JMLR: Workshop and Conference Proceedings 16 (2011) 113–126 Workshop on Active Learning and Experimental Design

Active Learning for Unbalanced Data in the Challenge with Multiple Models and Biasing

Yukun Chen YUKUN.CHEN@VANDERBILT.EDU
Subramani Mani* SUBRAMANI.MANI@VANDERBILT.EDU

Discovery Systems Lab, Department of Biomedical Informatics
*Department of Electrical Engineering and Computer Science**
Vanderbilt University
Nashville, TN 37232, USA

Editor: I. Guyon, G. Cawley, G. Dror, V. Lemaire, and A. Statnikov

Abstract

The common uncertain sampling approach searches for the most uncertain samples closest to the decision boundary for a classification task. However, we might fail to find the uncertain samples when we have a poor probabilistic model. In this work, we develop an active learning strategy called "Uncertainty Sampling with Biasing Consensus" (USBC) which predicts the unbalanced data by multi-model committee and ranks the informativeness of samples by uncertainty sampling with higher weight on the minority class. For prediction, we use Random Forests based multiple models that generate the consensus posterior probability for each sample as part of USBC. To further improve the initial performance in active learning, we also use a semi-supervised learning model that self labels predicted negative samples without querying. For more stable initial performance, we use a filter to avoid querying samples with high variance. We also introduce batch size validation to find the optimal initial batch size for querying samples in active learning.

1. Introduction

Active learning, unlike traditional supervised learning, takes into account the cost of labeling. When non-labeled data is abundantly available, active learners are seeking a way to maximize the learning performance while minimizing the cost of labeling samples from the data. In the active learning challenge (Guyon et al., 2011) our task is to develop active learning strategies to maximize the global score, which considers the overall learning performance as a function of the number of queried samples.

Although there are many types of active learning, this challenge uses pool-based active learning for classification scenarios to simulate its application in many domains where the size of available unlabeled data is large. In the challenge, we are given a data matrix where only one instance is provided with positive label. The prediction values and the instance(s) to be queried need to be uploaded to the system, which would return the true label(s) for the queried instances. Iteratively, we need to make new prediction and ask for new instance(s) to query until an ending criteria is met, for example, all training samples are

Table 1: Datasets in the Development Phase of Challenge in 6 domains (In feature type column "Feat Type", "b" represents binary, "c" continuous, and "m" mix of binary and continuous; the number (Num) of train and test samples is equal.)

Data Name	Domain Name	Feat Type	Feat Num	Sparse Rate	Missing Rate	Train/Test Num	Positive labels Rate
HIVA	Chemo-informatics	b	1617	90.88	0	21339	3.52
IBN_SINA	Handwriting recognition	m	92	80.67	0	10361	37.84
NOVA	Text processing	b	16969	99.67	0	9733	28.45
ORANGE	Marketing	m	230	9.57	65.46	25000	1.78
SYLVA	Ecology	m	216	77.88	0	72626	6.15
ZEBRA	Embryology	c	154	0.04	0.004	30744	4.58

queried. The prediction performance is evaluated according to area under the learning curve (ALC) which plots the AUC (area under the ROC curve) score for the prediction on all the samples with unknown labels as a function of the number of labels queried. For the ALC score computation, please refer our first reference paper (Guyon et al., 2011).

In order to maximize the ALC score, the global score in the challenge, we need to consider two major components of the active learning method: classification algorithm and querying algorithm. The querying algorithm is designed to find the most informative instance based on the prediction and/or some intermediate information during prediction. When the training set is small, or the missing value rate is high, or the feature dimension is large, an accurate classifier is needed to support querying algorithms to output the informative samples while the samples queried subsequently should further improve the accuracy of prediction. For a discussion of the basic active learning strategies, the reader is referred to the survey by Burr Settles (Settles, 2009). In an earlier paper (Chen and Mani, 2010), we have discussed the preliminary results on the development datasets by basic and newly proposed active learning strategies such as uncertainty sampling based and information density based querying methods, along with a classification voting committee of Nave Bayes', Support Vector Machines (SVM) and Random Forests. In this paper, we further investigate the uncertainty sampling based method and propose Uncertainty Sampling with Biasing Consensus (USBC) that improved our previous performance. Semi-supervised learning method is also incorporated when the training set is small, and batch size validation is proposed to find the minimal sufficient initial querying size.

The remainder of this paper is organized as follows. Section 2 describes the datasets in both development and final phases of the competition, and the preprocessing steps. Section 3 introduces all active learning strategies including USBC which consists of Random Forests based multiple models and uncertainty sampling based querying method, semi-supervised learning model and batch size validation. Section 4 presents the experiments and results for both development datasets and final datasets. In Section 5, we discuss the results and the strengths and the weaknesses of our methods. In Section 6, we summarize our work and point to some future directions.

Table 2: Datasets in the Final Phase of Challenge in 6 domains (In feature type column "Feat Type", "b" represents binary, "c" continuous, and "m" mix of binary and continuous; the number (Num) of train and test samples is equal.)

Data Name	Domain Name	Feat Type	Feat Num	Sparse Rate	Missing Rate	Train/Test Num	Positive labels Rate	Map Guess
A	?	m	92	79.02	0	17535	?	IBN_SINA
B	?	m	250	46.89	25.76	25000	?	ORANGE
C	?	m	851	8.6	0	25720	?	HIVA
D	?	b	12000	99.67	0	10000	?	NOVA
E	?	c	154	0.04	0.0004	32252	?	ZEBRA
F	?	m	12	1.02	0	67628	?	SYLVA

2. Datasets in the Challenge

The final data consists of 6 datasets, named A to F, with the same domains as the data in development phase, but the identity of the domains and the fraction of positive labels were purposely omitted. The performance result for each single upload was also unknown before the end of the challenge. Table 1 and Table 2 present the information for development data and final data, respectively.

Dataset ORANGE and dataset B have a large percentage of missing values. For variables with more than 50% of missing values, we considered the missing value as another state. For example, if a binary variable X has only ten percent valid values, we convert X into a variable with three states by including a missing state. For variables with no more than 50% missing values, we performed Gaussian imputation to impute the small number of the missing values randomly based on the distribution for non-missing values. For the datasets with very high number of features such as NOVA, HIVA, D and C, we performed preprocessing using Principle Component Analysis (PCA) to reduce the number of features to 100.

In the challenge, although the domain information of the final datasets was blocked, we could still broadly match them into 5 categories based on types of variables, size of variables, sample size, and number of missing values. For example, dataset Zebra and dataset E are considered in the same category because both datasets are continuous, which is a unique property convincing us to work on it separately; dataset NOVA and dataset D are in the same category because the number of features on both datasets is higher than the number of their samples, and feature types of both are binary; dataset ORANGE and dataset B are in the same category because both have a very high rate of missing values; dataset SYLVA and dataset F are similar because both have the largest number of samples; datasets IBN_SINA and HIVA and datasets A and C are considered similar because both have no obvious differences, but dataset C has a much higher number of variables and is more similar to HIVA. Our basic strategy is to find the best active learning model for each development dataset, and apply it to the final dataset in the corresponding category. The last column, Map Guess, in Table 2 summarizes our guess of the mapping between development data and final data.

3. Active Learning Strategies

Uncertainty Sampling with Bias Consensus (USBC) was the basic active learning algorithm we used for all datasets. This algorithm considers the posterior probability based on multiple models, the uncertainty value, as well as the bias factor based on the proportion of positive class in the training set at each iteration of active learning. It outputs the biasing uncertainty values for all instances, and the instances with the highest output value are queried for labeling. Two other strategies that we employed, semi-supervised learning support and batch size validation, are also introduced in this section.

The following notational convention is used for the description of our active learning strategies: The data feature matrix is denoted by $\mathbf{x}$; the outcome variable (class) column is denoted by $\mathbf{y}$; the model that generates the posterior probability of label y given data vector (instance) x is denoted by θ; the set of labeled and unlabeled training data is denoted by $\mathcal{L}$ and $\mathcal{U}$ respectively.

3.1. Random Forests Based Multiple Models

Random Forests Classifier (RF) proposed by Breiman (Breiman, 2001) is used as our basic classifier for training on the set of labeled samples $\mathcal{L}$. It is one of the best algorithms for learning predictive models with very low generalization error for most of the datasets in this challenge. But we also need to consider the variance of output generated by RF. Inspired by another basic active learning strategy query-by-committee (Seung et al., 1992), we used the ensemble of multiple Random Forests models as the prediction committee to reduce the variance by computing the consensus posterior probability (CPP) of the multiple models, computed by the following function for each sample:

$$CPP(x) = \frac{1}{M}\sum_{m=1}^{M} P(y = 1|x; \theta_{(m)}) \tag{1}$$

where $y = 1$ represents the positive label; M is the number of models; $\theta_{(m)}$ is the Random Forest model with index m. On the other hand, we could assess the informativeness of samples by comparing the variance of multi-model outputs. This is another reason why we use multiple Random Forests instead of single one with large number of trees.

In terms of the parameter selections for RF, the method of cross validation would not be sound because we have to start from a small training set in active learning. Based on the recommendation from Breiman, we used a large number of trees (*ntree*) and square root of the number of variables as the default size of randomly selected subset of variables. It is also supported by our experimental findings: the higher number of trees for the RF classifier can generate a higher average prediction score and lower variance in prediction. But we could not use a very high number of trees due to time and memory constraints. Therefore, we picked $ntree = 4000$ and the default size of randomly selected subset of variables.

In terms of the number of RF models M in the multiple models, the prediction based on the consensus probability would be more stable when M is high. Samples with low variance values based on these RF prediction models also benefited our sample selection in the next step in active learning. We could have more accurate variance information if we use a higher number for M, however due to time constraint, we picked $M = 5$ RFs with

the same parameters for our experiments. Additional discussion on the effect of variance information for active learning is included in Section 3.3.

3.2. Querying Methods

We focused on the uncertainty sampling based query method that could naturally solve the active learning problems based on its sound theoretical properties (Lewis and Gale, 1994). However, the active learning starts from a very small training set so we could hardly get a good model at the beginning. Looking for the true uncertain samples is sometimes as hard as finding a true decision boundary or model. Secondly, due to the fact that the negative class size is larger than the positive one, we most likely query more samples with negative labels based on the basic uncertainty sampling method. This would intensify the imbalance in the class distribution of training set and the classifier would tend to ignore the minority class at the beginning of the active learning process.

We prefer to query the minority class (positive samples in the challenge) because (1) we assume that a single sample with smaller-fraction label is more informative and (2) we would like to make the training set relatively balanced in the early iteration of active learning. We present our implementation of Least Confidence with Bias (LCB) even when the prior of fraction of positive labels is not available. LCB is able to query the samples that are close to the decision boundary and are more likely to belong to the minority class. We expected LCB to converge and output the most uncertain samples more quickly than the basic uncertainty sampling with no bias. Moreover, our method does not rely on the constant prior of positive label percentage from the original data. The bias factor only considers the real-time fraction of positive label from the current training samples.

Let us define the query method $Q(\mathbf{x})$, the function we use to assess how informative each instance is in the unlabeled pool U. x^* is selected as the most informative sample according to the basic query function:

$$x^* = \arg\max_{x \in \mathcal{U}} Q(\mathbf{x}) \tag{2}$$

Let pp be the percentage of positive label in the current training set. We define P_{max} for the binary-class problem as follows: P_{max} is the consensus posterior probability that outputs the highest informative value in function $Q^{LCB}(\mathbf{x}, pp)$. We can also say that the instance with CPP equal to P_{max} is most informative. The LCB function follows:

$$Q^{LCB}(\mathbf{x}, pp) = \begin{cases} \frac{CPP(\mathbf{x})}{P_{\max}}; \text{if} CPP(\mathbf{x}) < P_{\max} \\ \frac{1-CPP(\mathbf{x})}{1-P_{\max}}; \text{otherwise} \end{cases} \tag{3}$$

$$P_{max} = mean(0.5, 1 - pp) \tag{4}$$

The output curve of LCB with P_{max} of 0.60 and 0.70 (or $pp = 0.3$ and 0.1) is shown in Figure 1. If the two classes are balanced, $pp = 0.5$ and P_{max} is 0.5, which is equivalent to the original Least Confidence (LC) function. If there are fewer positive samples than negative in the dataset, LCB is more likely to query the uncertain samples with positive label.

During the active learning process, the bias factor pp would be adaptively assigned based on two conditions: the positive fraction and the performance of the model. For

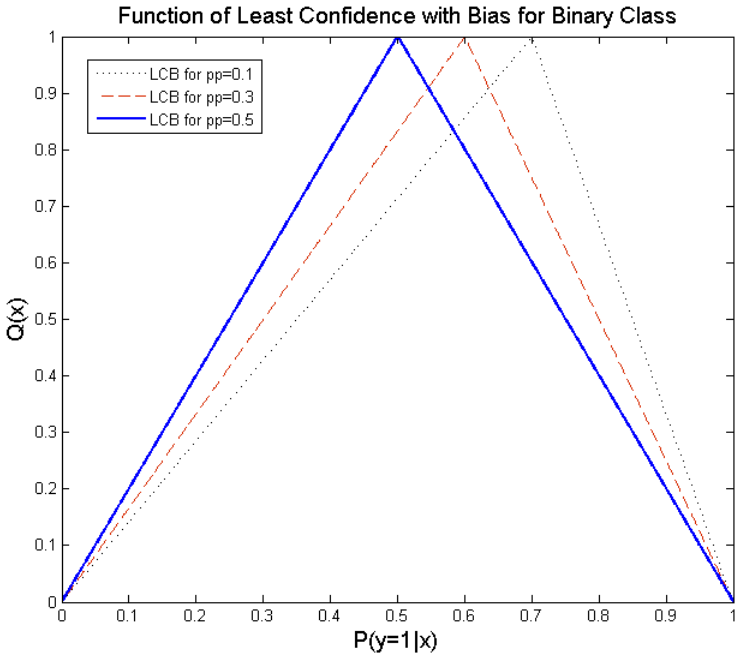


Figure 1: The function of least confidence with bias for different pp(Percentage of Positive label). If $pp = 0.5$, it is equivalent to original least confidence.

condition one, pp is just the positive fraction of current training set. But since we were given the prior knowledge that there are fewer positive samples than negative ones for all challenge datasets, it is not appropriate to bias to the negative class if the current fraction of positive samples is over 0.5. Thus we set $pp = 0.5$ if the positive fraction is over 0.5. For condition two, intuitively if our model is good enough, we would switch to the basic uncertainty sampling method. Although we could not obtain the performance metric in the final phase, we assume that when the size of the training set is sufficiently large, our model is adequate for predicting the most informative samples around the decision boundary. This user-defined threshold to switch between uncertainty sampling with and without bias is precisely assigned in Algorithm 2 in Section 4.

3.3. High-variance Filter

We considered the variance of posterior probabilities from multi-models to examine the factor of unstable performance. Although the consensus posterior probability was introduced, the beginning part of the learning curve still has a possibility of encountering a significant drop. For example, the point on the curve based on training with only two, four, eight, or sixteen training samples is much lower than the starting point. We did an independent experiment based on the variance from the multiple models. We queried and trained with two subsets of samples with the highest and lowest variance values of posterior probabil-

ities from multiple RF models based on previous prediction. It turns out that the AUC decrease or increase at the beginning of learning curve for the high-variance subset happens much more significantly and frequently than the low-variance one. Hence we introduced a filter to remove the samples with very high variance from the current querying list to increase the stability of performance. Eventually we prefer to query the samples with high informativeness and low variance in the early iteration of active learning. Let us define the filter variance threshold t such that samples with variance value greater than t will not be queried in the current iteration of active learning.

When we have a larger training set, the variance value for each sample would be gradually decreased since the individual model in the prediction committee would be more accurate and generate less disagreement with each other. We manually assign t for different development datasets such that there would be only one tenth of training samples which are not filtered out for querying at the first iteration of active learning.

3.4. Semi-supervised Learning Support

In our preliminary experiments, we only used cosine similarity function to do prediction by knowing just one positive sample. However, the performance was poor for most of the datasets since the cosine similarity is just not sufficient for label prediction. In the final phase, we performed an additional step for initial training: train the multiple Random Forests models with the positive seed along with a small number of predicted negative samples, which are predicted based on cosine similarity function. It generated better start points in the active learning framework for most of the development datasets. Here are the steps:

Algorithm 1: Semi-supervised learning for prediction based on one positive sample

Input: Data feature matrix $\mathbf{x}$ with one positive-labeled seed $y_1 = 1$
Output: CPP($\mathbf{x}$)

1. For all samples, estimate the cosine similarity to the positive-labeled seed;
2. Assign negative labels to K samples with the smallest cosine similarity values;
3. Train the multiple models with one given positive sample and K predicted negative samples and predict for other samples.

We also used this semi-supervised learning model to increase the size of training size by assigning more negative samples with the consensus posterior probability that was closest to negative label without purchasing the label. However, this approach could fail when the predicted negative samples are actually positive and the performance was adversely affected when we used too many predicted negative samples. Thus we need to limit the number of predicted negative samples in the early training.

3.5. Batch Size Validation

In the challenge, batch size (the number of instance to query at each iteration) is another user-defined factor. $B(i)$ represents the batch size at iteration i. Although we have proposed some ideas in order to improve the AUC score when the training set with labels is small, they may not work well uniformly for all datasets.

Batch size validation is proposed such that for each dataset we could find the smallest initial batch size by which our active learning method could generate the highest global score. The intuition is that we are unable to guarantee the performance when we just have a small number of purchased labels. The bad AUCs obtained with initial queries can adversely affect the global score, especially on the log2 scaling in the learning curve space, where the weight on each incremental query is decreased. The purpose of batch size validation for the classifier is to find the optimal initial batch size to reduce the chance of being negatively affected by overfitting when the training set is not large enough. The optimal initial batch size is the one which can achieve the highest global score in the development phase. In the final phase, we use the minimum sufficient initial batch size for final datasets accordingly.

This method can be applicable when we have a development dataset and final dataset, like the data in the challenge. For the active learning in general, we need to estimate the minimal initial batch size for the specific classifier and dataset, based on experience or an experiment such as batch size validation. Moreover, the idea could be extended to find not only the optimal initial batch size, but also the batch sizes in all other iterations.

4. Experiments and Results

4.1. Active Learning Procedure

Algorithm 2 in the next page presents the procedure we followed combining all the methods discussed earlier:

4.2. Results

We show the global scores in Table 3 and the corresponding learning curves in Figure 2 that represent our most recent performance in the development dataset based on the active learning procedure described in Section 4.1. In Figure 2, the x-axis for each graph represents the number of labels queried in Log2 domain, and the y-axis is AUC score.

We did batch size validation for only ZEBRA, IBN_SINA and NOVA and the results are shown in Figure 3. We did not do the same for ORANGE since Gaussian imputation for missing values could make the variance of output very high and unreliable. SYLVA obviously does not need the validation since the start point is already very high. And we were not sure if HIVA could exactly map to dataset A or C, so the default batch size was used.

For ZEBRA, the ALC score gradually increases when the initial batch size increases. It is obvious that our prediction model is constrained until we have a sufficiently large training set. The highest ALC score is achieved at almost the highest initial batch size. This score is about 20% more than the ALC with default initial batch size. So we applied this optimal initial batch size to final dataset E which is also with continuous features.

Algorithm 2: Active Learning Procedure with uncertainty sampling with biasing consensus (USBC)

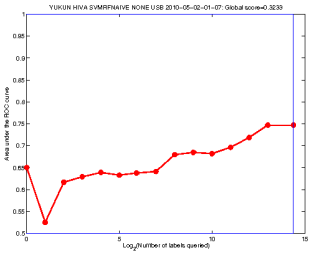
Input: Initial unlabeled set $\mathcal{U}$ containing data feature matrix $\mathbf{x}$ excluding the first labeled instance; the initial labeled set $\mathcal{L} = \{x_1, y_1 = 1\}$

Output: Global ALC score

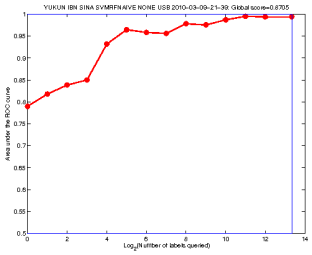
1. Initialization:
   (a) Run preprocessing steps (missing value imputation, PCA, etc) if needed;
   (b) Assign $B(i)$, the batch size as a function of iteration i, from 1 to i_{max} (the last iteration): the default is $B(i) = 2^{i-1}$, but it could be $B(i) = 2^{i+i_o}$, where i_o is 0, 1, 2, ..., i_{omax} (the max i_o that would query all samples at the beginning), depending on the batch size validation result;
2. CPP($\mathbf{x}$) = semi-supervised learning ($\mathbf{x}|\mathbf{y_1} = \mathbf{1}$) and output initial AUC;
3. Run $Q^{LCB}(\mathbf{x}, \mathbf{0.5})$ and query $B(1)$ sample(s) with the max Q($\mathbf{x}$), where $x \in \mathcal{U}$; update $\mathcal{U}$ and $\mathcal{L}$;
4. Run uncertainty sampling with biasing consensus (USBC) for i from 2 to i_{max}:
   (a) Add at most three predicted negative samples into the training sets (if activated);
   (b) Train by five RF models, predict for all unlabeled samples in $\mathcal{U}$ by CPP, and output AUC(i);
   (c) Run high-variance filter with parameter $t(i)$, temporally remove those $t(i)$ samples from $\mathcal{U}$ but added them back after current iteration (if activated);
   (d) Run $Q^{LCB}(\mathbf{x}, pp)$ and query $B(i)$ samples with the max Q($\mathbf{x}$) for true labels, where $x \in \mathcal{U}$, pp is the positive fraction for samples in $\mathcal{L}$ with two exceptions: if the positive fraction is larger than 0.5, $pp = 0.5$; if $|\mathcal{L}|$ is larger than $(|\mathcal{U}| + |\mathcal{L}|)/10$, $pp = 0.5$;
   (e) Update $\mathcal{U}$ and $\mathcal{L}$; loop over for $i = i + 1$ until $i > i_{max}$;
5. Output global ALC score

Table 3: The experimental results for development datasets

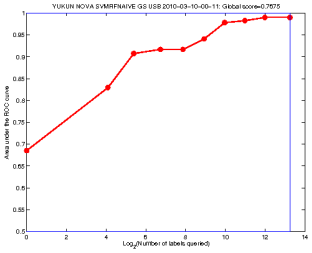
Data Name	ALC Score	AUC Score	Initial AUC Score	Initial Batch Size	Use Filter	Use Predicted Negative
HIVA	0.3233	0.7468 ± 0.79%	0.6502 ± 0.65%	1	No	No
IBN_SINA	0.8705	0.9960 ± 0.09%	0.7900 ± 0.28%	1	No	Yes
NOVA	0.7675	0.9940 ± 0.14%	0.6853 ± 0.38%	16	Yes	Yes
ORANGE	0.2037	0.7630 ± 1.11%	0.5170 ± 0.78%	1	No	Yes
SYLVA	0.9484	0.9990 ± 0.04%	0.8958 ± 0.22%	1	No	No
ZEBRA	0.5199	0.8318 ± 0.56%	0.6751 ± 0.48%	16384	No	No



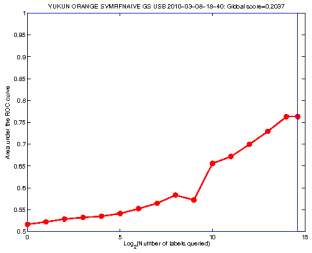
(*a*) Learning Curve HIVA



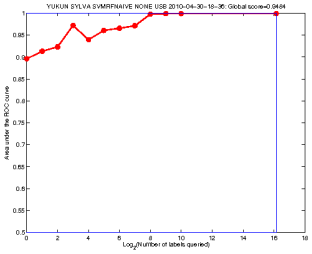
(*b*) Learning Curve IBN_SINA



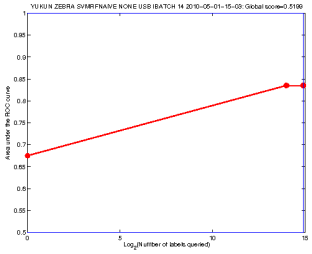
(*c*) Learning Curve NOVA



(*d*) Learning Curve ORANGE



(*e*) Learning Curve SYLVA



(*f*) Learning Curve ZEBRA

Figure 2: Learning Curves for Development Datasets.

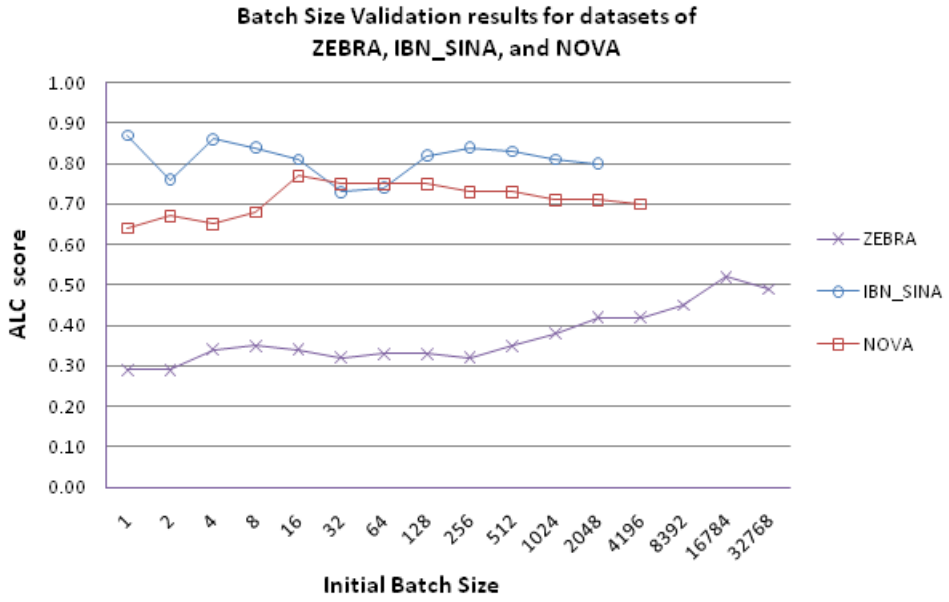


Figure 3: Batch size validation results for development datasets of ZEBRA, IBN_SINA, and NOVA

For IBN_SINA, the best ALC score was obtained when we used the default batch size. The ALC decreases with initial batch size increases.

For NOVA, the ALC score goes up when we increase the initial batch size to sixteen and then gradually decreases. Therefore initial batch size of sixteen is the minimum sufficient training set for the prediction of dataset NOVA by our multiple models. We applied this initial batch size to the final dataset D which also contains a very large number of binary features.

In Table 4, we show the final results for the datasets in the final phase. We compared our ALC scores and AUC scores (with initial AUC score) with the best ALC scores and AUC scores from participants in the challenge. Figure 4 shows the learning curves of our final results. In Figure 4, X-axis for each graph represents the number of labels queried in Log2 domain, and the y-axis is AUC score. These results are also available from the active learning challenge website: http://www.causality.inf.ethz.ch/activelearning.php?page=factsheet&id=157.

5. Discussion

Uncertainty sampling based querying method is highly dependent on the prediction model. Random Forest based prediction model may not be the best classifier for very-high-dimension, sparse and binary-feature datasets, like dataset ORANGE, HIVA, NOVA, B, C, and D. It did not perform well on datasets B and C. We actually anticipated the poor performance for datasets B and C because our multiple models did not have good prediction on OR-

Table 4: The results for final datasets

Data Name	Our ALC	Best ALC	Our AUC(Initial)	Best AUC	Initial Batch	Use Filter/ Pred-Neg	Final Rank
A	0.3609	0.6289	0.9615 ± 0.39% (0.7500)	0.9615	1	No/Yes	9
B	0.1297	0.3757	0.6484 ± 0.44% (0.5000)	0.7670	1	No/Yes	12
C	0.1876	0.4273	0.7715 ± 0.52% (0.4500)	0.8137	1	No/No	12
D	0.5390	0.8610	0.9554 ± 0.33% (0.4500)	0.9641	16	Yes/Yes	12
E	0.6266	0.6266	0.8939 ± 0.39% (0.7300)	0.9090	30000	No/No	1
F	0.7853	0.8018	0.9976 ± 0.09% (0.5500)	0.9990	1	No/No	3

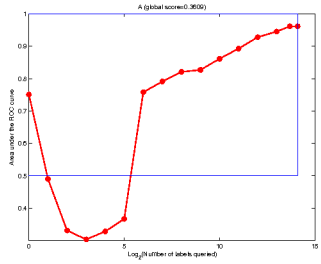
ANGE and HIVA which are similar to datasets B and C respectively. SVM based methods demonstrated reasonable performance for these types of datasets in this challenge from the other teams. However, Random Forest based prediction model turns out to be an excellent classifier for other datasets. Using our multiple Random Forests models for prediction based on training on almost all labeled training data, we obtained near-perfect performance on dataset F with over 99% AUC score and very good prediction on dataset A, D, and E.

We did not win on dataset D, which is similar to NOVA, because we had a very low initial AUC. It shows that our semi-supervised method did not work well in this type of datasets (see the initial AUC result for dataset B, C and D). However, the high-variance filter was effective on dataset D because the learning curve did not encounter any significant drop (see Figure 4(D)).

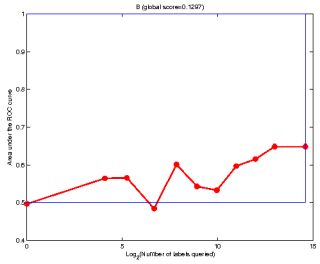
For dataset A, we did not perform well because our initial AUCs with no more than 32 training samples are lower than 0.5 (see Figure 4(A)). The use of predicted negative samples in the early iterations of active learning is not helpful and probably is the reason for our failure in dataset A as some of the predicted negative samples are actually positive. But when we have more than 32 training samples, our performance is better.

The uncertainty sampling with biasing consensus worked well on dataset F (with a score 0.02 less than the winner of dataset F). The learning curve (see Figure 4(F)) is what we expect: in general, the curve is exponentially increasing almost to the top at the beginning iterations and then maintaining the performance to the end. But our first four points on the learning curve were not good enough.

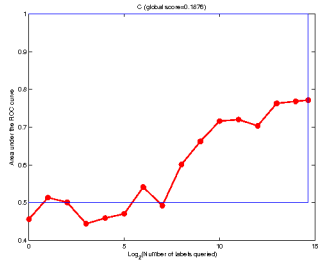
We won in the dataset E by querying 30000 out of 32252 training samples at the beginning. This is the winning strategy because of the batch size validation for ZEBRA, which is similar to dataset E. The semi-supervised learning that generates the starting point with 73% AUC also improved the global score on dataset E. The batch size validation result also indicates when to start active learning for the particular dataset and which method we need to use. On the other hand, we may not be able to use active learning process effectively. For example, it is hard to build an appropriate model with a small training set in dataset E. In this case, we would rather not use active learning but query all samples to achieve the best global score.



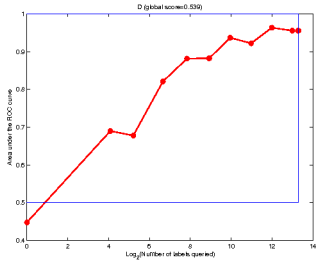
(a) Learning Curve A



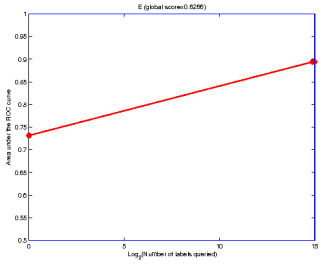
(b) Learning Curve B



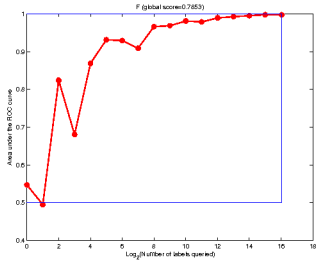
(c) Learning Curve C



(d) Learning Curve D



(e) Learning Curve E



(f) Learning Curve F

Figure 4: Learning Curves for Final Dataset A through F.

6. Conclusion and Future Work

In this paper, we presented our active learning strategies used in the final phase of active learning challenge. Apart from prediction model and query model, our strategy involved semi-supervised learning and batch size validation for active learning. However, we did not win more than one dataset like the other winners. Our methods need further evaluation using additional datasets. The active learning challenge is still an open problem to solve. One possible future direction to explore is to automatically assign batch size as a function of predictive performance and informativeness. We would also like to pursue other algorithmic methods that can consistently increase the learning curve and optimize the global score.

References

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

Y. Chen and S. Mani. Study of active learning in the challenge. 2010.

I. Guyon, G. Cawley, G. Dror, and V. Lemaire. Results of the active learning challenge. In *Active Learning Challenge*, volume 12, pages 19–45, 2011.

David D. Lewis and William A. Gale. 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, pages 3–12, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

H.S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the ACM Workshop on Computational Learning Theory*, pages 287–294, 1992.