2016 23rd International Conference on Pattern Recognition (ICPR)
Cancún Center, Cancún, México, December 4-8, 2016

A Benchmark of Classifiers on Feature Drifting Data Streams

Jean Paul Barddal, Heitor Murilo Gomes, Alceu de Souza Britto Jr., Fabrício Enembreck
Graduate Program in Informatics (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR)
{jean.barddal, hmgomes, alceu, fabricio}@ppgia.pucpr.br
Curitiba, Brazil

Abstract—**The ever increasing data generation confronts both practitioners and researchers on handling massive and sequentially generated amounts of information, the so-called data streams. In this context, a lot of effort has been put on the extraction of useful patterns from streaming scenarios. Learning from data streams embeds a variety of problems, and by far, the most challenging is concept drift, i.e. changes in data distribution. In this paper, we focus on a specific type of drift uncommonly assessed in the literature: feature drifts. Feature drifts occur whenever a subset of features becomes, or ceases to be, relevant to the concept to be learned. We propose and review several feature drifting data stream generators and use them to benchmark state-of-the-art data stream classification algorithms and their combination with drift detectors. Results show that, although drift detectors enable slight quicker recovery to feature drifts, best results are obtained by Hoeffding Adaptive Tree, the only learner that performs dynamic feature selection as streams progress.**

I. Introduction

The ever increasing data generation confronts both practitioners and researchers on handling massive amounts of data generated online, the so-called data streams. Examples of data streams include, but are not limited to, consumer click streams, telephone usage flows, multimedia data mining [1], computer networks intrusion detection [2] and stock market share exchanges [3].

In the context of streaming environments, several proposals were developed in the last years aiming at extracting useful patterns from these in both supervised [4], [5], [6] and unsupervised [7] fashions. In both cases, stream learning algorithms must process instances one at a time and inspect them only once. There is no restriction against storing instances for a short time, as long it does not jeopardize overall processing time nor maximum allowed memory space. Therefore, processing time and memory space are intimately related, since if the learning algorithms take too long to classify each instance, they will start accumulating in memory, until an overflow occurs and the system crashes [6].

Albeit latter restrictions, data streams are possibly ephemeral, it is, their underlying data distribution may shift through time. This enforces learners to either discard and learn a new model or adapt the existing model so it reflects the new reality.

More recently, authors in [8], [9] and [10] highlighted a specific kind of drift that is uncommonly tackled by existing learning algorithms, namely feature drift. Practically, a feature drift occurs whenever a subset of features becomes, or ceases to be, relevant to the learning task.

None of the early works into feature drifts provide an analysis into how feature drifts can be synthesized and what is their impact on prediction rates of state-of-the-art data stream learners. In this paper, we fill this gap by (i) presenting a variety of data stream generators capable of synthesizing feature drifts, and (ii) evaluating several state-of-the-art data stream learners.

As contributions of this paper, we cite:

1) A review and proposal of data generators capable of synthesizing feature drifts;
2) An extensive empirical evaluation of state-of-the-art data stream classifiers over feature drifting data streams; and
3) The provision of insights onto why feature selection is needed when learning from data streams.

We start this paper with a review of the classification task for data streams and the phenomenon of concept drift (Sec. II). Later, we formalize feature drifts accordingly to previous works [10], [9], [8] (Sec. III) and present data stream generators capable of synthesizing this type of drift (Sec. IV). Afterward, we benchmark state-of-the-art data stream classification algorithms and drift detectors in a variety of experiments (Sec. V), showing their behavior when confronted with feature drifts. Finally, we conclude this paper and discuss the need for future research into feature drifts adaptation (Sec. VI).

II. Data Streams, Classification and Concept Drift

Let $\mathcal{S} = [i_t = (\vec{x}_t, y_t)]_{t=0}^{t \to \infty}$ denote a data stream providing instances i_t each of which arriving at a timestamp t, where $\vec{x}_t$ is a d-dimensional feature vector belonging to a feature set $\mathcal{D} = [D_j]_{j=1}^{d}$, while $y_t \in Y$ is its ground-truth label.

The most common approach for extracting useful knowledge from data streams is classification. Classification is the task that distributes a set of instances into discrete classes accordingly to relations or affinities. Assuming a set of possible classes $Y = \{y_1, \ldots, y_c\}$, a classifier builds a model $f : \vec{x} \to Y$ that predicts for every unlabeled instance $\vec{x}_i$ its corresponding class y_i accurately [11]. Therefore, given a data stream $\mathcal{S}$, a classifier must produce and update a model f as new instances become available [11], [12].

Due to the temporal trait of data streams, these are expected to be ephemeral, thus, they are expected to undergo changes in their data distributions, a phenomenon named concept drift. Let Eq. 1 denote a concept, i.e. a set of prior probabilities of classes and class-conditional probability density function.

$$C = \bigcup_{y_j}^{Y} \{(P[y_j], P[\vec{x}|y_j])\} \tag{1}$$

Given a stream $\mathcal{S}$, instances are generated accordingly to C_t. If for every moment t_i we have $C_{t_i} = C_{t_{i-1}}$, the concept is said stable, otherwise, we have a concept drift.

The real-time response, single-pass processing and dealing with concept drifts are important traits that must be accounted for by state-of-the-art algorithms for data streams [6] and have been somewhat tackled by existing works [13], [14], [12]. Yet, recent researches in [8], [9] demonstrated that the above definition does not broaden a specific kind of drift that occurs when features become, or cease to be, relevant to the learning task, namely feature drift.

III. Problem Statement

Most of existing algorithms for data streams tackle the infinite length and drifting concept characteristics. However, not many attention has been given to a specific kind of drift: feature drifts. Conversely to conventional concept drifts, where changes in data distribution are claimed to occur inside the skewing of classes in ranges of features' values, feature drifts occur whenever a subset of features becomes, or ceases to be, relevant to the concept to be learned.

Until this point, the term "relevance" was used without a proper definition. In this paper we divide features in two types: relevant and irrelevant [9]. Assuming $S_i = \mathcal{D} \setminus \{D_i\}$ and $P[Y|S_i']$ to be a conditional probability of Y given a subset of features S_i', a feature D_i is deemed **relevant** *iff* Eq. 2 holds [15].

$$\exists S_i' \subset S_i, \text{ such that } P[Y|D_i, S_i'] \neq P[Y|S_i'] \tag{2}$$

Otherwise, the feature D_i is said **irrelevant**. In practice, if a feature that is statistically relevant is removed from a feature set, it will reduce overall prediction power since (i) it is strongly correlated with the class; or (ii) it forms a subset with other features and this subset is strongly correlated with the class [16].

Changes in the relevant subset of features enforce the learning algorithm to adapt its model to ignore the irrelevant attributes and account for the newly relevant ones [10]. Given a feature space $\mathcal{D}$ at a timestamp t, we are able to select the ground-truth relevant subset $\mathcal{D}_t^* \subseteq \mathcal{D}$ such that $\forall D_i \in \mathcal{D}_t^*$ Eq. 2 holds and $\forall D_j \in \mathcal{D} \setminus \mathcal{D}_t^*$ the same definition does not. A feature drift occurs if, at any two time instants t_i and t_j, $\mathcal{D}_{t_i}^* \neq \mathcal{D}_{t_j}^*$ betides.

Let $r(D_i, t_j) \in \{0, 1\}$ denote a function which determines whether Eq. 2 holds for a feature D_i in a timestamp t_j of the stream. A positive relevance ($r(D_i, t_i) = 1$) states that $D_i \in \mathcal{D}^*$ in a timestamp t_i. A feature drift occurs whenever the relevance of an attribute D_i changes in a timespan between t_j and t_k, as stated in Eq. 3.

$$\exists t_j \exists t_k,\ t_j < t_k,\ r(D_i, t_j) \neq r(D_i, t_k) \tag{3}$$

Changes in $r(\cdot,\cdot)$ directly affect the ground-truth decision boundary to be learned by the learning algorithm. Therefore, feature drifts can be posed as a specific type of concept drift that may occur with or without changes in the data distribution $P[\vec{x}]$ [8], [9]. We emphasize that feature drifts are indeed targeted by the generic concept drift formalization, however, most of existing works on concept drift detection and adaptation assume that the relevant subset of features remains the same and that drifts occur if certain values, or ranges of values, have their class distribution re-skewed.

As pointed out in [8] and [9], feature drifts are likely to occur in a variety of scenarios, but mainly on text stream scenarions, e.g. social media, SMS chats, online social networks (Facebook, Twitter) and e-mail spam detection systems.

As in conventional concept drifts, changes in $r(\cdot,\cdot)$ may occur during the stream. This enforces learning algorithms to detect changes in $\mathcal{D}^*$, discerning between features that became irrelevant and the ones that are now relevant and vice-versa. In order to overcome feature drifts, a learner must either (i) discard and derive an entirely new classification model that is consistent with relevant features; or (ii) adapt its current model to relevance drifts [10].

IV. Synthesizing Feature Drifts

The majority of existing data generators allow the creation of data streams with concept drifts. In contrast to feature drifts, conventional concept drifts are often synthesized by promoting changes in $P[\vec{x}|Y]$ of the prior and posterior concepts and the subset of relevant features remains the same and the drifts occur due class distribution changes in certain values, or ranges of values, of such features.

In this section, we survey and introduce data generators capable of inducing feature drifts as streams progress and present the drift framework adopted during experiments.

A. Generators

In this section, we survey and propose data stream generators capable of inducing feature drifts during streams.

SEA-FD. Described in [8], SEA-FD extends the SEA generator [17] and synthesizes streams with $d > 2$ uniformly distributed features, where $\forall D_i \in \mathcal{D}, D_i \in [0; 10]$ and $\mathcal{D}^* = \{D_\omega, D_\zeta\}$ is randomly chosen. As in [17], the class value y is given accordingly to Eq. 4, where θ is a user-given threshold.

$$y = \begin{cases} 1, & \text{if } D_\alpha + D_\beta \leq \theta \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

BG-FD. The Binary Generator with Feature Drift (BG-FD) is a new generator composed by boolean ($\{0, 1\}$) features and that possess three functions: BG1-FD, BG2-FD, and BG3-FD, all inspired in [18]. In BG1-FD, a random subset $\mathcal{D}^* \subset \mathcal{D}$ is relevant to the concept to be learned, where $|\mathcal{D}^*|$ is a parameter. Conversely, in BG2-FD and BG3-FD we have $\mathcal{D}^* = \{D_\alpha, D_\beta, D_\epsilon\}$. Instances' labels are defined as stated in Eqs. 5 (BG1-FD), 6 (BG2-FD) and 7 (BG3-FD) and labels are evenly likely to occur.

$$y = \begin{cases} 1, & \text{if } \bigwedge_{D_i \in \mathcal{D}^*} D_i \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$y = \begin{cases} 1, & \text{if } (D_\alpha \wedge D_\beta) \vee (D_\alpha \wedge D_\epsilon) \vee (D_\beta \wedge D_\epsilon) \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

$$y = \begin{cases} 1, & \text{if } (D_\alpha \wedge D_\beta \wedge D_\epsilon) \vee (\neg D_\alpha \wedge \neg D_\beta \wedge \neg D_\epsilon) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

RTG-FD. The original Random Tree Generator (RTG) builds a decision tree by randomly performing splits on features and assigning a random class label to each leaf. Instances are created by generating a random valued $\vec{x}$ and traversing the tree for its corresponding label. We propose an extension to this generator, namely RTG-FD, such that a random $\mathcal{D}^* \subset \mathcal{D}$ is relevant, where $|\mathcal{D}^*| < |\mathcal{D}|$ is a user-given parameter.

Asset Negotiation (AN). This generator was originally presented in [19], where the aim was to simulate drifting bilateral multi-agent system negotiation of assets. Assets are described by the following features: color, price, payment, amount and delivery delay. The task is to predict whether an opposing agent would or not be interested in an asset (binary classification problem). Feature drifts are synthesized with changes in the interest of an agent by modifying the concept through time based on five functions, each of which relying on a different subset of features.

B. Drift Framework

Feature drifts are synthesized accordingly to the sigmoid framework proposed in [11], where a drift is the change between two pure distributions given by two different concepts C_A and C_B, each built upon different relevant feature subsets. To model the probability that every new instance i_t drawn from $\mathcal{S}$ belongs to the concepts C_A or C_B, the framework follows a sigmoid given by Eq. 8, where $P[C_B]$ and $P[C_A]$ are the probabilities of i_t belonging to C_A or C_B, W is the drift window size, t is the current timestamp and t_0 is the time of the drift, when $P[C_A] = P[C_B] = 0.5$.

$$P[C_B] = |1 - P[C_A]| = \frac{1}{e^{W(t_0 - t)}} \tag{8}$$

V. Benchmark

In this section we assess the performance of several existing data stream classifiers (Sec. V-A) in synthetic streams generated accordingly to generators and framework stated in the previous section with a proper protocol (Sec. V-C).

A. Classifiers

This section briefly presents evaluated algorithms: 1-Nearest Neighbor (1NN), Updatable Naive Bayes (NB), Very Fast Decision Rules (VFDR), Very Fast Decision Tree (VFDT) and Hoeffding Adaptive Tree (HAT). These classifiers were chosen due to their different bias and their consistent implementation provided in the Massive Online Analysis (MOA) framework [20].

1) 1-Nearest Neighbor (1NN): 1-Nearest Neighbor maintains a fixed-sized buffer queue of labeled instances, each of which is used for classifying future unlabeled ones [20]. Instances are labeled accordingly to the instance $\vec{x}_j$ in a buffer that minimizes the Euclidian distance given by Eq. 9.

$$d(\vec{x}_i, \vec{x}_j) = \sqrt{\sum_{D_k}^{\mathcal{D}} (\vec{x}_i[D_k] - \vec{x}_j[D_k])^2} \tag{9}$$

2) Naive Bayes (NB): Naive Bayes is a probabilistic learner that assumes input features to be independent of one another. As new instances arrive, a probability contingency table is incremented accordingly features' values and labeling of each instance is given by the maximization of the conditional probability as follows: $\text{argmax}_{y_i \in Y} P[y_i] \prod_{D_j}^{\mathcal{D}} P[\vec{x}_t[D_j] \mid y_i]$.

3) Very Fast Decision Rules (VFDR): Very Fast Decision Rules learns ordered and/or unordered rules in the ***IF*** *antecedent* ***THEN*** *consequent* form, where *antecedent* is a conjunction of tests over features and *consequent* is a class label. It starts with an empty rule set and rules are grown and expanded accordingly to the minimization of the entropy of class labels Y of instances covered by each rule [21] if they satisfy the Hoeffding bound [22].

4) Very Fast Decision Tree (VFDT): It constructs a decision tree by using constant memory and time per instance by recursively replacing leaves with decision nodes as data arrives [23]. Different heuristic evaluation functions, e.g. Information Gain and Gini Impurity, are used to select features and to determine whether a split should be performed by also verifying the Hoeffding bound. Finally, VFDT assumes that the stream underlying distribution is stationary, i.e. no drifts occur.

5) Hoeffding Adaptive Tree (HAT): Hoeffding Adaptive Tree (HAT) is an extension of VFDT to deal with concept drifts [24]. Based on ADWIN change detector (Sec. V-B1), HAT verifies if the statistics of split nodes over features still meet the Hoeffding criterion. If there is an alternate better splitting feature, the entire subtree is replaced by a new leaf and this new subtree is grown with upcoming instances.

B. Drift Detectors

In this section, we state two recent and broadly used drift detectors presented in the literature. In our experiments, whenever a drift is detected, the classifier is reset so it can start learning an entirely new concept. This approach is, by far, the most widely used in data stream learning approaches [25], [26], [11]. These following drift detectors were chosen since they provided better results when compared to seminal works, e.g. DDM [25], EDDM [26], in a variety of applications.

1) Adaptive Sliding Window (ADWIN): The Adaptive Sliding Window change detector (ADWIN) keeps a variable-length window of recently seen items consistent with the hypothesis "there has been no change in the average output value (classification error) inside the window accordingly to a confidence bound δ" [11]. ADWIN is parameter- and assumption-free since it automatically detects and adapts to the current rate of change.

2) Exponentially Weighted Moving Average (EWMA): Exponentially Weighted Moving Average (EWMA) was proposed in [27], where misclassification rates are exponentially weighted accordingly to their position inside a sliding window. EWMA maintains three threshold levels: in-control, warning, and out-of-control. Given the overall misclassification rates inside the sliding window, the current system state varies inside these three thresholds. EWMA postulates that a concept drift occurs whenever the misclassification rates achieve the out-of-control level.

C. Experimental Protocol

Accuracy is measured accordingly to the Prequential test-then-train method [28]. Although claimed as pessimistic, Prequential is capable of monitoring models' performance over time when estimated over a sliding window. In the following experiments, the sliding window size was set to $1/100$ of the stream size. This size was selected to provide clear accuracy results obtained during the stream, as the ones presented in Figs 1. We refrain from providing results for both processing time and memory usage for the sake of brevity and since our focus is mainly on feature drift adaptation, which is translated onto accuracy.

Experiments encompass the usage of generators presented in Sec. IV. Synthesized streams have a length of $100,000$ instances, $|\mathcal{D}| = 50$ and $|\mathcal{D}^*| = 3$, with the exception of SEA-FD experiments where $|\mathcal{D}^*| = 2$ and $\theta = 7$ [8] and AN, where $|\mathcal{D}| = 5$. Streams with an (A) suffix contain 9 equally distributed abrupt ($W = 1$) feature drifts, while streams with a (G) contain 9 drifts at the same positions, however, these are gradual ($W = 1,000$).

Experiments were performed under the Massive Online Analysis (MOA) framework [20] in an Intel Xeon CPU E5649 @ 2.53GHz×8 and 16GB of RAM computer and statistical differences are verified accordingly to Friedman's and Nemenyi's non-parametric tests with $\alpha = 0.05$ [29].

D. Discussion

In Tab. I we present the Prequential accuracy obtained in experiments, where one can see that HAT presents higher accuracy in most of the experiments. With the aid of hypothesis tests, we determined that {HAT, NB-ADWIN, VFDT-ADWIN} are statistically superior to others (Fig. 2), thus highlighting the capability of HAT and drift detectors to overcome feature drifts. The results obtained are expected since HAT performs evaluations of features used in test nodes of the tree accordingly to drifts flagged by the ADWIN detector. This must be emphasized since HAT is the only algorithm capable of performing dynamic feature selection as the stream progresses, thus, corroborating the claims provided in [9].

Focusing on the usage of drift detectors, we ran statistical tests multiples times to verify if their usage sufficiently improved accuracy of algorithms to show statistical difference and verified that {1NN, 1NN-ADWIN} $\succ$ {1NN-EWMA}, {NB-ADWIN, NB-EWMA} $\succ$ {NB}, {HAT} $\succ$ {HAT-ADWIN, HAT-EWMA}, while for both VFDT and VFDR no statistical difference was found. Therefore, one can see that drift detectors enable classifiers to obtain higher accuracy, yet, the gain is not commonplace for all learners. For example, NB when combined to ADWIN, showed an average boost in performance of 11.32%, yet, it is still outperformed by HAT alone. On the other hand, drift detectors did not show impressive results when applied to the 1NN classifier. 1NN is a lazy learner that labels instances given data stored in a buffer, and in streaming scenarios, this buffer is a sliding window, which "forgets" older data automatically, thus adapting its pseudo-model.

More importantly, we refer to the results obtained by HAT, which obtained higher accuracy without drift detectors, showing that partial model resets are more interesting than full model resets. In contrast to full model resets, partial resets are beneficial since classifiers still possess a partial model to classify upcoming instances, even after drifts.

Results obtained also show that feature selection is beneficial for streaming environments. In the case of HAT, feature selection allowed higher prediction rates since subtrees given by split nodes over irrelevant features were replaced accordingly to the increase of error rates.

In Fig. 1 we present results obtained during the RTG-FD(G) experiment since it provided, in average, worst results. We group results accordingly to the classifiers used and plot results without the usage of drift detectors and with the usage of ADWIN and EWMA. From Figs. 1a through 1d, one can see that the usage of conventional drift detectors does not allow quick recovery to drifts. These results show the difficulty of detecting and adapting to feature drifts. Finally, we highlight the results plotted in Fig. 1e, where HAT performs better in certain drifts, due to its internal periodical feature evaluation.

VI. Conclusion

In this paper, we introduced and reviewed several feature-drifting data stream generators. Additionally, we benchmarked data stream classification algorithms with and without drift detectors in several feature drifting streams. Results obtained show that feature drift is a challenging trait of streams that must be accounted for by learning algorithms. Although the adoption of drift detectors allowed classifiers to overcome feature drifts quicker, they are still beaten by a Hoeffding Adaptive Tree (HAT). HAT is able to provide better results since it combines an embedded feature selection procedure and the adoption of a drift detector to verify if features used in split nodes are still relevant [9].

TABLE I: Average Prequential Accuracy (%) obtained during experiments.

Experiment	No drift detector					ADWIN					EWMA				
	1NN	NB	VFDT	VFDR	HAT	1NN	NB	VFDT	VFDR	HAT	1NN	NB	VFDT	VFDR	HAT
AN(A)	⊙ 87.09	77.12	91.47	⊙ 72.05	● 93.63	85.12	⊙ 92.49	⊙ 92.30	54.68	92.93	83.77	90.55	90.67	68.39	91.11
AN(G)	⊙ 87.05	77.13	91.57	⊙ 72.81	● 92.62	85.01	⊙ 91.23	⊙ 91.92	52.41	91.96	83.93	89.90	90.09	66.00	90.30
BG1-FD(A)	86.00	69.99	78.21	⊙ 72.67	94.00	⊙ 86.19	● 94.52	⊙ 94.32	51.55	⊙ 94.28	84.10	92.94	92.69	71.30	92.46
BG1-FD(G)	⊙ 85.65	69.98	78.25	71.91	● 93.31	84.59	⊙ 92.96	⊙ 92.93	52.49	92.62	83.08	92.05	91.98	⊙ 76.56	91.89
BG2-FD(A)	73.31	62.57	67.70	70.75	● 91.21	⊙ 73.37	⊙ 82.49	⊙ 82.09	53.06	80.53	69.40	81.33	80.36	⊙ 73.90	82.08
BG2-FD(G)	⊙ 72.98	62.60	67.27	68.20	● 88.98	72.09	⊙ 81.23	⊙ 80.86	58.56	79.55	69.25	80.36	80.35	⊙ 73.50	81.43
BG3-FD(A)	⊙ 65.96	54.96	⊙ 62.08	53.60	● 86.20	65.76	⊙ 59.98	60.41	50.81	60.00	61.87	57.49	60.45	⊙ 61.02	64.52
BG3-FD(G)	⊙ 65.86	54.73	60.14	54.26	● 82.03	65.42	⊙ 60.70	60.33	51.42	59.70	62.69	58.26	⊙ 60.39	⊙ 55.61	59.36
RTG-FD(A)	⊙ 57.21	55.41	55.65	56.08	● 65.24	57.09	⊙ 60.55	⊙ 60.28	⊙ 56.91	59.42	56.41	57.81	57.82	56.07	57.98
RTG-FD(G)	⊙ 57.11	55.41	55.68	55.86	● 63.25	57.03	⊙ 60.55	⊙ 60.00	⊙ 56.52	59.71	56.36	57.65	57.42	55.96	57.39
SEA-FD(A)	⊙ 75.28	79.83	80.82	⊙ 79.14	● 83.80	75.20	⊙ 82.68	⊙ 82.82	75.15	82.64	73.49	79.90	79.66	73.32	78.53
SEA-FD(G)	75.29	79.85	80.80	⊙ 78.22	● 83.59	⊙ 75.31	⊙ 82.90	⊙ 82.61	74.99	81.74	73.61	79.14	79.14	73.39	78.23

● stands for best result obtained by all classifiers in given experiment. ⊙ stands for best result obtained by each classifier in the given experiment.

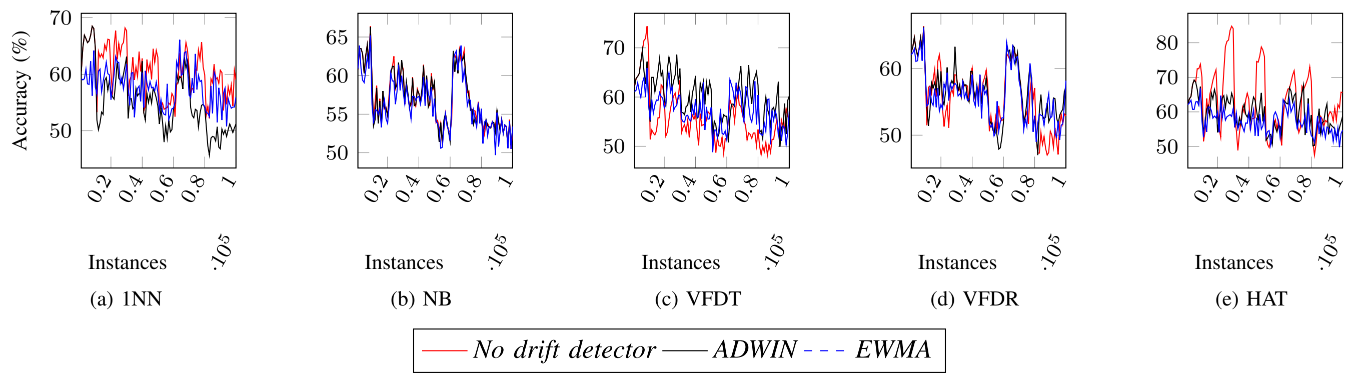

Fig. 1: Prequential accuracies (%) during RTG-FD(G) experiment.

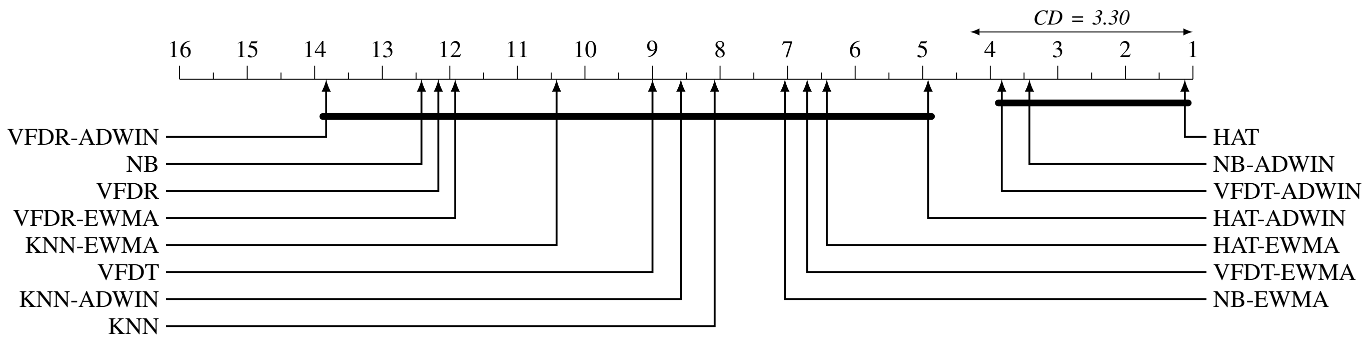

Fig. 2: Critical differences chart for comparison among all classifiers.

Accordingly to the results provided in this paper, we claim that there are the need and room for new methods for dynamic feature selection [30], [31], [32]. Future works envision the creation of adaptive feature selection filters for data streams. Filters would allow different biased classifiers to detect and adapt to feature drifts, thus boosting their accuracy during feature drifting streams, while diminishing processing time and memory space, due to lower dimensionality.

Acknowledgment

This project was financially supported by the *Coordenação de Aperfeiçoamento de Pessoal de Nível Superior* (CAPES) through the *Programa de Suporte à Pós-Graduação de Instituições de Ensino Particulares* (PROSUP) program.

References

[1] J. A. Silva, E. R. Faria, R. C. Barros, E. R. Hruschka, A. C. P. L. F. d. Carvalho, and J. a. Gama, “Data stream clustering: A survey,” *ACM Comput. Surv.*, vol. 46, no. 1, pp. 13:1–13:31, Jul. 2013.

[2] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu, “A framework for clustering evolving data streams,” in *Proceedings of the 29th International Conference on Very Large Data Bases - Volume 29*, ser. VLDB ’03. VLDB Endowment, 2003, pp. 81–92. [Online]. Available: http://dl.acm.org/citation.cfm?id=1315451.1315460

[3] J. P. Barddal, H. M. Gomes, and F. Enembreck, “Advances on concept drift detection detection in regression tasks using social networks theory,” *International Journal on Natural Computing Research*, pp. 1–14, 2015.

[4] A. Bifet, B. Pfahringer, J. Read, and G. Holmes, “Efficient data stream classification via probabilistic adaptive windows,” in *SAC*, 2013, pp. 801–806.

[5] J. P. Barddal, H. M. Gomes, and F. Enembreck, “SFNClassifier: A scale-free social network method to handle concept drift,” in *Proceedings of the 29th Annual ACM Symposium on Applied Computing (SAC)*, ser. SAC 2014. ACM, March 2014.

[6] J. a. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 44:1–44:37, Mar. 2014. [Online]. Available: http://doi.acm.org/10.1145/2523813

[7] J. P. Barddal, H. M. Gomes, and F. Enembreck, "SNCStream: A social network-based data stream clustering algorithm," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing (SAC)*, ser. SAC 2015. ACM, April 2015.

[8] ——, "Analyzing the impact of feature drifts in streaming learning," in *Proceedings of the 22th International Conference on Neural Information Processing*, ser. ICONIP 2015. Springer, November 2015.

[9] ——, "A survey on feature drift adaptation," in *Proceedings of the International Conference on Tools with Artificial Intelligence*. IEEE, November 2015.

[10] H.-L. Nguyen, Y.-K. Woon, W.-K. Ng, and L. Wan, "Heterogeneous ensemble for feature drifts in data streams," in *Advances in Knowledge Discovery and Data Mining*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7302, pp. 1–12. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30220-6_1

[11] A. Bifet, *Adaptive Stream Mining: Pattern Learning and Mining from Evolving Data Streams*, ser. Frontiers in Artificial Intelligence and Applications. IOS Press, 2010. [Online]. Available: http: //www.booksonline.iospress.nl/Content/View.aspx?piid=14470

[12] J. Gama, *Knowledge Discovery from Data Streams*, 1st ed. Chapman & Hall/CRC, 2010.

[13] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in *In SIAM International Conference on Data Mining*, 2007.

[14] A. Bifet, G. Holmes, and B. Pfahringer, "Leveraging bagging for evolving data streams," in *Machine Learning and Knowledge Discovery in Databases*, ser. Lecture Notes in Computer Science, J. L. BalcÃ¡zar, F. Bonchi, A. Gionis, and M. Sebag, Eds. Springer Berlin Heidelberg, 2010, vol. 6321, pp. 135–150.

[15] J. P. Barddal, H. M. Gomes, F. Enembreck, and B. Pfahringer, "(to appear) a survey on feature drift adaptation: Definition, benchmark, challenges and future directions," *Journal of Systems and Software*, vol. ?, no. ?, pp. ?–?, 2016.

[16] Z. Zhao, F. Morstatter, S. Sharma, S. Alelyani, A. Anand, and H. Liu, "Advancing feature selection research," *ASU feature selection repository*, pp. 1–28, 2010.

[17] W. N. Street and Y. Kim, "A streaming ensemble algorithm (sea) for large-classification," in *Proc. of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM SIGKDD, Aug. 2001, pp. 377–382.

[18] M. A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 359–366. [Online]. Available: http://dl.acm.org/citation.cfm?id=645529.657793

[19] F. Enembreck, B. C. Ávila, E. E. Scalabrin, and J.-P. A. Barthès, "Learning drifting negotiations." *Applied Artificial Intelligence*, vol. 21, no. 9, pp. 861–881, 2007. [Online]. Available: http://dblp.uni-trier.de/db/journals/aai/aai21.html#EnembreckASB07

[20] A. Bifet, G. Holmes, R. Kirkby, and B. Pfahringer, "MOA: Massive online analysis," *The Journal of Machine Learning Research*, vol. 11, pp. 1601–1604, 2010.

[21] J. Gama and P. Kosina, "Learning decision rules from data streams," in *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, 2011, pp. 1255–1260. [Online]. Available: http://ijcai.org/papers11/Papers/IJCAI11-213.pdf

[22] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, March 1963. [Online]. Available: http://www.jstor.org/stable/2282952?

[23] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '00. New York, NY, USA: ACM, 2000, pp. 71–80. [Online]. Available: http://doi.acm.org/10.1145/347090.347107

[24] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '01. New York, NY, USA: ACM, 2001, pp. 97–106. [Online]. Available: http://doi.acm.org/10.1145/502512.502529

[25] J. a. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Advances in Artificial Intelligence - SBIA 2004*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2004, vol. 3171, pp. 286–295.

[26] M. Baena-García, J. del Campo-Ávila, R. Fidalgo, A. Bifet, R. Gavaldá, and R. Morales-Bueno, "Early drift detection method," in *In Fourth International Workshop on Knowledge Discovery from Data Streams*, 2006.

[27] G. J. Ross, N. M. Adams, D. K. Tasoulis, and D. J. Hand, "Exponentially weighted moving average charts for detecting concept drift," *ArXiv e-prints*.

[28] J. Gama and P. Rodrigues, "Issues in evaluation of stream learning algorithms," in *Proc. of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM SIGKDD, Jun. 2009, pp. 329–338.

[29] G. W. Corder and D. I. Foreman, *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*. New Jersey: Wiley, 2009.

[30] H. Tsukioka and M. Kudo, "Selection of features in accord with population drift," in *Pattern Recognition (ICPR), 2014 22nd International Conference on*, Aug 2014, pp. 1591–1596.

[31] K. Wankhade, D. Rane, and R. Thool, "A new feature selection algorithm for stream data classification," in *Advances in Computing, Communications and Informatics (ICACCI), 2013 International Conference on*, Aug 2013, pp. 1843–1848.

[32] ——, "An overview of methods for feature selection based on mutual information for stream data classification," in *Communication Systems and Network Technologies (CSNT), 2013 International Conference on*, April 2013, pp. 630–634.