Implementation of Deep Orthogonal Hypersphere Compression for Anomaly Detection

Refaldi Intri Dwi Putra (48-237411)

Department of Information and Communication Engineering
The University of Tokyo

Abstract

Anomaly detection is important problem which has many applications in our daily lives. A technique to perform this is by using a deep learning model. The work from [Zhang et al., 2024] explored the discrepancy of the deep learning-based method for anomaly detection. Later, they proposed two novel methods named DOHSC and DO2HSC that can perform better than the existing methods. In this report, we summarize their work and implement their approach by ourselves.

1 About the Picked Paper

The paper is titled "Deep Orthogonal Hypersphere Compression for Anomaly Detection" by [Zhang et al., 2024] selected as a *Spotlight*[1] paper in The Twelfth International Conference on Learning Representations (ICLR) 2024, a top international conference in the field of artificial intelligence (AI). They proposed novel deep learning methods for anomaly detection, applicable for various data set structures such as image, tabular, and graph.

2 Reason to Pick the Paper

I pick this paper because of its high-quality work, the study begins with identifying problem in the existing method with clear manner and illustrative approach. Then, they analyzed the discrepancy of the existing methods from the theoretical analysis and empirical observations. Later, they proposed two novel methods based on the analysis which can be applicable for various data set structures, amplifying its impactfulness. Moreover, since its work contributes to the anomaly detection, it has many applications in various real-world problems. This is also the first time I read a paper about deep learning-based anomaly detection, so I want to challenge myself to understand it and trying to implement it by myself.

3 The Problem Setting

3.1 Basic Notation

In this report we use some basic notation and definition. Writing vector and matrix in a boldface like $\boldsymbol{v}, \boldsymbol{A}$ respectively. The vector norm (L2) denotes as $\|\boldsymbol{v}\|$ for any $\boldsymbol{v}$ and the Frobenius norm denotes by $\|\boldsymbol{A}\|_F = \sum_{i,j} A_{ij}$ for any $\boldsymbol{A}$. We denote $\boldsymbol{I}$ as the identity matrix. We denote $\mathbb{E}[\cdot]$ as the mean value. We refer **normal** data as a data that belongs to the decision region and **anomaly** data as a data that is outside the region. The separation between the normal and anomaly data is defined by a decision boundary (threshold) r and s denotes its anomalous score. The interest data is categorized based on one vs all other classes, so this problem is known as the **one-class anomaly detection**.

3.2 Deep Learning-based Anomaly Detection Problem

Consider a (feature) data matrix denotes as $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ with n instances and d features. By using an autoencoder[2], we can use the latent representation $\boldsymbol{Z} = f^{\text{enc}}_{\mathcal{W}}(\boldsymbol{X}) \in \mathbb{R}^{n \times k}$ to *initialize* a decision region's center $\boldsymbol{c} \in \mathbb{R}^k$. For example by calculating its mean such that $\boldsymbol{c} = \frac{1}{n}\sum_{i=1}^{n} f^{\text{enc}}_{\boldsymbol{W}}(\boldsymbol{x}_i)$, where $\boldsymbol{x}_i$ denotes the transpose of the i-th row of $\boldsymbol{X}$ and $f^{\text{enc}}_{\mathcal{W}}(\cdot)$ is an L-layer representation learning module with parameters $\mathcal{W} = \{\boldsymbol{W}_l, \boldsymbol{b}_l\}_{l=1}^{L}$. By using, this we want to detect the anomaly points by optimizing the decision boundary based on $\boldsymbol{Z}$.
Suppose that we have a distance from a learned representation $\boldsymbol{z}_i$ to the center $\boldsymbol{c}$ as $d_i = \|\boldsymbol{z}_i - \boldsymbol{c}\|$, where we can stack it as a vector $\boldsymbol{D} = \{d_i\}_{i=1}^{n}$. The decision boundary can be calculated by this objective function:

$$\hat{r} = \arg\min_{r} \mathcal{P}(\boldsymbol{D} \leq r) \geq \nu, \tag{1}$$

[1] https://openreview.net/forum?id=cJs4oE4m9Q
[2] https://en.wikipedia.org/wiki/Autoencoder

where $\mathcal{P}(\cdot)$ denotes the probability distribution and ν is the confidence level. From here we can detect an i-th anomaly data by calculating its anomalous score as

$$s_i = d_i^2 - \hat{r}^2, \tag{2}$$

where $s_i > 0$ denotes an anomaly data and vice versa for the normal data.

3.3 The Gap in the Existing Methods

3.3.1 The Discrepancy in the Existing Objective Function

The existing work called the Hypersphere Contraction optimization problem is formulated as follows:

$$\min_{\mathcal{W}} \frac{1}{n} \sum_{i=1}^{n} \|f_{\mathcal{W}}^{\text{enc}}(\boldsymbol{x}_i) - \boldsymbol{c}\|^2 + \frac{\lambda}{2} \sum_{l=1}^{L} \|\boldsymbol{W}_l\|_F^2, \tag{3}$$

where the first term is assumed to restric the gap between the representations as a hypersphere (Cartesian)[3], while the second term refers to the regularization in order to reduce the over-fitting.
It turns out that the assumption of the hypersphere represented by the objective function in equation 3 is not consistent with the learned decision boundary (ellipsoidical). This discrepancy leads to the suboptimal performance of the existing methods. The authors argued that there are two reasons for this: 1) the learned features have different variances, and 2) the learned features are correlated. These two reasons cannot be solved by optimizing the equation 3.

3.3.2 Soap-bubble phenomenon in sparsed high dimensional data

They pointed out that when the dimension of the data is high and sparse, the normal data is driving away from the center and leaves an inner hypersphere regions where there is no normal data. This phenomenon is called soap-bubble problem because we can imagine it looks like a bubble with two co-center sphere with different radius. Under the same objective function as equation 3, the anomaly data will be counted as normal data since this empty region is inside the boundary decision.

3.4 The Proposed Method

3.4.1 DOHSC and DO2HSC

Accordingly, they proposed two methods named Deep Orthogonal Hypersphere Compression (DOHSC) and Deep Orthogonal Bi-hypersphere Compression (DO2HSC) to tackle the problems. The following figures in figure 1 are screenshot from the paper, providing the algorithms of the two methods.
More specifically, the DOHSC method appends an orthogonal projection layer after the encoder to project the representation to be more aligned with the hypersphere following the problem described in 3.3.1. This means that each projected latent representation will be orthogonal to each other $\tilde{\boldsymbol{z}}_i \perp \tilde{\boldsymbol{z}}_j, i \neq j$. The projection layer is described as follows:

$$\tilde{\boldsymbol{Z}} = \text{Proj}_{\Theta}(\boldsymbol{Z}) = \boldsymbol{Z}\boldsymbol{W}^*, \quad \text{subject to} \quad \tilde{\boldsymbol{Z}}^\top \tilde{\boldsymbol{Z}} = \boldsymbol{I}_{k'} \tag{4}$$

where $\Theta := \{\boldsymbol{W}^* \in \mathbb{R}^{k \times k'}\}$ is the set of projection parameters, and k' is the projected dimension. In order to achieve this, they propose to use the singular value decomposition for efficiency such that:

$$\boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{V}^\top = \boldsymbol{Z}, \quad \boldsymbol{W} := \boldsymbol{V}_{k'}\boldsymbol{\Lambda}_{k'}^{-1}. \tag{5}$$

Assume that there are b samples in one batch, $\boldsymbol{\Lambda} = \text{diag}(\rho_1, \rho_2, ..., \rho_b)$ and $\boldsymbol{V}$ are the diagonal matrix with singular values and right-singular matrix of $\boldsymbol{Z}$, respectively. $\boldsymbol{V}_{k'} := [\boldsymbol{v}_1, ..., \boldsymbol{v}_{k'}]$ denotes *the first* k' right singular vectors, and $\boldsymbol{\Lambda}_{k'} := \text{diag}(\rho_1, ..., \rho_{k'})$. It is optimized by updating the original matrix $\boldsymbol{W}$ into a new matrix $\boldsymbol{W}^*$ during the training process. By doing so, in the DOSHC algorithm we have a projected $\tilde{\boldsymbol{Z}}$ and the boundary decision is calculated based on that.
Meanwhile, DO2HSC tackles the problem when the data set is sparsed and high dimensional as described in 3.3.2. Following that observation and the theoretical analysis, they proposed to build the decision boundary by using two hyperspheres with different radius. The first hypersphere is the inner hypersphere with decision boundary r_{min} and the second hypersphere is the outer hypersphere with decision boundary r_{max}. These two decision boundaries are initialized by using DOSHC method with different confidence level, one is $1 - \nu$ and another is ν.

[3] https://en.wikipedia.org/wiki/N-sphere

Algorithm 1 Deep Orthogonal Hypersphere Contraction (DOHSC)

Input: The input data $\mathbf{X} \in \mathbb{R}^{n \times d}$, dimensions of the latent representation k and orthogonal projection layer k', a trade-off parameter λ and the coefficient of regularization term μ, pretraining epoch $\mathcal{T}$, learning rate η.
Output: The anomaly detection scores $\mathbf{s}$.

Initialize the auto-encoder network parameters $\mathcal{W} = \{\mathbf{W}_l, \mathbf{b}_l\}_{l=1}^{L}$ and the orthogonal projection layer parameter Θ;
for $t \to \mathcal{T}$ **do**
  for each batch **do**
    Obtain the latent representation $\mathbf{Z} = f_{\mathcal{W}}^{\text{enc}}(\mathbf{X})$; ▷ Pretraining Stage
    Update the orthogonal parameter Θ of orthogonal projection layer by Eq. (3);
    Project the latent representation via Eq. (2);
    Calculate reconstruction loss via $\frac{1}{n}\sum_{i=1}^{n} \|f_{\mathcal{W}}^{\text{dec}}(\text{Proj}_{\Theta}(f_{\mathcal{W}}^{\text{enc}}(\mathbf{x}_i))) - \mathbf{x}_i\|^2$;
    Back-propagate the network, update $\mathcal{W}$ and Θ, respectively;
  end for
end for
Initialize the center of hypersphere by $\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n} f_{\mathcal{W}}^{\text{enc}}(\mathbf{x}_i)$;
repeat
  for each batch **do**
    Calculate anomaly detection loss via Optimization (4); ▷ Training Stage
    Repeat steps 4-6;
    Back-propagate the encoder network and update $\{\mathcal{W}\}_{l=1}^{L}$ and Θ, respectively;
  end for
until convergence
Compute decision boundary r by Eq. (5);
Calculate the anomaly detection scores $\mathbf{s}$ through Eq. (6);
return The anomaly detection scores $\mathbf{s}$.

Algorithm 2 Deep Orthogonal Bi-Hypersphere Compression (DO2HSC)

Input: The input data $\mathbf{X} \in \mathbb{R}^{n \times d}$, dimensions of the latent representation k and orthogonal projection layer k', a trade-off parameter λ and the coefficient of regularization term μ, pretraining epoch $\mathcal{T}_1$, iterations of initializing decision boundaries $\mathcal{T}_2$, learning rate η.
Output: The anomaly detection scores $\mathbf{s}$.

Initialize the auto-encoder network parameters $\mathcal{W} = \{\mathbf{W}_l, \mathbf{b}_l\}_{l=1}^{L}$ and the orthogonal projection layer parameter Θ;
for $t \to \mathcal{T}_1$ **do**
  for each batch **do**
    Repeat steps 4-8 of DOHSC; ▷ Pretraining Stage
  end for
end for
Update the orthogonal parameter Θ of orthogonal projection layer by Eq. (3);
Obtain the global orthogonal latent representation by Eq. (2);
Initialize the center of hypersphere by $\mathbf{c} = \frac{1}{n}\sum_{i=1}^{n} f_{\mathcal{W}}^{\text{enc}}(\mathbf{x}_i)$;
for $t \to \mathcal{T}_2$ **do**
  Repeat steps 13-17 of DOHSC; ▷ Pretraining Stage
end for
Compute decision boundary r of DOHSC by Eq. (5);
Initialize decision boundaries $r_{\max}$ and $r_{\min}$ via Eq. (7);
repeat
  for each batch **do**
    Obtain the latent representation $\mathbf{Z} = f_{\mathcal{W}}^{\text{enc}}(\mathbf{X})$; ▷ Training Stage
    Update the orthogonal parameter Θ of orthogonal projection layer by Eq. (3);
    Project the latent representation via Eq. (2);
    Calculate the improved total loss via Optimization (8);
    Back-propagate the network, update $\{\mathcal{W}\}_{l=1}^{L}$ and Θ, respectively;
  end for
until convergence
Calculate the anomaly detection scores $\mathbf{s}$ through Eq. (9);
return The anomaly detection scores $\mathbf{s}$.

Figure 1: The algorithm of DOHSC and DO2HSC, screenshot from the paper. The equation numbers are not the same as in this report, reader is advised to take a look to the original paper for more clarity. Overall the algorithm consists of initialization of the center $\boldsymbol{c}$ in pre-training stage(s), follows by fine tuning method. DOHSC has one pre-training stage while DO2HSC has two pre-training stages.

To show the difference between the existing method, and their proposed methods in terms of the formalism we show it in the Table 1 and provide the illustration depicted in the original paper in figure 2.

	Existing Method	DOSHC	DO2HSC
Learned representation	$\boldsymbol{z}_i, \boldsymbol{c} = \mathbb{E}[\boldsymbol{z}_i]$	$\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{c}} = \mathbb{E}[\tilde{\boldsymbol{z}}_i]$	$\tilde{\boldsymbol{z}}_i, \tilde{\boldsymbol{c}} = \mathbb{E}[\tilde{\boldsymbol{z}}_i]$
Distance (d_i)	$\|\boldsymbol{z}_i - \boldsymbol{c}\|$	$\|\tilde{\boldsymbol{z}}_i - \tilde{\boldsymbol{c}}\|$	$\|\tilde{\boldsymbol{z}}_i - \tilde{\boldsymbol{c}}\|$
Decision Boundary (ν)	$\hat{r}$	$\hat{r}$	$r_{\min}, r_{\max}$
Objective Function	$\min \frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{z}_i - \boldsymbol{c}\|^2 + \frac{\lambda}{2}\sum \|\boldsymbol{W}\|_F^2$	$\min \frac{1}{b}\sum_{i=1}^{b} \|\tilde{\boldsymbol{z}}_i - \boldsymbol{c}\|^2 + \frac{\lambda}{2}\sum \|\boldsymbol{W}\|_F^2$	$\min \frac{1}{b}\sum_{i=1}^{b} (\max\{d_i, r_{\max}\} - \min\{d_i, r_{\min}\}) + \frac{\lambda}{2}\sum \|\boldsymbol{W}\|_F^2$
Anomalous score	$s_i = d_i^2 - \hat{r}^2$	$s_i = d_i^2 - \hat{r}^2$	$s_i = (d_i - r_{\max}) \cdot (d_i - r_{\min})$

Table 1: The difference between the existing method and the proposed methods in terms of the formalism.

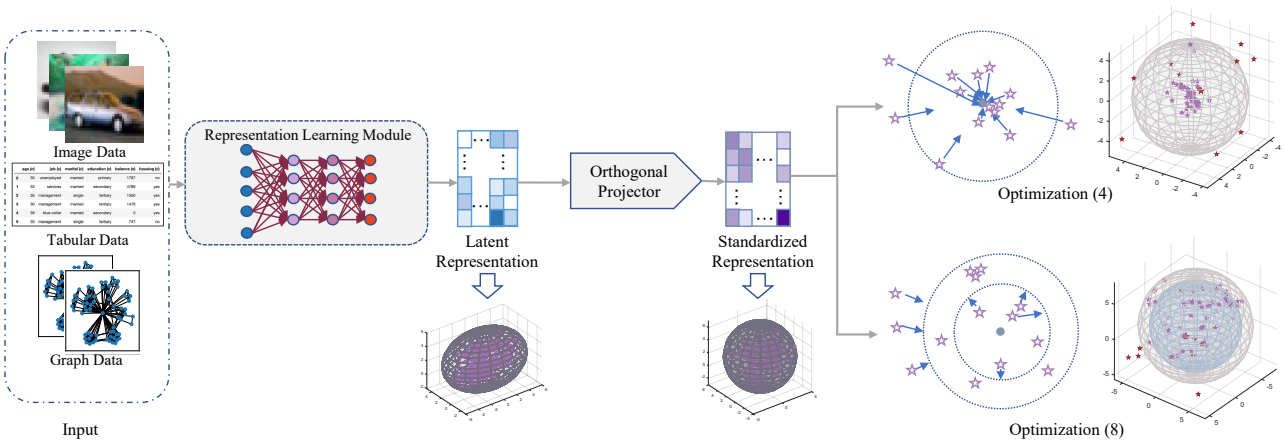

Figure 2: The illustration of the DOHSC in the right upper subfigure and DO2HSC in the right lower subfigure. Both share the same architecture up to projected representation. Notice that in the DO2HSC there are two spheres and two radii but DOHSC is just one. This figure is taken from the original paper so the equation numbers are not matching to this report's. Instead, one can refer to the table 1 or the paper.

3.5 Extension to Graph Data

The authors also extended their study to the graph data such that a set of graphs $\mathbb{G} = \{G_1, ..., G_N\}$ consists of N samples, the model will learn a k- dimensional and provide the decision boundary (soft). They maximize the mutual information $I(\cdot, \cdot)$ between the local ($\boldsymbol{h}$) and global representations ($\boldsymbol{H}$) in a batch. It uses the positive and negative samples to learn the representation. The procedure is the same with initialization of the center ($\tilde{\boldsymbol{c}}$) from the projected representation $\tilde{\boldsymbol{H}}$. The form of the objective function is similar to the non-graph data sets but there is a trade-off term with parameters λ due to the use of $I(\cdot, \cdot)$. We advise the reader to refer to the original paper for more details on their objective functions.

4 My Implementation

4.1 Workflow and questions

Workflow. Below is my interpretation of the algorithm in figure 1 that I implement by myself. First, we need to initialize a center $\boldsymbol{c}$ and saved the checkpoint of the autoencoder. Later, the encoder-part of the trained autoencoder can be loaded to the encoder that we use for either DOHSC or DO2HSC. In case, we perform DO2HSC, we also perform DOHSC so the training process is done in two stages in DOHSC and three stages in DO2HSC. The visualization of my workflow for the algorihms can be found in figure 3.

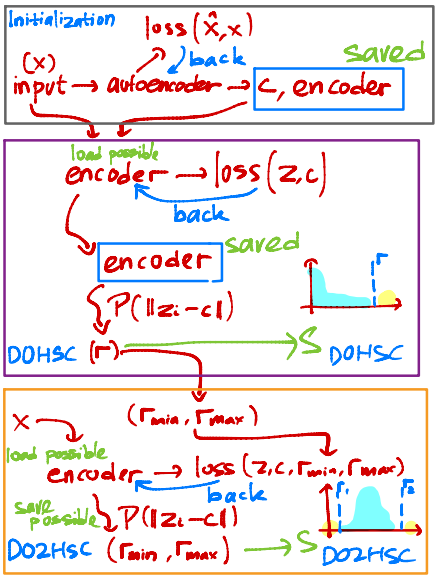

Figure 3: The workflow illustration for the implementation of DOHSC and DO2HSC. The training process is done in two stages in DOHSC and three stages in DO2HSC.

Questions. Meanwhile, there are some questions that I want to seek for the implementation. 1) Is their claim true and can I implement it by myself?, and 2) what is the effect of auto-encoder training and its obtained $\mathbf{z}$.

4.2 Basic Setting

4.2.1 Data

We will implement their algorithm in the data set named CIFAR-10 which consists of low-quality RGB images with 10 classes. Some representations of the figures and their labels can be seen in figure 4. Since this data set was also used in their report, we can compare our implementation with them.

4.2.2 Model

Since our data set is images data, we consider to use the convolutional neural network (ConvNet) as our backbone. There is also a consideration of using transformer model but it has been known that the vision transformer model is difficult to train for CIFAR-10 due to lacking inductive bias (invariant). Therefore, we justify to choose ConvNet due to its compatibility for CIFAR-10 and its simpler architecture.
Since, our models consist of autoencoder in the initial training and then we use the encoder part only for the anomaly detection, we construct our model with 3 layers convnet in the encoder where the image is downscaled

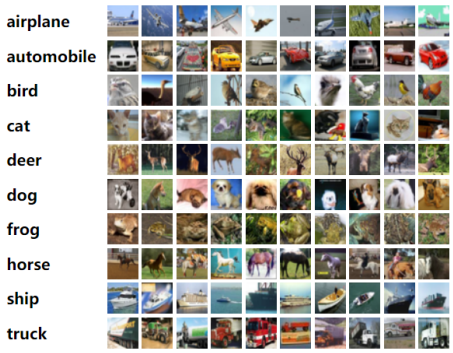

Figure 4: CIFAR-10 data set images and their 10-class labels. Source: https://www.cs.toronto.edu/~kriz/cifar.html

1/8 times, follow by the linear model to get the compressed representations ($\boldsymbol{Z}$) and then the decoder part where we use the deconvolution to upscale the images into its original size.
We use the save checkpoint for the initialization, the center $(\boldsymbol{c}, \tilde{\boldsymbol{c}})$, and the radius $r_{\max}, r_{\min}$ and load it based on the purpose by matching its model's module names. In order to compare the efficacy of their proposed model, we also train the non-projected layer version of the model. We also curious on the effect of the skip connection during the center initialization and its effect to the whole decision. Hence, in this implementation there are three variants of model that we train. The details are as follow 1) Type-a (no ortho layer), 2) Type-b := Type-a + ortho layer, and 3) Type-c := Type-b + skip connection. We note here, the paper compared type-b with type-a which was the existing work. The illustration of the models I implemented can be seen in figure 5.

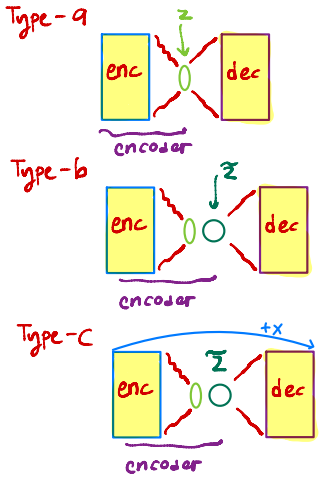

Figure 5: The illustration of the models that I implemented. Each model consists of 3 layers of ConvNet in the encoder, linear layer for the compressed representation, and 4 layers of deconvolution in the decoder. The ortho layer is added after the linear layer in the encoder for type-b. The skip connection is added for the type-c.

4.2.3 Training

To train the model, we follow the loss function described in the paper and get the decision based on the quantile of the distribution distance to the center. We set the learning rate for the autoencoder to be 10^{-3} with weight decay 10^{-5}, and for the encoder we set the learning rate as 10^{-2} with weight decay 10^{-4}. For the number of epochs, we use 20 epochs for DOHSC, for DO2HSC we use 300 epochs after 20 epochs of DOHSC. Both use 10 epochs for the autoencoder. For the optimizer, we use Adam optimization and use the learning rate scheduler.

4.2.4 Evaluation

For the evaluation, we use the same evaluation metrics by using the ROC-AUC scores which describe how likely the model can distinguish the normal and anomaly data. We use the scikit-learn library to calculate the ROC-AUC scores. We show the results in the table 2.

Table 2: The ROC-AUC scores of the implemented models on CIFAR-10 data set. The bold number denotes the highest score in each class.

	Type-a Baseline	Type-b DOHSC	Type-b DO2HSC	Type-c DOHSC	Type-c DO2HSC
airplane	0.625332	**0.807429**	0.550922	0.778600	0.494966
automobile	0.498241	0.778136	0.413177	**0.798894**	0.566402
bird	0.506461	**0.731340**	0.501813	0.705305	0.453072
cat	0.522005	0.767414	0.509876	**0.825998**	0.518482
deer	0.443229	0.677868	0.595395	**0.710667**	0.462096
dog	0.503729	0.536661	0.473181	**0.653098**	0.475568
frog	0.465518	0.619313	0.489470	**0.622623**	0.523069
horse	0.491382	0.619679	0.509680	**0.634519**	0.444832
ship	0.564833	0.703277	0.467833	**0.766374**	0.449965
truck	0.520972	0.772023	0.538794	**0.773195**	0.391118

5 Discussion

From our evaluation, we found that it is indeed their approach is better than the existing method by jus simply projecting the embedding to the hypersphere space. However, my implementation for DO2HSC did not give a better performance than the DOHSC. Probably, this is due to the learning setting such as ν to determine the bi-radii in my implementation is different than theirs.
Interestingly, the use of skip connection in the auto encoder can make the performance better in DOHSC method but not in DO2HSC. This might imply that the initialized center is actually important and the rate of the noise in the embedding also important as skip connection can denoise the representation. But, it is still not clear why the DO2HSC performance is getting worse when we use the skip connection.
Another comment is about the released implementation of their code for orthogonal projection seems to differ than what their paper described. When we used what was described by the paper $(1/\rho)$, the learning has become more difficult. My guess is that they want to penalizing this term to be as high as possible, but it needs some further clarification.
As for the future studies, the limitation of the paper is that, the approach does not consider multiple classes in the training data and then determine the outliers based on that. Probably in such settings, we need to build different centers $\mathbf{c_1}$ and $\mathbf{c_2}$. Then, the naive approach should be establishing the global center $\mathbf{c_g}$ and create a hyper-sphere from it, but this will include many abnormal data to the normal class. Henceforth, we should relax such assumption and probably rely more into the ellipsoidal to accommodate the multiple radii. But, such problem will create more complexity.
Another thought is in the stochastic gradient descent approach, gradient descent is based on vector gradient $\frac{\partial \vec{L}}{\partial \theta}$, which will update the next parameter as $\theta_{i+1} = \theta_i - \eta \frac{\partial \vec{L}}{\partial \theta}$ for a learning rate η. But notice that our loss function lives in the hypersphere space that may not optimal for such spherical surface assumption. In such consideration, it maybe worth it to try non-euclidean gradient descent for spherical embedding [4] in the future.
As the applications of anomaly detection are many, the one-class anomaly detection itself is already useful to many situations. For example, to identify whether a person is fraud or not we can rely on the anomaly detection considering fraud's activity will be much differ than the normal users. Hence, this method and technology can be the first weapon to separate normal users to the abnormal users. Later, investigation can be done further based on it.

6 Summary

In this report, I reviewed the aforementioned paper to tackle the anomaly detection problem by using a deep learning approach. The study showed that the existing methods possess a discrepancy between the learned decision boundary and the hypersphere assumption that leads to the suboptimal performance. They also showed that when the data is sparsed and high dimensional, the normal data is drifting away from the original causing a problem called soap-bubble. Then, they proposed two novel methods named DOHSC and DO2HSC that can overcame the identified problems respectively. I implemented their algorithms by myself and found my implemented algorithm works well in CIFAR-10 data set. I also explored the skip-connection variant model to observe the effect of the initialization center and the noise in the embedding. I also discussed the possible future studies based on my understanding.

[4] https://proceedings.neurips.cc/paper/2020/file/d9812f756d0df06c7381945d2e2c7d4b-Paper.pdf

References

[Zhang et al., 2024] Zhang, Y., Sun, Y., Cai, J., and Fan, J. (2024). Deep orthogonal hypersphere compression for anomaly detection. In *The Twelfth International Conference on Learning Representations*.