Evolutionary Multi-Objective Optimization based Ensemble Autoencoders for Image Outlier Detection

Zhaomin Chen[a,*], Chai Kiat Yeo[a], Bu Sung Lee[a], Chiew Tong Lau[a], Yaochu Jin[b]

[a] *Computer Network and Communication Graduate Lab*
School of Computer Science and Engineering
Nanyang Technological University, Singapore 639798
[b] *Nature Inspired Computing and Engineering*
Department of Computer Science
University of Surrey

Abstract

Image outlier detection has been an important research issue for many computer vision tasks. However, most existing outlier detection methods fail in the high-dimensional image datasets. In order to address this problem, we propose a novel image outlier detection method by combining autoencoder with Adaboost (ADAE). By ensembling many weak autoencoders, our method can better capture the statistical correlations among the features of normal data than the single autoencoder. Therefore, the proposed ADAE is able to determine the outliers efficiently. In order to reduce the many parameters in ADAE, we introduce the Sparse Group Lasso (SGL) constraint into the learning objective of ADAE. We combine Adagrad with Proximal Gradient Descent to optimize this additional learning objective. We also propose the multi-objective evolutionary algorithm to determine the best penalty factors of SGL. By evaluating on several famous image datasets, the detection results testify to the outstanding outlier detection performance of ADAE. The evaluation results also show SGL can make the detection model more compact while maintaining the similar detection performance.

*Corresponding author
Email addresses: chen0954@e.ntu.edu.sg (Zhaomin Chen), asckyeo@ntu.edu.sg (Chai Kiat Yeo), ebslee@ntu.edu.sg (Bu Sung Lee), asctlau@ntu.edu.sg (Chiew Tong Lau), yaochu.jin@surrey.ac.uk (Yaochu Jin)

Keywords: Image Outlier Detection, Autoencoder, Adaboost, Sparse Group Lasso (SGL), Mutli-objective Evolutionary Algorithm, Adagrad Proximal Gradient Descent (Ada-PGD)

1. Introduction

It has become more and more critical to build large training datasets for many computer vision tasks. Retrieving images from the Internet is one of the most useful methods [1, 2, 3]. However, the retrieved results often contain many irrelevant images. For example, when we input 'Flower' query in a search engine, it may return a few plant images, which introduce outliers to the 'Flower' category. Therefore, outlier detection and removal are essential to help us construct pure training dataset.

In order to detect outliers efficiently, many outlier detection mechanisms have been proposed. There exists a category of methods that utilizes the reconstruction error to detect outliers [4, 5, 6, 7]. They are all based on the assumption that there are strong statistical correlations among the features of normal data. By minimizing the reconstruction errors of normal training data, these reconstruction-based methods are able to capture these correlations. Therefore, normal data should have relatively smaller reconstruction errors than the outliers. Those data having large reconstruction errors are determined as outliers. In [4, 5], the authors adopted PCA to learn the feature correlations from normal data. The methods in [6, 7] learn the non-linear correlations from data by using autoencoder. However, for high-dimensional image data, it is very difficult for one single autoencoder to fully learn the feature correlations of normal data.

In this paper, we combine autoencoder with Adaboost (ADAE) to detect image outliers. By repeatedly training on the weighted normal dataset, ADAE is able to obtain a sequence of weak autoencoders. The final prediction model of ADAE is a weighted summation of all the weak autoencoders. As each successive autoencoder attempts to capture the correlations that the previous one has failed to capture, ADAE can better capture the image feature correlations than the

single autoencoder. In addition, we introduce the Sparse Group Lasso (SGL) constraint to the learning objective of the autoencoder in order to eliminate unnecessary parameters.

The main contributions of our work are threefold. Firstly, we propose a novel image outlier detection model which combines autoencoder with Adaboost. By evaluating on several popular image datasets, this model has been shown to have better detection performance than the other detection methods. Secondly, we further introduce the Sparse Group Lasso constraint into ADAE (ADAE-SGL) to obtain a compact detection model. In order to optimize this additional objective of ADAE-SGL, we combine Adagrad with Proximal Gradient Descent (Ada-PGD). Thirdly, in order to determine the best penalty factors of SGL, we reformulate the objective of ADAE-SGL into a bi-objective optimization problem. Therefore, by utilizing evolutionary multi-objective optimization method, we find the best penalty factors successfully. The evaluation results show that ADAE-SGL achieves similar detection performance while obtaining a compact detection model.

The rest of this paper is structured as follows. Section 2 presents a general discussion about the existing outlier detection approaches. In Section 3, we propose the novel image outlier detection algorithm. In Section 4, we first introduce the Sparse Group Lasso constraint into ADAE. Ada-PGD optimization algorithm is proposed here to optimize the new learning objective. In addition, by reformulating the learning objective into a bi-objective optimization problem, we utilize evolutionary multi-objective optimization method to determine the best penalty factors for SGL. In Section 5, we compare the detection performance of ADAE with other outlier detection algorithms by evaluating them on several famous image datasets. We also evaluate the effectiveness of ADAE-SGL. Session 6 concludes the paper.

2. Related Works

Researchers have proposed many outlier detection mechanisms in existing literature. Generally, there are four main categories among these mechanisms: statistical methods, neighbor-based methods, kernel-based methods and reconstruction-based methods.

Statistical methods [8, 9] attempt to fit the distributions on the training data. Then any data which has low probability under this distribution will be determined as an outlier. However, this kind of methods depends on the choice of distribution. An unsuitable distribution may result in a bad detection performance. Neighbor-based methods assume that normal data has relatively more neighbors than the outlier data [10, 11]. In [10], Breunig et al. adopted a density-based local outlier factor (LOF) to address this issue. In addition, by normalizing LOF, Kriegel et al. [11] propose the local outlier probabilities to detect outliers. However, the search for the nearest neighbors prohibits such methods to be applied to high-dimensional data due to the curse of dimensionality. High dimensional data will fool the algorithm to locate the improper neighbors, which will decrease the detection accuracy. Kernel-based methods [12, 13, 14] attempt to find a soft boundary which can surround the most normal data. In [12], Schölkopf et al. transformed this problem into the quadratic programming, so as to purposed the one class support vector machine (OCSVM). In [13], Azami et al. converted the OCSVM scores into the outlier probabilities. Quinn et al. [14] attempted to find the boundary based on a squared-loss function which is similar to OCSVM method. However, this kind of methods needs to calculate the kernels, which has high computational complexity. Thus, kernel-based methods are both time-consuming in training and testing.

Reconstruction-based methods assume that there are strong correlations in the features of the normal data. By minimizing the reconstruction errors of the normal training data, the reconstruction-based methods can capture these strong correlations. Therefore, the normal test data will have relatively smaller reconstruction errors, while outliers will have larger reconstruction errors. In

[4, 5], the authors adopt PCA to learn the strong correlations, and the method determines the data with larger reconstruction error as outliers. In [6], Sakurada et al. proposed an Autoencoder-based outlier detection method. As autoencoder can capture the nonlinear correlations as well as the linear correlations, it has better detection performance than PCA-based one. However, when applying the autoencoder to image outlier detection, the detection performance is not always very palatable. This is because the single autoencoder fails to fully capture the correlations among features, especially in the high-dimensional datasets, resulting in poor detection accuracy.

In this paper, we first address the image outlier detection problem by combining autoencoder with Adaboost. The overall idea of this algorithm is to sequentially apply the weak autoencoders to repeatedly modified versions of the training data. Therefore, by using an ensemble of these weak autoencoders, Adaboost-Autoencoder (ADAE) is able to better capture the statistical correlations among the features of the normal data so as to achieve better detection performance. In addition, by introducing Sparse Group Lasso constraint, we can generate the sparse solutions for the sake of making the detection model simpler. The best penalty factors of SGL are determined by the evolutionary multi-objective optimization method.

3. Adaboost-Autoencoder

In this section, we illustrate the Autoencoder-based outlier detection algorithm first. Then, we propose an extended image outlier detection algorithm which combines Adaboost with autoencoder.

3.1. General Autoencoder-based Outlier Detection

Autoencoder is a simple neural network which only has one hidden layer. Normally, it consists of two parts: the *encoder* and *decoder*

Encoder : Let us assume that the training set is $\{x_1, x_2, ..., x_n\}$, which are all d dimensional vectors ($x_i \in R^d$). The *Encoder* is trying to encode the original

data into m ($m < d$) number of neurons which make up the hidden layer. The activation output of the hidden neuron i is:

$$a_i = f_\theta(x) = s(\sum_{j=1}^{n} W_{ij}^{input} x_j + b_i^{input}) \tag{1}$$

where x is the input vector, s is a non-linear activation function, θ is the parameters $\{W^{input}, b^{input}\}$, W represents the encoder weight matrix with size $m \times d$ and b is a bias vector of dimensionality m. Therefore, the input vector is encoded into a lower-dimensional vector.

Decoder : The resulting hidden representation a is then decoded back to the original input space R^d. The mapping function is as follows:

$$x_i' = h_{\theta'}(a) = s(\sum_{j=1}^{n} W_{ij}^{hidden} a_j + b_i^{hidden}) \tag{2}$$

The parameter set of the decoder is $\theta' = \{W^{hidden}, b^{hidden}\}$.

We optimize the autoencoder to minimize the average reconstruction error with respect to θ and θ':

$$\theta^*, \theta'^* = \arg\min_{\theta,\theta'} \frac{1}{n} \sum_{i=1}^{n} \varepsilon(x_i, x_i{}') \tag{3}$$

where ε is the square error function.

As stated in [6], outlier detection using autoencoder is based on the assumption that there are strong correlations between the features of the normal data. In the training phase, we have the normal data as the training set. Hence, by minimizing the reconstruction errors, the autoencoder is able to find a subspace which can capture the statistical correlations of the normal data. In the test phase, we feed the data into this autoencoder and calculate their reconstruction errors. As the outliers are images from different semantic concepts, they are less likely to be reconstructed well, resulting in relatively larger reconstruction errors than the normal data.

As a result, by thresholding these reconstruction errors, we can easily identify the outlier image data:

$$c(x_i) = \begin{cases} normal & \varepsilon_i < \theta \\ outlier & \varepsilon_i > \theta \end{cases} \tag{4}$$

3.2. Combining Autoencoder with Adaboost

When applying the general autoencoder method to the image outlier detection, the detection results do not meet the expectation. This is because one autoencoder is not enough to capture all the statistical correlations of the normal data, especially in high-dimensional image dataset. In this subsection, we propose a novel image outlier detection algorithm which combines Adaboost with autoencoder. This Adaboost-Autoencoder (ADAE) outlier detection algorithm is presented in Algorithm 1.

Input: Dataset X of size n, the maximum number of iterations N, the initial weight $\vec{w}(w_i = 1/n)$

for *t = 1, ..., N* **do**

1. Train $t-th$ autoencoder (h_t) with the training set X and current weights $\vec{w}$: $\arg\min_h \frac{1}{n}\sum_i w_i \varepsilon(x_i, h(x_i))$
2. Calculate the adjust error e_i^t for each datum:
   Let $D_t = \max_{i=1}^n \varepsilon(x_i, h_t(x_i)) = \max_{i=1}^n \sum_j (x_{ij} - h_t(x_{ij}))^2$, then $e_i^t = \frac{\varepsilon(x_i, h_t(x_i))}{D_t}$
3. Calculate the average error of h_t: $e_t = \sum_{i=1}^{n} w_i^t e_i^t$
4. Calculate the error rate of h_t : $\beta_t = e_t/(1-e_t)$
5. Update the weight vector $w_i^{t+1} = w_i^t \beta_t^{1-e_i^t}/Z_t$

end

Output: $h_f(x) = \sum_{t=1}^{N} (\ln \frac{1}{\beta_t}) h_t(x)$

For AD-AE, the outlier score is $\varepsilon(x, h_f(x))$.

Algorithm 1: Adaboost-Autoencoder (ADAE) outlier detection

At the very beginning, initial weights $w_i = 1/n$ are assigned to each set of training data, so that the first autoencoder is trained on the data in the usual manner. For each successive iteration, the weight for each training data is modified according to its corresponding reconstruction error. Those data that have larger reconstruction error in the previous step have their weights increased, whereas the weights are decreased for those with smaller reconstruction error.

Each successive autoencoder is thereby forced to concentrate on those data that have not been reconstructed well by the previous ones. Actually, during the training process of ADAE, the autoencoders attempt to capture the correlations that previous ones fail to capture. Finally, all of these autoencoders are combined through a weighted summation. The weight for each autoencoder represents the contribution of each respective autoencoder to the final model.

By assembling all the autoencoders, we can better capture the statistical correlations among the features of the normal data. Therefore, ADAE can achieve much better outlier detection accuracy compared to the general Autoencoder-based method.

4. Multi-objective Optimization of Regularization with Sparse Group Lasso

As ADAE utilizes a sequence of autoencoders to capture the statistical correlations among the features of the normal image data, ADAE should have an excellent image outlier detection performance. However, it has a drawback. As it consists of many autoencoders, there are too many parameters in our model, which may result in large computational complexity for detecting image outliers. Therefore, in this section, we introduce the group sparsity regularization [15] during the training process to make the detection model compact.

By adding the regularization term, the training objective of each autoencoder becomes:

$$\min_{w} L(w) = \min_{w} \varepsilon(\{w\}, D) + \lambda\Omega(w) \tag{5}$$

where D is the training dataset, ε is the reconstruction loss parameterized by weights w, Ω is the pre-defined regularization term, and λ is the penalty term which balances the effects of loss function and regularization term.

4.1. Sparse Group Lasso

As we all know, the most famous regularization term is l_2 regularization [16], which is also denoted as "weight decay": $\Omega(w) \triangleq \|w\|_2^2$. Although l_2

regularization term does reduce the magnitude of weights, it cannot generate a sparse solution. The only way to obtain sparsity with weight decay is to set the weights to zero whose amounts are smaller than a pre-defined threshold. However, the threshold is not only difficult to choose, but also not suitable for the real application.

The second most common method is l_1-norm regularization term [17], which is to penalize the absolute value of the weights:

$$\Omega(w) \triangleq \|w\|_1 = \sum_k |w_k| \tag{6}$$

Apparently, l_1 norm is not differentiable at 0. Therefore, it is necessary to apply sub-gradient decent or proximal gradient descent [18] to optimize the objective function. As presented in [18], the proximal map of l_1 norm is the soft-thresholding operator. Thus, the update of this proximal gradient can lead to sparse solutions.

In order to make the neural network compact, we need to prune the unnecessary neurons whose incoming or outgoing weights are all 0. In this sense, l_1 norm is not suitable to obtain a compact neural network. In this paper, we adopt the Sparse Group Lasso (SGL) regularization term which is introduced in [15]. SGL consists of two parts: l_1 norm and group sparsity factor:

$$\Omega_{SGL}(w) \triangleq \mu\Omega_{GS}(w) + (1-\mu)\Omega_{l_1}(w) \tag{7}$$

where μ is the hyper-parameter for balancing the effects of l_1 norm and group sparsity term. Group sparsity term is defined as the summation of group l_2 norms:

$$\Omega_{GS}(w) \triangleq \sum_g \|W_g\|_2 = \sum_g \sqrt{\sum_i w_{g,i}^2} \tag{8}$$

where W_g is the weight matrix for group g. In our implementation, we put the outgoing weights from the same neuron into one group. Since l_2 norm has the effect that leads to similar value of weights in one group, minimizing this group sparsity term will result in complete zero of some groups' weights, so that we can remove such surplus neurons. Therefore, by adding this group sparsity

regularization term, the optimizer can decide which neuron is effective at each layer.

By adding l_1 norm, SGL can also guarantee the sparsity of weights among the remaining neurons after removing the surplus neurons. The illustrations of sparsity of l_1 norm, group sparsity and SGL are presented in Fig. 1. As we can see, SGL can remove some neurons completely. At the same time, it can result in a sparse solution for the other active neurons.

By introducing SGL, the training objective in Eq. 5 becomes:

$$\min_{w} L(w) = \min_{w} \varepsilon(\{w\}, D) + \lambda[\mu\Omega_{l_1}(w) + (1-\mu)\Omega_{l_{GS}}(w)] \tag{9}$$

Therefore, by utilizing this additional objective to train each autoencoder in ADAE, we can get a compact detection model, which we refer to as ADAE-SGL.

4.2. Adagra Proximal Gradient Descent (Ada-PGD)

As SGL contains l_1 and l_2 norms, SGL is not differentiable at some points. Therefore we utilize the proximal gradient descent to optimize the regularized training objective. Proximal gradient descent consists of two steps. Firstly, it computes an intermediate solution $w_{t+\frac{1}{2}}$ by performing gradient descent to the reconstruction loss error $\varepsilon(\{w\}, D)$. The second step is to compute the proximal map of SGL:

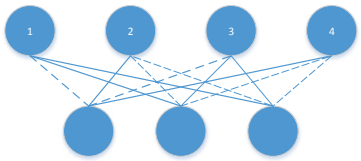
$$w_{t+1} = \arg\min\{\frac{1}{2}\left\|w - w_{t+\frac{1}{2}}\right\|^2 + \lambda\eta_t\Omega_{SGL}(w)\} \tag{10}$$

where η_t is the $t-th$ learning rate and λ is the penalty factor of regularization term.

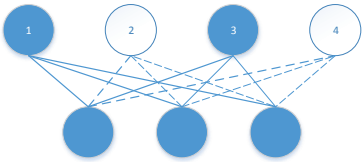
Based on the work in [18], the proximal mapping for the group sparsity is obtained as follows:

$$PG_{GS}(W, \eta) = (1 - \frac{\lambda\eta}{\|W_g\|_2})_+ W_g \tag{11}$$

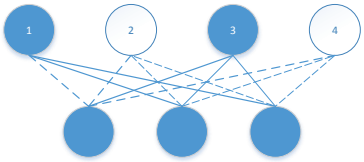
where $\|W_g\|_2$ is the l_2 norm of the weights of group g. From this proximal mapping, we can find out why group sparsity has the effect of completely removing some neurons. During this proximal update, whenever the l_2 norm of



(a) L_1 norm Sparsity



(b) Group Sparsity



(c) Sparse Group Lasso

Figure 1: **Illustration of Different Sparse Regularizers** The dash lines represent the connection with zero weights. The white circles denote the removed neurons.

some group is smaller than $\eta_{t+\frac{1}{2}}$, the weights in this group will be all set to zero, which leads to the removal of the entire group.

The proximal mapping for the l_1 norm is given as follows:

$$PG_{l_1}(W, \eta) = sign(W)[|W| - \lambda\eta_t]_+ \tag{12}$$

By combining Adagrad with this proximal gradient, we propose Adagrad Proximal Gradient Descent (Ada-PGD) algorithm to optimize our training objective. Adagrad is an adaptive learning rate optimization method proposed by [19]. The traditional Gradient Descent takes the learning rate constantly for all parameters, but Adagrad adjusts the learning rate for each individual parameter. For each parameter, its current learning rate is divided by the l_2-norm of the square summation of its previous gradients. Thus, if the l_2-norm is large, the learning rate will be small. Otherwise, the learning rate will be large. Therefore, Adagrad can adaptively control the learning rate of each parameter. The detailed Ada-PGD algorithm is presented in Algorithm 2.

Input: Dataset D, general learning rate η

while W *is not converging* **do**

  for *each mini-batch data* **do**

    1. Get the learning rate for the current step: $\eta_{curr} = \frac{\eta}{\sqrt{cache+\Delta}}$. The variable *cache* keeps track of the square summation of each parameter's gradient and Δ is a smoothing term to avoid division by zero.
    2. Update the parameters with gradient descent: $W_{t+\frac{1}{2}} = W_t - \eta_{curr}\nabla\varepsilon(W)$
    3. Update the parameters with Eq.11: $W_{t+\frac{1}{2},GS} = PG_{GS}(W_{t+\frac{1}{2}}, \eta_{curr})$
    4. Update the parameters with Eq.12: $W_{t+\frac{1}{2}} = PG_{l_1}(W_{t+\frac{1}{2},GS}, \eta_{curr})$

  end

end

Algorithm 2: Adagrad Proximal Gradient Descent Algorithm for Optimizing SGL Regularization

4.3. Evolutionary Multi-objective Optimization of the Training Objective with SGL

SGL contains two hyper-parameters λ and μ, which need to be determined in advance. Inspired by the work in [20], we adopt the Pareto-based learning approach to select the best λ and μ.

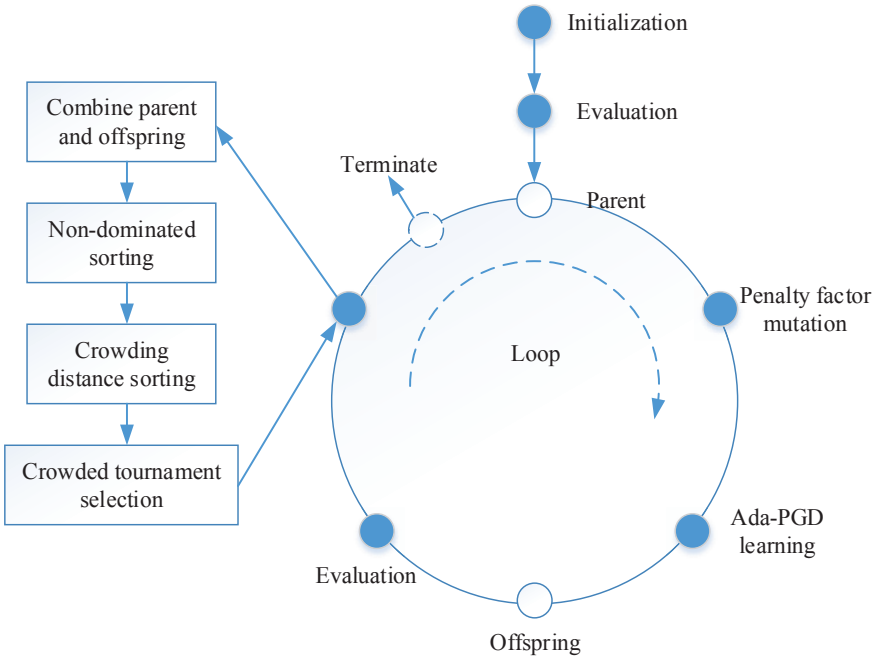


Figure 2: Framework for Multi-objective Optimization of SGL Using NSGA-II

The training objective in Eq. 9 can be reformulated into a Pareto-based multi-objective optimization problem:

$$\min_{w} L(w) = \min_{w}(\varepsilon(w), \Omega_{SGL}(w)) \tag{13}$$

Therefore, we can utilize Pareto-based learning framework to obtain the best sparse constraint penalty factors (λ and μ).

The detailed learning framework is presented in Fig. 2. As we attempt to find the best penalty factors, the mutation operations are only applied to vary λ and μ in this framework. After mutation, Ada-PGD is employed to optimize the

training objective Eq. 9. According to the evaluations of $\varepsilon(w)$ and $\Omega_{SGL}(w)$, we can select the population of the next generation. The selection method from Non-dominated Sorting Genetic Algorithm II (NSGA-II) [21] is utilized here. It mainly contains four steps. First, we generate offspring using simulated binary crossover and polynomial mutation. Parents are selected using a binary tournament selection leveraging between the non-dominated sorting and crowding distance. Combine the parent and offspring populations.. Second, the algorithm performs non-dominated sorting and calculate the crowding distance for the combined population. Third, the population is sorted according to the non-dominated sorting and the crowding distances. Fourth, NSGA-II selects the best half of the combined population as the parent of the next generation.

Finally, this Evolutionary Multi-Objective (EMO) based Optimization framework is able to achieve an amount of Pareto-optimal solutions that make up the Pareto Front. The knee point solution of the Pareto Front is selected as the final solution to determine the optimal λ and μ.

4.4. Conclusion

Fig. 3 presents the whole diagram of the ADAE detection model. By adopting the EMO-based optimization framework in Fig. 2, we can obtain the best penalty factors of SGL (λ_{opt} and μ_{opt}). Therefore, by feeding the deep learning features of images, we can utilize Ada-PGD to optimize the ADAE with SGL constraint. Finally, a compact image outlier detection model can be obtained.

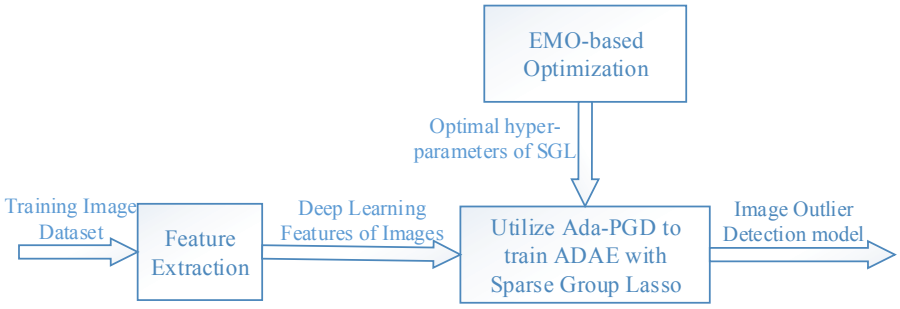


Figure 3: The Diagram of ADAE with Sparse Group Lasso Constraint

Table 1: Detail Information about all the Evaluation Datasets

	Trainset	Testset	
	Average number of Normal Images	Average number of Normal Images	Average number of Outlier Images
Caltech-101	500	300	460
Scene_data	200	100	500
STL-10	500	900	900
ImageNet	800	520	495

5. Evaluations

5.1. Image Datasets and Evaluation Metrics

We utilize the following image datasets to evaluate ADAE.

- **Caltech-101** [22] is an image dataset which consists of images of objects belonging to 101 categories. In our experiments, 10 categories of image data are used to evaluate our proposed algorithm. The size of each image is roughly 300×200 pixels.
- **Scene_data** [23] consists of fifteen natural scene categories. All the categories are included here for the evaluation, and each category contains about 300 image data on average.
- **STL-10** [24]is an image recognition dataset which contains 10 classes of image data. All the image data are utilized in our evaluation.
- **ImageNet Data** [25] is a large visual database designed for object recognition research. 10 classes of image data from the ImageNet website are extracted for our evaluation.

In our evaluation, the normal data are all selected from one category of images, whereas the outliers are the images of other categories. The training datasets contain only the normal data. The test dataset consists of both normal and outlier data. The detail information of all the evaluation datasets is listed in

Table 1. VGG [26] is utilized here to extract the deep learning features of each image in each dataset. In our evaluations, we just utilize the 4096-dimensional outputs of the first fully-connected layer in VGG.

In the following evaluations, we mainly adopt two evaluation metrics: AUC score and F1 score. AUC score is the area under the ROC curve. With different values of thresholds, we can obtain the ROC curve so as to get the AUC score. In addition, for all of these thresholds, we can also calculate their corresponding F1 scores. Among these F1 scores, we select the best F1 score for evaluation.

5.2. Methods to be compared

We also implement several famous outlier detection methods to compare with our proposed algorithm:

- Autoencoder (AE) As introduced in Sec. 3.1, the reconstruction errors are treated as outlier scores. For AE and every autoencoder in ADAE, we only set 20 hidden neurons and utilize the ReLU activation function.

- Local Outlier Factor (LOF) [10] By measuring the local density of a data point with respect to its neighbors, the authors in [10] introduced the local outlier factor (LOF). This LOF is a local factor which only takes into account a restricted neighborhood of each data point. Therefore, we utilize LOF scores as the outlier scores to classify the outlier data.

- Isolation Forest (IF) [27] As the outliers are few and different from the normal data, they are more susceptible to be isolated. Hence, the outliers can be isolated much closer to the root of IF. Therefore, we take the number of splittings required to isolate the image data as the outlier scores.

- One-class SVM (OCSVM) [12] Given a set of normal data, OCSVM attempts to find a soft boundary of that set. Therefore this boundary will help to classify the new point based on whether the point belongs to that set or not. In our evaluation, we take the distance to the boundary as the outlier score. The points inside the boundary will be assigned a negative

distance, whereas the points outside the boundary will have a positive distance.

- Least Square Approach (LS) [14] Based on a squared-loss function, the authors propose a probabilistic and nonparametric methods for outlier detection. It is a kernel-based method which is similar in essence to OCSVM. As the proposed objective function has a simple analytical solution, it is faster to train than OCSVM. For this method, we adopt the outlier probability as the outlier scores.

For all these methods, we can obtain their corresponding outlier scores. Based on these outlier scores, we can calculate the corresponding AUC and F1 scores. Therefore, we can compare their detection performance in terms of AUC and F1 scores.

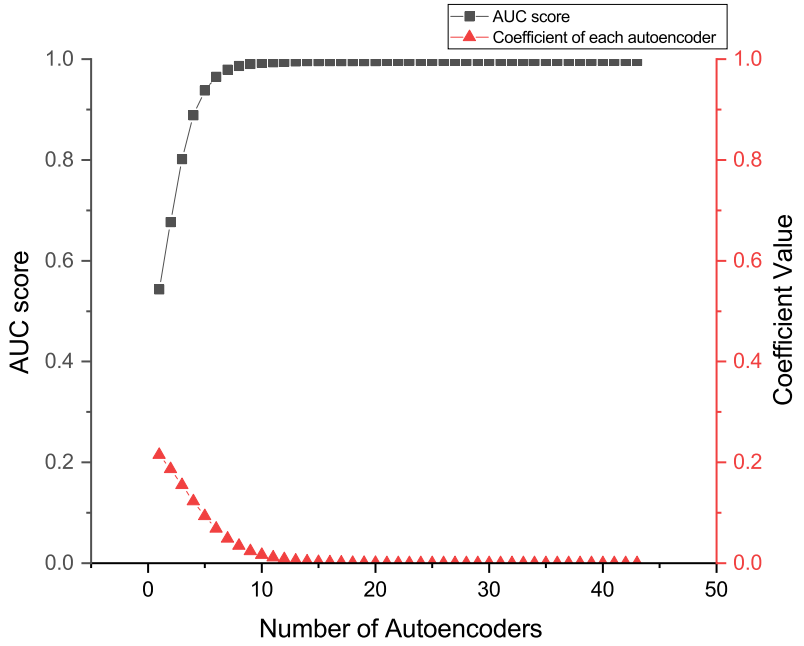


Figure 4: AUC scores and Coefficients for different number of autoencoders

5.3. Number of Iterations in ADAE

There is one hyper-parameter in ADAE that needs to be decided in advance: the number of iterations (N) in ADAE. Although more iterations can help ADAE increase the detection accuracy, it also increases the number of autoencoders which results in parameter redundancy of the detection model. Therefore, more iterations will result in greater computational costs in both training and testing.

Here we adopt the image data from 'goshawk' category in ImageNet as the example dataset. By evaluating ADAE algorithm on this example dataset, we can calculate the AUC scores for ADAE with different iteration numbers, which are presented in Fig. 4. As we can see, as the iterations of ADAE progresses, the AUC score steadily increases, reaching 0.996 after 20 iterations. 20 iterations are enough for ADAE to achieve good detection performance. In addition, Fig. 4 also shows the coefficient of each autoencoder. As the number of iterations increases, the coefficient keeps on decreasing. The coefficient for the $20th$ autoencoder is nearly 0, which means that the autoencoders after the $20th$ iteration have very little influence on the final detection model. Therefore, adding more iterations for ADAE only improves the detection accuracy slightly.

In conclusion, for the following evaluations, we just train ADAE with 20 iterations.

5.4. Evaluations of ADAE

In this subsection, we compare the detection performances of the different outlier detection algorithms by evaluating on the image datasets. Comparison results on each category of STL-10 dataset are presented in Fig. 5 and Fig. 6. The average AUC and F1 scores of the different methods are listed in Table 2. From these comparisons, we have the following observations:

- Among all these 6 detection methods, LOF has the worst detection performance. This is because LOF needs to calculate the local density of k nearest neighbors, which is strongly influenced by the curse of dimensionality [28]. In our experiments, we utilize 4096-dimensional deep learning

features to represent each image. Therefore, LOF cannot guarantee us a satisfactory detection result.

- The detection performance of AE is just slightly better than LOF, but is worse than the other 4 methods. This validates that one single autoencoder is not enough to capture the correlations of the features of the normal data.

- IF and OCSVM have similar detection performances, whereas LS performs slightly worse than them. However, as there are many hyper-parameters in OCSVM, it is very difficult to select the best hyper-parameters for the implementation. During the training process of IF, there is some randomness in IF. Therefore, for the data of some labels (e.g. 'bird'), IF performs badly.

- ADAE outperforms the other detection methods for all the categories. As we can see in Table 2, ADAE achieves the highest average AUC score and F1 score.

Table 3 also presents the comparison results on Caltech-101, Scene_data and ImageNet in terms of Precision, Recall and F1 scores. The scores are averaged over all the categories for each dataset. ADAE also outperforms other methods on these datasets.

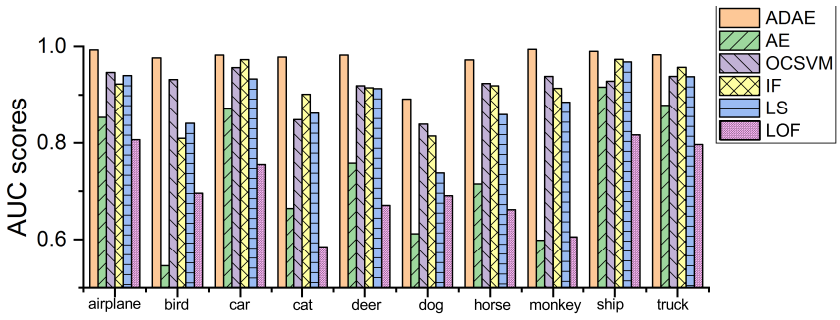


Figure 5: AUC score comparisons on each category in STL-10 dataset

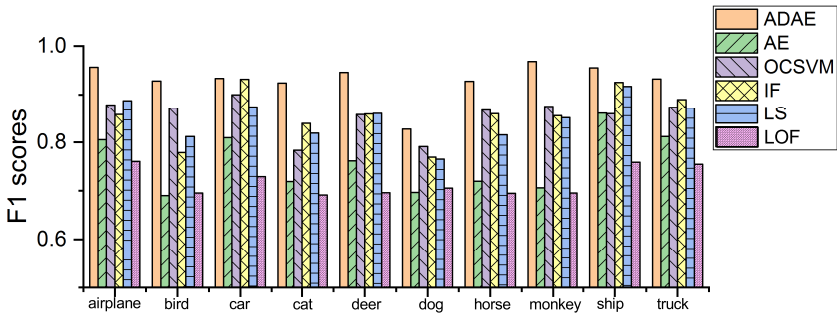


Figure 6: F1 score comparisons on each category in STL-10 dataset

Table 2: Average AUC and F1 scores on STL-10 Dataset

Methods	ADAE	AE	OCSVM	IF	LS	LOF
AUC Scores	0.9743	0.7411	0.9168	0.9096	0.8874	0.70833
F1 Scores	0.9299	0.7578	0.8555	0.8561	0.8470	0.7181

5.5. Evaluation of ADAE with SGL constraint (ADAE-SGL)

We also evaluate the detection performance of ADAE-SGL in this subsection. At the beginning, we need to determine the penalty factor of SGL (λ and μ) by adopting the Pareto-based learning framework presented in Fig. 2. Then, we compare the detection performance of ADAE-SGL with that of ADAE.

5.5.1. Pareto Front

We utilize the Pareto-based learning framework to obtain λ and μ by testing on "airplane" data in STL-10 dataset. Fig. 7 presents the Pareto Front which composed of 40 solutions.

For most multi-objective optimization problem, the knee point solution is preferred if no other user-specific or problem-specific preferences are available. Based on the Pareto Front in Fig. 7, the best compromise solution is selected at the knee point: $\lambda = 0.00563$ and $\mu = 0.9289253$. Therefore, in the following evaluations, we utilize this set of λ and μ as the penalty factors of SGL for each autoencoder in ADAE-SGL.

Table 3: Detection Results of Different Algorithms on Caltech-101, Scene_data and ImageNet. Evaluation metrics are averaged over all categories

Dataset	Caltech-101			Scene_data			ImageNet		
Metrics	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
ADAE	1.0000	1.0000	1.0000	0.9722	0.9780	0.9751	0.8709	0.9401	0.9050
AE	0.9703	0.9957	0.9828	0.8863	0.9820	0.9317	0.6036	0.9244	0.7186
OCSVM	0.9515	0.9804	0.9657	0.9081	0.9680	0.9371	0.7679	0.8857	0.8200
IF	0.9957	0.9978	0.9967	0.9257	0.9720	0.9483	0.7786	0.8962	0.8314
LS	0.9808	0.9978	0.9892	0.8853	0.9880	0.9338	0.7223	0.8646	0.7819
LOF	0.8740	0.9196	0.8962	0.6819	0.9560	0.7960	0.5060	0.8610	0.6374

5.5.2. Comparison with ADAE

By evaluating on Caltech-101, Scene_data, STL-10, and ImageNet datasets, we compare the detection results in terms of recall, precision and F1 score. In addition, we also introduce two sparse evaluation metrics to reveal the power of SGL: Neuro Sparsity and Weight Sparsity. The Neuro Sparsity is defined as the number of zeroed-out neurons over the total neuron number, whereas the Weight Sparsity is the proportion of weight parameters with zero values.

Table 4 presents the comparisons of ADAE and ADAE-SGL. As we can see, ADAE-SGL achieves similar or slightly worse detection performance than ADAE on all the datasets while only using 60% active neurons. It is be noted that the Weight Sparsity is larger than the Neuro Sparsity. This reveals that the SGL constraint not only generates some inactive neurons, but also generates the sparse solutions of weights among the remaining active neurons.

6. Conclusions

In this work, we address the problem of outlier detection in high-dimensional image datasets. We combine autoencoder with Adaboost to capture the statistical correlations among normal data's features. Moreover, we introduce the Sparse Group Lasso constraint to every autoencoder in order to obtain a com-

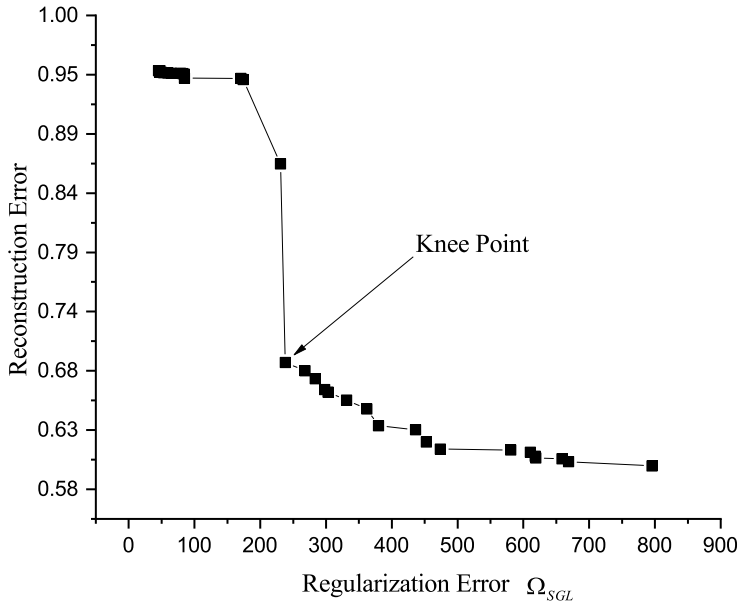


Figure 7: Pareto Front obtained for the "airplane" image data in STL-10 composed of 40 solutions

pact model. We propose Adagrad Proximal Gradient Descent to optimize the new training objective, and utilize Pareto-based learning framework to obtain the best penalty factors of SGL. By evaluating on several image datasets, our proposed detection algorithm not only has a compact detection model, but can also detect the image outliers efficiently.

List of Acronyms

ADAE	Adaboost-Autoencoder
Ada-PGD	Adagrad Proximal Gradient Descent
ADAE-SGL	Adaboost Autoencoder with Sparse Group Lasso Constraint

AE	Autoencoder
IF	Isolation Forest
LOF	Local Outlier Factor
LS	Least Square Approach
NSGA-II	Non-dominated Sorting Genetic Algorithm II
OCSVM	One-class SVM
PGD	Proximal Gradient Descent
SGL	Sparse Group Lasso

References

[1] T. Jin, Z. Liu, Z. Yu, X. Min, L. Li, Locality preserving collaborative representation for face recognition, Neural Processing Letters 45 (3) (2017) 967–979. doi:10.1007/s11063-016-9558-2.

[2] M.-L. Zhang, ML-RBF:RBF neural networks for multi-label learning, Neural Processing Letters 29 (2) (2009) 61–74. doi:10.1007/s11063-009-9095-3.

[3] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, Y. Zheng, Nus-wide: a real-world web image database from national university of singapore, in: Proceedings of the ACM International Conference on Image and Video Retrieval, ACM, 2009, p. 48. doi:10.1145/1646396.1646452.

[4] W. Qiu, Y. Wu, G. Wang, S. X. Yang, J. Bai, J. Li, A novel unsupervised anomaly detection based on robust principal component classifier, in: B. V. Dasarathy (Ed.), Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2006, SPIE, 2006. doi:10.1117/12.664383.

[5] H. Xu, C. Caramanis, S. Mannor, Outlier-robust pca: the high-dimensional case, IEEE Transactions on Information Theory 59 (1) (2013) 546–572. doi:10.1109/tit.2012.2212415.

[6] M. Sakurada, T. Yairi, Anomaly detection using autoencoders with nonlinear dimensionality reduction, in: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis, ACM, 2014, p. 4. doi:10.1145/2689746.2689747.

[7] J. T. Andrews, E. J. Morton, L. D. Griffin, Detecting anomalous data using auto-encoders, International Journal of Machine Learning and Computing 6 (1) (2016) 21.

[8] A. Dukkipati, D. Ghoshdastidar, J. Krishnan, Mixture modeling with compact support distributions for unsupervised learning, in: Neural Networks (IJCNN), 2016 International Joint Conference on, IEEE, 2016, pp. 2706–2713. doi:10.1109/ijcnn.2016.7727539.

[9] E. Eskin, Anomaly detection over noisy data using learned probability distributions, in: In Proceedings of the International Conference on Machine Learning, Citeseer, 2000.

[10] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, Lof: identifying density-based local outliers, in: ACM Sigmod Record, Vol. 29, ACM, 2000, pp. 93–104. doi:10.1145/335191.335388.

[11] H.-P. Kriegel, P. Kröger, E. Schubert, A. Zimek, LoOP: local outlier probabilities, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, 2009, pp. 1649–1652.

[12] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural Computation 13 (7) (2001) 1443–1471. doi:10.1162/089976601750264965.

[13] M. El Azami, C. Lartizien, S. Canu, Converting svdd scores into probability estimates: Application to outlier detection, Neurocomputingdoi:10.1016/j.neucom.2017.01.103.

[14] J. A. Quinn, M. Sugiyama, A least-squares approach to anomaly detection in static and sequential data, Pattern Recognition Letters 40 (2014) 36–40. doi:10.1016/j.patrec.2013.12.016.

[15] S. Scardapane, D. Comminiello, A. Hussain, A. Uncini, Group sparse regularization for deep neural networks, Neurocomputing 241 (2017) 81–89. doi:10.1016/j.neucom.2017.02.029.

[16] H. Zheng, J. Xie, Z. Jin, Heteroscedastic sparse representation based classification for face recognition, Neural processing Letters 35 (3) (2012) 233–244. doi:10.1007/s11063-012-9214-4.

[17] M. Tan, G. Pan, Y. Wang, Y. Zhang, Z. Wu, L1-norm latent svm for compact features in object detection, Neurocomputing 139 (2014) 56–64. doi:10.1016/j.neucom.2013.09.054.

[18] J. Duchi, Y. Singer, Efficient online and batch learning using forward backward splitting, Journal of Machine Learning Research 10 (Dec) (2009) 2899–2934.

[19] D. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[20] Y. Jin, B. Sendhoff, Pareto-based multiobjective machine learning: An overview and case studies, IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 38 (3) (2008) 397–415. doi:10.1109/tsmcc.2008.919172.

[21] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197. doi:10.1109/4235.996017.

[22] L. Fei-Fei, R. Fergus, P. Perona, Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories, Computer vision and Image understanding 106 (1) (2007) 59–70. doi:10.1109/cvpr.2004.383.

[23] S. Lazebnik, C. Schmid, J. Ponce, Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on, Vol. 2, IEEE, 2006, pp. 2169–2178. doi:10.1109/cvpr.2006.68.

[24] A. Coates, A. Ng, H. Lee, An analysis of single-layer networks in unsupervised feature learning, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 215–223.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE, 2009, pp. 248–255. doi:10.1109/cvprw.2009.5206848.

[26] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.

[27] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on, IEEE, 2008, pp. 413–422. doi:10.1109/icdm.2008.17.

[28] N. Kouiroukidis, G. Evangelidis, The effects of dimensionality curse in high dimensional knn search, in: Informatics (PCI), 2011 15th Panhellenic Conference on, IEEE, 2011, pp. 41–45. doi:10.1109/pci.2011.45.

Table 4: Comparisons of ADAE and ADAE-SGL

	Without Regularization					With Regularization				
	Precision	Recall	F1	Neuro Sparsity	weight sparsity	Precision	Recall	F1	Neuro Sparsity	weight sparsity
Caltech-101	1.0000	1.0000	1.0000	100%	100%	1.0000	1.0000	1.0000	40.10%	44.45%
Scene_data	0.9722	0.9780	0.9751	100%	100%	0.9741	0.9760	0.9750	39.37%	45.45%
STL-10	0.9219	0.9441	0.9299	100%	100%	0.9109	0.9411	0.9252	39.79%	44.00%
ImageNet	0.8709	0.9401	0.9050	100%	100%	0.8651	0.9442	0.9029	37.98%	41.49%