21st International Conference on Pattern Recognition (ICPR 2012)
November 11-15, 2012. Tsukuba, Japan

Label-Noise Reduction with Support Vector Machines

Sergiy Fefilatyev[1] (sfefilatyev@gmail.com), Matthew Shreve[1], Kurt Kramer[1], Lawrence Hall[1], Dmitry Goldgof[1], Rangachar Kasturi[1], Kendra Daly[2], Andrew Remsen[2], Horst Bunke[3]

[1]*Department of Computer Science and Engineering, University of South Florida*
[2]*College of Marine Science, University of South Florida*
[3]*Institute of Computer Science and Applied Mathematics, University of Bern*

Abstract

The problem of detection of label-noise in large datasets is investigated. We consider applications where data are susceptible to label error and a human expert is available to verify a limited number of such labels in order to cleanse the data. We show the support vectors of a Support Vector Machine (SVM) contain almost all of these noisy labels. Therefore, the verification of support vectors allows efficient cleansing of the data. Empirical results are presented for two experiments. In the first experiment, two datasets from the character recognition domain are used and artificial random noise is applied in their labeling. In the second experiment, a large dataset of plankton images, that contains inadvertent human label error, is considered. It is shown that up to 99% of all label-noise from such datasets can be detected by verifying just the support vectors of the SVM classifier.

1. Introduction

The presence of label-noise has an adverse effect on the performance of classifiers [15, 2]. In this paper, we present a procedure for cleansing training data that contains label-noise using a human in the loop.

The problem of mitigating the effects of the mislabeled instances of training data was previously approached from several directions. One well-studied approach was to filter unreliable training data before using the data to train a classifier. Such methods for filtering used the nearest neighbor algorithm [25, 22], AdaBoost [6], decision trees [15], and wide-margin classifiers [21]. More general (not-classifier specific) methods [3, 24] included an ensemble of classifiers voting in consensus or majority setups to filter the data. In all those approaches, the filtering was performed by comparing the classification results on the training set with the actual labels. While in many cases performance of classifiers increased after such filtering, these procedures could filter valid instances of training data which represented exceptions.

Another line of research focused on the problem of mitigation of negative effects of the presence of noise on the performance of the classifier without removing the noise instances from the training data. Different methods for that were focused on using AdaBoost [16] and large-margin classifiers. Many such methods [20, 13, 1] resulted in robust classifiers which demonstrated good handling of noisy data on several artificial and real datasets. The broad idea common to these methods is to minimize the overall error caused label noise.

In our approach, we are neither trying to automatically clean the training dataset, nor we are trying to automatically mitigate the poor performance of the classifier by optimization of underlying learning algorithm. Instead, we are considering the situation with a human in the loop, where an expert will examine a limited number of instances of training data to either confirm the label or correct it. We choose a small set of the training data represented by support vectors of the SVM classifier built on the original noisy training data as a subset that most likely contains all of the noise. In that, our approach is somewhat similar in idea to large margin-based active learning approaches where an instance of unlabeled data with important learning consequences is iteratively presented to an expert for labeling [5, 23, 17]. In their selection strategy, they use the distance to the current hyperplane to iteratively identify examples [11, 12] to present to a human for labeling. The output of each step, which is likely to be a support vector, is added to the current training set and is used to select another unlabeled instance in the next iteration. Our approach of active noise cleansing deals with

978-4-9906441-0-9

a labeled set assuming the presence of noise, requires only a few iterations for training of the SVM classifier, and, usually, achieves a major set of label corrections in the first iteration. We demonstrate the validity of this approach on two real datasets that contain artificially introduced label-noise, and also on a dataset that contains unintentional label error. We show that up to 99% of label-noise can be detected just by verifying the support vectors of the trained SVM classifier.

2. Algorithm

The algorithm is based on the hypothesis that incorrectly labeled data-points are likely to be chosen as support vectors. The argument for such behavior is supported by the optimization procedure for SVM parameters [4]. The process is based on the number of correctly separated samples of training data by a hyperplane, parameters of which are sought. Since the support vectors describe the hyperplane, it is likely that the more complex the boundary is the more support vectors are needed to describe it. The assumption used here is that the samples with incorrectly assigned labels infiltrate the samples with correctly assigned labels. Thus, the optimization process aimed at carving out the precise boundary for an accurate separation will create a more complex boundary compared to the case of the optimization of an SVM on the same data with all correct labels. Hence, by validating each support vector using the expert's knowledge, it is possible to detect (and update) a large number of incorrectly labeled training data by reviewing only a fraction of the training data. Remaining label-noise is potentially reduced even further in subsequent iterations of this process. Overall, the algorithm is described as follows:

1. Train a binary SVM classifier. If the training set contains multiple classes use a one-versus-one strategy.
2. Collect all examples that were identified as a support vector for each pair.
3. Have an expert validate each of these examples, updating its label if necessary.
4. Repeat *Steps 1 - 3* until no further label error is found.

In our implementation of the described algorithm, all SVM classifiers for each possible binary class combinations are created using the one-versus-one strategy. A class label is assigned by a majority vote. In the case of a tie among classes, the probability parameter of the SVM is used to select the class label. The feature selection process consists of two steps: initial SVM parameter tuning and binary feature selection. The parameters (γ, C, A) of the SVM are optimized by performing a grid-search with a given interval across the training dataset [19]. Using the SVM parameters determined in the first stage of the selection process, binary class feature selection (BFS) [10] is performed using wrappers [9, 18]. Each combination of features and SVM parameters was evaluated using 5-fold cross validation [26] and the classification accuracy on the training set was used to guide the selection process further. In cases, where the classification accuracy was equal for several evaluated sets, the correctness of probability (CPP) [14] was used to rank the sets.

All the examples that have been identified as support vectors were reviewed by the expert and re-labeled if necessary. This entire process was then repeated until none of the support vectors contain label error. In our simulated experiments below, two iterations were sufficient for removing 99% of all label noise.

3. Experiments

The validity of this approach was demonstrated empirically on two categories of datasets: datasets that contained artificially introduced label-noise, and also on one that contained unintentional label error.

3.1 Character Recognition Datasets

First, we evaluated the algorithm on data that contained artificially added label-noise. For this experiment, we chose two letter recognition datasets. The first dataset, UCI Letter Recognition dataset [8], had 20,000 instances, 16 features, and 26 classes (capital letters from A to Z). The classes in the dataset were approximately balanced, i.e. each class had approximately 700-800 instances each. We split the dataset into three parts. The training set had 12,000 instances randomly picked from the original dataset. The test partition had 4,000 instances, randomly picked from the original dataset minus the training set. The remaining part of the original set was left for a future validation.

After exploratory classification, we selected two classes, letters "H" and "R" that had the most confusion during classification. In the training set, for these two classes we introduced noise in the amount of 10%, 20%, and 30% the size of those classes (for three different experiments). The label-noise was introduced by switching labels between a pair of instances of these two classes. Thus, in an experiment with 10% label-noise, 10% of the instances of class "H" got the label "R", and 10% of the instances of class "R" got the label

UCI Letter Recognition Dataset								
Iteration	**Total # SV**	**Noise switches (both letters)**	**Class 'H' SV**	**Class 'H' noise SV detected**	**Class 'R' SV**	**Class 'R' noise SV detected**	**Total noise SV detected**	**% of noise detected (Accum.)**
1	463.2	90 (10%)	236.33	44.77	226.86	44.93	89.7	**99.6 %**
2	265.76	0.3 (after 1st iter.)	131.7	0.06	134.07	0.06	0.12	**99.9 %**
1	604.33	180 (20%)	307.76	89.433	296.56	89.63	179.06	**99.5%**
2	264.26	0.93 (after 1st iter.)	131.36	0.5	132.9	0.36	0.86	**99.9%**
1	707.36	270 (30%)	361.1	132.53	346.26	133.2	265.73	**98.4%**
2	268.3	4.26 (after 1st iter.)	133.2	2.4	135.1	1.7	4.1	**99.9%**

MNIST Dataset								
Iteration	**Total # SV**	**Noise switches (both digits)**	**Class '9' SV**	**Class '9' noise SV detected**	**Class '7' SV**	**Class '7' noise SV detected**	**Total noise SV detected**	**% of noise detected (Accum.)**
1	1072.2	200 (10%)	535.03	98.83	537.16	98.86	197.69	**98.8%**
2	419.7	2.41 (after 1st iter.)	196.56	0.9	223.13	0.9	1.8	**99.7%**
1	1378.36	400 (20%)	682	194.83	696.36	194.26	389.09	**97.2%**
2	422.6	10.91 (after 1st iter.)	207.4	5.03	215.2	5.4	10.43	**99.9%**
1	1582.83	600 (30%)	781.56	284.93	801.26	285.2	570.13	**95.0%**
2	487.166	29.87 (after 1st iter.)	236.8	14.46	250.36	14.26	28.72	**99.8%**

Table 1: Results of label-noise reduction on the the UCI Letter Recognition and MNIST datasets. Label-noise between the classes corresponding to letters 'H' and 'R' (for UCI dataset) and digits '9' and '7' (for MNIST) was randomly introduced in the amount of 10%, 20%, and 30%. Noise detection/correction was performed by reviewing the support vectors of the SVM classifier between 'H' and 'R' (for UCI) and '9' and '7' (for MNIST) over two iterations.

"H". Experiments with 20% and 30% noise had a similar setup. An SVM classifier between these two classes was built using an RBF kernel and SMO optimizer. The instances of the training data that were selected as the support vectors were examined by a human to verify the correctness of label-assignment. In order to exclude the possibility of a skewed result caused by a favorable random selection of instances of those classes undergoing a label switch, we repeated the experiment 30 times. Thus, 30 different training sets were created for each 10%, 20% and 30% label-noise setup and 30 SVM classifiers have been built for each of the three setups.

Table 1 shows the description of relevant parts of the classifier: number of instances used for support vectors (averaged from 30 different sets), number of noisy labels in the training set, number of noisy labels selected as support vectors as determined by a human after examination, and percentage of detected label-noise examples to all label-noise in the dataset. The results show 99% of the noisily labeled instances from the training set did show up in the support vectors. After correction of such label-noise in the dataset, a second iteration of the procedure removed almost all remaining label-noise.

The second dataset, MNIST Digit Recognition dataset [8], had 60,000 instances, 784 features (pixel values), 10 classes (digits from 0 to 9). The classes in the dataset are approximately balanced. We split the dataset into three parts. The selected 10,000 instances randomly picked from the original dataset for the training set. The test part had another 10,000 instances, randomly picked from the original MNIST dataset after the training set was removed.

Similar to the first experiment we selected the most confused classes, digits "9" and "7". In the training set for these two classes, we introduced 10%, 20%, and 30% noise for three experiments. In order to exclude the possibility of a skewed result caused by a favorable random selection of instances of those two classes, we repeated the experiment 30 times: built 30 SVM classifiers for each of the three setups between these two classes and reviewed the instances of training data that were selected as the support vectors.

Similarly, 99% of the noise examples introduced in the training set became the support vectors. Some difference in performance on the two datasets was observed in the first iteration of the algorithm. For the MNIST dataset the first iteration of the algorithm was less accurate. For example, for MNIST dataset with 30% of the noise there was still 5% of label-noise undetected. However, the second iteration of the algorithm brought the total performance to 99%, as seen with UCI

Letter Recognition dataset.

3.2 Plankton Dataset

The experiment with the Plankton dataset represented a more practical application of data cleansing, because the label-noise naturally occurred in the data. The dataset was created from the images collected for a project to discern oil droplets from plankton using the SIPPER platform (Shadow Imaging Particle Profiler and Evaluation Recorder) [7] during trips to the site of the Deepwater Horizon Oil Spill. The dataset had 36 classes. Most of the classes represented the plankton population, with the rest of the classes belonging to noise, air bubbles, and suspected oil droplets from the spill. The Plankton dataset was composed of 8,537 particles, which represented less than 0.5% of all image data during the cruise to the spill-affected area. The oil droplet class had 1,072 instances, comprising 12.49% of the particles in the dataset. The decision to label each particle was made based on a visual analysis of the particle, with the knowledge available to marine science experts. The noise in assigning labels was a result of fatigued human experts due to the large amount of required visual examination of similar looking particles.

The Plankton dataset was used as the training and test set for exploratory classification using a SVM classifier (ensemble consisting of 630 individual binary classifiers between all classes) to determine the class that had the most confusion with the class of primary interest - the oil droplet class. This class turned out to be the air-bubble class.

In the first part of the experiment, the support vectors of the individual binary SVM classifier corresponding to the classification decision of oil droplets vs. air-bubbles were examined in several iterations. If any of the support vectors belonging to the oil-droplet class were found to carry an incorrect label, the label was corrected and a new SVM classifier built on the updated Plankton dataset. The binary SVM classifier that was built on the initial Plankton dataset had 18 support vectors corresponding to the oil droplet class. After reviewing them, six support vectors were found to be mislabeled. After correcting the labels a new SVM classifier was built and the second iteration for reviewing the support vectors was performed. In total, three iterations of reviewing the support vectors were required to ensure that all the support vectors had all their labels correctly assigned, seven mislabeled particles were found during the first two iterations.

In the second part of the experiment, all 1,072 instances of the original uncorrected oil droplets from the Plankton dataset were independently examined by marine science experts to see if label-noise is evident. Seven instances of incorrectly assigned oil droplets were identified and they matched exactly the label-noise instances found using SVM-based approach. Thus, the same result for label-noise reduction was obtained by examining 51 instances of support vectors (in three iterations) as opposed to the full set of 1,072 instances, which is less than 5% of samples in the class.

Another interesting observation was that although in the first part of the experiment we only reviewed the support vectors of a single binary SVM classifier corresponding to the decision between the classes oil droplet vs. air bubble, it allowed detection of label-noise not necessarily related to air-bubbles. Out of the seven mislabeled oil droplets only four were air bubbles. The other three belonged to other classes, but they were still absorbed into support vectors of the described binary SVM classifier.

4. Conclusions

In this paper, we empirically showed that in order to help detect the examples of training data which have incorrectly assigned labels, the support vectors of the SVM classifier trained on such a noisy dataset need to be verified. By reviewing and, if necessary, updating the label of each support vector, it is possible to detect almost all (up to 99%) noise artificially applied to the character recognition datasets. On a dataset that contained images of plankton with inadvertent noise, we were able to detect all incorrect samples in the class of interest by reviewing only 5% of the data. Thus, the described approach helps to significantly reduce the effort needed to remove label-noise from data.

5. Acknowledgement

This research was made possible by a grant from BP/The Gulf of Mexico Research Initiative through the Florida Institute of Oceanography and Office of Naval Research grant N00014-07-0802.

References

[1] B. Biggio, B. Nelson, and P. Laskov. Support vector machines under adversarial label noise. In *Proceedings of Asian Conference on Machine Learning*, pages 97–112, 2011.

[2] C. Brodley and M. Friedl. Identifying and eliminating mislabeled training instances. In *Proceedings of International Conference on Artificial Intelligence*, pages 799–805, 1996.

[3] C. Brodley and M. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167, 1999.

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

[5] C. Campbell, N. Nello, and A. Alex. Query learning with large margin classifiers. In *Proceedings International Conference on Machine Learning*, pages 111–118, 2000.

[6] J. Cao, S. Kwong, and R. Wang. A noise-detection based adaboost algorithm for mislabeled data. *Pattern Recognition*, 2012.

[7] S. Fefilatyev, K. Kramer, L. Hall, D. Goldgof, R. Kasturi, A. Remsen, and K. Daly. Detection of anomalous particles from the deepwater horizon oil spill using the sipper3 underwater imaging platform. In *International Conference on Data Mining Workshops*, pages 741–748, 2011.

[8] P. Frey and D. Slate. Letter recognition using holland-style adaptive classifiers. *Machine Learning*, 6(2):161–182, 1991.

[9] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2):273–324, 1997.

[10] K. Kramer. *System for Identifying Plankton from the SIPPER Instrument Platform*. Ph.d. dissertation, University of South Florida, 2010.

[11] P. Mitra, C. Murthy, and S. Pal. Data condensation in large databases by incremental learning with support vector machines. In *Proceedings of International Conference on Pattern Recognition*, volume 2, pages 708–711, 2000.

[12] P. Mitra, C. Murthy, and S. Pal. A probabilistic active support vector learning algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(3):413–418, 2004.

[13] E. Niaf, R. Flamary, C. Lartizien, and S. Canu. Handling uncertainties in svm classification. In *IEEE Statistical Signal Processing Workshop*, pages 757–760, 2011.

[14] J. Platt et al. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3):61–74, 1999.

[15] J. Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[16] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine learning*, 37(3):297–336, 1999.

[17] G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *Proceedings of International Conference on Machine Learning*, volume 282, pages 839–846, 2000.

[18] H. Silva and A. Fred. Pairwise vs global multi-class wrapper feature selection. In *Proceedings of Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, volume=6, pages=1–6, year=2007*.

[19] C. Staelin. Parameter selection for support vector machines. Tech. rep. hpl-2002-354r1, Hewlett-Packard Company, 2003.

[20] G. Stempfel and L. Ralaivola. Learning svms from sloppily labeled data. *Artificial Neural Networks–ICANN 2009*, pages 884–893, 2009.

[21] J. Thongkam, G. Xu, Y. Zhang, and F. Huang. Support vector machine for outlier detection in breast cancer survivability prediction. *Advanced Web and Network Technologies, and Applications*, pages 99–109, 2008.

[22] I. Tomek. An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, (6):448–452, 1976.

[23] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

[24] S. Verbaeten and A. Van Assche. Ensemble methods for noise elimination in classification problems. In *Proceedings of International Conference on Multiple Classifier Systems*, pages 317–325, 2003.

[25] D. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, 2(3):408–421, 1972.

[26] I. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2005.