Self-supervised Detransformation Autoencoder for Representation Learning in Open Set Recognition

Jingyun Jia[0000−0003−0865−049X] and Philip K. Chan[0000−0002−3878−4205]

Florida Institute of Technology, Melbourne FL 32901, USA
jiaj2018@my.fit.edu pkc@fit.edu

Abstract. The objective of Open set recognition (OSR) is to learn a classifier that can reject the unknown samples while classifying the known classes accurately. In this paper, we propose a self-supervision method, Detransformation Autoencoder (DTAE), for the OSR problem. This proposed method engages in learning representations that are invariant to the transformations of the input data. Experiments on several standard image datasets indicate that the pre-training process significantly improves the model performance in the OSR tasks. Moreover, our analysis indicates that DTAE can yield representations that contain some class information even without class labels.

Keywords: Open set recognition · self-supervised learning · representation learning.

1 Introduction

Deep learning has shown great success in recognition and classification tasks in recent years. However, there is still a wide range of challenges when applying deep learning to the real world. Most deep neural networks and other machine learning models are trained under a static close-set scenario. However, the real world is more of an open-set scenario, in which it is difficult to collect samples that exhaust all classes. The problem of rejecting the unknown samples meanwhile accurately classifying the known classes is referred as Open Set Recognition (OSR) [1] or Open Category Learning [3]. The OSR problem defines a more realistic scenario and has drawn significant attention in applications such as face recognition [10], malware classification [6] and medical diagnoses [14].

In this paper, we bring self-supervised pre-training to the OSR problem and fine-tune the pre-trained model with different types of loss functions: classification loss and representation loss. Particularly, we propose Detransformation Autoencoder (DTAE) for self-supervision. DTAE consists of three components: an encoder, a decoder, and an input transformation module. The encoder encodes all transformed images to representations, and the decoder reconstructs the representations back to the original images before transformations. Compared to the traditional autoencoder, DTAE learns the representations that describe

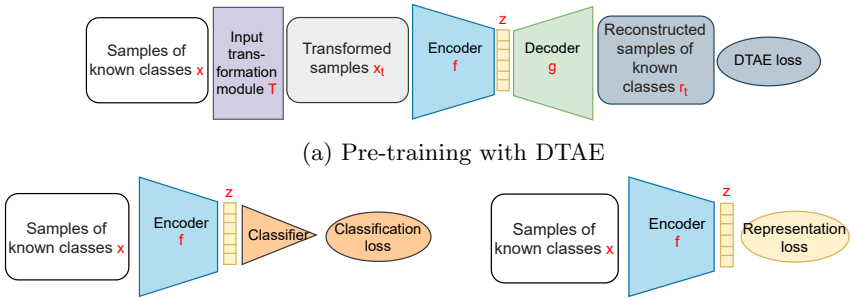
the pixels and are invariant to the transformations. Our contribution in this paper is threefold: First, we introduce DTAE as a self-supervised pre-training method for the OSR tasks. Second, our experiment results show that DTAE significantly improves the model performances for different down-streaming loss functions on several image datasets. Third, our analysis indicates that DTAE is able to capture some cluster information for both known and unknown samples even without class labels.

We organize the paper as follows. In section 2, we give an overview of related work. Section 3 presents the self-supervision method, DTAE, in pre-training for the OSR tasks. Section 4 shows that the pre-training process can significantly improve the model performance in several standard image datasets. Meanwhile, the models pre-trained with DTAE achieve the best performance in detecting the unknown class and classifying the known classes.

2 Related Work

We can divide neural network based OSR techniques into three categories based on the training set compositions. The first category borrows additional data as the unknown samples in the training set. To better discriminate known class and unknown class, Shu et al. [16] and Saito et al. [12] introduce unlabeled data during the training phase as the unknown class. The second category generates additional data as the unknown class, Ge et al. [4] introduce a conditional GAN to generate unknown samples followed by an OpenMax classifier. The third category does not use additional data. Bendale and Boult [1] propose OpenMax for the OSR problems. OpenMax adapts Meta-Recognition concepts to the activation patterns in the representation layer of the network and then estimates the probability of an input being from an unknown class. Hassen and Chan [6] propose ii loss for the OSR problem. It first finds the representations for the known classes during training and then recognizes an instance as unknown if it does not belong to any known classes. Jia and Chan [7] propose MMF as a loss extension to further separate the known and unknown representations for the OSR problem. CROSR in [18] trains networks for joint classification and reconstruction of the known classes to combine the learned representation and decision in the OSR task. Perera et al. [11] adopt a self-supervision framework to force the network to learn more informative features when separating the unknown class. Specifically, they used the output of the autoencoder as auxiliary features for the OSR task.

Self-supervision in representation learning generally uses a pretext task that is different from the primary task. The pretext task includes reconstructing the input based on a smaller number of features (autoencoders), classifying transformations such as rotations [5], intra-sample vs inter-sample transformations in contrastive loss [2], redundancy reduction in learned features from transformations [19]. In an Autoencoder, the "labels" are the input samples themselves, and the network learns the representations of the inputs by minimizing the dissimilarity between the input and output. Denoising autoencoder (DAE) corrupts



(a) Pre-training with DTAE

(b) Fine-tuning with classification loss (c) Fine-tuning with representation loss

Fig. 1: The two stage training process of the OSR problem.

the input samples first, then the network is trained to denoise corrupted versions of their inputs to reconstruct back to the their original forms. RotNet in [5] uses image rotations as the pretext task, and the model is trained on the classification task of recognizing the rotation classes. SimCLR in [2] introduces contrastive loss to improve the quality of learned representations. Given transformed samples, the contrastive loss reduces the intra-sample distances meanwhile increase the inter-sample distances. Barlow twins in [19] feeds distorted versions of a sample to two identical networks, and proposes an objective function that makes the cross-correlation matrix between their outputs as close to the identity matrix to minimize the redundancy between components of the representations.

Our proposed method uses a self-supervision approach to learning the features of the known classes without using additional unknown samples. Unlike DAE, our proposed method includes the original input samples in the training process. Moreover, we augment the input samples with different rotation transformations. The network learns the representations that are invariant to the transformations of the input data by decoding all the transformed images back to the original ones before transformations.

3 Approach

We propose a two-step training process (pre-training step and fine-tuning step) for the OSR problems, thus better separating different classes in the feature space. As illustrated in Figure 1, the training process includes two steps: 1) pre-training step uses detransformation autoencoder (DTAE) to learn features for all the input data; 2) fine-tuning step uses representation loss functions or classification loss functions to learn discriminative features for different classes.

3.1 Pre-training step

The objective of the self-supervised pre-training process is to learn some meaningful representations via pretext tasks without semantic annotations. The de-

sirable features should be invariant under input transformations, meanwhile, contain the essential information that can reconstruct the original input. We propose a detransformation autoencoder (DTAE) to learn representations by reconstructing ("detransforming") the original input from the transformed input. DTAE employs a transformation module and an encoder-decoder structure. While the encoder extracts the representations, the decoder reconstructs the original input from the learned representations.

The motivation of DTAE is to learn better representations for the OSR problem via encouraging intra-sample similarity and intra-class similarity of the learned representations. For example, if we have samples from "cat" class and "dog" class, given sample "cat1" and its transformation "cat1a", we can learn their representations z_{c1} and z_{c1a}. Similarly, we can learn the representations of "cat2" and "dog1" as z_{c2} and z_{d1}. The intra-sample similarity describes the similarity between the representations of the original input and its transformations, as z_{c1} and z_{c1a} in our example, and we denote this similarity as $sim(z_{c1}, z_{c1a})$. As the decoder in DTAE reconstructs the *same* original samples from the learned representations of *both* original and transformed samples, the learned representations are of high intra-sample similarity. Thus the learned representations are invariant to the transformations and contain important features of the samples. The intra-class similarity describes the similarity among the learned representations of the same class, as z_{c1} and z_{c2} in our example, and we denote this similarity as $sim(z_{c1}, z_{c2})$. The encoder-decoder structure in DTAE is a generative model that embeds crucial features in lower dimensions. Compared to a discriminate model, the representations learned by a generative model contain more comprehensive information to reconstruct the inputs. Thus, for a generative model, the learned representations of samples from the same class should be more similar than those of different classes. Overall, the desired representation space should satisfy $sim(z_{c1}, z_{c1a}) > sim(z_{c1}, z_{c2}) > sim(z_{c1}, z_{d1})$.

As shown in Figure 1a, in the pre-training stage with DTAE, the input transformation module T transforms any given data example x to several correlated views of the same example, denoted as $x_t = T(x)$. The network-based encoder $f(\cdot)$ extracts representation vectors from transformed data examples. Furthermore, decoder $g(\cdot)$ reconstructs the original data examples from the representation vectors. Let r_t denotes the reconstructed data example from transformed input x_t, then the detransformation loss function becomes:

$$\mathcal{L}(x, r_t) = \mathcal{L}(x, g(f(x_t))) \tag{1}$$

where $r_t = g(f(x_t))$. Specifically, we use MSE (Mean Squared Error) loss and have a total of M transformations, the loss function can be defined as:

$$\mathcal{L}_{\text{DTAE}} = \frac{1}{2} \sum_{t=0}^{M-1} \sum_{i=1}^{N} (x_i - r_{it})^2 \tag{2}$$

Each of the N data points has M transformations, and there are $M \times N$ data points after the input transformation module. In this work, we consider four

transformations for each data example, i.e. $t \in \{0, 1, 2, 3\}$ for all N input examples, resulting in $4N$ data points. For this paper, the four transformations in our experiments are rotations of an image: 0, 90, 180, and 270 degrees.

3.2 Fine-tuning step

While the pre-trained network can be fine-tuned by different loss functions, we focus on two types of loss functions in this paper: the classification loss and the representation loss. The objective of classification loss is to lower the classification error of the training data explicitly in the decision layers. One of the widely used classification loss functions is cross-entropy loss. The objective of representation loss functions is to learn better representations of training data. The representation loss functions are normally applied to the representation layers, such as triplet loss [15] and ii loss [6].

The fine-tuning network shares the same encoder and representation layer with the pre-training network. However, compared with the pre-training process, the fine-tuning process does not contain the input transformation module, which means the training examples are sent directly into the encoder. Moreover, instead of connecting to a decoder, the representation layer connects to a classification loss function or a representation loss function as shown in Figure 1b and Figure 1c. In this work, we consider both classification loss (cross-entropy loss) and representation loss (triplet loss [15] and ii loss [6]) in the OSR task.

3.3 Open Set Recognition (OSR)

A typical OSR task solves two problems: classifying the known classes and identifying the unknown class. From the representation level, the instances from the same class are close to each other, while those from different classes are further apart. Under this property, we propose the outlier score:

$$outlier\_score(x) = \min_{1 \le i \le C} \|\mu_i - z\|_2^2, \tag{3}$$

Where z is the learned representation of test sample x, μ_i is the representation centroid of the known class i. There are multiple ways to set the outlier threshold. Here, we sort the outlier score of the training date in ascending order and pick the 99 percentile outlier score value as the outlier threshold. Then, for the C known classes, we predict the class probability $P(y = i|x)$ for each class. When a network is trained on classification loss, the $P(y = i|x)$ is the output of the classification layer. Whereas in the case of a network without classification layer such as Figure 1c, we calculate $P(y = i|x)$ as:

$$P(y = i|x) = \frac{e^{-\|\mu_i - z\|_2^2}}{\sum_{j=1}^{C} e^{-\|\mu_j - z\|_2^2}} \tag{4}$$

In summary, a test instance is recognized as "unknown" if its outlier score is greater than the threshold t, otherwise it is classified as the known class with the highest class probability:

$$y = \begin{cases} unknown, & \text{if } outlier\_score(x) > t \\ \underset{1 \le i \le C}{\operatorname{argmax}} P(y = i|x), & \text{otherwise} \end{cases} \tag{5}$$

4 Experimental Evaluation

We evaluate the proposed pre-training method: Detransformation Autoencoder (DTAE) with simulated open-set datasets from the following datasets.
MNIST [9] contains 60,000 training and 10,000 testing handwritten digits from 0 to 9, which is 10 classes in total. Each example is a 28x28 grayscale image. To simulate an open-set dataset, we randomly pick six digits as the known classes participant in the training, while the rest are treated as the unknown class only existing in the test set.
Fashion-MNIST [17] is associated with 10 classes of clothing images. It contains 60,000 training and 10,000 testing examples. Same as the MNIST dataset, each example is a 28x28 grayscale image. To simulate an open-set dataset, we randomly pick six digits as the known classes participant in the training, while the rest are treated as the unknown class for testing.
CIFAR-10 [8] contains 60,000 32x32 color images in 10 classes, with 6,000 images per class. There are 50,000 training images and 10,000 test images. We first convert the color images to grayscale and randomly pick six classes out of the ten classes as the known classes, while the remaining classes are treated as the known class only existing in the test set.

4.1 Evaluation Network Architectures and Evaluation Criteria

In the proposed method, we use self-supervision in the pre-training stage, and then in the second stage, we fine-tune the pre-trained model with two types of loss functions: classification loss and representation loss. Specifically, we use the cross-entropy loss as the example of classification loss, and use ii loss [6] and triplet loss [15] as the examples of representation loss. We first trained the model from scratch as a baseline (no pre-training) for each loss function and compared it with the corresponding fine-tuned models after self-supervision. Second, to evaluate our proposed self-supervision technique DTAE, we compare the model performance using DTAE with traditional Autoencoder (AE) and RotNet [5] in the pre-training stage. We also compare the proposed method with OpenMax [1] to show that it is effective to OSR problems.

Figure 1a illustrates the network architecture of the DTAE. Moreover, the hyper-parameters are different based on datasets. For the encoder of the MNIST and the Fashion-MNIST datasets, the padded input layer is of size (32, 32), followed by two non-linear convolutional layers with 32 and 64 nodes. We also use

Table 1: The average ROC AUC scores of 30 runs at 100% and 10% FPR of OpenMax and a group of 5 methods (without pre-training as baseline, pre-training with AE, RotNet, DTAE and TAE) for each of the 3 loss functions (ce, ii, triplet). The underlined values are statistically significantly better than the baselines via t-test with 95% confidence. The values in bold and in brackets are the highest and the second-highest values in each group.

		MNIST		Fashion-MNIST		CIFAR-10	
FPR		100%	10%	100%	10%	100%	10%
OpenMax		0.9138	0.0590	0.7405	0.0160	0.6750	0.0060
ce	No pre-training	0.9255	0.0765	0.7175	0.0300	0.5803	0.0070
	AE	0.9410	**0.0805**	0.7346	0.0300	0.6114	[0.0084]
	RotNet	0.9367	0.0769	0.7364	[0.0316]	[0.6124]	0.0083
	DTAE (ours)	**0.9523**	[0.0801]	**0.7490**	**0.0324**	**0.6183**	**0.0086**
	TAE	[0.9477]	0.0799	[0.7389]	0.0298	0.6012	0.0075
ii	No pre-training	**0.9578**	0.0821	0.7684	0.0399	0.6392	0.0103
	AE	0.9560	**0.0828**	0.7636	0.0377	0.6320	0.0098
	RotNet	0.9530	0.0813	[0.7703]	[0.0404]	[0.6478]	[0.0106]
	DTAE (ours)	[0.9566]	[0.0825]	**0.7802**	**0.0410**	**0.6520**	**0.0108**
	TAE	0.9515	0.0815	0.7657	0.0387	0.6214	0.0091
triplet	No pre-training	0.9496	0.0750	0.7160	0.0211	0.6106	0.0089
	AE	**0.9563**	**0.0772**	0.7254	0.0220	0.6251	0.0090
	RotNet	0.9342	0.0702	[0.7435]	**0.0252**	[0.6285]	[0.0095]
	DTAE (ours)	[0.9543]	[0.0758]	**0.7441**	[0.0234]	**0.6327**	**0.0096**
	TAE	0.9531	0.0757	0.7271	0.0215	0.6114	0.0081

the max-polling layers with kernel size (3, 3) and strides (2, 2) after each convolutional layer. We use two fully connected non-linear layers with 256 and 128 hidden units after the convolutional component. Furthermore, the representation layer is six dimensions in our experiments. The representation layer is followed by a decoder, which is the reverse of the encoder in our experiments. We use the Relu activation function and set the Dropout's keep probability as 0.2. We use Adam optimizer with a learning rate of 0.001. The encoder network architecture of the CIFAR-10 experiment is similar to the MNIST dataset, except the padded input layer is of size (36, 36). We use batch normalization in all the layers to prevent features from getting excessively large. And as mentioned in section 3.3, we use contamination ratio of 0.01 for the threshold selection. The encoder and representation layer maintain the same architecture and hyper-parameters in the fine-tuning network. Meanwhile, the decoder is replaced with different fully connected layers associated with different loss functions.

We simulate three different groups of open sets for each dataset then repeat each group 10 runs, so each dataset has 30 runs in total. When measuring the model performance, we use the average AUC scores under 10% and 100% FPR (False Positive Rate) for recognizing the unknown class. We chose the 10% FPR limit as higher FPR is generally undesirable, particularly when negative instances are much more abundant than positive instances. We measure the F1 scores for known and unknown classes separately as one of the OSR tasks is to classify the known classes. Moreover, we perform t-tests with 95% confidence in the AUC scores and F1 scores to see if the proposed DTAE pre-training method can significantly improve different loss functions.

Table 2: The average F1 scores of 30 runs of OpenMax and a group of 5 methods (without pre-training as baseline, pre-training with AE, RotNet, DTAE and TAE) for each of the 3 loss functions (ce, ii, triplet). The underlined values show statistically significant improvements (t-test with 95% confidence) comparing to the baselines. The values in bold and in brackets are the highest and the second highest values in each group.

		MNIST			Fashion-MNIST			CIFAR-10		
		Known	Unknown	Overall	Known	Unknown	Overall	Known	Unknown	Overall
OpenMax		0.8964	0.7910	0.8814	0.7473	0.5211	0.7150	0.6456	0.5407	0.6307
ce	No pre-training	0.7596	0.7561	0.7591	0.6858	0.5591	0.6677	0.5672	0.3697	0.5390
	AE	0.7735	0.7894	0.7757	0.7264	0.5481	0.7009	0.5729	0.4605	[0.5569]
	RotNet	[0.8931]	[0.8447]	[0.8862]	0.7117	**0.5694**	0.6914	0.5616	**0.4729**	0.5489
	DTAE (ours)	**0.8967**	**0.8579**	**0.8912**	[0.7335]	[0.5692]	[0.7100]	**0.5911**	[0.4728]	**0.5742**
	TAE	0.8804	0.8420	0.8749	**0.7482**	0.5364	**0.7179**	[0.5815]	0.3889	0.5540
ii	No pre-training	0.9320	0.8833	0.9250	0.7720	0.5870	0.7456	0.6206	0.3570	0.5829
	AE	**0.9387**	**0.8950**	**0.9325**	0.7669	0.5745	0.7394	0.6241	0.2527	0.5711
	RotNet	0.9300	0.8761	0.9223	**0.7771**	**0.6108**	**0.7533**	**0.6442**	[0.3980]	[0.6090]
	DTAE (ours)	[0.9344]	[0.8885]	[0.9279]	[0.7768]	[0.6064]	[0.7524]	[0.6421]	**0.4252**	**0.6111**
	TAE	0.9308	0.8830	0.9240	0.7625	0.5869	0.7374	0.6135	0.2103	0.5559
triplet	No pre-training	0.9103	0.8302	0.8989	0.7491	0.5055	[0.7208]	0.5798	0.4515	0.5614
	AE	[0.9144]	0.8356	[0.9032]	0.7505	0.5051	0.7154	[0.6086]	[0.4800]	0.5902
	RotNet	0.9012	0.8182	0.8893	[0.7514]	[0.5376]	[0.7208]	0.6037	**0.4978**	[0.5886]
	DTAE (ours)	**0.9166**	**0.8513**	**0.9073**	**0.7558**	**0.5459**	**0.7259**	**0.6205**	0.4724	**0.5993**
	TAE	0.9126	[0.8387]	0.9021	0.7472	0.5092	0.7132	0.5926	0.4220	0.5682

4.2 Experimental Results

Model performance We compare the model performances of cross-entropy loss, ii loss, and triplet loss with and without pre-training. Table 1 are the averaged ROC AUC scores of the model performances in three datasets under different FPR values. Comparing "RotNet", "DTAE" and "AE" rows with "No pre-training" rows, we observe that using self-supervision techniques for pre-training significantly improved the model performance. The results also show that our proposed self-supervision method DTAE achieves the top two ROC AUC scores for all the cases. Moreover, with our proposed pre-training method, all three loss functions perform better than OpenMax in 5 out of 6 cases (3 datasets$\times$2 FPR limits).

To evaluate the detransformation component of DTAE, we performed an ablation study on our method without detransformation, which is denoted as TAE. Although both DTAE and TAE use transformed instances as input, TAE reconstructs the transformed instances as output, while DTAE reconstructs the original instances as output. Comparing the "TAE" rows and "DTAE" rows, we observe that the detransformation component in DTAE plays a key role in improving the model performance. That is, our results indicate that learning features invariant to transformations, via detransformation, can yield more effective features than those learned from reconstructing the same samples.

Table 2 shows that the OSR performances of different methods are measured by F1 scores in known and unknown class domains. We first calculate the F1 scores for each known class and the unknown class, then average all the

Table 3: The training time (in seconds) for the self-supervision methods in different datasets.

	AE	RotNet	DTAE
MNIST	75	132	137
Fashion-MNIST	70	118	145
CIFAR-10	86	147	182

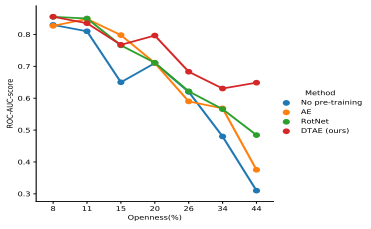


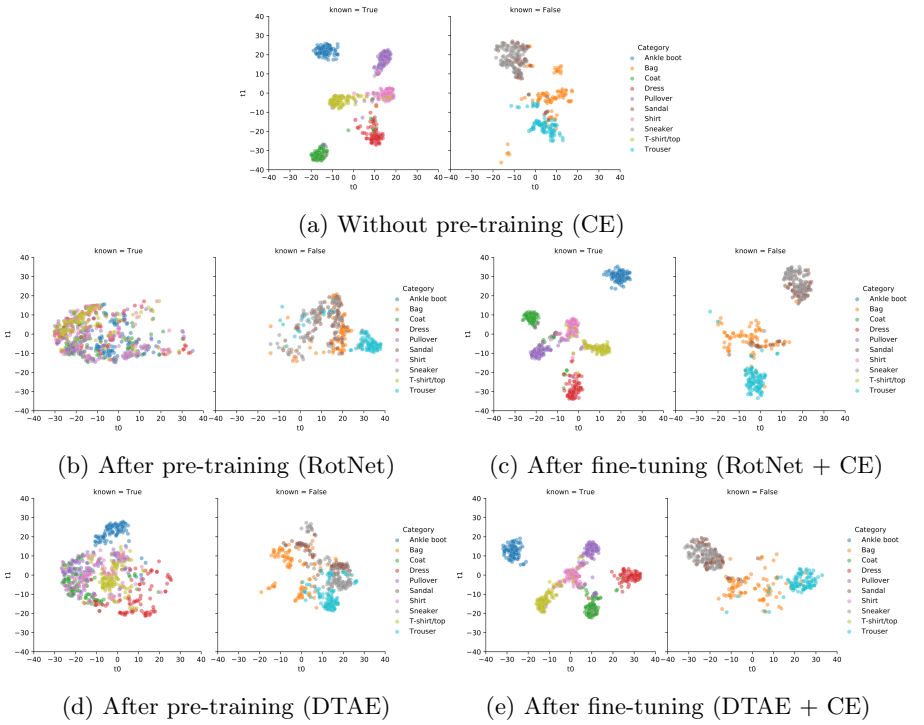
Fig. 2: AUC-ROC scores against varying Openness.

classes as the Overall F1 scores. The results show that models with pre-training achieve statistically significant improvements. Moreover, Our proposed method also achieves the top two F1 scores in 26 out of 27 cases (3 loss functions×3 datasets×3 domains).

Training time While the pre-training step benefits the model performances and does not affect the final model complexity and inference time, it takes extra time during the training phase. Table 3 shows the comparison of the training time of the self-supervised networks in different datasets via NVIDIA RTX 2080. Because RotNet and DTAE both include transformed data as input, they took a longer training time than AE. We observe that DTAE takes a slightly longer training time than RotNet. The reason is that the network structure of DTAE is more complex than that of RotNet. While both RotNet and DTAE share the same encoder and representation layer structures, RotNet uses a softmax layer after the representation layer. Meanwhile, DTAE connects the representation layer with a decoder module. The decoder is the reverse of the encoder, which contains more layers than a softmax layer and needs a longer time in the forward and backward propagations.

Openness study We also study the model performances against vary Openness [13]. Let n_{train} be the number of known classes participant in the training phase, with n_{test} denotes the number of classes in the test set, and n_{target} denotes the number of classes to be recognized in the testing phase. Openness can be defined as: $Openness = 1 - \sqrt{\frac{2 \times n_{train}}{n_{test} + n_{target}}}$.

In our experiments with the Fashion-MNIST dataset, we use all the ten classes in testing phase ($n_{test} = 10$) and varying the number of known classes from 2 to 9 ($n_{train} = 2, \ldots, 9$) in the training phase, and remaining classes together are treated as the unknown class to be recognized along with the known classes during inference ($n_{target} = n_{train} + 1$). That is, the openness is varied from 8% to 44%. We evaluate the AUC ROC scores of different models using cross-entropy loss: without pre-training (baseline), pre-training with AE, pre-training with RotNet, and pre-training with our proposed DTAE. The results are shown in Figure 2. We observe that the three different models have similar performances when the openness is small. However, the AUC ROC scores of the baseline (No pre-training) degrade rapidly as the Openness increases. Moreover,



(a) Without pre-training (CE)

(b) After pre-training (RotNet)

(c) After fine-tuning (RotNet + CE)

(d) After pre-training (DTAE)

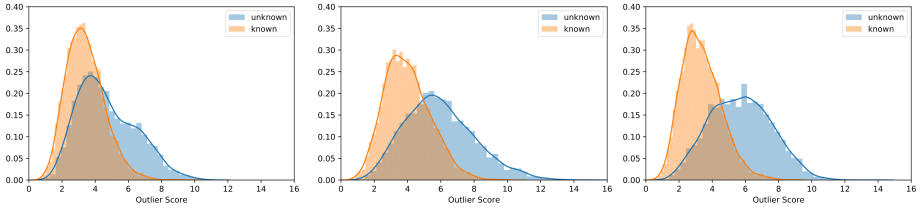
(e) After fine-tuning (DTAE + CE)

Fig. 3: The t-SNE plots of the Fashion-MNIST test set using cross-entropy loss. The left subplots are the representations of the known class, and the right plots are the representations of the unknown classes.

the trend is alleviated by pre-training with self-supervision methods. Overall, the model pre-trained with DTAE is relatively more robust to openness and achieves the best performance.

4.3 Analysis

To analyze the differences in representations after pre-training and after fine-turning, we plot 1000 samples from the Fashion-MNIST test set in Figure 3. In these experiments, classes "T-shirt/top", "Pullover", "Dress", "Coat", "Shirt" and "Ankle boot" are known classes while the remaining classes are unknown and absent from the training set. Figure 3a shows the t-SNE plot of the representations learned from cross-entropy loss without pre-training. Figures 3b and 3c are the learned representations of the model pre-trained by RotNet and fine-tuned by cross-entropy in different stages. Figures 3d and 3e are the learned representations of the model pre-trained by DTAE and again, fine-tuned by cross-entropy. From all the final representations of the three models in Figures 3a, 3c and 3e, we observe overlaps between the known class "Ankle boot" (blue) and one



(a) Without pre-training (CE)

(b) After fine-tuning (RotNet + CE)

(c) After fine-tuning (DTAE + CE)

Fig. 4: The distributions of outlier scores for the known and unknown classes of the Fashion-MNIST dataset in different experiments using cross-entropy loss.

component of the unknown class "Sneaker" (gray) as well as class "Dress" (red) and class "Trouser" (cyan). And the pre-training reduces the overlaps between "Shirt" (pink) and "Bag" (orange). Moreover, for the representations after pre-training, it shows that the representations learned by DTAE in Figure 3b are more separable than those learned by RotNet in Figure 3d for different classes. Note that DTAE, similar to RotNet, is not provided with class labels, but it can find representations that are more separable among the classes than RotNet. Moreover, we find that the representations learned by DTAE contain more fashion (target) information than those learned by RotNet (see supplementary material: `https://tinyurl.com/4amjev3m` for more details).

Figure 4 shows distributions of the outlier scores in experiments on the Fashion-MNIST test set. Compared with the model without pre-training in Figure 4a, the pre-training steps in Figures 4b and Figure 4c increase the outlier scores in the unknown classes, which pushes their score distributions further away from the known classes. The fact that there are fewer overlaps between the known classes and the unknown class makes them more separable. The results indicate that the model pre-trained with DTAE has the fewest overlaps between the known and unknown classes.

5 Conclusion

In this work, we introduce the self-supervision technique to OSR problems. We provide experiments across different image datasets to measure the benefits of the pre-training step for OSR problems. Moreover, we have presented a novel method: Detransformation Autoencoder (DTAE) for self-supervision. The proposed method engages in learning the representations that are invariant to the transformations of the input data. We evaluate the pre-trained model with both classification and representation loss functions. The experiments on several standard image datasets show that the proposed method significantly outperforms the baseline methods and other self-supervision techniques. Our analysis indicates that DTAE can yield representations that contain some target class information even without class labels.

References

1. Bendale, A., Boult, T.E.: Towards open set deep networks. In: Proc. of the IEEE conf. on computer vision and pattern recognition. pp. 1563–1572 (2016)
2. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.E.: A simple framework for contrastive learning of visual representations. In: Proc. of the 37th Intl. Conf. on Machine Learning, ICML 2020. pp. 1597–1607
3. Dietterich, T.G.: Steps toward robust artificial intelligence. AI Magazine **38**(3), 3–24 (2017)
4. Ge, Z., Demyanov, S., Garnavi, R.: Generative openmax for multi-class open set classification. In: British Machine Vision Conf. (2017)
5. Gidaris, S., Singh, P., Komodakis, N.: Unsupervised representation learning by predicting image rotations. In: 6th Intl. Conf. on Learning Representations, ICLR 2018
6. Hassen, M., Chan, P.K.: Learning a neural-network-based representation for open set recognition. Proc. SIAM Intl. Conf. Data Mining pp. 154–162 (2020)
7. Jia, J., Chan, P.K.: MMF: A loss extension for feature learning in open set recognition. In: Intl. Conf. on Artificial Neural Networks, Proc. Part II. pp. 319–331 (2021)
8. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
9. LeCun, Y., Cortes, C., Burges, C.J.: The MNIST database (1999), `http://yann.lecun.com/exdb/mnist/`
10. Ortiz, E.G., Becker, B.C.: Face recognition for web-scale datasets. Comput. Vis. Image Underst. **118**, 153–170 (2014)
11. Perera, P., et al.: Generative-discriminative feature representations for open-set recognition. In: 2020 IEEE/CVF Conf. on Computer Vision and Pattern Recognition. pp. 11811–11820
12. Saito, K., Yamamoto, S., Ushiku, Y., Harada, T.: Open set domain adaptation by backpropagation. In: Proc. of the European Conf. on Computer Vision (ECCV). pp. 153–168 (2018)
13. Scheirer, W.J., de Rezende Rocha, A., Sapkota, A., Boult, T.E.: Toward open set recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence **35**(7), 1757–1772 (2013)
14. Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U., Langs, G.: Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In: Information Processing in Medical Imaging - 25th Intl. Conf. pp. 146–157 (2017)
15. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Conf. on Computer Vision and Pattern Recognition. pp. 815–823. IEEE (2015)
16. Shu, L., Xu, H., Liu, B.: Unseen class discovery in open-world classification. arXiv preprint arXiv:1801.05609 (2018)
17. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms (2017)
18. Yoshihashi, R., Shao, W., Kawakami, R., You, S., Iida, M., Naemura, T.: Classification-reconstruction learning for open-set recognition. In: Conf. on Computer Vision and Pattern Recognition. pp. 4016–4025 (2019)
19. Zbontar, J., Jing, L., Misra, I., LeCun, Y., Deny, S.: Barlow twins: Self-supervised learning via redundancy reduction. In: Proc. of the 38th Intl. Conf. on Machine Learning, ICML 2021, Virtual Event. vol. 139, pp. 12310–12320. PMLR