Learning Discriminative Reconstructions for Unsupervised Outlier Removal

Yan Xia[1] Xudong Cao[2] Fang Wen[2] Gang Hua[2] Jian Sun[2]
[1]University of Science and Technology of China
[2]Microsoft Research

Abstract

We study the problem of automatically removing outliers from noisy data, with application for removing outlier images from an image collection. We address this problem by utilizing the reconstruction errors of an autoencoder. We observe that when data are reconstructed from low-dimensional representations, the inliers and the outliers can be well separated according to their reconstruction errors. Based on this basic observation, we gradually inject discriminative information in the learning process of autoencoder to make the inliers and the outliers more separable. Experiments on a variety of image datasets validate our approach.

1. Introduction

Building large training datasets has become critical for many computer vision tasks, especially with the rapid development of deep learning, which is data-hungry. Retrieving images from search engines [22, 5, 14, 25] is one of the most effective solutions. However, retrieved results often contain many outliers that are irrelevant to the query intent, as shown in Fig. 1. Therefore, automatically removing outliers can greatly help us to construct large scale datasets, or at least significantly reduce the required labeling cost.

Automatically removing outliers from unlabeled data operates in an unsupervised mode. For this problem, methods in literature explicitly or implicitly make an assumption that inliers are located in dense areas while outliers are not. The dense areas can be estimated by statistical methods [6, 27, 11], neighbor-based methods [17, 3, 7], and reconstruction-based methods [23, 26]. For example, the methods in [23, 26] compute PCA projections on data, and those having large projection variances are determined as outliers. Since PCA can be interpreted as minimizing the reconstruction error of training data, we treat the methods in [23, 26] as reconstruction-based methods for unsupervised outlier removal.

In this paper, we also use reconstruction error to infer underlying dense areas of data. But different from [23, 26], we adopt the autoencoder [2, 18] to learn reconstructions. Note that there exists works [10, 16, 20] that use autoencoder for a similar but fundamentally different task — novelty detection (or anomaly detection). In novelty detection, training data are all positive, and it is straightforward to train a normal profile using autoencoder. Our work is inspired by but different from these works, since in the problem of unsupervised outlier removal, training data are unlabeled, and it is more challenging to learn models from noisy data.

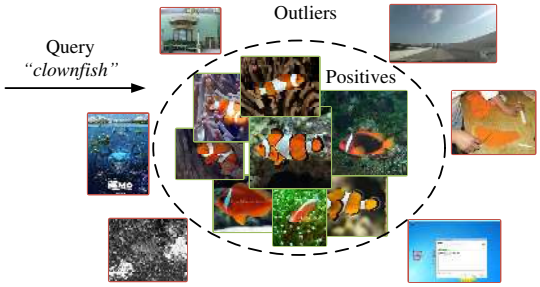


Figure 1. Example images returned by an image search engine when querying "clownfish". Those consist of positive images (relevant to the query intent) and outliers.

As the first step of our work, we find the autoencoder itself is a simple yet effective tool for unsupervised outlier removal. We observe that when data are compressed into low-dimensional representations and then reconstructed by an autoencoder, the inliers (also called positive examples) tend to have smaller reconstruction errors than the outliers. Based on this, one can conveniently identify the images with large reconstruction errors as outliers. We were both surprised and excited in finding that simply thresholding the reconstruction errors can lead to competitive results with most existing methods.

Based on this finding, we further make the reconstruction error even more discriminative. This is achieved by introducing self-learned discriminative information during the learning procedure of an autoencoder. Instead of minimizing the reconstruction errors of *all* data, our idea is to minimize the reconstruction errors *only from the positives*. By doing so, the reconstruction errors of positive data are even smaller while those of outliers are not.

Because the true labels of the positive data are unknown, we implement the above idea in an iterative and "self-paced" manner. In one step, we estimate data as "positive" or "outlier" according to their reconstruction errors. In the other step, we update network parameters in the autoencoder by reducing the errors of the "positives", resulting in more discriminative reconstructions. Discriminative reconstructions benefit labeling and, in turn, accurate labels help with learning reconstructions. These two steps promote each other and we cycle them iteratively until convergence.

In addition to its simplicity and superior performance, our approach also possesses the following merits.

- It is robust to unknown outlier ratios (the proportion of outliers in the entire noisy set). Our method can adaptively handle data with a large range of outlier ratios (*e.g.*, as high as 70 %), without any parameter tuning.
- It is flexible to design. For various image representations (raw pixels, hand-craft features, or pre-learned features), our approach is flexible to involve linear or non-linear neurons in the autoencoder. This is in contrast to most existing methods that deal with non-linear data using kernel functions.
- It is efficient to learn. The learning of our method can be well conducted by mini-batch gradient descent. Each learning iteration only needs a small mini-batch of training samples, so that we can handle large scale data in a streaming fashion and leverage the off-the-shelf parallel computation framework on CPU/GPU.

2. Related Work

Unsupervised outlier removal has been extensively studied both inside and outside the computer vision literature. Statistical methods [6, 27, 11] fit parametric distributions on data, and outliers are identified as those having low probability under the learned distributions. Neighbor-based methods [17, 3, 7] assume positive data have close neighbors while outliers are far from each other. The one-class svm method [21, 7] just treats all training data as positive and the origin point as negative, and learns a max-margin classifier to arbitrate outliers. The key underlying assumption of these methods can be summarized as: positive data are more densely distributed than outliers.

There also exists a category of methods that use reconstruction error for unsupervised outlier removal. The methods in [23, 26] learn PCA projections from data, and those having large projection variances are treated as outliers. We notice reconstruction error has also been used for novelty detection [10, 16, 20], which is a related but fundamentally different task with unsupervised outlier removal: the training data in novelty detection are all positive, while that in unsupervised outlier removal are quite noisy.

Most recently, Liu *et al.*[14] propose an unsupervised one-class learning (UOCL) method that outperforms all the above methods. UOCL utilizes manifold regularization, balanced soft labels and a max-margin classifier. With the use of these ingredients, UOCL has shown state-of-the-art performance for unsupervised outlier removal.

3. Autoencoder for Outlier Removal

In this section, we show the reconstruction error in an autoencoder is discriminative and can be used for unsupervised outlier removal. We further make the reconstruction error more discriminative and propose our method in the following Sec. 4.

3.1. Reconstruction error is discriminative

Suppose we have a set $\{x_1, ..., x_n\}$ where x_i is an image representation (*e.g.*, raw pixels or a feature vector). We apply an autoencoder to first compress x into a low-dimensional intermediate representation and then map it back to a reconstructed copy, $f(x)$. The form of an autoencoder $f(\cdot)$ is a neural network, with hidden linear or non-linear neurons. The reconstruction error of x_i is the squared loss: $\epsilon_i = \|x_i - f(x_i)\|^2$. The autoencoder can be learned by minimizing the average reconstruction error:

$$\mathcal{J}(f) = \frac{1}{n}\sum_{i=1}^{n}\epsilon_i = \frac{1}{n}\sum_{i=1}^{n}\|f(x_i) - x_i\|^2. \qquad (1)$$

How does the learned autoencoder relate to the task of unsupervised outlier removal? Before giving our explanations, we first present our main empirical observation: *the reconstruction errors of positive data are always smaller than that of outliers.* In other words, the error ϵ_i is a good indicator of whether a datum x_i is an inlier or outlier.

In fact, this phenomenon stems from the nature of autoencoder. When an autoencoder compresses noisy data into low-dimensional representations, it cannot well reconstruct *every* datum because the low-dimensional intermediate layer(s) performs like an information bottleneck. Therefore, in order to minimize the overall reconstruction error, an autoencoder has to find the representations that can capture statistical regularities of training set [1]. In the case of removing outliers from noisy images, positive data are image features from a same semantic concept, while outliers are scattered. So the positives have more regularities in their distribution. Therefore, they are more likely to be reconstructed well, resulting in relatively smaller reconstruction errors.

Besides the above empirical explanation, we further study the reason why an autoencoder can produce discriminative error by looking into its learning procedure.

Effective gradient magnitude. In an autoencoder, network

parameters can be updated by gradient descent:

$$\bar{g} = \frac{d\mathcal{J}}{df} = \frac{1}{n}\sum_{i=1}^{n} g_i = \frac{2}{n}\sum_{i=1}^{n}(f(x_i) - x_i), \qquad (2)$$

where $g_i = 2(f(x_i) - x_i)$ is the gradient contributed by x_i.

The above gradient shows that, when an autoencoder updates its parameters, the gradient is averaged over all training data. Therefore, for each single datum, the overall gradient ($\bar{g}$) is different from the gradient on this point (g_i). This means an autoencoder does not try to reduce the error of *every* single datum, but the overall error.

Then we wonder how much the error is reduced for a single datum. This can be measured by the effective gradient magnitude, *i.e.*, the projection of the overall gradient on the direction of this datum's gradient:

$$g_i^{\text{effect}} = \frac{< g_i, \bar{g} >}{|g_i|} = |\bar{g}| cos\theta(g_i, \bar{g}), \qquad (3)$$

where $\theta(g_i, \bar{g})$ is the angle between the two vectors.

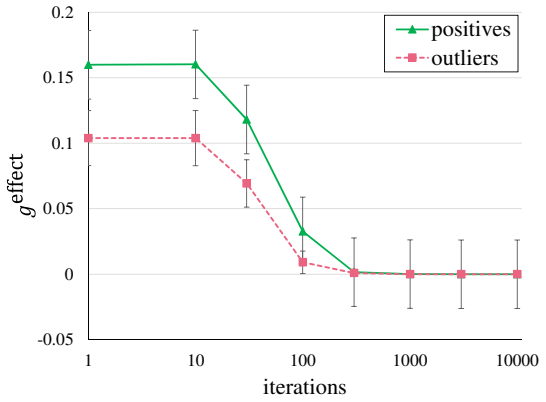
On noisy training set, the overall gradient is more likely to be dominated by the positives. First, outliers are usually not as many as the positives in real tasks. Second, even though the outliers are more than half, positive data still have chances to dominate the gradient. This is because outliers are arbitrarily scattered throughout the feature space, resulting in counterbalancing gradient directions; while positive data are densely distributed, and their gradient directions are relatively more consistent. However, the positives will not occupy the dominant position when there are too many outliers in the noisy set.

In case positive data dominate the direction of the overall gradient, the angle $\theta(g_i, \bar{g})$ for a positive datum is more often smaller than it is for an outlier, leading to larger effective gradient magnitude g_i^{effect}. In other words, the overall gradient puts more efforts on reducing the errors of positive data. Therefore, an autoencoder trained on noisy data is more likely to reconstruct the positives better.

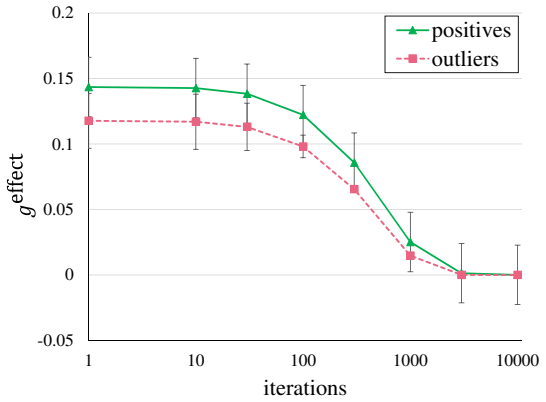
An example case. We validate the above reasoning by experiments. To simulate a noisy dataset, we use images with the same semantic meanings downloaded from the ImageNet website[1] as the positives, and randomly sample outliers from 1.2 million ILSVRC2012 [19] images. Images are represented by 2048-dimensional deep learning features extracted from a pre-trained CNN model.[2] The concept "car wheel", which contains 1,981 positive images and equal number of outliers, is used as an example set here, as well as the rest illustrative experiments in this paper.

[1] http://image-net.org/

[2] We use the network introduced in [12], which contains five convolutional layers and two fully connected layers. The 2048-dimensional outputs of the first fully connected layer are used as image features.



(a) train auto-encoder on a set with 50% outliers



(b) train auto-encoder on a set with 70% outliers

Figure 2. The effective gradient magnitude (Equation 3) in an autoencoder, averaged over positive training data and outliers respectively. The training set has (a) 50% and (b) 70% outliers.

We train an autoencoder having one hidden layer with 32 linear neurons on this example set. In the training procedure, we compute the effective gradient magnitude on each datum, and average them on positive data and outliers respectively. The result is shown in Fig. 2 (a). We can see positive data always have larger effective gradient magnitudes than outliers, until training converges.

The above result is on a set with half outliers. We further show the result on a set with 70% outliers[3] in Fig. 2 (b). We can see even when outliers are more than half, the autoencoder still put more efforts on reducing the errors of positive data, showing its robustness to outlier ratios.

By putting more efforts on reducing the errors of positive data, an autoencoder is prone to reconstruct the positives better. We also show an example case here. On the example set with half outliers, we train auto-encoders with various intermediate dimensionalities, and plot the results in Fig. 3. We can see the average reconstruction error of positive data is consistently lower than that of outliers, no matter what the intermediate dimensionality is.

[3] The positive data are the same as in previous example, and outliers are still randomly sampled from ILSVRC2012 images, at a ratio of 70%.

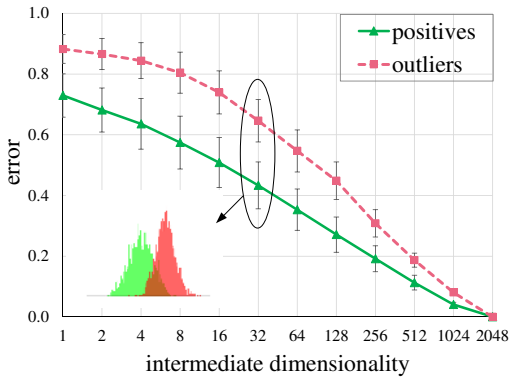


Figure 3. The average reconstruction errors in an autoencoder on the example set with half outliers. The thumbnail shows the error distributions when the intermediate dimensionality is 32.

3.2. Autoencoder — a baseline method

Based on the discriminative reconstruction error of an autoencoder, it is straightforward to label data having large errors as outliers. Specifically, after an autoencoder is learned, we can apply any clustering algorithm (such as k-means [15]) to partition reconstruction errors into two clusters. Samples in the cluster with a large average error are identified as outliers.

4. Learning Discriminative Reconstructions

4.1. Basic idea

Although the reconstruction error is discriminative, the error distributions of positives and outliers may still be significantly overlapped, especially at high outlier ratios (as shown in Fig. 4 (a)). In these cases, simply thresholding reconstruction errors will lead to many misclassifications.

To deal with this problem, we propose to only reconstruct the positives. If we can do this, the overall gradient in an autoencoder is only contributed by the positives. Then the reconstruction errors of positive data are minimized while those of outliers are not, leading to more separable error distributions. We verify this idea with an experiment. Suppose we have groundtruth labels of the sample set. We train an autoencoder only on the positive part and plot the error distributions in Fig. 4 (b). From this figure we can tell, by only reconstructing the positive data, the autoencoder produces more separable error distributions, compared with the one trained on all data (Fig. 4 (a)).

However, it is a chicken-and-egg problem because we do not know which part of the data is positive or outlier at the very beginning. Nevertheless, we can tackle this problem by alternating the following two steps:

- *Discriminative labeling*: estimate positives from entire noisy data based on current reconstruction errors.
- *Reconstruction learning*: update the autoencoder to reduce reconstruction errors of the "positives" and enlarge the separability of error distributions.

As a preview, Fig. 4 (c) shows the error distributions by iterating the above two steps. We can see the reconstruction errors of positive data and outliers become much more separable with this method and the separability is almost as good as that obtained by using the groundtruth (Fig. 4 (b)).

4.2. Algorithm

As mentioned, our algorithm works by iterating two steps: discriminative labeling and reconstruction learning. We outline our algorithm in Fig. 5 and detail it as follows.

Discriminative Labeling. This step aims to estimate data labels, *i.e.*, classify x_i into either positive label $y_i = 1$ or outlier label $y_i = 0$, according to its reconstruction error ϵ_i. We achieve this goal by optimizing the following objective:

$$\min_{y} \; h = \frac{\sigma_w}{\sigma_t}, \tag{4}$$

where σ_w is the summation of within-class variances which is desired to be small, and σ_t is the total variance. Specifically, $\sigma_w = \sum_{y_i=1}(\epsilon_i - c^+)^2 + \sum_{y_i=0}(\epsilon_i - c^-)^2$ and $\sigma_t = \sum(\epsilon_i - c)^2$, where c^+, c^- and c are the mean reconstruction errors of positive data, outliers, and the entire noisy data, respectively. We normalize σ_w by the total variance σ_t, in order to make h dimensionless.

Since ϵ_i is scalar, labeling data can be translated into sorting the reconstruction errors and then finding a cut-off threshold. Therefore, the objective in Equation 4 can be trivially optimized by linearly scanning an optimal threshold. Then we label each sample x_i by comparing its reconstruction error ϵ_i against the obtained optimal threshold.

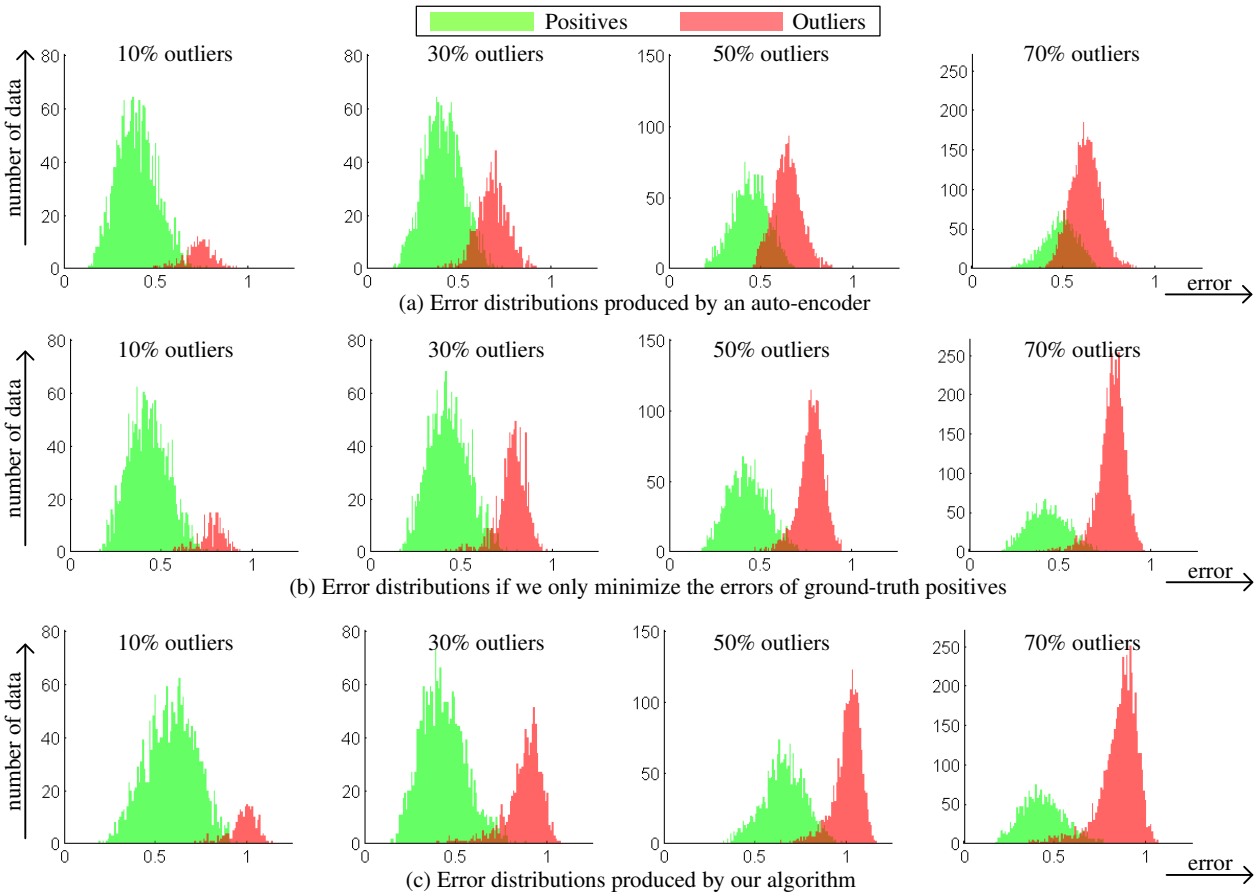
Reconstruction Learning. When data labels are provided by the labeling step, we want to reduce the reconstruction errors of the positives to make error distributions more separable. With this in mind, we design the loss function as:

$$\mathcal{L}(f) = \frac{1}{n^+} \sum_{y_i=1} \epsilon_i + \lambda h, \tag{5}$$

where the first term measures the averaged reconstruction error of data labeled as positive, and the second term (h is given in Equation 4) represents the separability of error distributions. The parameter λ controls the tradeoff between the two terms.[4]

The loss $\mathcal{L}$ can be reduced by gradient descent, which can be easily integrated into the back-propagation procedure of the autoencoder. The gradient of $\mathcal{L}$ with respect to

[4]We find the final solution is insensitive to the value of the parameter λ. So we fix it to 0.1 for the sake of simplicity in all our experiments.



(a) Error distributions produced by an auto-encoder

(b) Error distributions if we only minimize the errors of ground-truth positives

(c) Error distributions produced by our algorithm

Figure 4. Reconstruction error distributions on datasets with various outlier ratios. Data are reconstructed by: (a) an autoencoder, (b) an autoencoder that only reconstructs groundtruth positives and (c) our algorithm.

the network f is given by:

$$\begin{aligned}\frac{d\mathcal{L}}{df} &= \sum_{i=1}^{n} \frac{\partial \mathcal{L}}{\partial \epsilon_i} \times \frac{d\epsilon_i}{df} \\ &= \sum_{i=1}^{n} (\frac{y_i}{n^+} + \lambda \frac{\partial h}{\partial \epsilon_i}) \cdot 2(f(x_i) - x_i).\end{aligned} \tag{6}$$

This gradient has clear intuitions. The factor of the first term, $\frac{y_i}{n^+}$, means only positive data are considered when learning reconstructions ($y_i = 0$ for outliers). The factor of the other term, $\frac{\partial h}{\partial \epsilon_i}$, explicitly makes error distributions more separable. Both terms help to enlarge the separability of error distributions, leading to more discriminative reconstructions.

These two steps, discriminative labeling and reconstruction learning, are performed alternatively until the labels stay unchanged.[5] In this manner, network parameters in f are updated gradually and data labels are refined step by step. Finally, we perform the discriminative labeling step on all data to finish the algorithm.

[5]We did not observe any divergence or oscillation in our experiments.

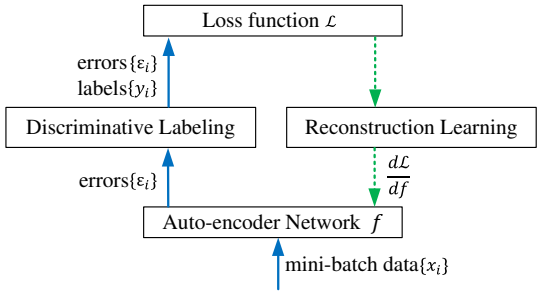


Figure 5. Flow chart of our algorithm.

4.3. Discussions

Learning procedure. To illustrate the algorithm, we plot the evolution of reconstruction errors on the example set in Fig. 6. Some key stages in the learning procedure are:

- At the beginning, network parameters are randomly initialized, and positive data and outliers have nearly the same error distributions (Fig. 6 (a)). So in this stage, reconstruction error is not discriminative, and the labeling step can only produce noisy labels: false positives are

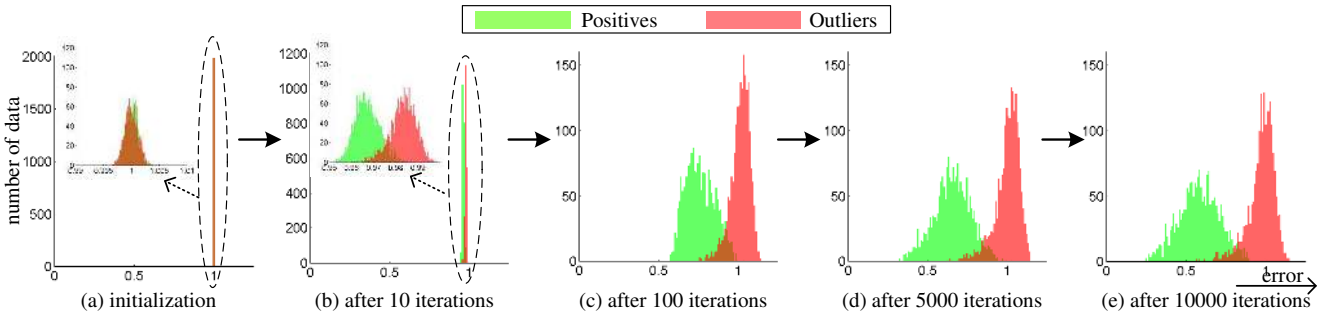


Figure 6. The evolution of reconstruction errors in the learning procedure of our algorithm.

roughly in the same proportion to the outliers in the entire training set. This is the worst case for labeling. With noisy labels, our algorithm works in the same manner as an autoencoder, *i.e.*, tries to reconstruct all training data.

- After a few iterations, error distributions start to be separable (Fig. 6 (b)) due to the discrimination power of autoencoder, as discussed in Sec. 3. The labeling step begins to produce more accurate labels, *i.e.*, the percentage of false positives is less than the outlier ratio. Consequently, the reconstruction step updates network parameters towards the direction of better reconstructing positives.
- The better reconstructions, the smaller errors for positives. Meanwhile, most outliers are not well reconstructed and even be pushed away by the discriminative term h in our loss function. Then the reconstruction error becomes more discriminative (Fig. 6 (b) to Fig. 6 (c)).
- After positives and outliers are already well separated, more iterations only make them more distant, but not change their labels (Fig. 6 (d) to Fig. 6 (e)). In this situation, data labels become stable and iteration ends.

Properties. As can be seen from the learning procedure, our algorithm does not require a pre-specified outlier ratio. Actually, it is quite robust to outlier ratios, as an advantage inherited from the autoencoder (discussed in Sec. 3). Moreover, as long as the autoencoder can provide discriminative reconstruction errors, our algorithm can make them more discriminative. Fig. 4 (a)(c) compare the results of an autoencoder without/with discriminative reconstruction, at various outlier ratios. From this figure we can see: first, the discriminative reconstruction can significantly improve the autoencoder; second, even at a high outlier ratio of 70%, our algorithm still produces decent results.

Note that our algorithm does not impose any restriction on the design of the autoencoder. Therefore, our algorithm is flexible for various data representations. For example, when input data are raw images, the network can be designed to have multiple hidden layers with non-linear neurons(as in [9]); for deep learning features, we can even design f as a single-hidden-layer network with linear neurons.

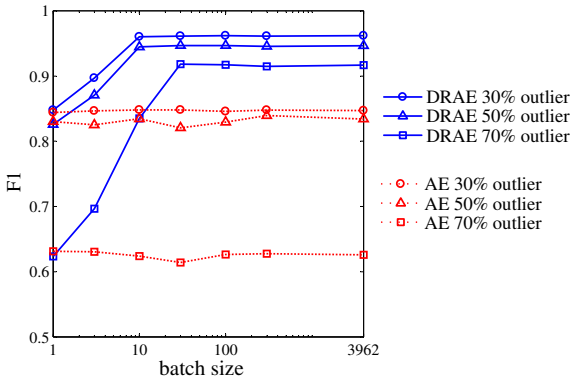


Figure 7. Labeling accuracy *vs.*batch size. AE stands for the autoencoder method and DRAE stands for our algorithm.

This flexibility is not possessed by some existing methods, where non-linearity is only involved by kernel functions.

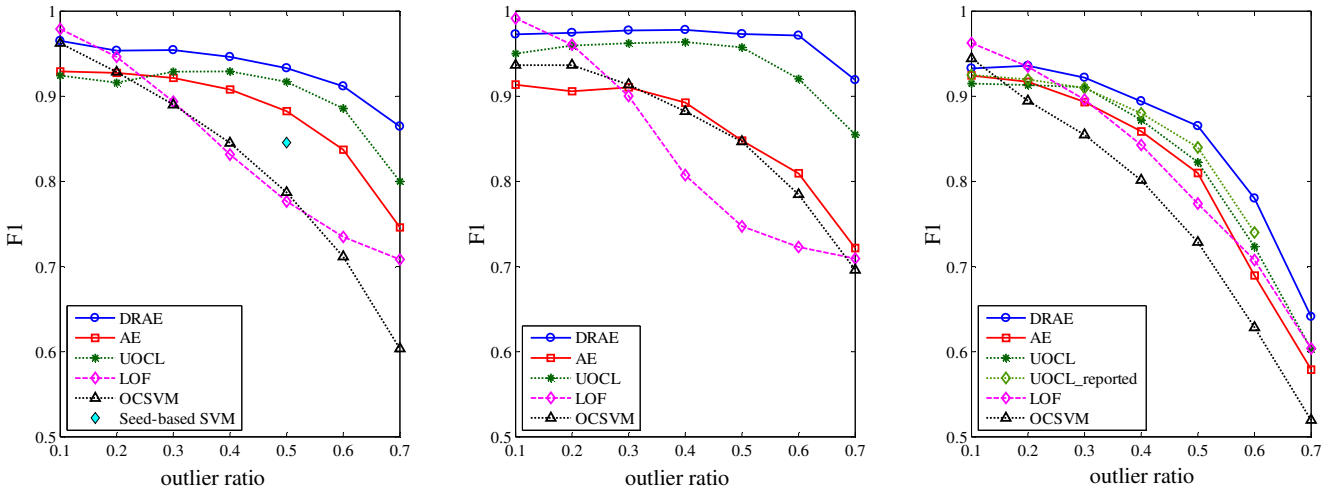
Last, mini-batch gradient descent makes our algorithm efficient. As shown in Fig. 7, on the example set with 3,962 samples, we can safely use batch size as small as 100 for various outlier ratios.[6] Thus, we can use the highly optimized CPU/GPU computation framework and are ready to handle large scale data.

5. Experiments

5.1. Datasets and features

- **ImageNet-16** [25]. This dataset consists of images in 16 semantic concepts from the ImageNet website, and each concept contains about 5,000 images on average. For each concept, we randomly sample outliers from 1.2 million ILSVRC2012 images [19] with the proportion from 10% to 70%.
- **Caltech-101** [8] was used for the task of unsupervised outlier removal by [14]. In this dataset, 11 concepts with 100-800 positive images are used in the experiments. For

[6] In Fig. 7, the labeling accuracy is measured by $\text{F1}=\frac{2\cdot\text{Precision}\cdot\text{Recall}}{\text{Precision}+\text{Recall}}$. Precision is the proportion of correctly identified positives relative to the number of identified positives, and recall is the proportion of correctly identified positives relative to the total number of positives in one set.



(a) ImageNet-16: deep learning feature (b) Caltech-101: deep learning feature (c) Caltech-101: LLC feature

Figure 8. Comparisons on (a) ImageNet-16 with deep learning features, (b) Caltech-101 with deep learning features and (c) Caltech-101 with LLC features. F1 scores are averaged over all concepts in one dataset.

dataset	ImageNet-16			Caltech-101			Caltech-101		
feature	deep learning feature			deep learning feature			LLC feature		
measure	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
UOCL	0.899	**0.940**	0.917	0.950	0.968	0.957	0.755	0.916	0.822
DRAE	**0.927**	**0.940**	**0.933**	**0.970**	**0.981**	**0.973**	**0.790**	**0.931**	**0.865**

Table 1. Comparisons between UOCL and DRAE when outlier ratio is 50% on each dataset. Scores are averaged over all concepts.

each concept, outliers are randomly sampled from all the other concepts with a proportion from 10% to 70%.

- **MNIST** [13] contains 60,000 handwritten digits from 0 to 9. For each category of digit, we simulate outliers as randomly sampled images from other categories.
- **Google-30** [14] has 30 concepts (about 500 images per concept) of images crawled from search engines. Outliers in this set are real images that are irrelevant to a corresponding textual query.

In most of our experiments, images are represented by features extracted from a seven-layer CNN [12], which is pre-trained on ImageNet. We use the 2048-dimensional outputs of the first fully-connected layer as image features. On Caltech-101, we also extract locality-constrained linear codes (LLC) [24] for the comparison with existing methods. On MNIST, we directly use raw pixels as inputs.

5.2. Methods to be compared

Since our method is to learn discriminative reconstructions in an autoencoder, we call our method *DRAE*. We compare the following methods with it:

- Autoencoder (AE). This is our baseline method as introduced in Sec. 3. Notably, with linear neurons and squared loss, an autoencoder learns the same subspace as PCA [2]. So we omit the comparisons with using PCA to learn reconstructions.
- Unsupervised one-class learning (UOCL) [14]. We reproduce the results of UOCL by strictly following the details (*e.g.*, using Gaussian kernels and the number of neighbors) in [14].
- One-class svm (OCSVM) [21]. We use the implementation of LibSVM [4]. In this method, the amount of identified outliers is proportional to a parameter v. Since the outlier ratio is unknown before training, we just set v based on ground-truth.
- Seed-based SVM [25]. We directly compare the results reported in [25].
- Local outlier factor (LOF) [3]. In this method, we set the neighbor number as 10% of the dataset size and use groundtruth outlier ratio to determine outliers as those with highest outlier factors.

5.3. Results on ImageNet-16/Caltech-101/Google-30

On these datasets, images are represented by deep learning features. The architecture of autoencoder has one hidden layer with 32 linear neurons, and the batch size is 100 in AE and DRAE. F1 score is used for evaluating accuracy. Comparisons on ImageNet-16 and Caltech-101 are shown in Fig. 8, from which we have the following observations.

- Comparing AE with LOF and OCSVM, we can see even a naive version of autoencoder outperforms LOF and

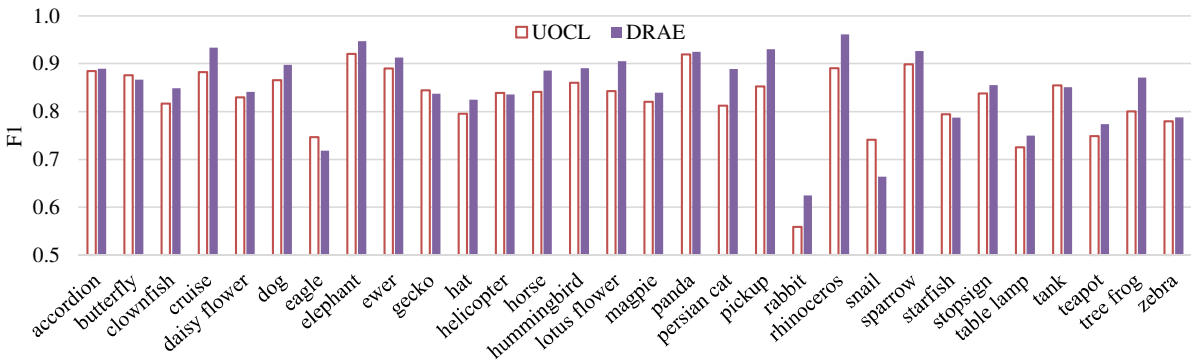


Figure 9. Comparisons on each concept in Google-30 dataset

OCSVM. Note that the outlier ratio is even pre-given for these two methods. This indicates utilizing reconstruction error for unsupervised outlier removal is vaild.

- DRAE significantly outperforms AE for all outlier ratios on both datasets. This validates our idea of learning more discriminative reconstructions.
- DRAE also outperforms the state-of-the-art UOCL method, especially at high outlier ratios.
- Even though the outlier ratio is as large as 70%, DRAE still achieves a high F1 score (0.865), showing its robustness for outlier ratios.

On Caltech-101, we also use LLC features for a fair comparison with UOCL. From Fig. 8 (c), the performance of UOCL in our implementation is very close to that reported in [14]. The minor difference may come from different codebooks and parameters in extracting LLC features, and different mixtures of outliers.

We further report the average precision and recall values of DRAE and its best competitor UOCL in Table. 1, on datasets with 50% outliers. It can be seen DRAE outperforms UOCL in various measures.

Fig. 9 shows the result on Google-30 dataset. DRAE outperforms UOCL on 23 concepts out of the total 30 concepts, and the average F1 scores of the two methods are 0.849 and 0.826, respectively.

5.4. Results on MNIST

For the raw pixel inputs of MNIST, we use four hidden layers (784-1000-500-250-30) and sigmoid neurons [9] in the autoencoder. The performance is shown in Fig. 10 (a), where DRAE still achieves the best performance.

Note that UOCL performs poor on the raw pixel input. We argue that this is because the non-linearity in UOCL solely relies on the applied kernel functions. In contrast, the non-linearity in DRAE is introduced by the powerful neural network. To verify this, we train an autoencoder with the same architecture as before using the whole training set of MNIST, and use the outputs of the fourth hidden layer (30-dimensional) as image features. Fig. 10 (b) shows the new result.[7] Here, DRAE has one linear hidden layer, while OCSVM and UOCL adopt Gaussian kernels. As can be expected, the gaps between UOCL/OCSVM and DRAE become smaller due to the using of better non-linear features. This demonstrates that the autoencoder can simultaneously play two roles: feature learning and reconstruction.

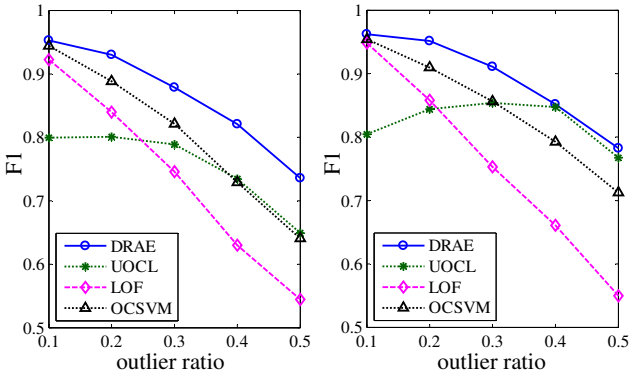


Figure 10. Comparisons on MNIST dataset when (a) images are represented by raw pixels and (b) images are represented by outputs of the fourth hidden layer in a pre-trained autoencoder.

6. Conclusion

In this paper, we focus on the problem of automatically removing outliers from noisy data. We present that an autoencoder itself is a simple yet effective tool for the unsupervised outlier removal task. Moreover, we further make this tool more powerful by learning more discriminative reconstructions. Our method, which is robust to outlier ratios, flexible to design, and efficient to learn, would benefit constructing large scale datasets in the further.

[7] In Fig. 10 (b), the F1 of UOCL at 10% outlier ratio is lower than it is at 40% outlier ratio. This is because in UOCL, the soft labeling balances the amount of identified positives and outliers, but in turn this can be a disadvantage when groundtruth labels are not balanced. This disadvantage could be remedied by its manifold regularization, but not the case here.

References

[1] Y. Bengio. Learning deep architectures for ai. *Foundations and trends in Machine Learning*, 2009. 2

[2] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological cybernetics*, 59(4-5):291–294, 1988. 1, 7

[3] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *International Conference on Management of Data*, 2000. 1, 2, 7

[4] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011. 7

[5] X. Chen, A. Shrivastava, and A. Gupta. Neil: Extracting visual knowledge from web data. In *ICCV*, 2013. 1

[6] E. Eskin. Anomaly detection over noisy data using learned probability distributions. In *ICML*, 2000. 1, 2

[7] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo. A geometric framework for unsupervised anomaly detection. In *Applications of data mining in computer security*. Springer, 2002. 1, 2

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *CVPR*, 2004. 6

[9] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006. 6, 8

[10] N. Japkowicz, C. Myers, M. Gluck, et al. A novelty detection approach to classification. In *International Joint Conference on Artificial Intelligence*, 1995. 1, 2

[11] J. Kim and C. D. Scott. Robust kernel density estimation. *The Journal of Machine Learning Research*, 2012. 1, 2

[12] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 3, 7

[13] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998. 7

[14] W. Liu, G. Hua, and J. R. Smith. Unsupervised one-class learning for automatic outlier removal. In *CVPR*, 2014. 1, 2, 6, 7, 8

[15] J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967. 4

[16] L. Manevitz and M. Yousef. One-class document classification via neural networks. *Neurocomputing*, 2007. 1, 2

[17] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD Record*, 2000. 1, 2

[18] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 1988. 1

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv:1409.0575*, 2014. 3, 6

[20] M. Sakurada and 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*, page 4. ACM, 2014. 1, 2

[21] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 2001. 2, 7

[22] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. *TPAMI*, 2011. 1

[23] M.-L. Shyu, S.-C. Chen, and K. Sarinnapakorn. A novel anomaly detection scheme based on principal component classifier. *IEEE Foundations and New Directions of Data Mining Workshop*, 2003. 1, 2

[24] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, pages 3360–3367, 2010. 7

[25] Y. Xia, X. Cao, F. Wen, and J. Sun. Well begun is half done: generating high-quality seeds for automatic image dataset construction from web. In *ECCV*, 2014. 1, 6, 7

[26] H. Xu, C. Caramanis, and S. Mannor. Outlier-robust pca: the high-dimensional case. *Information Theory, IEEE Transactions on*, 2013. 1, 2

[27] K. Yamanishi, J.-I. Takeuchi, G. J. Williams, and P. Milne. On-line unsupervised outlier detection using finite mixtures with discounting learning algorithms. In *Knowledge Discovery and Data Mining*, 2000. 1, 2