2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Using HDDT to avoid instances propagation in unbalanced and evolving data streams

Andrea Dal Pozzolo, Reid Johnson, Olivier Caelen,
Serge Waterschoot, Nitesh V Chawla and Gianluca Bontempi

Abstract— **Hellinger Distance Decision Trees [10] (HDDT) has been previously used for static datasets with skewed distributions. In unbalanced data streams, state-of-the-art techniques use instance propagation and standard decision trees (e.g. C4.5 [27]) to cope with the unbalanced problem. However it is not always possible to revisit/store old instances of a stream. In this paper we show how HDDT can be successfully applied in unbalanced and evolving stream data. Using HDDT allows us to remove instance propagations between batches with several benefits: i) improved predictive accuracy ii) speed iii) single-pass through the data. We use a Hellinger weighted ensemble of HDDTs to combat concept drift and increase accuracy of single classifiers. We test our framework on several streaming datasets with unbalanced classes and concept drift.**

HDDT, Hellinger distance, Unbalanced data, Data streams, Concept drift, Fraud detection.

I. Introduction

The explosion of data available everyday has increased the amount of data to process. When data arrive as a continuous stream of transactions, it is impossible to store all the observations. Therefore there is the need of tools that are able to process streams of data as soon as they arrive.

In data streams, the data distribution may change over the time. For this reason several techniques have been developed to deal with concept drift [30].

When the class distribution of a dataset is skewed, dataset is said to be unbalanced. In the static learning setting, the problem of learning in the case of unbalanced data has been widely explored [22]. Learning from non-stationary data streams with skewed class distribution is, however, a relatively recent domain. State-of-the-art techniques have addressed this problem by propagating minority observations between batches, with C4.5 decision tree being the most common algorithm used [7], [16], [20], [24]. All these techniques retain previous minority class instances in order to combat class imbalance. In this paper we have used Hellinger Distance Decision tree (HDDT) [9] as a base learner for data streams. This choice has allowed us to avoid instance propagation and produce superior performances in terms of predictive accuracy, computational time and resources needed.

In order to combat concept drift we have used a batch-ensemble model combination based on Hellinger Distance and Information Gain as in [24]. This choice has proved to be beneficial in the presence of changing distributions in the data. We have tested our framework with different types of datasets: unbalanced datasets without known concept drift, artificial datasets with known concept drift and a highly unbalanced credit card fraud dataset with concept drift.

Andrea Dal Pozzolo and Gianluca Bontempi are with the Machine Learning Group, Computer Science Department, Faculty of Sciences ULB, Université Libre de Bruxelles, Brussels, Belgium. (email: {adalpozz, gbonte}@ulb.ac.be).

Reid Johnson and Nitesh V Chawla are with the Data Inference Analytics and Learning Lab, Computer Science and Engineering Department, University of Notre Dame, Notre Dame IN, USA. (email: {rjohns15, nchawla}@nd.edu).

Olivier Caelen and Serge Waterschoot are with the Fraud Risk Management Analytics, Worldline, Belgium. (email: {olivier.caelen, serge.waterschoot}@worldline.com).

Research is supported by the Doctiris scholarship of Innoviris, Belgium and in part by the NSF Grant ECCS-0926170, US.

II. Unbalanced datasets problem

Learning from unbalanced datasets is a difficult task, since most learning algorithms are not designed to cope with a large difference between the number of cases belonging to different classes [2]. The unbalanced nature of the data is typical of many applications such as medical diagnosis, text classification and oil spill detection. Credit card fraud detection [25], [11], [27] is another well-known instance of a highly unbalanced problem since (fortunately) the number of fraudulent transactions is typically much smaller than legitimate ones.

In the literature, traditional methods for classification with unbalanced datasets rely on sampling techniques to balance the dataset [22]. In particular we can distinguish between methods that operate at the data and algorithmic levels [5].

At the data level, balancing techniques are used as a pre-processing step to rebalance the dataset or to remove the noise between the two classes, before any algorithm is applied. Data level techniques have the advantage of leaving the algorithms unchanged so that any algorithms can be tested. A well-known technique consists of *under* sampling the majority class by removing observations at random until the dataset is balanced [14]. *Undersampling* does not take into consideration any specific information in removing observations from the majority class, yet it is easy to implement and to understand.

At the algorithmic level, the classification algorithms themselves are adapted to deal with the minority class detection. An example of a classification algorithm designed for the unbalanced problem is HDDT, which will be discussed in section IV.

In this paper we will consider only binary classification tasks with unbalanced class distribution. We will call the

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

majority class negative (coded as $-$ or 0) and the minority class as positive (coded as $+$ or 1).

III. Hellinger Distance

Originally introduced to quantify the similarity between two probability distributions [28], the Hellinger distance has been recently proposed as a splitting criteria in decision trees to improve the accuracy in unbalanced problems [9], [10]. In the context of data streams, it has produced excellent results in detecting classifier performance degradation due to concept drift [8], [24].

Let (Θ, Q, λ) denote a measure space [19] where P denotes the set of all probability measures on Q that are absolutely continuous with respect to a probability measure λ. Consider two probability measures $P_1, P_2 \in P$. The Hellinger distance is defined as:

$$d_H(P_1, P_2) = \sqrt{\int_\Theta \left(\sqrt{\frac{dP_1}{d\lambda}} - \sqrt{\frac{dP_2}{d\lambda}} \right)^2 d\lambda} \quad (1)$$

Note that $d_H(P_1, P_2)$ does not depend on λ. If we replace λ with a different probability measure with respect to which both P_1 and P_2 are absolutely continuous, $d_H(P_1, P_2)$ remains the same.

For compactness we can rewrite equation 1 as:

$$d_H(P_1, P_2) = \sqrt{\int_\Theta \left(\sqrt{dP_1} - \sqrt{dP_2} \right)^2} \quad (2)$$

In the case of discrete distributions of a countable space Φ, the previous formula boils down to the following:

$$d_H(P_1, P_2) = \sqrt{\sum_{\phi \in \Phi} \left(\sqrt{P_1(\phi)} - \sqrt{P_2(\phi)} \right)^2} \quad (3)$$

Hellinger distance has several properties:

- $d_H(P_1, P_2) = d_H(P_1, P_2)$ (symmetric)
- $d_H(P_1, P_2) >= 0$ (non-negative)
- $d_H(P_1, P_2) \in [0, \sqrt{2}]$

d_H is close to zero when the distributions are similar and close to $\sqrt{2}$ for distinct distributions.

IV. Hellinger Distance Decision Trees

Starting from equation 3, Cieslak and Chawla [9] derive a new decision tree splitting criteria based on Hellinger distance that is skew insensitive. They start from the assumption that all numerical features are partitioned into p bins, so that the resulting dataset is made of only categorical variables. Then for each feature f, they compute the distance between the classes over all of the feature's partitions. In the case of a binary classification problem, where f_+ denotes the instances of the positive class and f_- the negatives, the Hellinger distance between f_+ and f_- is:

$$d_H(f_+, f_-) = \sqrt{\sum_{j=1}^{p} \left(\sqrt{\frac{|f_{+j}|}{|f_+|}} - \sqrt{\frac{|f_{-j}|}{|f_-|}} \right)^2} \quad (4)$$

where j defines the j^{th} bin of feature f. At each node of the tree, $d_H(f_+, f_-)$ is computed for each feature and then the feature with the maximum distance is used to split. The authors of [9] recommend to leave the tree unpruned and to use Laplace smoothing for obtaining probabilities from leaf frequencies. Note that the class priors do not appear explicitly in equation 4, which means that class imbalance ratio does not influence the distance calculation.

V. Learning from unbalanced and drifting data streams

A. Concept drift in data streams

In many real-work applications (e.g. intrusion detection, spam detection, fraud detection, etc.), the data is not available all at once, but it is received over time in streams of batches (e.g., daily internet usage dumps). In this scenario, the challenge is to use all the information up to a specific time step t to predict new instances arriving at time step $t+1$ [20].

However, the concept to learn can change due to non-stationary distributions. This problem is known as concept drift or non-stationary learning [30]. Let us define as Ω the function generating a data stream. For two points in time t and z such that $t \neq z$, in the case of concept drift we have $\Omega_t \neq \Omega_z$. This means that the assumption that the training and testing batches come from the same distribution may not hold. Since Ω is usually unknown, we cannot predict concept drift. An assumption traditionally accepted in data mining is that the class distribution remains constant. In streaming data, class prevalence can instead change over time, which means that in a stream one class can become over- or under-represented.

Let us define $X_t = \{x_0, x_1, ..., x_t\}$ as the set of labeled observations available at time t, where x_i is an n-dimensional vector. For a new unlabelled instance x_{t+1} we can train a classifier γ on X_t and predict $P(c_i|x_{t+1})$, the probability that the instance belongs to class c_i.

Using Bayes' theorem we can write $P(c_i|x_{t+1})$ as:

$$P(c_i|x_{t+1}) = \frac{P(c_i)P(x_{t+1}|c_i)}{P(x_{t+1})} \quad (5)$$

Since $P(x_{t+1})$ is the same for all classes c_i we can remove $P(x_{t+1})$:

$$P(c_i|x_{t+1}) = P(c_i)P(x_{t+1}|c_i) \quad (6)$$

Kelly [23] argues that concept drift can occur from a change in any of the terms in equation 6, namely:

- $P(c_i)$, class priors.
- $P(x_{t+1}|c_i)$, distribution of the classes.
- $P(c_i|x_{t+1})$, posterior distributions of class membership.

Change in $P(c_i)$ can cause well-calibrated classifiers to become miscalibrated. A change in the class priors can alter class distribution to the point of making the distribution unbalanced. Concept drift due to $P(x_{t+1}|c_i)$ affects the distribution of the observations within the class, but leaves the class boundary unchanged [20]. When $P(c_i|x_{t+1})$ changes,

there is a change in the class boundary that makes any previously learnt classifiers biased. The latter is the worst type of drift, because it directly affects the performance of a classifier, as the distribution of the features, with respect to the class, has changed [20].

B. Hellinger distance as weighting ensemble strategy

In evolving data streams it is important to understand how similar two consecutive data batches are in order to decide whether a model learnt on a previous batch is still valid. Lichtenwalter and Chawla [24] propose to employ Hellinger distance as a measure of the distance between two separate batches. Let us define as B^t the batch at time t used for training and B^{t+1} as the subsequent testing batch. First numeric features are discretized into equal-width bins, then Hellinger distance between B^t and B^{t+1} for a given feature f is calculated as:

$$HD(B^t, B^{t+1}, f) = \sqrt{\sum_{v \in f} \left(\sqrt{\frac{|B^t_{f=v}|}{|B^t|}} - \sqrt{\frac{|B^{t+1}_{f=v}|}{|B^{t+1}|}} \right)^2} \tag{7}$$

where $|B^t_{f=v}|$ is the number of instances of feature f taking value v in the batch at time t, while $|B^t|$ is the total number of instances in the same batch.

Equation 7 does not account for differences in feature relevance. In general, feature distance should have a higher weight when the feature is relevant, while a small weight should be assigned to a weak feature. Making the assumption that feature relevance remains stable over time, Lichtenwalter and Chawla [24] suggest to use the information gain to weight the distances.

For a given feature f of a batch B, the Information Gain (IG) is defined as the decrease in entropy E of a class c:

$$IG(B, f) = E(B_c) - E(B_c|B_f) \tag{8}$$

where B_c defines the class of the observations in batch B and B_f the observations of feature f.

For the testing batch we cannot compute $IG(B, f)$, as the labels are not provided, therefore the feature relevance is calculated on the training batch. The authors define a new distance function that combines IG and HD as:

$$HDIG(B^t, B^{t+1}, f) = HD(B^t, B^{t+1}, f) * (1 + IG(B^t, f)) \tag{9}$$

$HDIG(B^t, B^{t+1}, f)$ provides a relevance-weighted distance for each single feature. The final distance between two batches is then computed taking the average over all the features.

$$D_{HDIG}(B^t, B^{t+1}) = \frac{\sum_{f \in B^t} HDIG(B^t, B^{t+1}, f)}{|f \in B^t|} \tag{10}$$

The authors suggest to learn a new model as soon as a new batch is available and store all the models. The learnt models are then combined into an ensemble where the weights of the models are inversely proportional to the batches' distances. The lower the distance between two batches the more similar is the concept between them. In a streaming environment with concept drift we should expect good performances on the current batch from models learnt on similar concepts. With this reasoning in mind, the ensemble weights should be higher for smaller distances. The authors suggest to use the following transformation:

$$weights_t = D_{HDIG}(B^t, B^{t+1})^{-b} \tag{11}$$

where b represents the ensemble size.

C. Related work in unbalanced data streams

The online learning problem typical of streaming data has attracted a lot of attention in research, however little has been done to tackle the streaming problem where the class distribution is skewed [20]. Credit card fraud detection is an example of a data source that suffers from class imbalance. The number of frauds occurring in each chunk of the stream is usually less than 1% [12]. As the type of fraudulent activity can evolve in time, the learning strategy has to adapt to concept drift. State-of-the-art methods for unbalanced data streams with concept drift combine ensemble methods with sampling.

Gao's framework [16], [17] addresses the unbalanced problem of a chunk by propagating past minority class observations and undersampling the majority class. The positive examples are accumulated along the stream until they represent 40% of the observations. When this happens, the oldest positive examples are replaced by the new minority class observations. This propagation method ignores the similarity of the minority class instance to the current concept, relying only on its similarity in time.

REA [7] and SERA [6] proposed by Chen and He propagate to the last chunk only minority class that belong to the same concept using Mahalanobis distance and a *k*-nearest neighbors algorithm.

With Learn++.NIE [13], Ditzler and Polikar extend their own Learn++.NSE [15] method for unbalanced datasets. For each batch they create different balanced subsets using undersampling and then combine models learnt on each balanced subset.

Lichtenwalter and Chawla [24] suggest to propagate not only positives, but also observations from the negative class which are misclassified in the previous chunk to increase the boundary definition between the two classes.

Hoens and Chawla in HUWRS.IP [20] adapt HUWRS [21] for unbalanced streams introducing an instance propagation mechanism based on a Naïve Bayes classifier. Naïve Bayes is used to select old positive instances which are relevant to the current minority class context. This method relies on finding instances that are similar to the current minority class context. In some cases of rapid drift, however, such instances may not be available.

Wang, Minku and Yao [29] propose Sampling-based Online Bagging (SOB) to deal with unbalanced data streams. Their algorithm, is essentially a modification of Online

Bagging [26], in which the sampling rate of the instances belonging to one class is determined adaptively based on the current imbalance status and classification performance. The problem with this approach is that it is not designed to handle concept drifts as it aims to maximize G-mean greedily over all received examples [29].

VI. Experimental setup

Many of the data streaming frameworks for concept drift and unbalanced data use C4.5 [27] decision tree as the base learner [21], [20], [24], [16]. In our experiments we compared the results of C4.5 to the Hellinger Distance Decision Tree (HDDT) with the parameters suggested in [9] (unpruned and Laplace smoothing). The comparison is done using different propagation/sampling methods and model combinations (ensemble vs. single models).

In an unbalanced data stream, for each batch/chunk, the positive class examples represent the minority of the observations. Each batch can be considered as a small unbalanced dataset, permitting all the techniques already developed for static unbalanced datasets to be implemented. In a streaming environment, however, it is possible to collect minority observations from previous batches to combat the class skewness. For our experiments we considered instance propagation methods that assume no sub-concepts within the minority class. In particular we used Gao's [16] and Lichtenwalter's [24] propagation methods presented in section V-C and two other benchmark methods (UNDER and BL):

- SE (Gao's [16] propagation of rare class instances and undersampling at 40%)
- BD (Lichtenwalter's Boundary Definition [24]: propagating rare-class instances and instances in the negative class that the current model misclassifies.)
- UNDER (Undersampling: no propagation between batches, undersampling at 40%)
- BL (Baseline: no propagation, no sampling)

The first two methods can be considered as oversampling methods since the minority proportion in the batches is augmented. From now on, for simplicity we will call all the previously discussed instance propagation methods *sampling* strategies.

For each of the previous sampling strategies we tested:

- HDIG: D_{HDIG} weighted ensemble.
- No ensemble: single classifier.

In the first case, an ensemble is built combining all models learnt with weights given by equation 11.

In the second case, we use the model learnt in the current batch to predict the incoming batch. This option has the advantage of being faster as no models are stored during the learning phase.

VII. Experimental results

In all our experiments we reported the results in terms of AUROC (area under the ROC curve) as it is *de facto* standard in unbalanced problems [4]. The framework was implemented in Java and we used the Weka [18] implementation of C4.5 and HDDT.

A. Datasets

In our experiments we used different types of datasets. We used benchmark UCI datasets [1] to first study the unbalanced problem without worrying about concept drift. These datasets are not inherently sequential and exhibit no concept drift; we render them as data streams by randomizing the order of instances and processing them in batches as in [24]. Then we used the MOA [3] framework to generate some artificial datasets with drifting features to test the behavior of the algorithms under concept drift. Finally we used a real-world credit card dataset which is highly unbalanced and whose frauds are changing in type and distribution. This dataset contains credit card transactions from online payment between the 5^{th} of September 2013 and the 25^{th} of September 2013, where only 0.15% of the transactions are fraudulent. It was provided by a Belgian payment processor, but for confidentially reasons we cannot reveal more about this data.

TABLE I
Datasets

Name	Source	Instances	Features	Imbalance Ratio
Adult	UCI	48,842	14	3.2:1
can	UCI	443,872	9	52.1:1
compustat	UCI	13,657	20	27.5:1
covtype	UCI	38,500	10	13.0:1
football	UCI	4,288	13	1.7:1
ozone-8h	UCI	2,534	72	14.8:1
wrds	UCI	99,200	41	1.0:1
text	UCI	11,162	11465	14.7:1
DriftedLED	MOA	1,000,000	25	9.0:1
DriftedRBF	MOA	1,000,000	11	1.02:1
DriftedWave	MOA	1,000,000	41	2.02:1
Creditcard	FRAUD	3,143,423	36	658.8:1

B. Results

We first tested the different sampling strategies using HDDT and C4.5. One the left side of Figure 1 we see the results where D_{HDIG} distance discussed in section V-B is used to weight the models from different batches according to equation 11. On the right are the results where only the model of the current batch is used for prediction. The columns indicate the batch mean AUROC for each strategy averaged over all UCI datasets. This means that for each dataset we computed the mean AUROC over all batches and then average the results between all datasets. In general we notice that HDDT is able to outperform C4.5. For each sampling method we see that the ensembles counterpart of the single models have higher accuracy.

In Figure 2 we display the average computational time. As expected, when a single classifier is used the framework is much faster, but it comes at the cost of lower accuracy (see Figure 1). When UNDER sampling is used in the framework we have the smallest computational time, as it uses a subset of the observations in each batch and no instances are propagated between batches.

Figure 3 shows the results for the datasets with concept drift generated using the MOA framework. HDIG-based

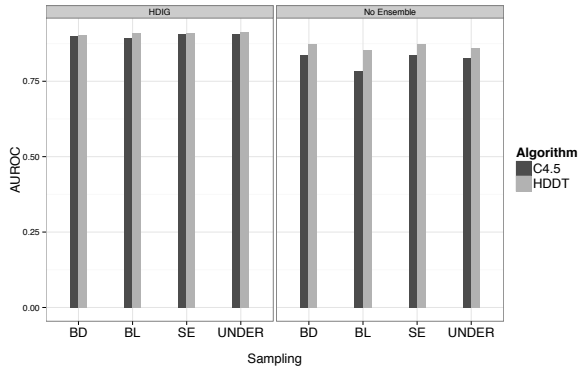

Fig. 1. Batch average results in terms of **AUROC** (higher is better) using different sampling strategies and batch-ensemble weighting methods with C4.5 and HDDT over all **UCI datasets**.

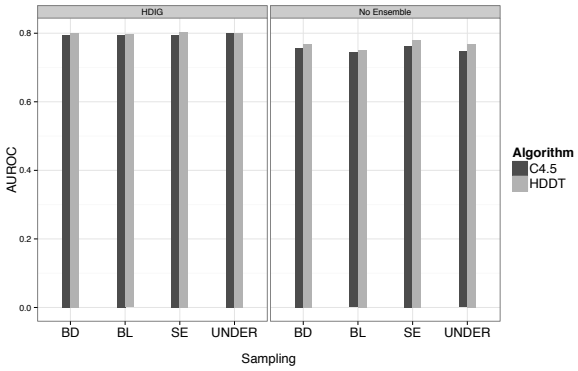

Fig. 3. Batch average results in terms of computational **AUROC** (higher is better) using different sampling strategies and batch-ensemble weighting methods with C4.5 and HDDT over all drifting **MOA datasets**.

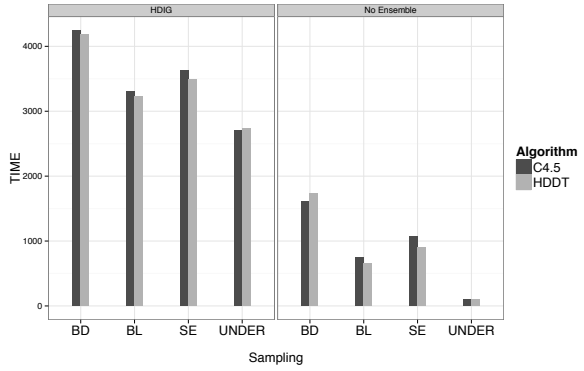

Fig. 2. Batch average results in terms of computational **TIME** (lower is better) using different sampling strategies and batch-ensemble weighting methods with C4.5 and HDDT over all **UCI datasets**.

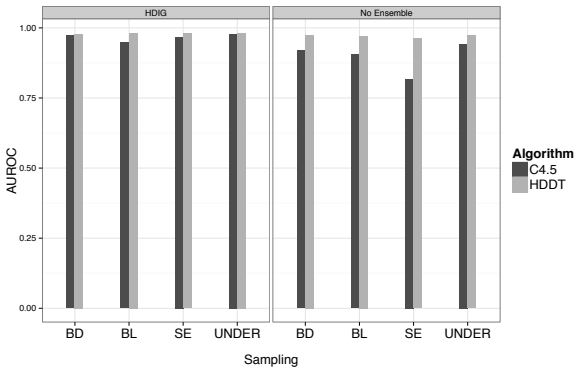

Fig. 4. Batch average results in terms of **AUROC** (higher is better) using different sampling strategies and batch-ensemble weighting methods with C4.5 and HDDT over **Credit card dataset**.

ensembles return better results than a single classifier and HDDT again gives better accuracy than C4.5.

Figure 4 displays the results on the Credit card dataset. This dataset is a good example of an unbalanced data stream with concept drift. Once again HDDT is always better than C4.5, however the increase in performance given by the ensemble is less important than the one registered with the UCI datasets. From Figure 4, it is hard to discriminate the best strategy, as many of the them have comparable results.

Figure 5 shows the sum of the ranks for each strategy over all the chunks. For each chunk, we assign the highest rank to the most accurate strategy and then sum the ranks over all chunks. Let $r_{s,k} \in \{1, ..., S\}$ be the rank of strategy s on chunk k and S be the number of strategies to compare. The strategy with highest AUROC in k has $r_{s,k} = S$ and the one with the lowest has $r_{s,k} = 1$. Then the sum of ranks for the strategy s is defined as $\sum_{k=1}^{K} r_{k,s}$, where K is the total number of chunks. The higher the sum, the higher the number of times one strategy is superior to the others.

The strategy with the highest sum of ranks (*BL_HDIG_HDDT*) combines BL with HDIG ensembles of HDDTs. BL method leaves the batches unbalanced, which means that the best strategy is actually the one avoiding instance propagation/sampling. A paired t-test on the ranks was then used to compare each strategy with the best. Based on this test, we saw that the strategy with the second-highest sum of ranks (*UNDER_HDIG_HDDT*) is not significantly worse than the first. Compared to the first, this strategy implements UNDER sampling at each batch instead of BL. Figure 5 confirms that HDDT is better than C4.5. The C4.5 implementation of the winning strategy (*BL_HDIG_C4.5*) is significantly worse than the best (*BL_HDIG_HDDT*). The same happens for the second best ranking strategy (*UNDER_HDIG_HDDT* ranks higher than *UNDER_HDIG_C4.5*).

VIII. Conclusion

To our knowledge, our work is the first to evaluate the use of the HDDT tree algorithm for streaming data. Many of the state-of-the-art techniques use the C4.5 algorithm combined with sampling or instance propagation to balance the batches before training. We have shown that when used

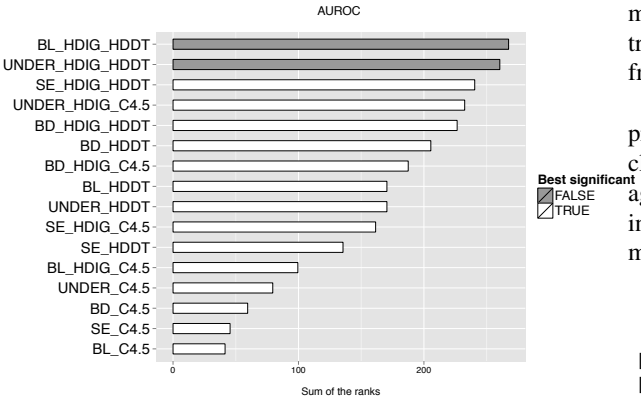

Fig. 5. Comparison of different strategies using the sum of ranks in all chunks for the **Credit card dataset** in terms of **AUROC**. In gray are the strategies that are not significantly worse than the best having the highest sum of ranks.

in data streams, HDDT without sampling typically leads to better results than C4.5 with sampling. Thus, HDDT can offer better performance than C4.5, while actually removing sampling from the process.

The removal of the propagation/sampling step in the learning process has several benefits:

- It allows a single-pass approach (the observations are processed as soon as they arrive, avoiding several passes throughout the batches for instance propagation).
- It reduces the computational cost/resources needed (this is important since with massive amounts of data it may no longer be possible to store/retrieve old instances).
- It avoids the problem of finding previous minority instances from the same concept (in the case of a new concept in the minority class, it may not be possible to find previous observations to propagate).

For these reasons we think our framework is more efficient than state-of-the-art methods for unbalanced data streams.

We have used artificial datasets to test how different strategies work under concept drift. The use of HDIG as an ensemble weighting strategy has increased the performances of the single classifiers, not only in artificial datasets with known drift (MOA datasets), but even in datasets whose distribution is assumed to be more or less stable (UCI datasets).

Finally, we tested our framework on a proprietary dataset containing credit card transactions from online payment. This is a particularly interesting dataset, as it is extremely unbalanced and exhibits concept drift within the minority class. HDDT performs very well when combined with BL (no sampling) and UNDER sampling. An important feature of these basic sampling strategies is the fact that frameworks implementing them are much faster (see Figure 2) since no observations are stored from previous chunks. When these two sampling strategies give comparable results, the practitioner could prefer UNDER sampling as it is more memory efficient since it uses a reduced part of the batch for training. By using undersampling, however, a lot of instances from the majority class are not considered.

In our experiments we compared our framework with propagation/sampling methods that consider the minority class as a single cluster. Future work will investigate propagation methods such as REA, SERA and HUWRS.IP seen in section V-C that are able to deal with sub-concepts in the minority class.

REFERENCES

[1] D. N. A. Asuncion. UCI machine learning repository, 2007.

[2] G. Batista, A. Carvalho, and M. Monard. Applying one-sided selection to unbalanced datasets. *MICAI 2000: Advances in Artificial Intelligence*, pages 315–325, 2000.

[3] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer. Moa: Massive online analysis. *The Journal of Machine Learning Research*, 99:1601–1604, 2010.

[4] N. V. Chawla. Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook*, pages 853–867. Springer, 2005.

[5] N. V. Chawla, N. Japkowicz, and A. Kotcz. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter*, 6(1):1–6, 2004.

[6] S. Chen and H. He. Sera: selectively recursive approach towards nonstationary imbalanced stream data mining. In *Neural Networks, 2009. IJCNN 2009. International Joint Conference on*, pages 522–529. IEEE, 2009.

[7] S. Chen and H. He. Towards incremental learning of nonstationary imbalanced data stream: a multiple selectively recursive approach. *Evolving Systems*, 2(1):35–50, 2011.

[8] D. A. Cieslak and N. V. Chawla. Detecting fractures in classifier performance. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 123–132. IEEE, 2007.

[9] D. A. Cieslak and N. V. Chawla. Learning decision trees for unbalanced data. In *Machine Learning and Knowledge Discovery in Databases*, pages 241–256. Springer, 2008.

[10] D. A. Cieslak, T. R. Hoens, N. V. Chawla, and W. P. Kegelmeyer. Hellinger distance decision trees are robust and skew-insensitive. *Data Mining and Knowledge Discovery*, 24(1):136–158, 2012.

[11] P. Clark and T. Niblett. The cn2 induction algorithm. *Machine learning*, 3(4):261–283, 1989.

[12] A. Dal Pozzolo, O. Caelen, S. Waterschoot, and G. Bontempi. Racing for unbalanced methods selection. In *Proceedings of the 14th International Conference on Intelligent Data Engineering and Automated Learning*. IDEAL, 2013.

[13] G. Ditzler and R. Polikar. An ensemble based incremental learning framework for concept drift and class imbalance. In *Neural Networks (IJCNN), The 2010 International Joint Conference on*, pages 1–8. IEEE, 2010.

[14] C. Drummond, R. Holte, et al. C4. 5, class imbalance, and cost sensitivity: why under-sampling beats over-sampling. In *Workshop on Learning from Imbalanced Datasets II*. Citeseer, 2003.

[15] R. Elwell and R. Polikar. Incremental learning of variable rate concept drift. In *Multiple Classifier Systems*, pages 142–151. Springer, 2009.

[16] J. Gao, B. Ding, W. Fan, J. Han, and P. S. Yu. Classifying data streams with skewed class distributions and concept drifts. *Internet Computing*, 12(6):37–49, 2008.

[17] J. Gao, W. Fan, J. Han, and S. Y. Philip. A general framework for mining concept-drifting data streams with skewed distributions. In *SDM*, 2007.

[18] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[19] P. R. Halmos. *Measure theory*, volume 2. van Nostrand New York, 1950.

[20] T. R. Hoens and N. V. Chawla. Learning in non-stationary environments with class imbalance. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–176. ACM, 2012.

[21] T. R. Hoens, N. V. Chawla, and R. Polikar. Heuristic updatable weighted random subspaces for non-stationary environments. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 241–250. IEEE, 2011.

[22] N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[23] M. G. Kelly, D. J. Hand, and N. M. Adams. The impact of changing populations on classifier performance. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 367–371. ACM, 1999.

[24] R. N. Lichtenwalter and N. V. Chawla. Adaptive methods for classification in arbitrarily imbalanced and drifting data streams. In *New Frontiers in Applied Data Mining*, pages 53–75. Springer, 2010.

[25] L. Olshen and C. Stone. Classification and regression trees. *Wadsworth International Group*, 1984.

[26] N. C. Oza. Online bagging and boosting. In *Systems, man and cybernetics, 2005 IEEE international conference on*, volume 3, pages 2340–2345. IEEE, 2005.

[27] J. R. Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[28] C. R. Rao. A review of canonical coordinates and an alternative to correspondence analysis using hellinger distance. *Questiió: Quaderns d'Estadística, Sistemes, Informatica i Investigació Operativa*, 19(1):23–63, 1995.

[29] S. WANG, L. L. MINKU, and X. YAO. Online class imbalance learning and its applications in fault detection. *International Journal of Computational Intelligence and Applications*, 12(04), 2013.

[30] I. Zliobaite. Learning under concept drift: an overview. Technical report, Overview, Technical report, Vilnius University, 2009 techniques, related areas, applications Subjects: Artificial Intelligence, 2009.