An Active Learning Approach for Ensemble-based Data Stream Mining

Rabaa Alabdulrahman[1], Herna Viktor[1] and Eric Paquet[1,2]
[1]School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada
[2]National Research Council of Canada, Ottawa, Canada

Keywords: Online Learning, Data Streams, Active Ensemble Learning. Oracle.

Abstract: Data streams, where an instance is only seen once and where a limited amount of data can be buffered for processing at a later time, are omnipresent in today's real-world applications. In this context, adaptive online ensembles that are able to learn incrementally have been developed. However, the issue of handling data that arrives asynchronously has not received enough attention. Often, the true class label arrives after with a time-lag, which is problematic for existing adaptive learning techniques. It is not realistic to require that all class labels be made available at training time. This issue is further complicated by the presence of late-arriving, slowly changing dimensions (i.e., late-arriving descriptive attributes). The aim of active learning is to construct accurate models when few labels are available. Thus, active learning has been proposed as a way to obtain such missing labels in a data stream classification setting. To this end, this paper introduces an active online ensemble (AOE) algorithm that extends online ensembles with an active learning component. Our experimental results demonstrate that our AOE algorithm builds accurate models against much smaller ensemble sizes, when compared to traditional ensemble learning algorithms. Further, our models are constructed against small, incremental data sets, thus reducing the number of examples that are required to build accurate ensembles.

1 INTRODUCTION

Recently, there has been a surge of interest in the development of data stream algorithms that are not only accurate, but that are also fast and efficient in terms of resources allocation. This research has wide application in many areas. For instance, pocket (or mobile) data mining where the number of resources may be limited is relevant in scenarios such as emergency response, security and defense. In such a setting, the data are often incomplete and contains missing (or late arriving) labels. Further, green data mining, which aims to reduce the data mining processes' carbon footprint, is an important growth area in this era of Big Data. In both these setting, labelling all the data is both expensive and impractical.

Ensemble learning, where a number of so-called base classifiers are combined in order to build a model, has shown much promise when used in the online data stream classification setting. However, a number of challenges remain. It follows that the labelling process is costly and that missing (or incorrect) labels may hinder the model construction process. To this end, the use of active learning, where the user is in-the-loop, has been proposed as a way to extend ensemble learning (Sculley, 2007b, Sculley, 2007a, Chu et al., 2011). Here, the hypothesis is that active learning would increase the accuracy, while reducing the ensemble size and potentially decreasing the number of examples needed to build accurate model. That is, active ensembles potentially build accurate models against much smaller ensemble sizes, when compared to traditional ensemble learning approaches. Further, the models are constructed against smaller data sets, which imply that the wait time before a user is presented with a model is more likely to be reduced. This holds much benefit in scenarios such as emergency response and defense, where reducing decision makers' wait times are of crucial importance.

This paper introduces the active online ensemble (AOE) algorithm that extends online Bagging and online Boosting ensembles with an active learning component. In our approach, the human expert

Alabdulrahman, R., Viktor, H. and Paquet, E.
An Active Learning Approach for Ensemble-based Data Stream Mining.
DOI: 10.5220/0006047402750282
In *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2016) - Volume 1: KDIR*, pages 275-282
ISBN: 978-989-758-203-5

(oracle) is presented with small sets of examples for labelling. The proposed algorithm is tested on streams of instances, which is suitable for scenarios where new instances need to be classified one at a time, i.e. an incremental and online learning setting. In this scenario, the goal is to achieve high performance (in terms of accuracy) while utilizing as few labelled examples as possible.

This paper is organized as follows. The next section presents related works. We detail our active online ensemble method in Section 3. Section 4 describes the experimental evaluation. Finally, Section 5 concludes the paper.

2 RELATED WORK

Classifiers construct models that describe the relationship between the observed variables of an instance and the target label. However, as stated above, in a data stream setting, the labels may often be missing, incorrect or late arriving. Further, labelling involves domain expertise and may be costly to obtain.

Predictive models can be generated using classification methods. However, the produced model's accuracy is highly related to the labelled instances in the training set. Incorrectly classified instances can result in inaccurate, or biased models. Further a data set may be imbalanced, where one class dominates another. One suggested solution is to use active learning to guide the learning process (Stefanowski and Pachocki, 2009, Muhivumundo and Viktor, 2011). This type of learning tends to use the most informative instances in the training set.

Active learning studies how to select the most informative instances by using multiple classifiers. Generally, informative examples are identified as the ones that cause high disagreement among classifiers (Stefanowski and Pachocki, 2009). Thus, the main idea is using the diversity of ensemble learning to focus the labelling effort. This usually works by taking some information of the data from the users, also known as the *oracles*. In other words, the algorithm is initiated with a limited amount of labelled data. Subsequently, it passes them to the learning algorithm as a training set to produce the first classifier. In each of the following iterations, the algorithm analyses the remaining unlabelled instances and presents the prediction to the *oracle* (human expert) in order to label them. These labelled examples are added to the training set and used in the following iteration. This process is repeated until the user is satisfied or until a specific stopping criterion is achieved.

Past research in active learning mainly focused on the pool-based scenario. In this scenario, a large number of unlabelled instances need to be labelled. The main objective is to identify the best subset to be labelled and used as a training set (Sculley, 2007a, Chu et al., 2011). Hence, the basic idea behind active learning stems from the Query-by-Committee method, which is a very effective active learning approach that has wide application for labelling instances. Initially, a pool of unlabelled data is presented to the oracle, which is then selected for labelling. A committee of classifiers is trained and models are generated based on the current training data. The samples used for labelling are based on the level of disagreement in between the individual classifiers. In pool-based scenarios, the unlabelled data are collected in the candidate pool. However, in a data stream setting, maintaining the candidate pool may prove itself to be challenging as a large amount of data may arrive at high speed.

One of the main challenges in data stream active learning is to reflect the underlying data distribution. Such a problem may be solved by using active learning to balance the distribution of the incoming data in order to increase the model accuracy (Zliobaite et al., 2014). The distribution is adapted over time by redistributing the labelling weight as opposed to actively labelling new instances. Learn++ is another algorithm proposed by (Polikar et al., 2001) that employ an incremental ensemble learning methods in order to learn from data streams.

Also, traditional active learning methods require many passes over the unlabelled data, in order to select the informative one (Sculley, 2007a). This can create a storage and computational bottleneck in the data stream setting and big data. Thus, the active learning process needs to be modified for the online setting.

Another scenario is proposed by (Zhu et al., 2007) to address the data distribution associated with the data stream. Recall that in a data stream there is a dynamic data distribution because of the continuous arriving of data. In data stream mining, it is unrealistic to build a single model based on all examples. To address this problem, (Zhu et al., 2007) proposed an ensemble active learning classifier with the goal of minimizing the ensemble variance in order to guide the labelling process. One of the main objectives of active learning is to decide the newly arrived instances labels. According to the proposed framework in (Zhu et al., 2007), the correct labelling helps to reduce the overall classifier ensemble variance and error rate.

It follows that having a user-in-the-loop to guide and to label new instances may be costly and time inefficient (Zliobaite et al., 2014). However, it is very beneficial in scenarios where the absence of labelled data has a high incidence over the accuracy. In such scenarios, the use of a labelled instances pool is proposed. Here, the algorithms query for information from the domain expert and the answered queries result in building trained models (Chu et al., 2011). By using these models, the algorithm is able to bootstrap, and thus to label new data.

In our work, we extend this idea of maintaining an instance pool, where a small number of unlabelled instances are provided to the oracles. Additionally, we combine ensembles with online learning methods, as discussed in the following section.

3 ACTIVE ONLINE ENSEMBLE (AOE) ALGORITHM

In this research we developed two active online learning algorithms, namely active online Bagging (AOBagging) and active online Boosting (AOBoosting). Both of these approaches extend ensemble-based learning methods, namely Bagging and Boosting.

3.1 Bagging and Boosting Ensembles

Ensemble learning refers to the process of combining multiple models, such as classifiers or experts into a committee, in order to solve a computational problem. The main objective of using ensemble learning is to improve the model performance, such as classification and predictions accuracy (Read et al., 2012). This happens because if a single classifier predicts the wrong class value, the ensemble method takes into consideration the entire vote from all the trained classifiers. In this case, if one is incorrectly classified, the other correctly classified results will overcome it. In our framework, two active ensemble learning methods are used, namely Query-by-Bagging and Query-by-Boosting. These two methods are based on the well-known Bagging and Boosting ensemble methods, as summarized below.

The Bagging algorithm (also known as bootstrap aggregating (Breiman, 1996)), trains each classifier on a random sampled subset that is uniformly generated from the original data set by random sampling with replacement. The final prediction is made based on the different hypotheses resulting from different learners. Finally, averaging the output of the resulting hypotheses provides the final prediction. The Boosting algorithm also resamples the data but with a uniform distribution. Rather, each hypothesis resulted from different learners is weighted. The final prediction is based on a majority-weighted decision (Mamitsuka and Abe, 2007).

In the active ensemble learning setting, the Query-by-Bagging and Query-by-Boosting methods extend the idea of Query-by-Committee, where instances are selected from a pool. That is, the oracle is responsible for choosing the selected data, rather than the ensemble learner. Hence, instead of the random sampling as used by traditional Bagging and Boosting, the oracle is responsible for choosing a small number of labelled examples to create the first predictive model. Incrementally, additional examples may be chosen by the oracle, to be labelled based on the knowledge gained from the previous model. The oracle's involvement helps to improve the efficiency and reflect positively on the model accuracy.

In this setting, the total number of example must be known beforehand (Bifet and Kirkby, 2009). The probability of choosing an example to be added to the bootstrap follows a Binomial distribution (Bifet and Kirkby, 2009) whereas in Boosting, the number of examples must be known in order to calculate the weights associated with the instances (De Souza and Matwin, 2013) It follows that Query-by-Bagging and Query-by-Boosting inherits these limitations since they originated from the original Bagging/Boosting ensemble methods. Therefore, it is not suitable to directly apply these two active learning methods in a stream setting.

To address the previous mentioned limitation, we turn our attention to online, incremental stream learners. Specifically, Oza and Russell created incremental versions of the previously introduced Bagging and Boosting methods, namely OzaBag and OzaBoost (Oza, 2005). To handle data streams, Oza and Russell noted that the number N of examples is infinite ($N \rightarrow \infty$). In order to calculate the weight without the need of knowing the data set size, they updated the calculation by using a Poisson distribution (λ) which associate a probability to each example to be used in the training process (Bifet and Kirkby, 2009). In this method, the Poisson distribution (λ) parameter assigned to each example is increased if the base model misclassifies the instances and decreased if it correctly classifies it. Then, the new value is presented to the next model (Oza, 2005). We adopt the OzaBag and OzaBoost methods during our active learning phase.

The following subsections detail our active online ensemble (AOE) algorithm. As mentioned before, ensemble learning allows the algorithm to combine multiple classifiers in order to improve the classification or prediction accuracy. Further, online active learning uses a small number of labelled instances to guide the process of labelling the unlabelled ones. We present the most informative instances to the oracle (human expert) and then use the oracle's feedback to simultaneously build the trained models. That is, our active online ensemble learning framework uses the diversity of ensemble learning to create a number of models.

The overall workflow of our method is shown in Figure 1. The whole framework can be divided into two main stages. In the first stage, we utilize the previously introduced active learning methods, namely Query-by-Bagging and Query-by-Boosting. This is followed by online learning.

3.2 Active Ensemble Learning

Initially, we utilize ensemble learning methods to construct our initial classification model using the initial labelled training data. The class of the newly arrived unlabelled examples is predicted using the initial model. Next, the oracle evaluates the predicted values. Instances with high prediction probabilities are chosen and labelled by the oracle and then appended to the training set. Using the ensemble learning in this step guides the oracle toward the most informative labelled example by using multiple classifiers.

As a first step, we proceed to train ensembles of classifiers against the current window of the data stream. A set of models is constructed for each window (or data set) from the initial (labelled) data. Using the resulting models, the test data T_i is evaluated against the model. The outcome is a prediction value for each unlabelled example in the test set. These prediction values are subsequently presented to the oracle who chooses X examples from each class. Here, the oracle determines the numbers of examples that are chosen. Typically, the aim is to limit this number to range in between 10 and 20. In the current implementation, the oracle chooses 10 instances from each class and appends them to the training data set. This number was set by inspection. That is, the oracle is presented with the predictive probability of an unlabelled instance belonging to a class. Subsequently, the oracle selects the examples with the highest prediction probability. He/she proceeds to label these examples and append them to the original training data. Adding these newly labelled examples results in the new accumulated data set F_i that is tested repeatedly in the second stage and used to guide the learning process.

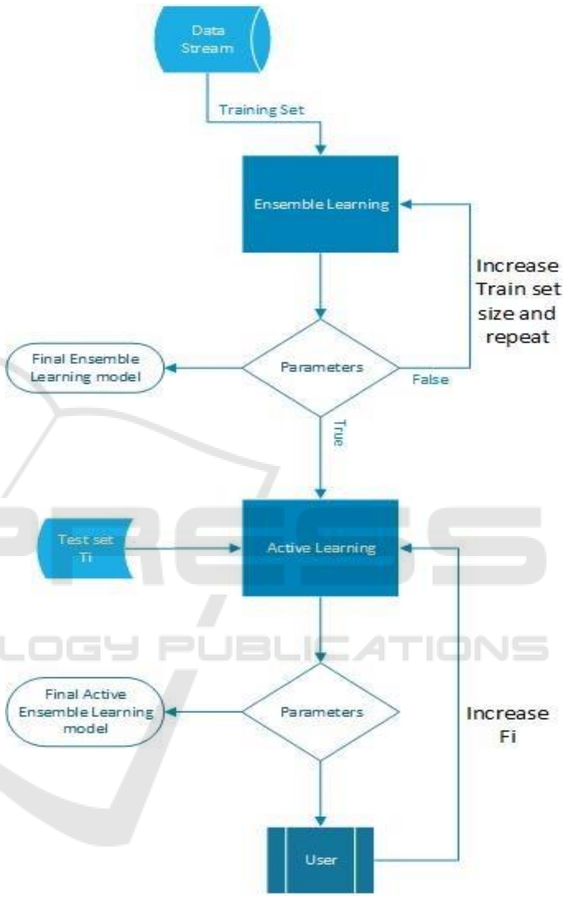

Figure 1: Active Online Ensemble Process.

3.3 Active Online Ensemble Learning

The second part of the AOE method involves online learning. Here, the augmented training data, as obtained from active learning, is used to build incremental models. As mentioned earlier, the oracle actively selects the informative examples and adds them to the training data. This results in the predictive model, which is subsequently fed to the online methods, namely OzaBag and OzaBoost. The predicted models from the previous stage are used to guide the learning process. Any new instances coming in the stream are labelled using the online ensemble methods resulting in improved classification accuracy.

Algorithm 1: Active Online Ensemble Learning.

Input:
h_i: Hypothesis obtained from the Ensemble Learning;
D_i: Training data resulted from the Ensemble Learning;
A: Active learning method (Online Boosting or Online Bagging);
T_i: Test set;
C_i: Base classifier;
F_i: Labelled data;
N: Ensemble size;
M: Number of models in the ensemble;
X: labelling set size (default 10)

Initiate $F_i = D_i$
For all training examples in do
1- **Test** T_i according to A on (h_i, C_i)
2- **Calculate** probabilities and select P_i
3- **Present ranked** P_i to Oracle Oracle confirm label of X instance from each class (with highest P_i value)
4- **Update** F_i with output from step 2: $F_i = F_i \cup X$
5- **Apply** A to (F_i, A)

Output the final hypothesis according to A

4 EXPERIMENTS

We conducted our experiments on a desktop with an Intel®(R) Core™(TM) i5-2410M CPU @ 2.30 GHz processor and with 8.00 GB of RAM.

We used four benchmarking data sets namely Waveform, Spambase, Chess and Australian, as summarized in Table 1. These data sets were obtained from the UCI Machine Learning repository (Lichman, 2013). The Spambase data set consists of emails, which were classified into Spam or Non-spam. Specifically, this data set includes a collection of 4,601 e-mails from the postmaster and individuals who filed spam. Also, the collection of non-spam emails came from filed work as well as personal emails. The Chess data set represents a chess endgame, where a pawn on A7 is one square away from queening. The task is to determine whether the player who plays with the White Chess pieces is able to win (or not). The overall size of this data set is 3196 with 36 attributes and it contains no missing values. The Waveform data set is formed from 5000 records having a threefold classification, corresponding to three classes of waves. Finally, the Australian data set contains information about credit card applications. In this data set, all the attributes' names and values have been altered in order to protect the confidentiality of the users. It consists of various data types.

For the implementation of our algorithms, we used the Weka and MOA Data Mining environments (Witten and Frank, 2005). MOA was specifically designed for data streams mining (Bifet and Kirkby, 2009). We evaluated the performances of our system against two based learners, namely the Hoeffding tree (HT) algorithm and the k-Nearest Neighbors (kNN) method. The model built with kNN is highly sensitive to the choice of *k*. For this reason, the values of *k* were determined by cross-validation, as suggested by Ghosh (Ghosh, 2006). Recall that we extended Oza's online versions of Bagging and Boosting. We also use these two algorithms in our comparisons.

Initially, each data set is normalized, a feature selection is performed and a reduced version is produced (Bryll et al., 2003). The attribute selection method was performed with Ranker with an Information Gain Attribute evaluator. This method process each attribute independently and is robust against missing values (Bryll et al., 2003). We utilized different ensemble sizes (10, 20, 25, 50, and 100 respectively). In all cases, the data sets were partitioned into test sets (90% to 95% of the instances) and training sets (5% to 10% of the instances). The test set was divided into 16 subsets of equal size. The test sets were randomly selected from the original test set. In all cases, the number of unlabelled instances selected from the test set and presented to the oracle, at specific time, was set to 10. These numbers were determined by inspection in order to avoid over-fitting. The MOA data stream generator was used in our work.

4.1 Accuracy versus Ensemble Size

Recall that one of the goals of this study is to determine the effect of ensemble sizes, when incorporating active learning into the ensemble learning process. For both AQBag and AQBoosting, the kNN classifier's accuracy is generally the best, in terms of accuracy, except for three experiments against the SpamBase data set where the Hoeffding tree has the best performances. The origin of such

Table1: Data sets used in our experimentation.

Data set	Size	#Attributes	#Classes	Data distribution	Data characteristic	Attribute characteristics	Missing values
Spambase	4601	57	2	Class 0=39.4%	Multivariate	Integer, Real	Yes
				Class 1= 60.5%			
Waveform	5000	40	3	Class0 = 33.84%	Multivariate, data generator	Real	No
				Class1 = 33.06%			
				Class2 =33.1%			
Chess	3196	36	2	Class0 = 52% Class1 = 48%	Multivariate	Categorical	No
Australian	690	14	2	Class0 = 44.5%	Multivariate	Categorical, Integer, Real	Yes
				Class1 = 55.5%			

behaviour is to be found in the very nature of kNN. Indeed, the classifier is a lazy learner that does not process the instances until the arrival of new unlabelled one. However, Observing new instance requires only updating the distance database. Therefore, the increase of the average classification time as the ensemble size increases is noticeable.

In most cases the active learning algorithms are able to build an accurate model, using small training sets, as shown in Table 3. This table shows that, for the Spambase, Waveform and Chess data sets, that the percentages of instances used during active learning are less than 21%. In the case of the Australian data set, the active learning process was only able to construct accurate models after 56% of the instances were labelled. Also, larger ensembles were needed for this data set. In general, our further analysis shows that active online learning often leads to smaller, more compact ensemble sizes than traditional ensemble learning, as shown in Table 4.

The only exception is in the case of the Bagging algorithm, when applied to the Australian data set. The results shown in Table 2 are the error rates for each data set for different ensemble sizes. As mentioned earlier, the test set is divided into 16 subsets. Therefore, the resulting error rate is the average of the error rates over the 16 subsets.

We further evaluated the results when using active learning (or not). The results indicate that, for our experiments, the active Bagging ensembles are generally smaller, than online Bagging ensembles.

In summary, a benefit of our approach is that the training sizes are smaller than the ones used by counterpart online versions. That is, we are able to

Table 2: Active Online Ensemble Learning - Summary of Results.

Data sets	Ensemble-size	AOBagging (HT)	AOBagging (kNN)	AOBoosting (HT)	AOBoosting (kNN)
Spambase	**10**	**3.8182%**	6.0000%	5.8182%	3.2727%
	20	4.1818%	8.5455%	6.3636%	5.6364%
	25	4.3636%	6.1818%	4.9091%	8.5455%
	50	4.3636%	5.8182%	5.4545%	**2.7273%**
	100	4.1818%	7.4545%	5.6364%	8.0000%
Waveform	**10**	16.2245%	**11.2245%**	16.6327%	12.3469%
	20	16.0204%	12.5510%	17.1429%	13.4694%
	25	16.3265%	16.0204%	15.9184%	**11.7347%**
	50	16.3265%	11.6327%	18.2653%	13.1633%
	100	16.0204%	11.6327%	16.7347%	15.1020%
Chess	**10**	8.5938%	**2.8125%**	3.7500%	**2.5000%**
	20	7.1875%	3.2813%	5.3125%	3.7500%
	25	8.1250%	2.9688%	3.5938%	3.9063%
	50	9.3750%	5.3125%	4.6875%	3.7500%
	100	7.3438%	5.4688%	4.0625%	4.3750%
Australian	10	5.9126%	3.3419%	7.1979%	4.6272%
	20	5.9126%	3.3419%	7.4550%	**3.8560%**
	25	5.9126%	3.8560%	8.4833%	5.6555%
	50	5.9126%	3.5990%	8.7404%	4.8843%
	100	5.9126%	**3.0848%**	7.7121%	4.1131%

Table 3: A summary of training subsets used.

	Spam-base	Wave-form	Chess	Australia
Data set size	4601	5000	3196	690
Training set size: first iteration	230	500	320	69
Training set size: last iteration	550	980	640	389
% Instances used	11.95	19.60	20.03	56.38

Table 4: Best results based on ensemble sizes with and without active learning.

Data set	Bagging	Boosting	AO Bagging	AO Boosting
Spambase	50	100	10	50
Wave form	25	25	10	25
Chess	20	25	10	10
Australian	50	25	100	20

build accurate models against smaller, incrementally growing training seta. This holds and advantage, especially in a big data setting.

4.2 Impact of Active Learning on Learning Time

Intuitively, the incorporation of active learning into an ensemble learning environment (batch as well as online) implies an additional step, as it involves the user-in-the-loop. It follows that this may lead to overhead in terms of time. The results are shown in Tables 5 and 6. The time does not include the time involved in the manual labelling process` by the oracle

The table shows that, in general, active learning does not significantly influence the model construction time. Rather, in many cases, the active learning process results in comparable and even faster times, as in the case of AOBagging using Hoeffding trees. An exception occurs when combining the kNN base learner with the Online Boosting approach. In this classifier, the distance between new instances to all the labelled sets is calculated and added to the distance matrix after each arrivals of a new instance. In addition, for each calculation, the algorithm needs to scan the entirety of the data which have been store so far in order to complete the calculation. Further, the fact that the Boosting algorithm demands the calculation of the instances weight after each new arrival has a negative impact on the classification of the new input neighbours (De Souza and Matwin, 2013).

4.3 Discussion

The presented framework provides valuable guidelines to data mining practitioners who aim to determine when to use the active learning process. It follows that active learning is essential in domains where very few labels exist. Considering the evaluated result and the data sets used in this work, we conclude that if the ensemble size is an important parameter, then online active learning may also be a good choice. That is, in a cost-sensitive learning setting where the size of the models is of importance, going toward the active ensemble route may be worthwhile. Overall, active online ensemble learning did not add a noticeable value to the model's accuracy. In most cases, it resulted in the same accuracy as in the ensemble learning or even increased the error rate. However, active learning leads to smaller ensembles, which may again be beneficial in a cost-sensitive learning setting.

Table 5: Average classification time for the ensemble methods measured in seconds.

	Bagging		Boosting	
Ensemble size	HT	kNN	HT	kNN
10	0.043	0.009	0.048	0.269
20	0.063	0.019	0.048	0.429
25	0.067	0.016	0.054	0.580
50	0.116	0.028	0.052	0.955
100	0.228	0.033	0.070	2.302

Table 6: Average classification time for the Active Online methods measured in seconds.

	AOBagging		AOBoosting	
Ensemble size	HT	kNN	HT	kNN
10	0.029	0.017	0.041	84.605
20	0.047	0.021	0.060	160.31
25	0.053	0.021	0.071	177.30
50	0.099	0.033	0.138	335.46
100	0.193	0.066	0.281	588.10

We further investigated the optimal size of the training sets. The result shows the classifiers' ability to be trained with a smaller set of data. Also, we were able to increase the performances of the classifier by only adding ten new classified instances from each class to each new accumulated training set.

5 CONCLUSION AND FUTURE WORK

This paper introduced an online active learning framework for data stream mining. In our work, we extended the online versions of Bagging and Boosting ensembles, in order to facilitate labeling of streaming data. Our results indicate that the active learning process requires smaller ensembles in order to obtain the same levels of accuracy than ensembles where the user in not in the loop. This is a promising result, especially from a cost-sensitive learning point of view. Our future research will involve additional experimental evaluation in order to investigate the decision points as to when to include active learning into a data stream. It follows that data streams are susceptible to concept drift. Our work did not explicitly address this issue and we plan to do so in the future.

REFERENCES

Bifet, A. & Kirkby, R. 2009. Data Stream Mining a Practical Approach. The University of Waikato: Citeseer.

Breiman, L., 1996. Bagging predictors. *Machine learning,* 24(2)**,** pp.123-140.

Bryll, R., Gutierrez-osuna, R. & Quek, F., 2003. Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets. *Pattern recognition,* 36(6)**,** pp.1291-1302.

Chu, W., Zinkevich, M., Li, L., Thomas, A. & Tseng, B., 2011. Unbiased online active learning in data streams. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 195-203). ACM.

De Souza, E. N. & Matwin, S., 2013, May. Improvements to Boosting with Data Streams. In *Canadian Conference on Artificial Intelligence* (pp. 248-255). Springer Berlin Heidelberg.

Ghosh, A. K., 2006. On optimum choice of k in nearest neighbor classification. *Computational Statistics & Data Analysis,* 50(11)**,** pp.3113-3123.

Lichman, M., 2013. UCI Machine Learning Repository [https://archive.ics.uci.edu/ml]. University of California, School of Information and Computer Science. Irvine, CA.

Mamitsuka, H. & Abe, N., 2007. Active ensemble learning: Application to data mining and bioinformatics. *Systems and Computers in Japan,* 38(11)**,** pp.100-108.

Muhivumundo, D. & Viktor, H., 2011. Detecting Data Duplication through Active Ensemble-based Learning. *The IEEE African Conference on Software Engineering and Applied Computing (ACSEAC 2011).* Cape Town, South Africa: IEEE.

Oza, N. C., 2005, October. Online bagging and boosting. In *2005 IEEE international conference on Systems, man and cybernetics* (Vol. 3, pp. 2340-2345). IEEE.

Polikar, R., Upda, L., Upda, S. S. & Honavar, V., 2001. Learn++: An incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* 31(4)**,** pp. 497-508.

Read, J., Bifet, A., Pfahringer, B. & Holmes, G., 2012, October. Batch-incremental versus instance-incremental learning in dynamic and evolving data. In *International Symposium on Intelligent Data Analysis* (pp.313-323). Springer Berlin Heidelberg.

Sculley, D., 2007a, August. Online Active Learning Methods for Fast Label-Efficient Spam Filtering. In *CEAS* (Vol. 7, p.143).

Sculley, D., 2007b, August. Practical learning from one-sided feedback. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 609-618). ACM.

Stefanowski, J. & Pachocki, M., 2009. Comparing performance of committee based approaches to active learning. *Recent Advances in Intelligent Information Systems***,** pp.457-470.

Witten, I. H. & Frank, E., 2005. Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann.

Zhu, X., Zhang, P., Lin, X. & Shi, Y., 2007, October. Active learning from data streams. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*(pp. 757-762). IEEE.

Zliobaite, I., Bifet, A., Pfahringer, B. & Holmes, G., 2014. Active learning with drifting streaming data. *IEEE transactions on neural networks and Learning Systems,* 25(1)**,** pp.27-39.