Batch-Mode Active Learning by Using Misclassified Data

Tengyu Sun and Jie Zhou

Department of Automation, Tsinghua University, Beijing 100084, China
sty-le@163.com, jzhou@tsinghua.edu.cn

Abstract. In this paper, we proposed a batch-mode active learning strategy which makes use of the misclassified data. The proposed algorithm first trains a classifier using the labeled data. Then in the active learning step, unlike many existing algorithms querying unlabeled samples close to the decision boundary, ours queries samples close to the data misclassified by the current classifier. In order to incorporate the diversity into the batch-mode querying, the proposed algorithm clusters the unlabeled samples near the misclassified data and queries the samples closest to each cluster center. Experimental results on real world datasets show that the proposed algorithm has a satisfying performance.

Keywords: batch-mode active learning, Adaboost, k-means

1 Introduction

Active learning has become an interesting topic in the machine learning community as more and more training data are easy to collect but hard to label. Active learning not only aims at reducing the work of human annotation but also intends to train a better classifier by selecting a better training set. There have been many works in the field of active learning during the past 20 years [1]. Among them, some study the theories behind the active learning algorithms [2–4], and others try to use the theory to solve real world problems [5, 6]. Both theoretic and application results show that active learning is a promising solution for large-scale data annotation problems.

Traditional active learning algorithms query unlabeled samples one by one. But when used in practice, batch-mode querying is better. We can query N samples together at one time instead of the serial querying. To use a serial querying algorithm in a batch-mode-querying way, simply querying the top N results in the serial querying is not the best strategy because there may be redundancy among the top best samples. There have been some algorithms addressing this problem [7–9]. Xu et. al. proposed an algorithm to cluster samples within the margin of the SVM where is considered as the uncertain area of the classifier [9]. The resulting cluster centers give a better approximation of the distribution of the data in the uncertain area than the N closest-to-boundary points do.

In this paper, we proposed an active learning algorithm that also uses the clustering result to help batch-mode querying. But unlike Xu's algorithm, we

cluster the unlabeled samples near the data misclassified by the current classifier, because we think not only the samples close to the decision boundary but also the samples near the misclassified data are uncertain. Querying uncertain samples far from the decision boundary can overcome the sampling bias brought by just querying samples close to the decision boundary. So, our algorithm first trains a classifier using the labeled samples, and then clusters the unlabeled samples near the misclassified data and queries the samples closest to each cluster center. Experimental results on real world datasets show that our algorithm outperforms the algorithms just querying samples close to the boundary.

The remainder part of the paper is arranged as follow: in the next section, we first give an intuitive example which motivates our algorithm, and then we show the details of the proposed algorithm. Experimental results on real world datasets will be shown in Section 3. At last, we conclude the paper and discuss our future work in Section 4.

2 Query by Misclassified Data

2.1 An Intuitive Example

Many active learning algorithms repeatedly train a classifier using the labeled dataset, and then query the unlabeled sample closest to the current decision boundary. This is mainly because they think it is the most uncertain sample. As for the samples far away from the boundary, they think the predictions given by the classifier are reliable, so don't need to query. Under certain circumstances, there are also theoretical guarantees behind this kind of algorithms. For example, when the classifier is SVM and the data is linear separable, the sample closest to the decision boundary is expected to shrink the version space the most [10]. However in the non-separable case, this kind of query strategies may bring bad results. This phenomenon has been discussed in [11] and is called as sampling bias. The key point is that actually the predictions of the samples far from the boundary are not that reliable, especially near a misclassified sample, because the current decision boundary may not be optimal. We may over trust the predictions on the unlabeled samples, and get stuck in a local minimal solution. In fact, what are truly reliable is the labels of the data misclassified by the current classifier. If we find more samples that look like the misclassified data, we could 1) tell whether a misclassified sample is a noise data, 2) improve the performance of the classifier.

From the Fig. 1 below, we can see that there are some unlabeled samples in the circle that are far away from the decision boundary, so the prediction results on them should be trustable. But they are near a negative point which is misclassified by the current boundary. If we use a nearest neighbor classifier, they should belong to the negative class instead of the positive class given by the current classifier. So if we query these labels, we probably get 3 kinds of results: 1) all the samples are negative which means the current decision boundary needs a big change; 2) about half of the samples are negative and half positive, this probably means the optimal decision boundary may lie in this area; 3) all the

samples are positive which makes the misclassified sample more likely a noise point. All the 3 kinds of results are of benefit to the training of the classifier. Note that there are also some misclassified data close to the decision boundary so querying samples near the misclassified data may also result in querying samples close to the decision boundary, which is of help in some case.

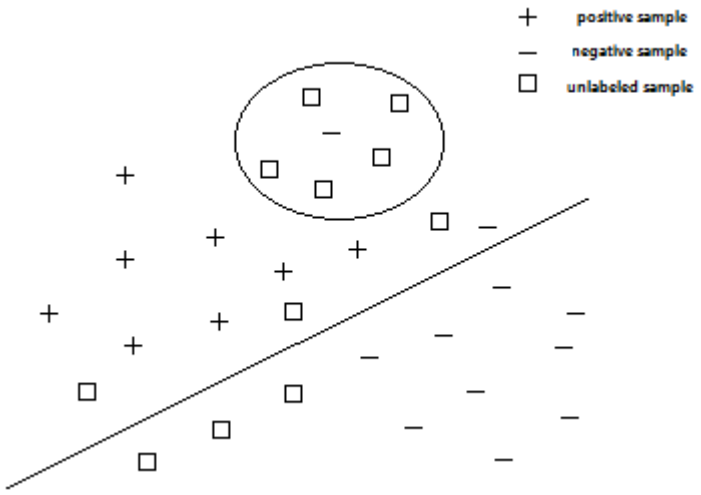

Fig. 1. An intuitive example shows the use of misclassified data

2.2 The Algorithm

From the example above, we have seen the use of the misclassified data. Now we discuss the details of our algorithm. The flow diagram of our algorithm is shown in Fig. 2. Our algorithm repeatedly trains a classifier using the labeled data, then clusters the samples near the misclassified data and queries the samples closest to each cluster center, at last updates the labeled dataset and retrains the classifier. The pseudo code of the algorithm is listed as below.

```
Algorithm 1: Query By Misclassified data (QBM)
Input: labeled dataset L = {(x_i, y_i)}_{i=1}^{l},
       unlabeled dataset U = {x_i}_{i=l+1}^{l+u}, querying batch size N.
repeat until meet some stop criterion:
    train a classifier h using the labeled dataset L.
    let the misclassified dataset be M = {x_i | (x_i, y_i) ∈ L,  h(x_i) ≠ y_i}.
    Q = ∅.
    for each x_i in U
        if one of x_i's k nearest points in L is some point x_j in M
            w_i = L(h(x_j), y_j) / (1 + ||x_i − x_j||^2).
            Q = Q ∪ {(x_i, w_i)}.
        end
    end
    cluster Q into N clusters,
    querying N points closest to each cluster center.
    update L and U with queried samples.
```

The clustering procedure in the active learning part of the algorithm can be viewed as a vector quantization of the candidate querying set Q, since the size of Q is usually larger than the batch size N. So we use the cluster centers to represent the set Q. When we cluster Q, we want the cluster centers are more close to the misclassified data which have a higher misclassified loss. So we give each unlabeled sample x_i in Q a weight w_i and use a weighted clustering algorithm. The weight w_i is proportion to the loss function's value of the misclassified data that x_i close to and inverse proportion to the distance to the misclassified data. Specifically, it equals $\frac{L(h(x_j),y_j)}{1+\|x_i-x_j\|^2}$, where x_i is the unlabeled point, x_j is the misclassified data that x_i close to, and $L(h(x_j), y_j)$ is the loss function. Thus a point close to a misclassified data with a high loss has a high weight. To handle the weighted data, we use a weighted clustering algorithm which is an variant version of the standard k-means algorithm. We modify the object function of the k-means algorithm to $J = \sum_i w_i \parallel x_i - m_k \parallel^2$ so that the cluster centers are more close to the data with higher weights. The pseudo code of the weighted k-means algorithm is listed as below.

```
Algorithm 2: Weighted K-means
Input: weighted dataset Q = {(x_i, w_i)}, number of clusters K.
    random initial K clusters C_1, C_2, ..., C_K.
    calculate J = Σ_i w_i || x_i − m_{c_i} ||^2, where x_i ∈ C_{c_i},
    and the cluster centers m_k = (Σ_{c_i=k} w_i x_i) / (Σ_{c_i=k} w_i), k = 1, 2, ..., K.
    repeat until meet some stop criterion:
        for each (x_i, w_i) in Q
            c_i = arg min_k  w_i || x_i − m_k ||^2
        end
        update J and new cluster centers m_k according to
        the new assignmnet.
```

Note that sometimes when the labeled dataset is small or the dataset is separable by the classifier, the size of Q may be much smaller than N, or even be zero. If this happens, we use a random sampling in U to select the rest querying samples, since the random sampling can best preserve the distribution of the input dataset. When faced with real world data and the initial labeled samples are sufficient, this kind of situation usually would not happen.

As for the classifier h in the training procedure, we could use any classifier that can be formulated as a solution of minimizing a certain kind of loss function. In the experiments, we choose the Adaboost with a fisher linear classifier and the SVM as the basic classifiers. The corresponding loss functions for the misclassified data are $L(h(x), y) = \exp(-yh(x))$ and $L(h(x), y) = \max(1-yh(x), 0)$. Experimental results will be shown in next section, which prove our algorithm's good performance on real world datasets.

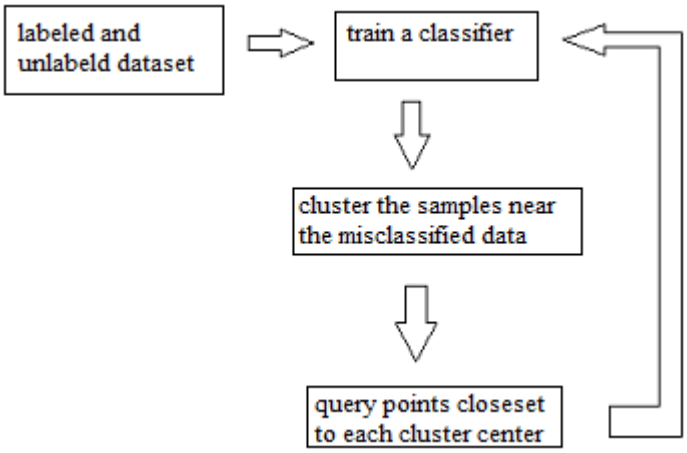

Fig. 2. Flow diagram of QBM

3 Experiment

3.1 Nomao Challenge

Nomao dataset[1] is the dataset used in ALRA workshop challenge (Active Learning in Real-world Applications) [12]. Nomao is a famous place search engine. It collects data about places from the Internet and merges the data referring to the same place. Each item in the dataset describes the similarity of two spots collected from the Internet. The goal is to decide whether these two spots refer

[1] http://archive.ics.uci.edu/ml/datasets/Nomao

to the same place or not. The dataset contains 34465 samples with missing values. Each sample has a label +1 or -1, indicating that the two spots refer to the same place or not. Thus it can be formulated as a binary classification problem. We randomly choose 30000 samples as the training samples and the left 4465 as test samples. We use two kinds of classifiers as the basic classifier h used in training procedure, namely SVM and Adaboost with fisher linear classifier. For each classifier, we compare 3 different active learning algorithms on the dataset: the QBM algorithm (Algorithm 1), the uncertainty sampling [13] which queries samples closest to the decision boundary, and the random querying which randomly selecting samples in the unlabeled set. Each algorithm is carried out 10 times with 3000 randomly selected samples in training set as the initial labeled set L and the rest as unlabeled set U. The query batch size is 1000. Results are shown in Fig. 3 & 4. The solid line with circle on it is the result of QBM, the solid line with cross is for random querying and the one with square is for the uncertainty sampling [13].

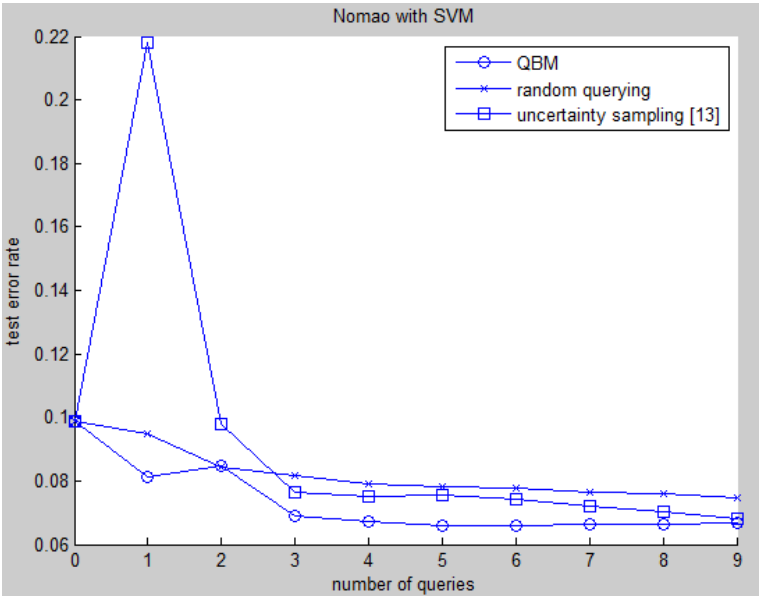

Fig. 3. Experimental results of 3 algorithms on Nomao dataset with SVM as the basic classifier

We can see from the results that the QBM algorithm generally gets a better result under the same times of querying. In the Fig. 3, we can see that in the first 5 rounds of querying QBM trains a better classifier than random querying and uncertainty sampling. As the number of queries grows, the difference between QBM and uncertainty sampling gets smaller, this is mainly because there are fewer unlabeled samples left and the difference between two labeled sets becomes

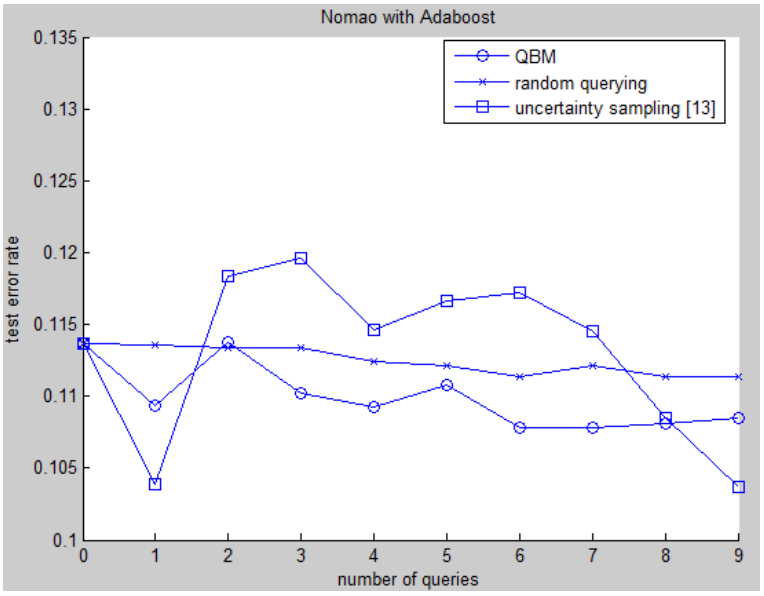

Fig. 4. Experimental results of 3 algorithms on Nomao dataset with Adaboost as the basic classifier

small. From Fig. 4, we see uncertainty sampling suffers from sampling bias, it gets a better results at the beginning, but in the coming queries, the result gets worse. On the contrary, QBM performs better in the first 5 queries. Comparing Fig. 3 with Fig. 4, we can see SVM with QBM performs the best on this dataset.

3.2 20 Newsgroups

20 Newsgroups dataset[2] is a newsgroup documents dataset with 11256 training samples and 7489 test samples. The documents from different newsgroups have different topics. So it is a multiple classes classify problem, but we use it as several one-to-one binary classification problems. We also compare the 3 active learning algorithms with the 2 different basic classifiers as in the previous section. Followed [14], we select 2 hardest pairs of topics to classify. The training set is about 1000 samples each, and the test set has around 700 samples. The query batch size is 100. Experiments are carried out 10 times with random initial labeled set of size 100.

The results are shown as below. Like in the previous section, the solid line with circle is the result of QBM, the solid line with cross is for random querying and the one with square is for the uncertainty sampling [13]. We can see that QBM is generally better than other algorithms except in the Fig. 7 where the uncertainty sampling is better during the middle part of the queries but defeated

[2] http://people.csail.mit.edu/jrennie/20Newsgroups/

by QBM at last. It is noticeable to see that sometimes active learning algorithms fails to random querying as shown in the Fig. 7.

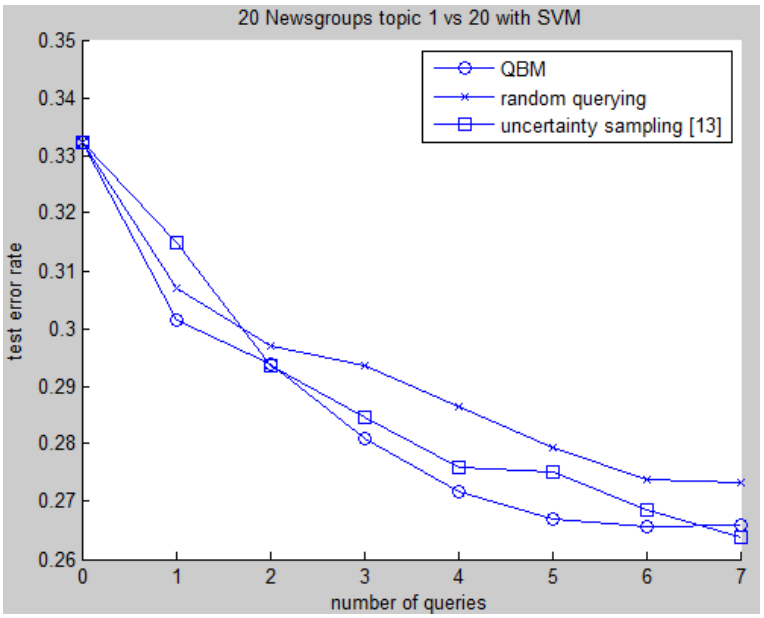

Fig. 5. Experimental results of 3 algorithms on topic alt.atheism vs talk.religion.misc with SVM as the basic classifier

4 Conclusion and Future Work

We proposed a new active learning algorithm to make use of the misclassified data. The experimental results show that it yields good results on real world datasets. It is worth being mentioned that the algorithm has some connections with the Adaboost algorithm, but there have not been some theoretical analyses on that. Our future work will focus on it.

Acknowledgements

This work was partly supported by the Natural Science Foundation of China under Grant 61020106004 and 61021063.

References

1. Settles, B.: Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2009)

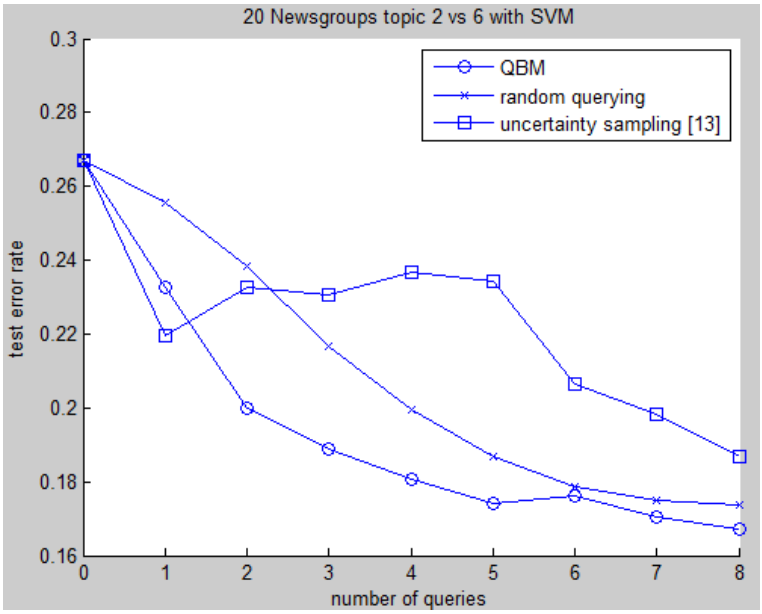

Fig. 6. Experimental results of 3 algorithms on topic comp.graphics vs comp.windows.x with SVM as the basic classifier

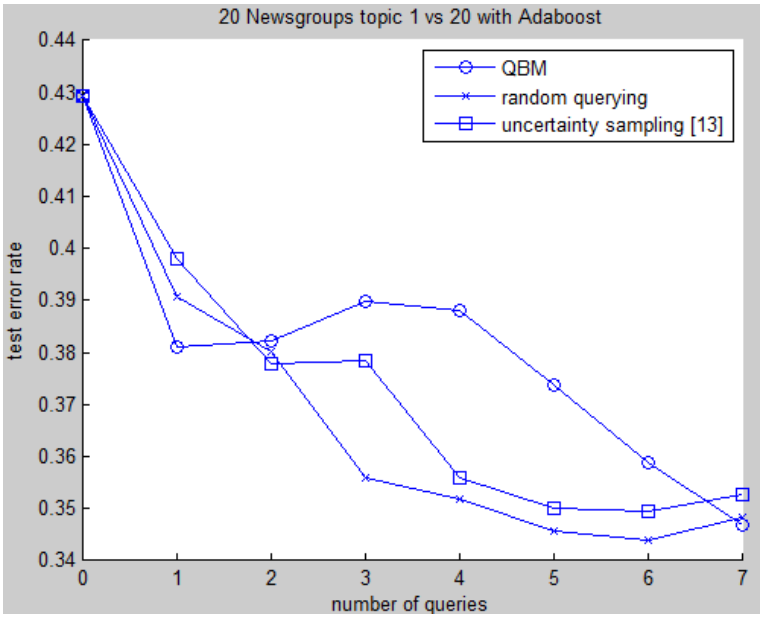

Fig. 7. Experimental results of 3 algorithms on topic alt.atheism vs talk.religion.misc with Adaboost as the basic classifier

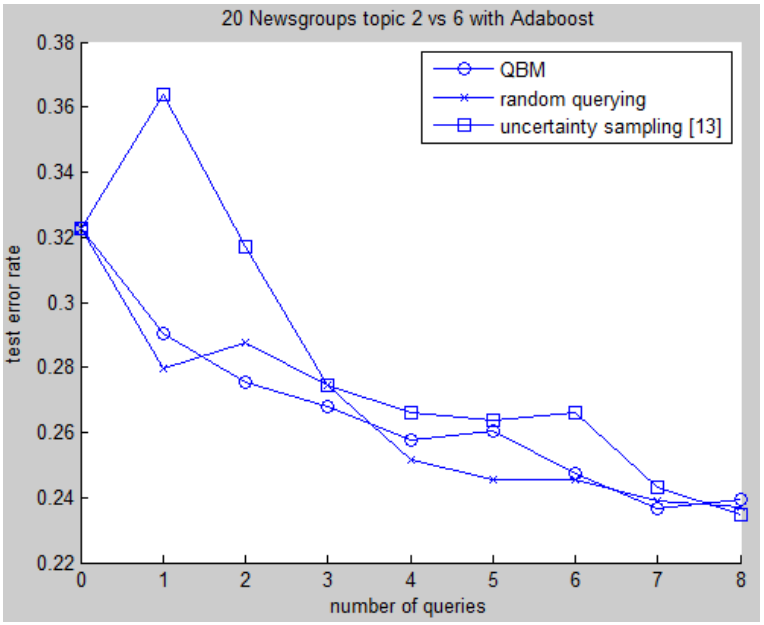

Fig. 8. Experimental results of 3 algorithms on topic comp.graphics vs comp.windows.x with Adaboost as the basic classifier

2. Freund, Y., Seung, H.S., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. Mach. Learn. **28**(2-3) (September 1997) 133–168
3. Dasgupta, S.: Analysis of a greedy active learning strategy. In: In Advances in Neural Information Processing Systems, MIT Press (2004) 337–344
4. Balcan, M.F., Hanneke, S., Vaughan, J.W.: The true sample complexity of active learning. Machine Learning **80**(2-3) (2010) 111–139
5. Tomanek, K., Olsson, F.: A web survey on the use of active learning to support annotation of text data. In: Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing. HLT '09, Stroudsburg, PA, USA, Association for Computational Linguistics (2009) 45–48
6. King, R.D., Whelan, K.E., Jones, F.M., Reiser, P.G.K., Bryant, C.H., Muggleton, S.H., Kell, D.B., Oliver, S.G.: Functional genomic hypothesis generation and experimentation by a robot scientist. Nature **427**(6971) (January 2004) 247–252
7. Brinker, K.: Incorporating diversity in active learning with support vector machines. In: In Proceedings of the 20th International Conference on Machine Learning, AAAI Press (2003) 59–66
8. Hoi, S.C.H., Jin, R., Zhu, J., Lyu, M.R.: Batch mode active learning and its application to medical image classification. In: In Proceedings of the 23rd International Conference on Machine Learning, Morgan Kaufmann (2006) 417–424
9. Xu, Z., Yu, K., Tresp, V., Xu, X., Wang, J.: Representative sampling for text classification using support vector machines. ecir 03 (2003)
10. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. J. Mach. Learn. Res. **2** (March 2002) 45–66

11. Dasgupta, S.: The two faces of active learning. In: Proceedings of the 12th International Conference on Discovery Science. DS '09, Berlin, Heidelberg, Springer-Verlag (2009) 35–35
12. Candillier, L., Lemaire, V.: Design and analysis of the nomao challenge - active learning in the real-world. In: Proceedings of the ALRA : Active Learning in Real-world Applications, Workshop ECML-PKDD 2012, Friday, September 28, 2012, Bristol, UK. (2012) to appear
13. Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval. SIGIR '94, New York, NY, USA, Springer-Verlag New York, Inc. (1994) 3–12
14. Schohn, G., Cohn, D.: Less is more: Active learning with support vector machines. In: Proceedings of the Seventeenth International Conference on Machine Learning. ICML '00, San Francisco, CA, USA, Morgan Kaufmann Publishers Inc. (2000) 839–846