE branch becomes

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{z}^c, \boldsymbol{z}^s) = p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^c, \boldsymbol{z}^s)p(\boldsymbol{z}^c)p(\boldsymbol{z}^s) \tag{4}$$

The generative distribution $p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^c, \boldsymbol{z}^s)$ is set as a Gaussian distribution with its mean given by the decoder output. The priors $p(\boldsymbol{z}^c)$ and $p(\boldsymbol{z}^s)$ are both Gaussian distributions with zero mean and unit variance. Then, the inference model becomes

$$q_{\boldsymbol{\phi}}(\boldsymbol{z}^c, \boldsymbol{z}^s|\boldsymbol{x}) = q_{\boldsymbol{\phi}}(\boldsymbol{z}^c|\boldsymbol{x})q_{\boldsymbol{\phi}}(\boldsymbol{z}^s|\boldsymbol{x}) \tag{5}$$

The posteriors for $\boldsymbol{z}^c$ and $\boldsymbol{z}^s$ are set to Gaussian distributions $q_{\boldsymbol{\phi}}(\boldsymbol{z}^c|\boldsymbol{x}) = \mathcal{N}(\mu^c_{\boldsymbol{\phi}}(\boldsymbol{x}), \sigma^c_{\boldsymbol{\phi}}(\boldsymbol{x}))$ and $q_{\boldsymbol{\phi}}(\boldsymbol{z}^s|\boldsymbol{x}) = \mathcal{N}(\mu^s_{\boldsymbol{\phi}}(\boldsymbol{x}), \sigma^s_{\boldsymbol{\phi}}(\boldsymbol{x}))$ whose mean and variance are given by the outputs of encoder networks. Then, the loss function of the VAE becomes

$$\begin{aligned}\mathcal{L}_{VAE} = &\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{z}^c, \boldsymbol{z}^s|\boldsymbol{x})}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x}|\boldsymbol{z}^c, \boldsymbol{z}^s)] \\ &+D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}^c|\boldsymbol{x}) \parallel p(\boldsymbol{z}^c)) + D_{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{z}^s|\boldsymbol{x}) \parallel p(\boldsymbol{z}^s))\end{aligned} \tag{6}$$

Similarity loss. In order to make common features encode invariant factors in an input image pair, we introduce the following similarity loss between the pair of common features extracted from $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$:

$$\mathcal{L}_{sim} = D(q_{\boldsymbol{\phi}}(\boldsymbol{z}^c|\boldsymbol{x}_A) \parallel q_{\boldsymbol{\phi}}(\boldsymbol{z}^c|\boldsymbol{x}_B)) \tag{7}$$

where D defines a statistical distance between latent variables. There are various types of similarity metric that can be used for D. A simple candidate is L2 or L1 distance between centroids $\boldsymbol{\mu}^c(\boldsymbol{x}_A)$, $\boldsymbol{\mu}^c(\boldsymbol{x}_B)$ of the two posteriors. However, as shown in Figure 3, when the posterior distributions have different variance along each latent dimension, the distance between centroids do not reflect the distance between distributions. Therefore, we used a kind of Mahalanobis distance as follows:

$$\mathcal{L}_{sim} = \frac{1}{M}\sum_{i=1}^{M}\frac{(\mu^c_i(\boldsymbol{x}_A) - \mu^c_i(\boldsymbol{x}_B))^2}{\sigma^c_i(\boldsymbol{x}_A)\sigma^c_i(\boldsymbol{x}_B)} \tag{8}$$

Here, μ^c_i and σ^c_i represent the i-th element of the mean and the standard deviation of the posterior distribution, and M is the dimension of the latent variable. The metric measures scaled distance along each latent dimensions according to its variances. The experimental results and a comparison between various kind of distance metrics are shown in Section 4.5.

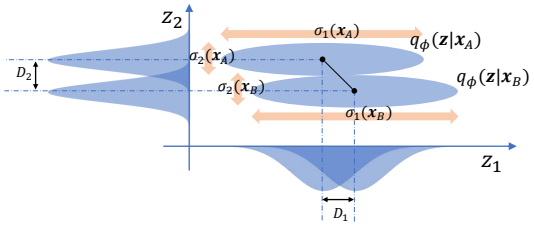

Figure 3. Illustration of posterior distributions in 2D case. Even if the distances in each dimension D_1, D_2 are the same, distributions are farther away in z_2 axis because of the smaller variance.

Activation loss. One problem with the similarity constraints is that there exists a trivial solution. The constraints can be completely satisfied by setting the mean vectors of common features to all zeros. In this case, all the information in the input are encoded by the specific features and the common features do not represent any information. To avoid this, we introduce another loss to encourage activation of the common features:

$$\mathcal{L}_{act} = \mathcal{L}_{sparsity} + \mathcal{L}_{invmax} \tag{9}$$

Activation loss consists of two parts: sparsity loss and invmax loss:

$$\mathcal{L}_{sparsity} = \sum_{i=1}^{d}(s\log m_i + (1-s)\log(1-m_i)) \tag{10}$$

$$\mathcal{L}_{invmax} = \frac{1}{B}\sum_{k=1}^{B}(\max_i |\mu^k_i|)^{-1} \tag{11}$$

Here, μ^k_i is an i-th element of the mean vector, and k represents sample index in a mini-batch. m_i is the average of $|\mu^k_i|$ in the mini-batch, *i.e.*, $m_i = \sum_{k=1}^{B}|\mu^k_i|$. $\mathcal{L}_{sparsity}$ means that through a mini-batch, every unit should be activated to s in average (s is a hyperparameter), and $\mathcal{L}_{invmax}$ means that at least one unit should be activated for each sample.

3.4. Fine-tuning

Now we have acquired the encoder for extracting common and specific features separately. As the next step, we build a event detector network $C_{\boldsymbol{\psi}}$ on the learned common features $\boldsymbol{\mu}^c_A$ and $\boldsymbol{\mu}^c_B$ extracted from each image in a pair.

$$\boldsymbol{y} = C_{\boldsymbol{\psi}}(\boldsymbol{\mu}^c) \quad \text{where} \quad \boldsymbol{\mu}^c = [\boldsymbol{\mu}^c_A, \boldsymbol{\mu}^c_B] \tag{12}$$

Here, [*,*] represents a concatenation of two vectors. We used cross-entropy loss to train the classifier on a ground truth label $\boldsymbol{t}$.

$$\mathcal{L}_{fine} = \boldsymbol{t}\log\boldsymbol{y} + (1-\boldsymbol{t})\log(1-\boldsymbol{y}) \tag{13}$$

In the fine-tuning phase, classifier parameters $\boldsymbol{\psi}$ and encoder parameters $\boldsymbol{\phi}$ are jointly trained. Because common

features represents image contents that are invariant to trivial events, a robust event detector can be effectively trained even with a small amount of labels. During the fine-tuning phase, negative samples are randomly under-sampled to have the same number of samples as positives.

4. Experiments

In this section, we verified the effectiveness of our method on four change detection datasets: Augmented MNIST, ABCD, PCD, and WDC dataset. We used Augmented MNIST for comparing our method with other methods such as Mathieu et al. [17] which are commonly evaluated on relatively simple datasets (*e.g.,* MNIST). After that we evaluated our method on in-the-wild datasets (ABCD, PCD, and WDC dataset). While all the datasets originally contained many positive samples, we limited the available positive samples to simulate a class-imbalance scenario. The numbers of positive and negative samples used in the experiments are listed in Table 1. In Section 4.4, we conducted the qualitative evaluation by visualizing learned features. Finally, in Section 4.5, we investigated several design choices of our model.

4.1. Datasets

Augmented MNIST. To validate the proposed model, we set a problem of detecting a change of digit from a pair of samples in MNIST. An input image pair is labeled as positive if the digits in the pair are different and labeled as negative if they are the same. For source images, we use three variants of MNIST [29]: MNIST with rotation (MNIST-R), background clutter (MNIST-B), and both (MNIST-R-B).

ABCD dataset. The ABCD dataset [10] is a dataset for detecting changes in buildings from a pair of aerial images taken before and after a tsunami disaster. The task is to classify whether the target buildings were washed away by the tsunami or not. Training and test patches were resized and cropped in advance such that the target buildings were in the center (*i.e.,* we used "resized" patches as used in [10]).

PCD dataset. The PCD dataset [21] is a dataset for detecting scene changes from a pair of street view panorama images. For each pair, pixel-wise change masks are provided as ground truth. In this work, we solved the change mask estimation problem by conducting patch-based classification. First, input patch pairs of size 112×112 were cropped from original images, then they were labeled as positive if the center area of size 14×14 was purely changed pixels and labeled as negative if the center area was purely unchanged pixels. In the testing phase, we cropped the patch pairs in a sliding manner, and overlaid the classifier outputs to create a heatmap of change probabilities. The heatmap was then binarized using a threshold of 0.5, which results in change mask estimation.

Table 1. Number of positive and negative samples used in each dataset. All the negative samples were used for representation learning. In fine-tuning, both the negative and positive samples were used.

	Training		Testing	
	#negatives	#positives	#negatives	#positives
Aug. MNIST	100,000	50 / 500 / 32,000	50,000	50,000
ABCD	3374	5 / 50 / 3378	847	845
PCD	56718	50	-	-
WDC	250,000	50 / 500	1934	1934

Table 2. Comparison to the anomaly detection methods on Augmented MNIST dataset. For all models, only negative samples were used during training.

	MNIST-R	MNIST-B	MNIST-R-B
AE-rec [32]	54.27	54.48	51.36
VAE-rec [1]	57.24	53.27	50.7
CAE-l2 [2]	55.14	55.74	50.29
MLVAE [5]	60.72	59.70	52.75
Mathieu et al. [17]	58.34	60.31	52.16
VAE w/o sim.	54.95	56.44	52.02
VAE w/ sim. (ours)	**71.66**	**82.55**	**62.23**

WDC dataset. In order to evaluate our method on a more large scale dataset, we prepared a new change detection dataset. This dataset is for detecting newly constructed or destructed buildings from a pair of aerial images of Washington D.C. area. The dataset contains images of multiple years (1995, 1999, 2002, 2005, 2008, 2010, 2013 and 2015). They have 16 cm resolution, and covers over 200 km^2 for each year. We automatically annotated changes in buildings by comparing the building footprints produced at different years. All the images and the footprints are acquired from open data repository hosted by the Government of District of Columbia [28]. For more detail about the dataset, please refer to the supplementary material.

4.2. Experimental setup

Baselines. For comparison, we built several baseline models for handling the class-imbalance problem. (1) *Random under/over-sampling*: a straightforward approach for class-imbalance problem is under-sampling of major class instances or over-sampling of minor class instances. For each sampling schemes, we trained a siamese CNN (the state-of-the-art architecture for image comparison tasks). (2) *Transfer learning*: transfer learning is considered to be effective when the number of available labels are limited. We transferred weights from the ImageNet pre-trained models, and fine-tuned it with under-sampling scheme. (3) *Disentangled representation learning methods*: For comparison with the state-of-the-art representation learning models, we tried [5, 17] to acquire common features. In the original formu-

Table 3. Change detection accuracies on Augmented MNIST dataset. The number of positive samples were varied from 50 to 32,000. Each result is given in terms of the mean and standard deviation obtained by 10 training runs using different training subsets.

	#Labels	Under samp.	Over samp.	MLVAE [5]	Mathieu et al. [17]	VAE w/o sim.	VAE w/ sim. (ours)
MNIST-R	50	50.63(0.31)	50.47(0.44)	57.22(1.39)	61.09(1.20)	51.55(0.43)	**79.65(4.42)**
	500	60.05(3.10)	61.84(1.37)	79.15(0.90)	77.78(0.74)	64.74(1.31)	**89.73(0.56)**
	32000	94.82(0.21)	95.49(0.15)	95.68(0.17)	95.85(0.23)	95.76(0.09)	**95.94(0.15)**
MNIST-B	50	50.69(0.61)	50.38(0.16)	59.33(2.25)	58.79(2.66)	52.67(1.44)	**82.16(0.37)**
	500	52.04(1.52)	52.27(2.80)	72.26(0.96)	75.16(1.09)	73.56(2.24)	**84.69(0.42)**
	32000	94.92(0.21)	93.28(0.15)	95.67(0.10)	94.47(0.29)	**96.25(0.06)**	96.05(0.13)
MNIST-R-B	50	50.30(0.11)	50.37(0.08)	51.61(0.67)	51.19(0.51)	50.32(0.28)	**60.58(1.60)**
	500	50.35(0.12)	50.47(0.19)	56.21(0.27)	53.10(0.93)	52.39(0.49)	**62.68(0.46)**
	32000	79.04(0.25)	75.94(0.80)	78.73(0.26)	78.55(1.17)	80.92(0.41)	**81.54(0.57)**

Table 4. Change detection accuracies on the ABCD, WDC and PCD dataset. On the column of ABCD and WDC dataset, accuracies are presented for different numbers of positive samples. On PCD dataset, the performance is reported for three evaluation metrics (Accuracy, mIoU, and IoU for positive class). The number of positive samples used for PCD dataset is 50. Each result is given in terms of the mean and standard deviation obtained by 10 training runs using different training subsets.

	ABCD			WDC		PCD		
	#Labels 5	50	All	#Labels 50	500	Acc.	mIoU	IoU
Under samp.	61.14(11.61)	64.05(17.16)	95.24(0.20)	53.12(4.56)	51.72(3.03)	73.28(3.10)	56.27(3.32)	47.95(2.20)
Over samp.	60.88(13.58)	54.05(11.78)	92.91(0.39)	52.02(3.37)	52.09(4.80)	**80.52(3.48)**	60.88(3.68)	44.92(3.49)
Transfer	77.39(7.30)	88.17(0.75)	**96.03(0.19)**	61.32(1.73)	71.07(3.04)	75.59(2.58)	58.74(2.77)	49.60(2.18)
MLVAE [5]	65.36(5.19)	86.31(1.80)	95.33(0.19)	**63.58(1.59)**	74.70(0.77)	76.88(1.22)	60.13(1.50)	50.55(1.75)
Mathieu et al. [17]	64.73(5.41)	77.66(2.11)	91.79(0.21)	60.54(2.80)	71.55(0.69)	73.71(3.55)	56.63(3.59)	48.02(2.13)
VLAE w/o sim.	67.32(6.51)	86.69(1.79)	95.18(0.14)	59.41(1.68)	74.17(1.05)	77.22(1.75)	60.49(2.27)	50.73(2.70)
VLAE w/ sim. (ours)	**78.52(5.01)**	**89.70(0.77)**	95.60(0.14)	63.25(0.86)	**75.70(0.66)**	78.20(1.96)	**61.66(2.23)**	**51.77(1.84)**

lation of [17], the discriminator requires class labels as its additional input. However, since we have no access to the class labels, we used image pair instead (*i.e.,* discriminate real-generated and real-real pairs). (4) *Anomaly detection methods*: we also tried several anomaly detection methods from [1, 2, 32]. To apply the methods, images in each pair are concatenated and regarded as a single data point. The models are trained using only negative (i.e. normal) data, and rare events are detected as outliers.

Model architecture for representation learning. We built two architectures: one for Augmented MNIST dataset and another for the rest of the datasets. For Augmented MNIST dataset, the encoder had a simple architecture of "C-P-C-P-C-H", where C, P, and H represent convolution, max-pooling, and hidden layer, respectively. Here, the hidden layer consists of four branches of convolutional layers, each of which extract mean and log-variance of specific and common features. For the rest of the dataset, in order to model complex real-world scenes, we used a hierarchical latent variable model proposed in [35], where a particular image is modeled by a combination of multiple latent variables with different levels of abstraction. Specifically, we used a model with 5 hidden layers in the experiments. Because target events are often related to high-level image contents, the common features were extracted only on the top two hidden layers. For both architectures above, the decoder part was set to be symmetric to its encoder. For the detailed architecture and the hyper-parameter settings, please refer to the supplementary materials.

Model architecture for fine-tuning. In the fine-tuning phase, we attached an event detector consisting of three fully-connected layers. The dimensions of the layers were 100-100-2 for Augmented MNIST dataset and 2048-2048-2 for the rest of the datasets. During fine-tuning, the learning rate of the pre-trained encoder part was down-weighted by a factor of 10.

4.3. Quantitative results

Table 3 shows the results for Augmented MNIST dataset. When labels were scarce, the proposed method outperformed the other models by a large margin. By comparing the models with and without similarity loss ("VAE w/o sim." and "VAE w/ sim."), we can conclude that the proposed similarity loss is essential to learn better representations for the change detection task. The performance improvement is especially remarkable with 50 labels, where the proposed model improved by approximately 20-30% compared to the baselines.

In Table 2, we also compare our method to several anomaly detection methods. In this case, we did not train the event detector. Instead, we detected change events by applying k-means clustering to the distance between common features. In the table, the proposed method outperforms the other models.

Table 4 shows the results for the ABCD, WDC and PCD datasets, respectively. Also, for these in-the-wild datasets,

(a) Source images and ground truth (b) Estimated change mask (c) Source images and ground truth (d) Estimated change mask

Figure 4. Examples of mask estimation results on the PCD dataset. From top to bottom, the figures in the columns *b* and *d* shows the result of "Under samp.", "Transfer", and "VLAE w/ sim. (ours)", respectively.

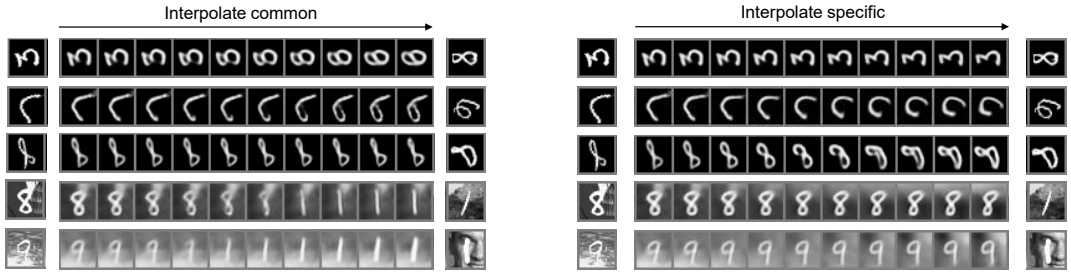

Figure 5. Results of feature interpolation analysis. **(a)** Interpolation of common features. **(b)** Interpolation of specific features.

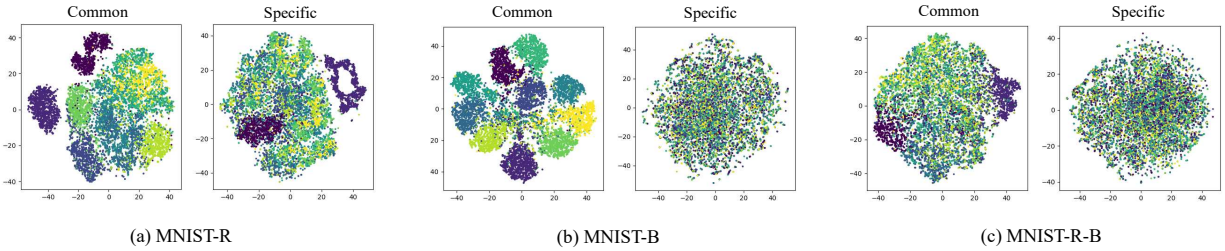

Figure 6. Results of t-SNE visualization for common and specific features. The color of each plot represents digit classes.

the proposed method outperformed the other baselines. Figure 4 compares the estimated change mask for baseline models and that of the proposed model. We see that the baseline models are sensitive to illumination changes or registration errors in roads or buildings. Clearly, they suffer from false alarms created by trivial events. On the other hand, much of the false alarms were successfully suppressed in the output of the proposed model.

4.4. Visualization of Latent Variables

In this subsection, we investigate what is encoded in common features and specific features by visualizing them. *Interpolation*: we generated a sequence of images by linearly interpolating image representations between pairs of images. To independently investigate the learned semantics of common and specific features, the features were interpolated one at a time while fixing the others. Figure 5 shows the result of visualization on Augmented MNIST dataset. When common features were interpolated between different digits, the digit classes in the generated sequences gradually changed accordingly, while the other factors (*i.e.,* rotation, styles, and background) were unchanged. On the other hand, when specific features are interpolated, rotation angles or background patterns are changed accordingly, while the digit classes remained the same. The result shows that the common features extract information about digit classes, but they are invariant to the variation observed in the same digit pairs. *2D visualization:* we visualized the learned features by t-SNE [30]. Figure 6 shows the visualization results for common and specific features. In this figure, the same color plots correspond to the same digits. We see that the common features are more informative about digit classes compared to the specific features.

We also conducted the above visualization for the rest of the datasets. However, for the real-world complicated scenes, it was difficult to achieve clear disentanglement. Specifically, we observed that the activations of the units in the common features are degenerated to a certain value.

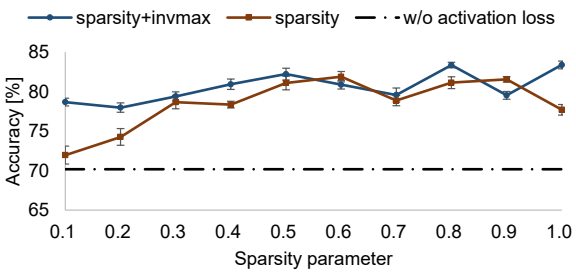

Figure 7. Analysis of the effect of activation loss. Results for different sparsity parameter values are shown with red plots (without invmax-loss) and blue plots (with invmax-loss). The error bar shows the standard deviations of the accuracies for 10 runs with different training subsets

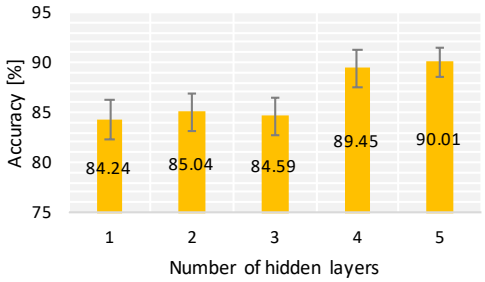

Figure 8. Sensitivity analysis of the number of hidden layers. The error bar shows the standard deviations of the accuracies for 10 runs with different training subsets.

4.5. Ablation Study

Effect of activation loss. To investigate the effect of activation loss (Eq. (9)), we conducted sensitivity analysis on the sparsity parameter and existence of $\mathcal{L}_{invmax}$. Figure 7 shows the results for MNIST-R. From these results, we can draw several conclusions. First, a sparsity parameter of approximately 0.5 seems to be suitable because the performance becomes unstable in terms of the choice of parameter value at larger than 0.5. Secondly, the use of $\mathcal{L}_{invmax}$ boosts performance. Lastly and most importantly, regardless of the parameter choices, the presence of activation loss improves the performance.

Choice of the distance function for similarity loss. Here, we investigate several choices of distance function in Eq. (7). Table 5 compares six types of distance function evaluated on the MNIST-R and ABCD datasets. We found that both Mahalanobis distance and Jeffreys divergence are suitable choices. This result supports our intuition that we should consider not only the mean vector of the latent distribution but also the shape of the distribution.

Importance of hierarchical latent variables. Here, we investigate the effect of using hierarchical latent variable models for representation learning. In this analysis, the hidden layers were eliminated one by one from the proposed model in order of the lowest layer to the highest. Figure 8 shows the results on the ABCD dataset. In the figure, the models with 4 and 5 hidden layers perform better. This result shows the importance of extracting hierarchical latent variables for rare event detection of complicated real-world scenes.

Table 5. Comparison of the choices of distance function used in similarity loss.

	MNIST-R	ABCD
L2	82.16(0.64)	90.13(1.31)
L1	79.14(0.90)	89.01(1.44)
Cosine	60.94(0.97)	89.63(1.51)
MMD	62.30(0.50)	89.69(1.35)
Jeffrey's Divergence	86.90(0.32)	89.85(0.98)
Mahalanobis	89.73(0.56)	89.70(0.77)

5. Conclusion

We proposed a novel representation learning method to overcome the class-imbalance problem in rare event detection tasks. The proposed network learns the two separated features related to image contents and other nuisance factors from only low-cost negative samples by introducing a similarity constraint between the image contents. The learned features are utilized in the subsequent fine-tuning phase, where rare event detectors are learned robustly. The effectiveness of the proposed method was verified by the quantitative evaluations on the four change detection datasets. For the evaluations, we created a large-scale change detection dataset using publicly available data repository. In addition, the qualitative analysis on Augmented MNIST showed that the model successfully learns the desired disentanglement.

The disentanglement of the proposed method is still insufficient for complicated scenes in the real world, due to degenerated solution observed in the common features. The performance of our method will be greatly improved with the clearer feature disentanglement. A possible next step to achieve this is to avoid degenerated solution by introducing adversarial training as used in [17], or maximizing mutual information between the common feature and input images [8]. Also, in the future, we intend to apply the learned invariant features to various types of event detection tasks including change mask estimation and change localization.

Acknowledgement

This paper is based on results obtained from a project commissioned by the New Energy and Industrial Technology Development Organization (NEDO). We thank Nevrez Imamoglu for the discussions and suggestions about the distance metrics. Also, we thank Motoki Kimura for his help in preparing WDC dataset.

References

[1] J. An and S. Cho. Variational autoencoder based anomaly detection using reconstruction probability. *SNU Data Mining Center, Tech. Rep.*, 2015. 5, 6

[2] C. Aytekin, X. Ni, F. Cricri, and E. Aksu. Clustering and unsupervised anomaly detection with l2 normalized deep auto-encoder representations. *arXiv preprint arXiv:1802.00187*, 2018. 5, 6

[3] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, pages 584–599. Springer, 2014. 1

[4] O. Barnich and M. Van Droogenbroeck. ViBe: A universal background subtraction algorithm for video sequences. *IEEE Transactions on Image Processing*, 20(6):1709–1724, 2011. 2

[5] D. Bouchacourt, R. Tomioka, and S. Nowozin. Multi-Level Variational Autoencoder: Learning Disentangled Representations from Grouped Observations. *arXiv preprint arXiv:1705.08841*, 2017. 2, 5, 6

[6] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan. Domain Separation Networks. *NIPS*, 2016. 2

[7] M. Chen, L. Denoyer, and T. Artieres. Multi-view data generation without view supervision. *ICLR*, 2018. 2

[8] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. *NIPS*, 2016. 2, 8

[9] Z. Chen, X. Sun, L. Wang, Y. Yu, and C. Huang. A deep visual correspondence embedding model for stereo matching costs. In *ICCV*, 2015. 1

[10] A. Fujita, K. Sakurada, and T. Imaizumi. Damage Detection from Aerial Images via Convolutional Neural Networks. *MVA*, 2017. 5

[11] A. Fujita, K. Sakurada, T. Imaizumi, R. Ito, S. Hikosaka, and R. Nakamura. Damage Detection from Aerial Umages via Convolutional Neural Networks. In *MVA*, 2017. 1

[12] L. Gueguen and G. Street. Large-Scale Damage Detection Using Satellite Imagery. *CVPR*, 2015. 2

[13] S. Khan, X. He, F. Porikli, M. Bennamoun, F. Sohel, and R. Togneri. Learning deep structured network for weakly supervised change detection. *IJCAI*, 2017. 2

[14] S. H. Khan, X. He, F. Porikli, M. Bennamoun, F. Sohel, and R. Togneri. Learning Deep Structured Network for Weakly Supervised Change Detection. In *IJCAI*, 2017. 1

[15] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. *ICLR*, 2014. 3

[16] T. Kulkarni, W. Whitney, P. Kohli, and J. Tenenbaum. Deep Convolutional Inverse Graphics Network. *NIPS*, 2015. 2

[17] M. Mathieu, J. Zhao, P. Sprechmann, A. Ramesh, and Y. LeCun. Disentangling factors of variation in deep representations using adversarial training. *NIPS*, 2016. 2, 5, 6, 8

[18] A. Odena, C. Olah, and J. Shlens. Conditional Image Synthesis With Auxiliary Classifier GANs. *arXiv preprint arXiv:1610.09585*, 2016. 2

[19] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014. 3

[20] K. Sakurada and T. Okatani. Change Detection from a Street Image Pair using CNN Features and Superpixel Segmentation. In *BMVC*, 2015. 1

[21] K. Sakurada and T. Okatani. Change Detection from a Street Image Pair using CNN Features and Superpixel Segmentation. *BMVC*, 2015. 5

[22] N. Siddharth, B. Paige, J.-W. van de Meent, A. Desmaison, N. D. Goodman, P. Kohli, F. Wood, and P. H. S. Torr. Learning Disentangled Representations with Semi-Supervised Deep Generative Models. *NIPS*, 2017. 2

[23] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2015. 1

[24] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. *CVPR*, 1999. 2

[25] S. Stent, R. Gherardi, B. Stenger, and R. Cipolla. Detecting Change for Multi-View, Long-Term Surface Inspection. In *BMVC*, 2015. 1

[26] Y. Sun, Y. Chen, X. Wang, and X. Tang. Deep learning face representation by joint identification-verification. In *NIPS*, pages 1988–1996. Curran Associates, Inc., 2014. 1

[27] L. Tran, X. Yin, and X. Liu. Disentangled Representation Learning GAN for Pose-Invariant Face Recognition. *CVPR*, 2017. 2

[28] http://opendata.dc.gov/pages/dc-from-above. DC GIS program. 5

[29] http://www.iro.umontreal.ca/~lisa/twiki/bin/view.cgi/Public/MnistVariations. Augmented MNIST. 5

[30] L. van der Maaten and G. Hinton. Visualizing Data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 7

[31] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, pages 499–515. Springer, 2016. 1

[32] Y. Xia, X. Cao, F. Wen, G. Hua, and J. Sun. Learning discriminative reconstructions for unsupervised outlier removal. In *ICCV*, 2015. 5, 6

[33] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. 1

[34] J. Zbontar and Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *JMLR*, 17(1), 2016. 1

[35] S. Zhao, J. Song, and S. Ermon. Learning Hierarchical Features from Generative Models. *ICML*, 2017. 6