Handling Label Noises in Dataset

Ziming Ma Zhiyuan Zhao Xiaochen Zong

Abstract

Label noises are common in datasets, and they have negative effects on learning of various models. Our work focuses on testing the influence on training of noises, reducing such influence and explaining why certain models are robust to such noises. We designed experiments to test the influence of a different proportions of noises on a models, including fully connected neuron network and AlexNet and give explanations; developed a few methods to clean noises, including using neuron network and SVM to distinguish and clear noisy labels and stopping training at the appropriate time to reduce negative effects.

Introduction

Datasets are paramount for training machine learning models. However, in real life, datasets of clean noisy labels are rare, and label noises caused by human error are quite common. Meanwhile, training with noises also illustrates some rules in machine learning. In this study, we focus on handling these noises, building robust and accurate models, as well as trying to interpret some noticeable facts.

Related work

Rolnick et al. (2018) did thorough experiments to test the influence and demonstrated remarkably high test performance after training on corrupted data from various datasets. For example, on MNIST they obtained test accuracy above 90 percent after the number of false labels is 100 times of correct labels. They credit this surprising outcome to the fact that for large batch sizes increasing proportion of false labels is merely decreasing the magnitude of the true gradient during gradient descent, rather than altering its direction, since false labels cancel out each other in training. This also leads to their suggestion that choosing larger batch sizes and downscaling the learning rate to mitigate noises' effect. SVM can be used to cleanse samples with noisy labels from the dataset, which is shown by [7, 8]. Our experiments to cleanse label noise will reproduce the methods used in those paper, and we will try to make some improvement based on those methods.

Test of influence

We performed three individual experiments to test the effects of noisey labels on learning.

1. We add exogenous noisy labels to MNIST. In this case, data points in original training set are not altered. For the model we chose 2-fully-connected-layer neuron network, which we call Simple NN. We recorded the ratio of noisy labels to original training data in Figure 1. We assume original MNIST are nearly 100% correctly labeled.

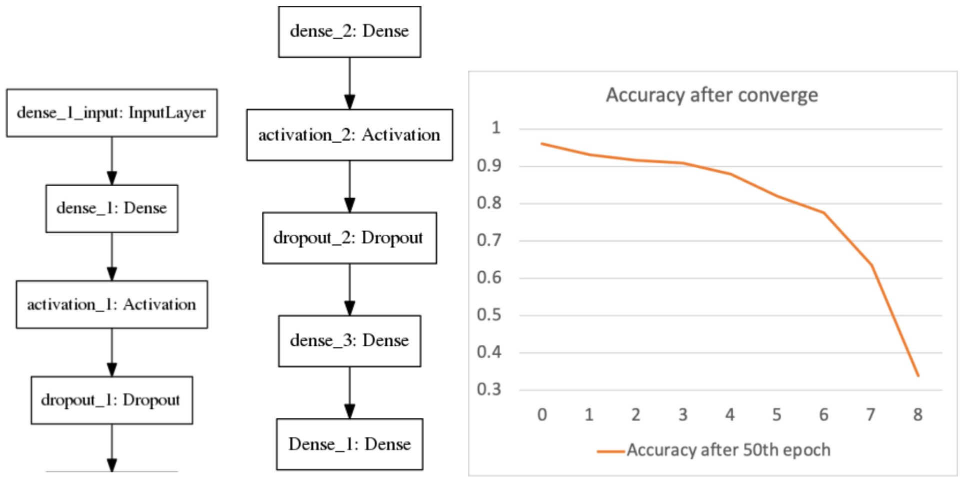


Figure 1: Architecture of Simple NN and its accuracy curve as the ratio of noisy labels increases.

2. We altered certain proportion of labels in MNIST. For the model we still chose Simple NN. We recorded the proportion of noisy labels in training data in Figure 2.
3. Still we altered certain proportion of labels in MNIST. For the model we still chose AlexNet. We recorded the proportion of noisy labels in training data in Figure 3.

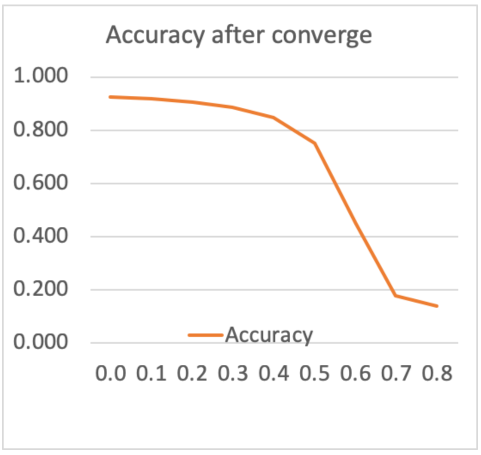


Figure 2: accuracy curve of Simple NN as the proportion of noisy labels increases

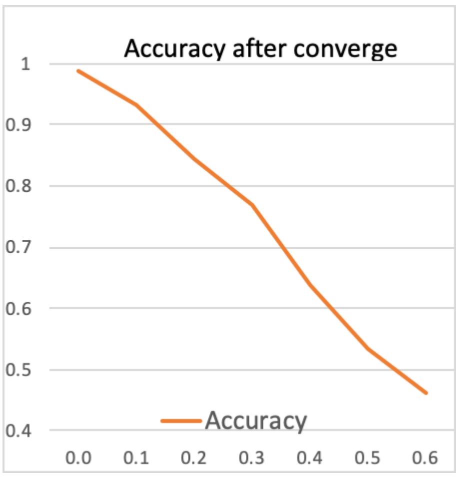


Figure 3: accuracy curve of AlexNet as the proportion of noisy labels increases.

From experiment results we find that generally, the larger proportion of noisy labels in training set, the worse learning outcome. Meanwhile, if we compare the performance of Simple NN to AlexNet under the same experiment layout, we find while Simple NN retained the accuracy over 80 percent when the noises proportion reached 0.4, AlexNet's accuracy is linearly related to the proportion.

A possible explanation of this comparison is that complex models tend to overfit, i.e. learn details in noisy examples and distinguish them from correct examples, while simpler models have better generalization ability.

Explanation of high accuracy under large noises proportion

To give an explanation of why models retained high accuracy when 30 percent of data points in original dataset are mislabeled, or when 3 times of noise labels are added to training data, we calculated the distribution of labels for a given type of picture.

When noisy labels take up 30% of training data, for a particular kind of pictures, e.g. pictures with character "2" on it, the distribution of their labels is shown in Figure 4.

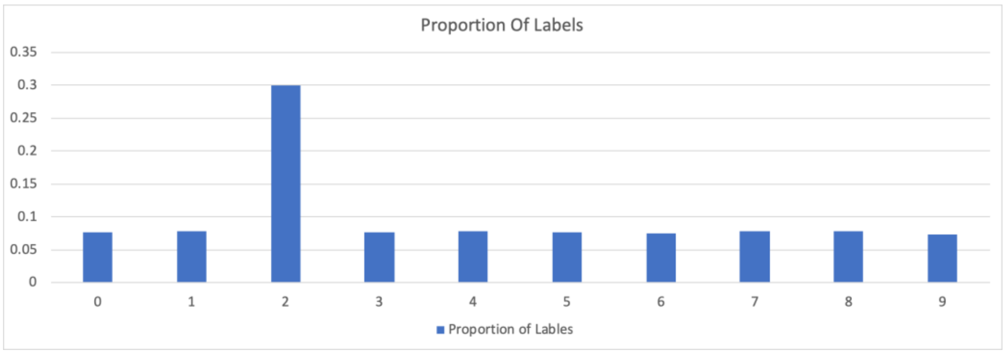


Figure 4: the distribution of labels for a given type of picture when 30 percent of data points in original dataset are mislabeled.

When noise labels 3 times original data are added to training set, for a particular kind of pictures, e.g. pictures with character "2" on it, the distribution of their labels is shown in Figure 5.

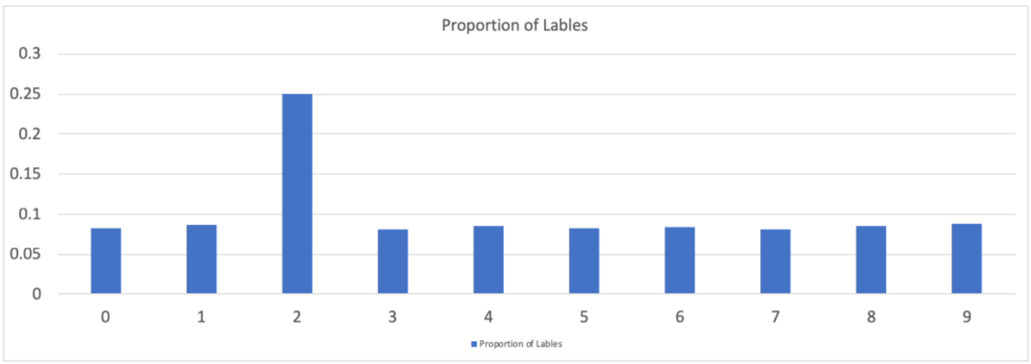


Figure 5: the distribution of labels for a given type of picture when noise labels 3 times original data are added to training set.

From the plots, we observe that in both cases, correct labels are still dominant in distributions. Considering the "cancelling out" effect[3] of noisy labels, this domination give models possibility to learn correct labels of given picture.

Perhaps the concept of information entropy gives a better description of this observation:

$$H_s = \sum_{i=1}^{n} p_i I_e = -\sum_{i=1}^{n} p_i \log_2 p_i$$

It can be proved that, to maximize information entropy H, we need to include as many classes in the dataset as possible, while keeping the distribution of each label uniform. MNIST achieves this. In MNIST, there are ten possible outcomes for one picture. The more outcomes, the lower proportion of incorrect examples in one label. This works like “diluting” incorrect labels. During training, these false labels would “cancel out” each other.

Handling noisey labels

We developed four methods to handle noisy labels in datasets.

1. Use simpler models as Pre-trained model to identify false labels.

Assume we have MNIST, but 20% of the total 70000 examples are corrupted by uniformly distributed label noise. We train Simple NN on training set of 60000, reaching 90.6% accuracy. Then we use it to predict examples in test set of 10000, in which 2000 are noises. For those have different labels from our prediction, we regard them as noisy labels and pick them out.

As a result, 2613 examples are picked out from the test set of 10000, very few of which are correctly labeled but wrongly picked. A rough estimation is that 99 percent of false labels are distinguished by the model. This is a pretty decent outcome.

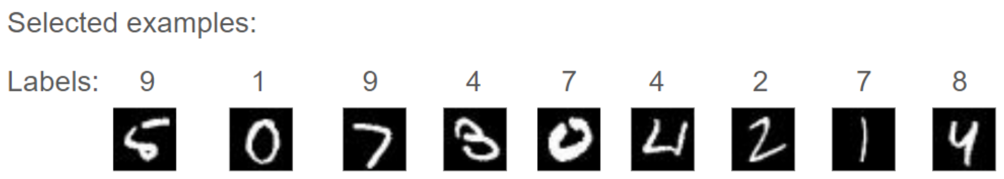


Figure 6: Selected examples picked out by the simpler model.

2. Use simpler models to find ambiguous training pictures.

With the trained Simple NN in last experiments, we get its prediction of probabilities of each labels(0~9) for the entire training set of 60000. If a example has the largest probability < 0.38 while the second largest > 0.1, we regard it as ambiguous. As a result, 130 example were picked out.

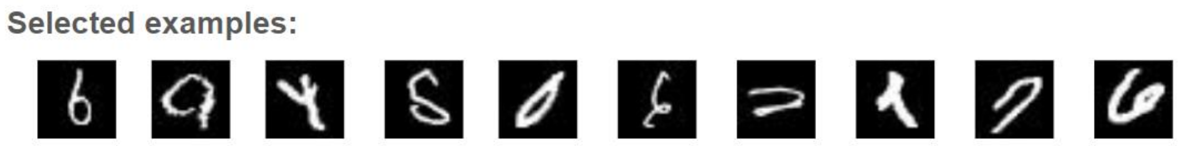


Figure 7: Selected examples picked out as ambiguous by the simpler model.

As we can see, most pictures picked by model are indeed ambiguous, even hard to recognize by human eyes.

3. Utilize SVM to clean mislabeled training data.

Using SVM combining with human review is an efficient way to cleanse training data that contains label noise. The method is based on the hypothesis, mislabeled examples are more likely to be chosen as support vectors, which is supported by [9]. The SVMs used in the experiments are implemented by using the scikit-learn python machine learning library.

3.1 Clean Noisy Label by Two-class SVM (TCSVM)

This experiment is based on the following algorithm, which is given by [8]:

1) Train a binary SVM classifier using the training examples.
2) Collect and review all the support vectors.
3) Correct the labels of the mislabeled examples in the support vectors.
4) Repeat Steps 1-3 until no further noisy label is detected.

The dataset we use in this experiment contains 4000 training samples which equally come from class 4 and class 9 of the MNIST dataset. We introduce 10%, 20% and 30% label noise by flipping label (4 ↔ 9). “4” and “9” are two most confused classes in MNIST, this dataset imitates the situation of human error when labeling. The result shows that in all the 3 noise rates, more than 99% samples with label noise can be corrected after 3 iterations, and most of them can be detected in the first iteration. The results are according with [8], and they are listed in the appendix of this paper.

3.2 Clean Noisy Labels With One-Class SVM (OCSVM)

The algorithm is same as the above one, except that the label noise will be removed instead of being corrected. The dataset we use contains 3600 training samples randomly selected from the class “4” of the MNIST dataset. And we introduce 10%, 20% and 30% uniform label noise to the dataset. This dataset imitates a more general situation of label noise. For the 10% noise rate, we can remove 94.72% label noise by reviewing 50.06% samples of the dataset. With the increasing of the noise rate, an increasing number of samples needed to be reviewed to remove 95% label noise. The results are according with [7], while our performance is a little better. The details will be shown in the appendix of this paper.

The SVM classifier has the property to capture mislabeled samples as its support vector. So, the SVM can be used to detect the label noise. However, because this method needs human reviewing, it is not suitable for cleansing label noise for a large dataset. We tried some experiments to get rid of the reviewing procedure, and the details will be shown in the appendix of this paper.

4. Stop training early to prevent overfitting on complex models(e.g., AlexNet).

As shown in previous experiment, the biggest issue with complex models(AlexNet in this case) when training with noise is overfitting. Here, we are able to prevent it through early stop.

For dataset we used MNIST, in which noisy labels take up 30% and other 70% are correct labels.
First, we configure hyper-parameters in the same way as relatively successful model Simple NN: for the last two fully-connected layer, we choose dense = 500 and dropout = 0.5.

Then we try to stop training at the very time when:

a. Validation accuracy reaches highest. Or:
b. Proportion of correct examples × Validation accuracy = Training Accuracy.

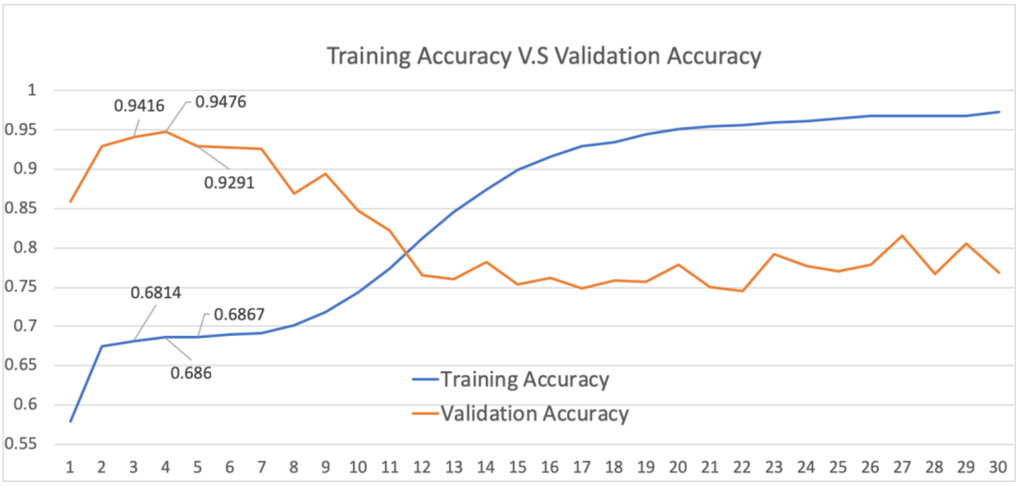


Figure 8: AlexNet's accuracy change on MNIST as training epoch increases.

In this case, when validation accuracy reached highest at 0.9476, Training accuracy reached 0.686. Since 0.9476×0.7 = 0.66332, formula b was roughly satisfied. So at this epoch, it is the right time to stop training. If we continue training, it is clear to see that as training accuracy increases, validation accuracy drops, which is an apparent signal of overfitting.

Conclusion

While larger proportion of noisy labels leads to lower test accuracy, simpler models are more robust to noises than complex models because simpler ones are less likely to overfit. One of the explanation of high accuracy under large amount of noise is among all labels of one training example, correct labels still dominate. We also developed a few methods to handle noisy labels, including using pre-trained simple models to distinguish false labels and ambiguous pictures, cleaning noisy labels with SVM, and stop training early to prevent overfitting on complex models.

Future work

Based on our work, it is possible to develop an iterative procedure to clean noisy labels while building highly accurate complex models: Split the entire training set evenly into into two parts, A and B. Train AlexNet on A and stop training at the right time to gain highest validation accuracy, then use this model to clean B; then train a new AlexNet on cleaned B to gain highest validation accuracy, and use it to clean A, and so on. Finally, within a few iteration, we will get a pretty clean dataset as well as a highly accurate AlexNet model.

Reference

[1] Amnon Drory, Shai Avidan and Raja Giryes, “On the Resistance of Neural Nets to Label Noise”, arXiv:1803.11410 [cs.LG], 30 Mar 2018.

[2] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht and Oriol Vinyals, “Understanding deep learning requires rethinking generalization”, arXiv:1611.03530 [cs.LG], 26 Feb 2017.

[3] David Rolnick, Andreas Veit, Serge Belongie and Nir Shavit, “Deep Learning is Robust to Massive Label Noise”, arXiv:1705.10694 [cs.LG], 26 Feb 2018.

[4] Frédéric Branchaud-Charron, Fariz Rahman, Fariz Rahman, Taehoon Lee, Keras: Deep Learning for humans, https://github.com/keras-team/keras/blob/master/examples/mnist_cnn.py

[5] Feisiqi, Into to Keras: Hello Keras on MNIST-Handwritting Recognition, https://blog.csdn.net/sdust_dx/article/details/80365674

[6] N. Natarajan I. Dhillon P. Ravikumar A. Tewari "Learning with noisy labels" Proc. Neural Inf. Process. Syst. pp. 1196-1204 2013.

[7] Rajmadhan Ekambaram, “Label Noise Cleaning Using Support Vector Machines”, 26 Feb 2017.

[8] Sergiy Fefilatyev, Matthew Shreve, Kurt Kramer, Lawrence Hall, Dmitry Goldgof, Rangachar Kasturi, Kendra Daly, Andrew Remsen, Horst Bunke, “Label-Noise Reduction with Support Vector Machines”, 15 Nov. 2012.

[9] C. Burges. “A tutorial on support vector machines for pattern recognition”. Data mining and knowledge discovery, 2(2):121–167, 1998.

[10] Yao Wang, Influence and recognition of noisy labels in deep learning, https://blog.csdn.net/wangyao_bupt/article/details/77485553

Appendix

Iteration	Total # SV	% Cumulative #SV	Noise Rate	% of noise detected (Accum.)	Total noise SV detected
1	1652	41.30%	400 (10%)	99.50%	398
2	640	41.90%	400 (10%)	99.70%	400
1	2466	61.65%	800 (20%)	97.50%	780
2	576	62.12%	800 (20%)	99.00%	792
1	3072	76.8%	1200 (30%)	97.00%	1164
2	668	77.2%	1200 (30%)	98.00%	1176

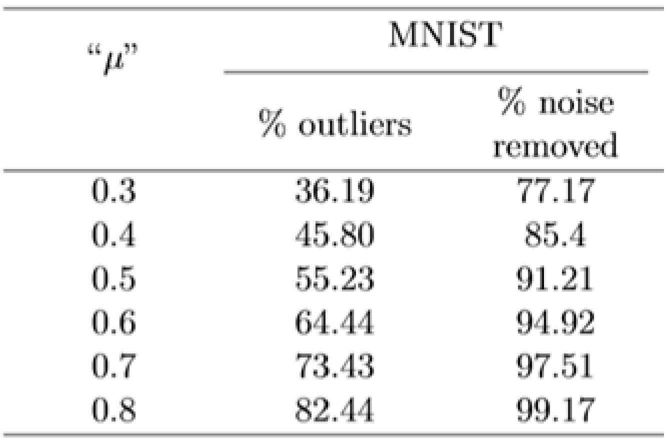
Table 1. Results of our TCSVM used to detect label noise. We use RBF kernel, select C as 8 and select gamma as 1/(number of samples in the dataset).

Iteration	Total # SV	Noise Rate	% of noise detected (Accum.)	Total noise SV detected
1	1072.2	200 (10%)	98.80%	197.69
2	419.7	200 (10%)	99.70%	199.49
1	1378.36	400 (20%)	97.20%	389.09
2	422.6	400 (20%)	99.90%	399.52
1	1582.83	600 (30%)	95.00%	570.13
2	487.166	600 (30%)	99.80%	598.85

Table 2. Results of the TCSVM from [8], used to detected label noise from a dataset with 2000 samples from MNIST dataset. This is the average result of 30 experiments. Our result in table 1 is according with this result.

"μ"	Total # SV	% Cumulative #SV	Noise Rate	% of noise detected (Accum.)	Total noise SV detected
0.5	1802	50.06%	360 (10%)	94.72%	341
0.5	1801	50.03%	720 (20%)	90.56%	652
0.55	1984	55.11%	720 (20%)	92.92%	669
0.6	2161	60.03%	720 (20%)	95.00%	684
0.6	2161	60.03%	1080 (30%)	89.72%	969
0.65	2343	65.08%	1080 (30%)	92.22%	996
0.7	2523	70.08%	1080 (30%)	96.02%	1037

Table 3. Results of our OCSVM used to detect label noise. We use RBF kernel and select gamma as 1/(number of samples in the dataset).



"μ"	MNIST	
	% outliers	% noise removed
0.3	36.19	77.17
0.4	45.80	85.4
0.5	55.23	91.21
0.6	64.44	94.92
0.7	73.43	97.51
0.8	82.44	99.17

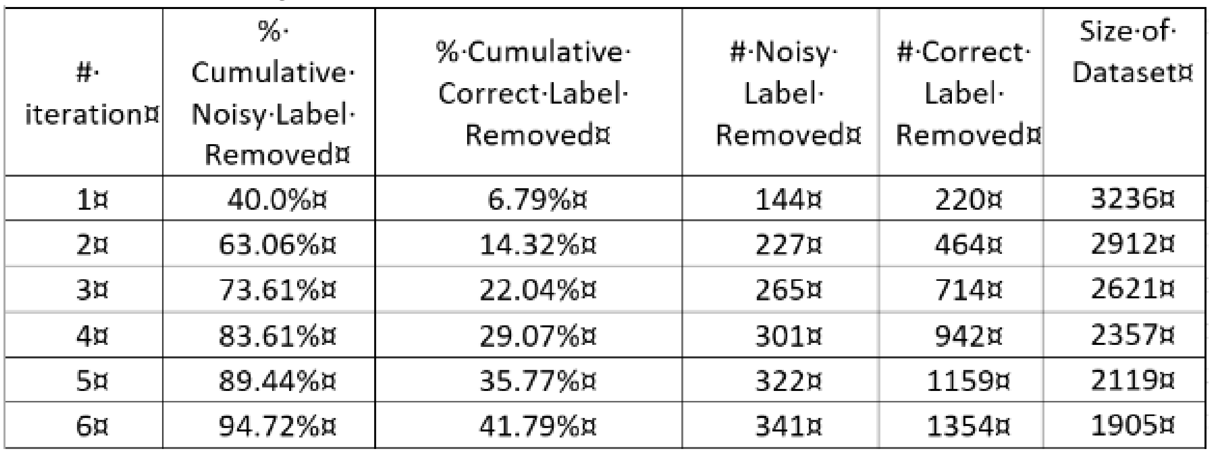
Table 4. Results of the TCSVM from [8], used to detected label noise from a dataset with 1000 samples from MNIST dataset. 900 samples are from class "4", 100 samples are from class "9" and class "7" (50 each). The RBF kernel is used.

3.3 Attempt to get rid of reviewing

In order to get rid of human review when using SVM to detect label noise, we implement the following experiment. The experiment is based on this algorithm:

1) Train OCSVM classifier using the training examples.
2) Remove all the outliers from the dataset
3) Train a new OCSVM classifier with the updated dataset
4) Use the new classifier to classify the removed outliers.
5) Remove the '-1' examples and keep the '1' examples.
6) Repeat Steps 1-5 until no further noisy label is found.

We use the same dataset as we used in 3.2, and the noise rate is 10%. And the results are shown in the following table.



# iteration	% Cumulative Noisy Label Removed	% Cumulative Correct Label Removed	# Noisy Label Removed	# Correct Label Removed	Size of Dataset
1	40.0%	6.79%	144	220	3236
2	63.06%	14.32%	227	464	2912
3	73.61%	22.04%	265	714	2621
4	83.61%	29.07%	301	942	2357
5	89.44%	35.77%	322	1159	2119
6	94.72%	41.79%	341	1354	1905

Table 5. Results of our OCSVM used to detect label noise without human review. We use RBF kernel and select gamma as 1/(number of samples in the dataset).

From the result we can see, most of the samples with label noise can be detected and removed by our method. However, a great number of clean samples are also removed from the dataset. A possible explanation is that the OCSVM classifier is not able to classify the outliers well. We can combine our method with the original method which is mentioned in 3.2 to cleanse the label noise. First, we can cleanse about 50% of the label noise by using our method without human review. Then use the original method from 3.2 to cleanse the rest label noise. In this case, although we still need the reviewing procedure, the total number of samples we need to review will be decreased.