# Rare Event Detection using Disentangled Representation Learning

Hamaguchi, R.; Sakurada, K; Nakamura, R.

AIST

Mar 1, 2024

Presented by Scott Sun from Duke B&B

# ChatGPT artwork

prompt: “generate a picture about the process of disentangled representation learning using a variational autoencoder”

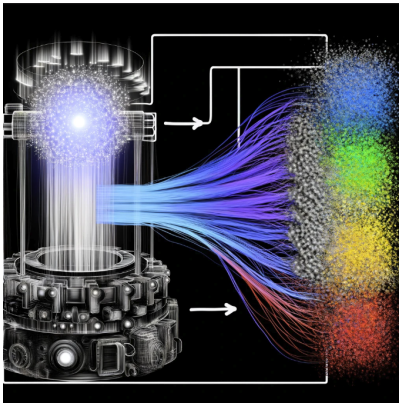

Figure: ChatGPT art

# Introduction

The authors propose a VAE-based architecture for disentangled representation learning, which aims to address rare event problem in classification task

**Goal:** to address imbalance classification with **disentangled representation** learning under the framework of **image similarity estimation**

- Why rare event is a problem? The lack of data makes it difficult for models to learn difference between classes. Without enough samples from the minority class, models tend to give false positive for trivial signals in the controls or overlook the rare events.
- In image similarity estimation, images are paired up, and the task is to detect if the image pair is homogeneous or heterogeneous (example in next slide).
- Using a generative model to recover the data generating process, we are able to obtain latent representations that capture the inherent nature of the observed, which can be extremely helpful for downstream tasks
- Via disentangled representation learning, a generative model gains interpretability and can generate controllable samples

# Introduction

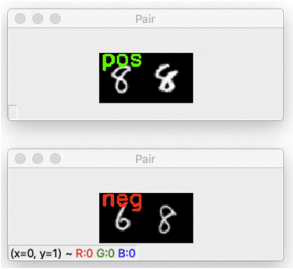


Figure: MNIST training data for image similarity estimation

# Background: VAE & Reparameterization trick

Recall that the goal of Variational Inference (VI) is to find a $q_\phi(z)$ to approximate the posterior $p_\theta(z|x)$. VAE, extending from VI, replaces $q_\phi(z)$ with $q_\phi(z|x)$, which can be seen as the distribution for encoder.

$$\mathcal{L}_{\text{VAE}} \equiv -\text{ELBO} = -\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right] + \text{KL}\big(q_\phi(z|x)\|p(z)\big)$$

To perform BP on $\phi$ (parameter of the encoder), a common trick people do is reparameterization. With normality assumption on $q_\phi(z|x)$

$$z = \mu_\phi(x) + \sigma_\phi(x) \odot \epsilon \quad \text{where } \epsilon \sim N(0, I)$$

# Method: Setup overview

In this setup, latent representations are grouped into two parts: *common* and *specific*.

- common: features that represent the content of the image (i.e. digit), which are invariant to trivial signals
- specific: features that represent the trivial signals that does not identify the content (i.e. scale, brightness, etc.)

After learning good common representations, we then use them as features of a classification task.

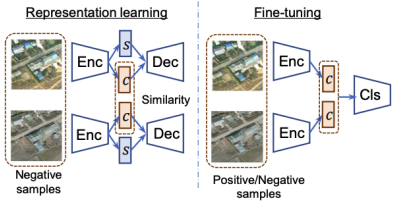


Figure: Two-stage training for disentangled representation learning

# Method: Setup overview

In this setup, latent representations are grouped into two parts: *common* and *specific*.

- common: features that represent the content of the image (i.e. digit), which are invariant to trivial signals
- specific: features that represent the trivial signals that does not identify the content (i.e. scale, brightness, etc.)

After learning good common representations, we then use them as features of a classification task.

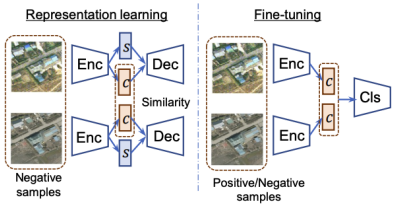


Figure: Two-stage training for disentangled representation learning

sidenote: I don't think common is a good name as it can be misleading; common here means common within the class

# Method: Representation learning (pre-training)

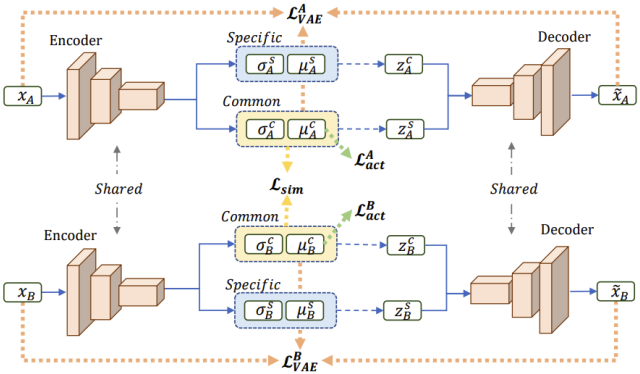


Figure: Disentangled representation learning architecture.

# Method: Representation learning (pre-training)

disentangled VAE loss

$$\mathcal{L} = -\mathbb{E}_{q_\phi(z|x)}[\log_\theta(x|z)] + KL\big(q_\phi(z^c|x)\|p(z^c)\big) + KL\big(q_\phi(z^s|x)\|p(z^s)\big)$$

where $z = (z^c, z^s)$ and assume priori independence between the two parts. Other regularization term:

- similarity/difference loss: promote similarity between $q_\phi(z^c_A|x_A)$ & $q_\phi(z^c_B|x_B)$
  Recommended: Mahalanobis distance, Jefferey divergence (aka symmetric version of KL)
  Not Recommended: L1, L2 on $\mu$'s
- activation loss: $q_\phi(z^c|x)$'s parameters are not trivial (i.e. 0)
  - sparsity: on average, every latent factor should be activated to $s$

$$\mathcal{L}_{\text{sparcity}} = \sum_{i=1}^{d}(s\log m_i + (1-s)\log(1-m_i)) \quad \text{where} \quad m_i = \sum_{k=1}^{B}|\mu_i^k|$$

  - inv-max: at least one factor should be activated for each sample

$$\mathcal{L}_{\text{invmax}} = \frac{1}{B}\sum_{k=1}^{B}(\max_i |\mu_i^k|)^{-1}$$

# Method: Fine-tuning

Instantiate a classifier and calculate $y = C_\psi(\mu_\phi^c)$ for similarity estimation, where $\mu^c = [\mu_\phi(x_A)^c, \mu_\phi(x_B)^c]$. The training task is a binary classification problem with BCE as the optim function.

In the fine-tuning phase, classifier parameter $\psi$ and encoder parameter $\phi$ are jointly trained.

Recall that in the pre-training stage, we only use the negative-negative to learn/initialize the disentangled representation. During the fine-tuning stage we use both homogeneous and heterogeneous pairs as training data.

# Experiment overview

| | 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 |

Figure: Benchmark datasets

1. Aug. MNIST: An input image pair was labeled as positive if the digits in each image were different and labeled as negative if they were same. For source images, we used three variants of MNIST: MNIST with rotation (MNIST-R), background clutter (MNIST-B), and both (MNIST-R-B)
2. ABCD: dataset for detecting changes in buildings from a pair of aerial images.
3. WDC: dataset for detecting newly constructed or destructed buildings from a pair aerial images.
4. PCD: dataset for detecting scene changes from a pair of street view panorama images; solved the change mask estimation problem by conducting patch-based classification (semantic segmentation problem)

# Result: Aug. MNIST

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. [19] | 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)** |

Figure: in the paper they call it “change detection accuracies”, to me that seems like sensitivity

# Result: Aug. MNIST

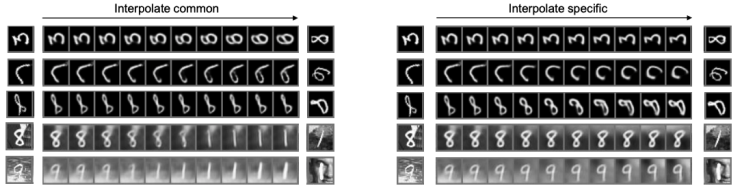


Figure 4. Results of feature interpolation analysis. **(a)** Interpolation of common features. **(b)** Interpolation of specific features.

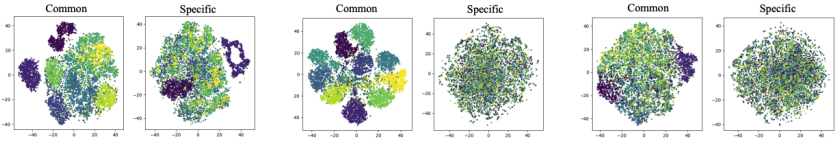


Figure 5. Results of t-SNE visualization for common and specific features. The color of each plot represents digit classes.

Figure: Visual analysis on the disentangled representation of MNIST data

# Recommendations

**Is it worth reading?** Yes.

- introduce rare event problem from a *image similarity estimation* task's perspective; in our clinical rare disease setting, we can also reformat the training data in this way (this seems like a contrastive learning framework to me)
- good extension from VAE representation learning
- the method is intuitive and more straightforward to understand compared to some other advanced $\beta$-VAE approaches

**Is it worth implementing?** Yes.

- the result about the interpolation in latent space looks really promising to me
- although there is no public github repo, the architecture and loss functions are simple (compared to MGVAE...)