On the Application of GP to Streaming Data Classification Tasks with Label Budgets

Ali Vahdat[1], Aaron Atwater[1], Andrew R. McIntyre[1], and Malcolm I. Heywood[1]

[1]*Faculty of Computer Science, Dalhousie University, Halifax, NS. Canada*

Abstract

A framework is introduced for applying GP to streaming data classification tasks under label budgets. This is a fundamental requirement if GP is going to adapt to the challenge of streaming data environments. The framework proposes three elements: a sampling policy, a data subset and a data archiving policy. The sampling policy establishes on what basis data is sampled from the stream, and therefore when label information is requested. The data subset is used to define what GP individuals evolve against. The composition of such a subset is a mixture of data forwarded under the sampling policy and historical data identified through the data archiving policy. The combination of sampling policy and the data subset achieve a decoupling between the rate at which the stream passes and the rate at which evolution commences. Benchmarking is performed on two artificial data sets with specific forms of sudden shift and gradual drift as well as a well known real-world data set.

1 Introduction

Streaming data introduces additional challenges to the classification task that change the nature of the task in fundamental ways. A short list of potential factors of interest to this research includes, but is not limited to:

Non-stationary process: The process generating the data itself might well be subject to concept drift (gradual change) or shift (sudden change). This is straightforward to address if all the data is labeled. However, this would imply that it is possible to provide such labels at the rate that the data is received. In general this is not possible, indeed, there might be a considerable cost associated with labeling.

Any time operation: Data is received on a continuous basis; whereas a machine learning (ML) algorithm should be capable of providing 'predictions' at any point in time. Dividing the data into independent training and test partitions implies knowledge about when the stream is sufficiently stationary in order to identify consistent training and test partitions, which is not possible in general. Instead, ML for streaming needs to make decisions regarding what data to use for training while always having a champion individual available for classification.

Requesting labels: Typically, some form of a windowed interface[1] is assumed in order to provide bounds on what new data can potentially be used for training purposes. Additional decisions are necessary to decide how much of this data to request labels for. Naturally, if the ML can establish that no change has appeared between the content of consecutive windows, then there is no need to request labels. In short, a balance needs achieving between too little training (in which case the ML model goes stale) versus too much (which implies that too many labels are requested).

Class imbalance: Given that only a small fraction of the data stream is available at any point in time (cf., the windowed interface to the data stream), then it is not possible to a priori stratify content. Thus, it is quite likely that the content of any data window will not be representative all possible classes. Indeed, when classes are imbalanced, it is likely that data window content will be degenerate (consist of a single class alone). Current approaches to class imbalance in GP assume an offline (batch) approach e.g., [5], thus do not directly transfer to the streaming data scenario.

In this work we are interested in pursuing an evolutionary approach capable of addressing all three of the above properties simultaneously. In particular, we focus on the issue of change detection with label budgets in order to explicitly make label requests.

[1]One of two generic window interfaces are typically assumed: 1) a sliding window – data shuffled in and out under a first-in, first-out data structure of finite length, or; 2) non-overlapping windows of finite length.

Change detection represents a label free process by which changes to the underlying process generating stream content are detected. There are at least two general aspects to the **change detection** question (detailed further in Section 2). In the first case a statistical characterization might be assumed relative to: 1) the input stream; 2) some property of the classifier architecture; or 3) the decision boundary of the classifier. All of these scenarios are potential solutions for detecting a general drift or shift to prior classification boundaries after which **label requests** would be made. However, when a previously learnt class is associated with an as yet unseen class and / or previously learnt classes are switched (as might happen when user preferences shift), there is no reference for model based change detection. This means that label requests should be driven through both knowledge of appropriate statistical characterization(s) as well as purely stochastically in order to capture switches to previously learnt classifier behaviour. Moreover, there is a limit to the number of label requests, irrespective of the source of the request i.e., a **finite label budget**.

In this work we establish a basic framework for addressing these questions under genetic programming (GP), Section 3. Such a framework consists of: 1) the sliding window interface to the stream which defines the subset of data 'available' at any point in time; 2) a sampling policy for determining which exemplars from the current sliding window are actually selected for training purposes; 3) a data subset against which a generation of the evolutionary algorithm is performed, and; 4) a data archiving policy for determining how much (if any) of the data subset is retained between generations. Naturally, it is the sampling policy that defines when exemplars labels are requested, whereas the role of the archiving policy is to retain exemplars that support the identification of 'good' GP individuals i.e., a Pareto archiving policy. A classifier is always available for providing a class label, care of the data subset that provides the basis for always being able to select a champion classifier without reference to additional label information.

Section 5 performs a benchmarking study using two artificial data sets characterized by non-stationary (gradual) drift and (sudden) shift. A third real-world data set known to include concept drift is also included. The well known Massive Online Analysis (MOA) toolbox provides a baseline classifier with labelling budget. To our knowledge this is the first time that GP has been benchmarked under the conditions of streaming data with a labelling budget. A discussion of the studies findings and future work presented in Section 6.

2 Background and Related Work

There has been a significant body of work dedicated to streaming data scenarios under non-evolutionary ML frameworks e.g., [33, 19, 6, 20]. For brevity we will focus on the classification task alone (as opposed to clustering, item set mining or forecasting / prediction) and concentrate on the issue of change detection which lies at the centre of building ML frameworks capable of operating under label budgets. As noted in the introduction, there are three broad categories of interest, outlined as follows.

Characterizing properties specific to the model of classification: represents an approach in which properties specific to a class of ML are measured and compared to a prior reference characterization. This is particularly appropriate for decision tree architectures in which the frequency of leaf node utilization is often the property characterized [17, 24]. Naturally, the limitation in this method is that it is specific to a particular ML architecture as opposed to any ML representation.

Characterizing properties of the input data: implies that the focus of change detection is now on characterizing behaviour relative to sliding window content. Generally two sliding windows are employed. The content of a reference window is compared to that of the most recent sliding window. The principle design decision revolves around what statistic to employ e.g., Chernoff bounds [26], entropy [11, 38], Kullback–Leibler divergence [35], Hoeffding bounds [7], Fractal correlation dimension [18] or Hellinger divergence metric [15]. One significant drawback of pursuing such an approach is that it is often necessary to label the data (i.e., metrics are estimated class-wise). In addition, statistics estimated relative to the input stream are also subject to as loss of effectiveness as the dimensionality increases [40].

Characterizing properties of the label space: implies that the 'behaviour' of the classifier output is the focus of measurements. Various authors have proposed statistical characterizations of class boundary information cf., classifier confidence [30, 36]. For example, changes to classifier certainly might be detected using thresholds [31], or changes to the number of confident predictions detected [27]. Naturally, such characterizations of classifier behaviour are independent of the ML framework and potentially applicable to GP as well [34].

A remaining drawback of any of the above approaches is that they are unable to detect when a previously encountered input behaviour, $P(x)$, is associated with a new or different class label. For example, under this scenario a label space characterization would still associate $P(x)$ with the previous label, or $P(y|x)$. Likewise an input data characterization would not register any change either, as there has been no change to $P(x)$. Under

these scenarios generating label requests uniformly (i.e., independently of the measurable properties) has been shown to be surprisingly effective [41]; as have hybrid approaches combining label space and uniform sampling [37]. Section 3 will discuss how such findings can be factored into an approach for applying GP to streaming data under limited label budgets.

From the particular perspective of GP there has not been a concerted effort to address streaming data requirements. The largest body of EC research has been directed towards EC in 'dynamic environments', and applications to finance in particular [14]. Financial applications generally take the form of evolving a trading agent to make buy, hold or sell decisions given some summary of the current and historical trading conditions. State-of-the-art in this case evolve both the temporal features used to characterize local historical information as well as some form of decision tree used to determine what action to apply. Unlike the streaming task of interest to this work there is no concept of a label budget. Instead trading agents are either evolved on a continuous basis [14, 25], periodically retrained [39], or make use of loss criteria to dynamically re-trigger training events [32].

Other developments from evolutionary computation (EC) include the application of learning classifier systems (LCS) to dynamic environments. Dam *et al.*, discuss the adaption of learning rates versus reinitializing the population for noisy versus non-stationary environments under the assumption that all the data is labelled [10]. Behdad and French propose an online version of LCS in which the exploit and explore cycles are reversed [4]. Moreover, various probabilistic heuristics are proposed for filling in known 'gaps' in label information. Finally, we note a recent work in which a k-NN formulation was assumed such that PSO could be used to adapt the direction in which k-NN prototype classifiers migrated under streaming data classification [9]. Again, the data must be completely labeled a priori.

3 Generic GP streaming data architecture

The general architecture proposed for applying GP to streaming data under finite labelling budgets is summarized by Figure 1. The initial sliding window (SW) represents the set of exemplars that can be indexed (for training) at time, t, or $SW(t)$; whereas the sampling policy determines which exemplars are actually utilized for training purposes, or $Gap(i) \in SW(t)$; where $|Gap(i)| \leq |SW(t)|$. Only the $|Gap(i)|$ exemplars chosen are added to the Data Subset ($DS(i)$) and have their corresponding labels requested. Note also that the rate at which the sliding window updates, or t, need not be the same as the rate at which samples are taken, or i. Instead, the rate of sampling *from* the sliding window, i, defines the rate at which the Data Subset ($DS(i)$) and therefore the GP population is updated. With each new generation i, $|Gap(i)|$ exemplars are identified from the current $SW(t)$ position, and a corresponding number of exemplars from $DS(i)$ are replaced by the content of $Gap(i)$. Naturally, the degree of differentiation between stream and DS updates sets the label budget ratio.

The scheme assumed for prioritizing DS content for replacement is defined by a **Data Archiving Policy**, in this case Pareto archiving will be assumed e.g., [13]. Specifically, Pareto archiving is used to identify exemplars that distinguish between the performance of GP classifiers. Such a set of exemplars are said to be non-dominated [13]. One of the drawbacks of assuming a Pareto archiving policy, however, is that the archive of exemplars distinguishing between different GP classifiers can potentially increase to $P^2 - P$; where P is the size of the GP population. This would have implications for the overall efficiency of the algorithm. Hence, we limit the size of DS to a suitable finite value and employ a DS exemplar diversity / aging heuristic [1, 2]. The process for choosing exemplars from DS for replacement now takes one of the following forms, depending on which condition is satisfied:

- The number of exemplars forming distinctions is less than or equal to $|DS| - |Gap|$. Let the exemplars from DS that do not support distinctions be $\bar{d}$. As $|\bar{d}| \geq |Gap|$ then the DS exemplars replaced by $Gap(i)$ are selected from $\bar{d}$ alone.
- The number of exemplars forming distinctions is more than $|DS| - |Gap|$. Any exemplars not forming distinctions ($\bar{d}$) will be replaced. In addition $|Gap| - |\bar{d}|$ exemplars forming distinctions will be replaced (potentially resulting in the loss of GP classifiers i.e., no longer identified as being non-dominated). The exemplars forming distinctions are now ranked in accordance with how many other points form the same distinction and how long an exemplar has been in the archive [1, 2]. In effect exemplars supporting: 1) unique distinctions see more priority than those forming more common distinctions i.e., a form of fitness sharing or diversity maintenance, and 2) older exemplars are more likely to removed in favour of those forming more recent distinctions.

Such preference schemes were previously shown to be useful under GP streaming classification, albeit without label budgeting [3, 2, 1].

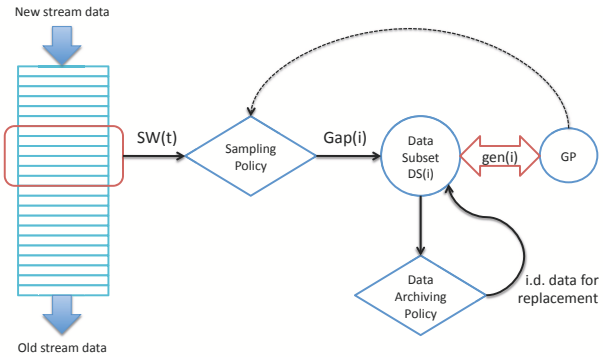

Figure 1: Components of generic architecture for applying GP to streaming data under a label budget

In the case of the **Sampling Policy**, Section 2 highlighted that two basic sources have been identified from the wider literature: stochastic sampling versus using classification confidence information. Classifier confidence information implies that as the certainty of the class label suggested by a classifier decreases (approaches ambiguity), then request a label. In the case of the stochastic source of label requests, this is performed uniformly relative to exemplars that are classified with certainty. The objective being to confirm that cases which are classified with certainty have not undergone some form of shift into a different class. Moreover, we also note that even requesting labels with a uniform probability (under a label budget) is often better than more sophisticated heuristics [41]. In this work we will assume the uniform sampling heuristic under a label budget for the purpose of establishing our baseline performance. Future work will introduce mechanisms to include classifier confidence, where given that multiple GP individuals are evolved, it remains to be seen how much significance is given to a single GP classifier confidence versus that of, say, the archive of non-dominated classifiers.

Finally, in order to provide **anytime classifier operation**, it is necessary to have a champion individual responsible for predicting labels for stream data at any point in time. Thus, relative to the current content of $DS(i)$ the non-dominated GP individual maximizing class-wise detection rate (DR) is assumed to be representative of the wider population. DR is estimated as follows: $\frac{1}{C}\sum_{j=[1,...,C]} DR(j)$; where C is the number of classes present in $DS(i)$. $DR(j) = tp/(tp + fn)$; where tp and fn denote true positive and false negative counts w.r.t. class j. Note that in limiting the available candidate GP classifiers to the non-dominated set, we reduce the likelihood of selecting degenerate classifiers. This is particularly important because stream data is not stratified, whereas it is highly likely that stream data is imbalanced. Hence, it is generally not possible to make any guarantees regarding the distribution of classes appearing in any sliding window location.

The specific form of GP assumed takes the form of Symbiotic Bid-Based GP (SBB) and therefore benefits from the ability to perform task decomposition (construct a classifier as a team of programs). Studies have demonstrated that SBB is more effective than monolithic GP under (stationary) classification tasks [29], as well as under data sets with large attribute spaces [16] and data sets with high degrees of class imbalance [28]. Aside from the additional transparency of the resulting solutions, pursuing a GP teaming approach also provides an elegant solution to multi-class classification. Readers are referred to the earlier SBB papers for details of selection / variation operators and instruction set [28, 16].

4 Experimental Methodology

This section begins by establishing the approach to benchmark dataset selection (Section 4.1). Section 4.2 discusses parameterization issues for the GP approach. Section 4.3 outlines the approach adopted to performance evaluation. Finally, the alternate streaming classifier (an adaptive form of Naive Bayes with label budgeting) is summarized (Section 4.4).

4.1 Data sets

Three data sets will be employed for the purposes of benchmarking: 1) two artificially created and therefore with known degrees of non-stationary behaviour,[2] and; 2) a well known real world data set, Electricity [23], that

[2]Publicly available at `http://web.cs.dal.ca/~mheywood/Code/SBB/Stream/StreamData.html`

Table 1: Benchmarking dataset properties

Discrete non-stationary concept shift [41]	
# of Classes per concept	5
# Attributes (# irrelevant attributes)	7 (1)
# Exemplars per 'data block'	500,000
# Exemplars over entire stream (S_{max})	6,500,000
Continuous non-stationary concept drift [17]	
# of Classes per concept	3
# Attributes (# irrelevant attributes)	11 (1)
# Exemplars per 'data block'	1,000
# Exemplars over entire stream (S_{max})	150,000
Electricity data set [23]	
# of Classes	2
# Attributes	8
# Exemplars over entire stream (S_{max})	45,312

has frequently been employed for streaming data benchmarking tasks. The basic properties of the data sets are summarized by Table 1.

Discrete concept shift dataset: adopts the approach taken in [41]. The Dataset Generator tool[3] is used to construct decision trees that specify a partitioning of the attribute space into a 5-class classification task based on randomly generated thresholds. Data is generated with a uniform p.d.f. and then assigned a class using the decision tree. A total of two concept generator decision trees (C1, C2) are used to create two sources of data. A single stream of data is then constructed block-wise with data integrated from each of the two concept generator decision trees.

Let each block consist of 500,000 exemplars, as sampled from the concept generator tuple of the form: $\langle C1\%, C2\% \rangle$. A total of 13 blocks describe the transition from purely $C1$, to purely $C2$ as follows: $\langle 100, 0 \rangle$, $\langle 100, 0 \rangle$, $\langle 100, 0 \rangle$, $\langle 90, 10 \rangle$, $\langle 80, 20 \rangle$, ... $\langle 0, 100 \rangle$. As the first 10% of the stream is made available for initial model construction, generating the first three blocks of data using concept $C1$ implies that the initial models never see anything other than concept $C1$. Moreover, blocks composed from a mixture of the two concepts assume a uniform p.d.f.

Continuous concept drift dataset: adopts the approach taken in [17]. Hyperplanes are defined in a $d = 10$ dimensional space. Initial values of the hyperplane parameters are selected with uniform probability. Every $1,000$ exemplars, half of the parameters are considered for modification with a 20% chance of change. Class labels are allocated as a function of hyperplanes exceeding a class threshold. See [1] for a detailed description of the construction of this data set.

Electricity: characterizes the rise and fall of electricity demand in New South Wales (Australia) using consumption and price information for the target and neighbouring regions [23]. As such it is a two class data set (demand will either increase or decrease relative to the previous period), moreover, the distribution of classes is very imbalanced.

4.2 Parameterization of GP

The following approach will be adopted regarding the experimental methodology assumed for benchmarking GP algorithms under streaming data:

Model initialization is performed using the first 10% of the stream. This implies that relative to Figure 1, the first 10% of the generations are performed with the sliding window stationary over the initial 10% of the data set. Thereafter, the sliding window interface assumes a first-in, first-out data structure of fixed depth.

A **sliding window** of length S_{max}/i_{max} exemplars is assumed after model initialization. The remainder of the stream passes through at a constant rate (S_{max} is the ultimate length of the stream and i_{max} is the total generation count). The window is used to define the pool from which the new $|Gap(i)|$ training exemplars are sampled and their label are requested. Note that in effect this parameterization results in a *non-overlapping* sliding window. However, GP is evaluated w.r.t. the content of the Data Subset (Figure 1). Hence, there is still a 'gentle' turnover

[3]Gabor Melli. The 'datgen' Dataset Generator. `http://www.datsetgenerator.com/`

in new to old exemplar content between consecutive generations, or $\frac{|Gap|}{|DS|}$ new exemplars are introduced at each generation, where $|Gap| = 20$ and $|DS| = 120$.

Label budget is the ratio of points whose labels are requested over the stream length, or $\frac{i_{max} \times |Gap|}{S_{max}}$. In other words only $i_{max} \times |Gap|$ exemplars are requested for their label in a stream of length S_{max}. Thus, under the artificial concept shift data set (Table 1) with $S_{max} = 6,500,000$ and $i_{max} = 1,000$, then 6,500 exemplars pass between updates of the GP population, all of which are retained in the sliding window for potential sampling by the Sampling Policy. In the parameterization assumed here only $|Gap| = 20$ exemplars are sampled by the Sampling Policy to be used for training at each generation Thus, the Label Budget in this example would be $\frac{1,000 \times 20}{6,500,000} \approx 0.3\%$. Given the rather different stream lengths of the three benchmarking data sets (Table 1), two different parameterizations for i_{max} will be assumed per data set as follows:

- Concept shift data set: $i_{max} \in \{1,000; 10,000\}$;
- Concept drift and Electricity data sets: $i_{max} \in \{500; 1,000\}$

Different tapped delay line depths will be investigated in conjunction with GP. The number of taps will either be 1 (special case of no history) or always consist of 8 taps. Separate experiments will be performed with skip periods of 1, 2, 4 or 8. For example, the skip period of 2 corresponds to data instances of $[t, t-2, t-4, ..., t-14]$ where t is the index of the exemplar for which a class label is desired. Increasing the skip period increases the depth of data indexed, but reduces the resolution given a fixed number of taps. Support for a tapped delay line implies that GP indexes a matrix of #attributes $\times$ #taps and is therefore independent of the streaming framework details (Section 3).

Any remaining parameters follow from earlier work with SBB on streaming data [1, 2].

4.3 Evaluation

For comparison purposes we make use of two forms of comparator classifier: a "no change" classifier [8] and an Adaptive Naive Bayes model with label budgeting [37]; the latter is described in Section 4.4. The no change classifier was proposed as a naive 'devils advocate' approach to classification in which access to perfect label information is assumed. Specifically, it assumes that the label for exemplar t is the same as that for exemplar $t-1$, and continues with this 'prediction' until the label changes; after which the latest change in label is assumed. In short, the no change classifier is a one-bit state machine that makes no use of attribute information at all. It is known to be particularly effective (given complete access to label information) when the data stream consists of bursts of exemplars carrying the same class label [8]. However, as the number of classes and / or the degree of interleaving of class information increases, performance decreases. As such a no change classifier provides a 'feel' for how much implicit class variation exists in a stream.

Performance evaluation takes the form of an accumulative moving average metric, or prequential accuracy [12], where this is the default metric assumed by the MOA toolbox as well as being more generally used in the ML streaming literature [22]. Specifically, the prequential accuracy at exemplar t in the stream is estimated relative to all past $t-1$ exemplars as well as exemplar t, or

$$preq_t = \frac{(t-1) \times preq_{t-1} + R(t)}{t} \tag{1}$$

where $R(t) = 1$ denotes a true classification of instance t, and $R(t) = 0$ denotes otherwise. The ratio of time indexes acts as a weighting factor, enforcing a decay for older updates [22]. The resulting prequential accuracy takes the form of a curve, although current benchmarking practice also tends to compare against the final prequential accuracy estimate for $t = S_{max}$.

4.4 Adaptive Naive Bayes model

The second comparator classifier is documented in a recent study of streaming data classification under label budgets and drift detection [37], and has been made available in the Massive Online Analysis (MOA) toolbox.[4] Specifically, the Naive Bayes classifier with budgeted active learning and drift detection under the prequential evaluation task is employed. The drift detection mechanism selected was the DDM (Drift Detection Method) algorithm given by Gama *et al.*, [21] with default values for threshold (1) and step parameters (0.01). The 'random'

[4]MOA prerelease 2014.03; `http://moa.cms.waikato.ac.nz/overview/`

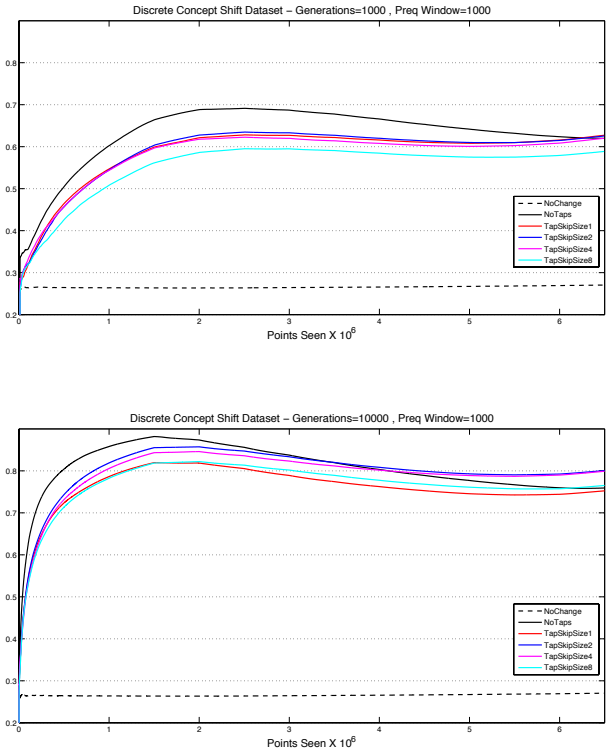

Figure 2: GP Prequential accuracy during stream. Artificial dataset with sudden concept shift. First 10% of stream (0.65×10^6 exemplars) are used to construct the initial model. Top: Label budget of $\approx$ 0.3% or $i_{max} = 1,000$; Bottom: Label budget of $\approx$ 3% or $i_{max} = 10,000$

active learning strategy was selected as it provided the baseline in [37] and is closest to the stochastic sampling policy adopted in this work. Finally the label budget parameter was varied over the range $[0.05, 0.5]$ in 0.05 level increments.

Under this configuration a new classifier is built when the current classifier's performance begins to degrade; the current classifier is then replaced when drift is explicitly detected by the DDM. The active learning with budgeting is managed by the random selection policy where incoming points are queried for labels with the same uniform random probability as the budget parameter itself.

5 Results

Figures 2, 3 and 4 provide a behavioural summary for how prequential accuracy (eqn. (1)) develops under GP during the stream w.r.t. different labelling budgets and support for various tapped delay line configurations (Section 4.2) over the three data sets (Section 4.1). In all cases each curve is the result of averaging the performance of GP over 50 runs for each configuration. Clearly a larger label budget always improves prequential accuracy, however, under the concept shift data set (Figure 2) more labels are necessary in order to provide a benefit when using a tapped delay line. This is natural as the changes are sudden, thus after the change, the data is stationary until the next sudden change. Moreover, there was a sensitivity to the taps used in the delay line under the concept shift data set, with only tap skips (2 and 4) resulting in performance improvements. Conversely, including taps was always beneficial to the concept drift and electricity data sets, with a more prominent effect under electricity (Figure 4). We also note that on the concept shift data set some regression in prequential accuracy occurs after the initial (10%) pre-training period (Figure 2). Conversely, progress on the concept drift and electricity data sets is generally monotonically increasing (Figures 3 and 4).

These figures also illustrate the performance of the "no change" classifier (Section 4.3). Both the artificial data sets illustrate the inherent weaknesses of such a classifier, with somewhat less than one class correctly classified on the concept shift data set (Figure 2), and a continuous decay experienced under concept drift (Figure 3). In the latter case, the no change classifier experiences an initial labelling success, but thereafter decays as no particular pattern of behaviour is evident in the labels.

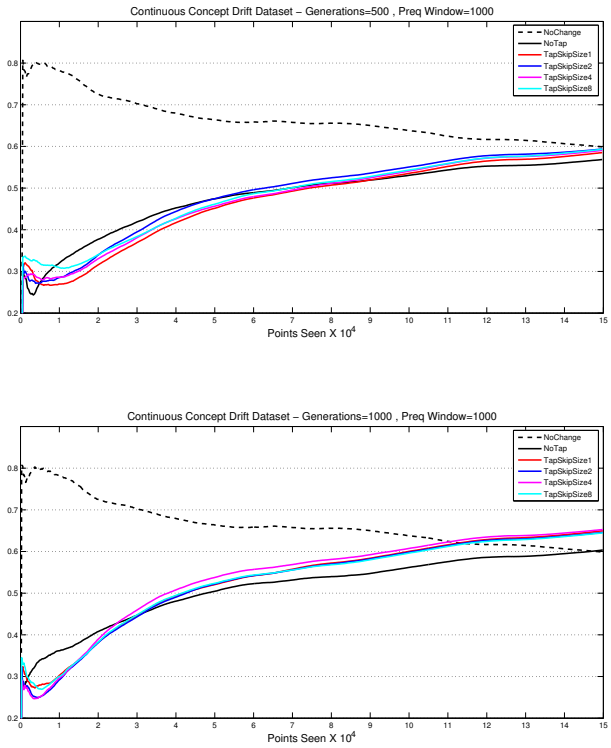

Figure 3: GP Prequential accuracy during stream. Artificial dataset with gradual concept drift. First 10% of stream (1.5×10^4 exemplars) are used to construct the initial model. Top: Label budget of $\approx 6.7\%$ or $i_{max} = 500$; Bottom: Label budget of $\approx 13.3\%$ or $i_{max} = 1,000$

Table 2 summarizes the resulting prequential accuracy in terms of mean and standard deviation for each tapped delay line configuration, label budget and data set. Note that label budgets are expressed in terms of the total stream length (Section 4.2). At each generation no more than 20 exemplars are sampled from the stream (Section 4.2). As noted above, there is a general preference for including the tapped delay line, with best cases most frequently occurring for a skip size of 4 (and no preference for a skip size of 1).

Results for the second baseline classifier, the adaptive Naive Bayesian (ANB) framework for streaming data classification under label budgets as implemented in the MOA toolkit (Section 4.4) are summarized in Figure 5. In this case label budgets are specified as a percentage of the entire stream in increments of 5 between 5 and 50 percent. These results should therefore be compared with the the GP results summarized in Table 2. Under the concept shift data set, a 3% label budget is sufficient for GP to perform better than any configuration of ANB, whereas ANB performs better than GP under the concept drift data set (both outcomes are significant at the 99% confidence interval). In the case of the Electricity data set, ANB performs better at low label budgets, but as the label budget increases there is no statistically significant difference.

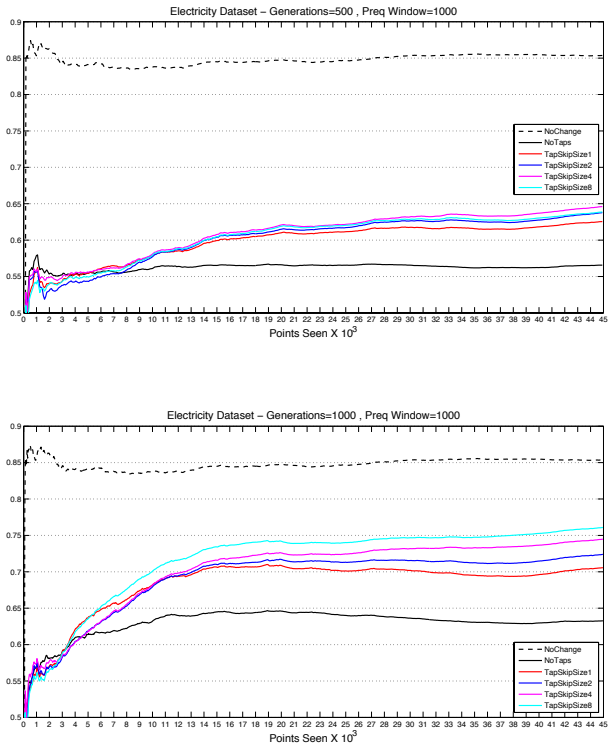

Figure 4: GP Prequential accuracy during stream. Electricity data set. First 10% of stream (4.5×10^3 exemplars) are used to construct the initial model. Top: Label budget of $\approx 22\%$ or $i_{max} = 500$; Bottom: Label budget of $\approx 44\%$ or $i_{max} = 1,000$

6 Conclusion

A framework is proposed and benchmarked for applying GP to streaming data classification tasks under limited label budgets. This effectively makes GP an online algorithm able to adapt on a continuous basis to data streams created by a non-stationary source. From the perspective of scalability we note that the cost of any single GP generation is set by the cost of Pareto archiving i.e., set by the size of the data subset. Future work will continue to develop the framework to utilize additional information when deciding which exemplars to request label information for i.e., classifier confidence (see dashed line in Figure 1).

Acknowledgments

The authors gratefully acknowledge support from the NSERC CRD program (Canada).

References

[1] A. Atwater. Towards coevolutionary genetic programming with Pareto archiving under streaming data. Master's thesis, Faculty of Computer Science, 2013.

[2] A. Atwater and M. I. Heywood. Benchmarking Pareto archiving heuristics in the presence of concept drift: diversity versus age. In *ACM Genetic and Evolutionary Computation Conference*, pages 885–892, 2013.

[3] A. Atwater, M. I. Heywood, and A. N. Zincir-Heywood. GP under streaming data constraints: a case for Pareto archiving? In *ACM Genetic and Evolutionary Computation Conference*, pages 703–710, 2012.

[4] M. Behdad and T. French. Online learning classifiers in dynamic environments with incomplete feedback. In *IEEE Congress on Evolutionary Computation*, pages 1786–1793, 2013.

[5] U. Bhowan, M. Johnston, M. Zhang, and X. Yao. Evolving diverse ensembles using genetic programming for classification with unbalanced data. *IEEE Transactions on Evolutionary Computation*, 17(3):368–386, 2013.

Table 2: Overall prequential accuracy of GP – Mean and standard deviation. Values in bold indicate a best case for each label budget.

Discrete Concept Shift data set				
	label budget ≈ 0.3%		label budget ≈ 3%	
TapSkip	mean	std. dev.	mean	std. dev.
none	**62.6**	5.0	75.4	5.3
1	62.0	3.3	75.9	6.0
2	62.4	3.8	80.5	4.6
4	61.8	4.1	**80.6**	4.7
8	59.2	5.1	77.1	5.5
Continuous Concept Drift data set				
	label budget ≈ 6.7%		label budget ≈ 13.3%	
none	57.4	2.7	60.3	2.2
1	57.6	3.3	65.2	2.6
2	59.8	3.8	65.0	3.3
4	59.4	4.2	**65.7**	2.4
8	**60.0**	3.2	64.6	2.6
Electricity data set				
	label budget ≈ 22%		label budget ≈ 44%	
none	56.6	2.0	63.4	1.3
1	62.3	2.4	71.1	3.2
2	63.1	2.6	72.3	2.8
4	**64.3**	3.3	75.4	3.6
8	63.2	3.3	**76.5**	3.6

[6] A. Bifet. *Adaptive Stream Mining: Pattern Learning and Mining from Evolving Data Streams*, volume 207 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2010.

[7] A. Bifet and R. Gavalda. Learning from time-changing data with adaptive windowing. In *SIAM International Conference on Data Mining*, pages 443–448, 2007.

[8] A. Bifet, I. Žliobaitė, B. Pfahringer, and G. Holmes. Pitfalls in benchmarking data stream classification and how to avoid them. In *Machine Learning and Knowledge Discovery in Databases*, volume 8188 of *LNCS*, pages 465–479, 2013.

[9] A. Cercantes, P. Isasi, C. Gagné, and M. Parizeau. Learning from non-stationary data using a growing network of prototypes. In *IEEE Congress on Evolutionary Computation*, pages 2634–2641, 2013.

[10] H. H. Dam, C. Lokan, and H. A. Abbass. Evolutionary online data mining: An investigation in a dynamic environment. In *Studies in Computational Intelligence*, volume 51, chapter 7, pages 153–178. Springer, 2007.

[11] T. Dasu, S. Krishnan, S. Venkatasubramanian, and K. Yi. An information-theoretic approach to detecting changes in multi-dimensional data streams. In *Proceedings of the Symposium on the Interface of Statistics*, 2006.

[12] A. P. Dawid. Statistical theory: The prequential approach. *Journal of the Royal Statistical Society-A*, 147:278–292, 1984.

[13] E. D. de Jong. A monolithic archive for pareto-coevolution. *Evolutionary Computation*, 15(1):61–93, 2007.

[14] I. Dempsey, M. O'Neill, and A. Brabazon. *Foundations in Grammatical Evolution for Dynamic Environments*, volume 194 of *Studies in Computational Intelligence*. Springer, 2009.

[15] G. Ditzler and R. Polikar. Hellinger distance based drift detection for non stationary environments. In *IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments*, pages 41–48, 2011.

[16] J. A. Doucette, A. R. McIntyre, P. Lichodzijewski, and M. I. Heywood. Symbiotic coevolutionary genetic programming: a benchmarking study under large attribute spaces. *Genetic Programming and Evolvable Machines*, 13(1):71–101, 2012.

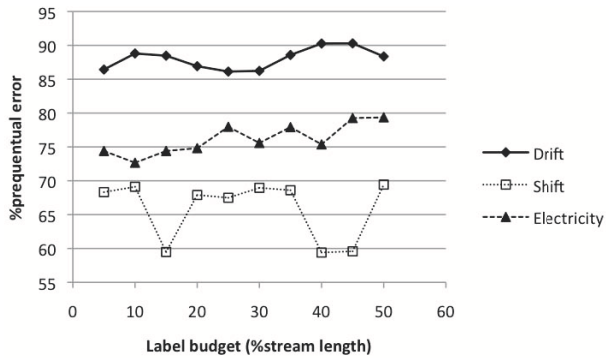

Figure 5: Overall prequential accuracy – MOA adaptive Naive Bayes.

[17] W. Fan, Y. Huang, H. Wang, and P. S. Yu. Active mining of data streams. In *Proceedings of SIAM International Conference on Data Mining*, pages 457–461, 2004.

[18] G. Folino and G. Papuzzo. Handling different categories of concept drift in data streams using distributed GP. In *European Conference on Genetic Programming*, volume 6021 of *LNCS*, pages 74–85, 2010.

[19] J. Gama. *Knowledge discovery from data streams*. CRC Press, 2010.

[20] J. Gama. A survey on learning from data streams: current and future trends. *Progress in Artificial Intelligence*, 1(1):45–55, 2012.

[21] J. Gama, P. Medas, G. Castillo, and P. Rodrigues. Learning with drift detection. In *Advances in Artificial Intelligence*, volume 3171 of *Lecture Notes in Computer Science*, pages 286–295, 2004.

[22] J. Gama, R. Sebastião, and P. Rodrigues. On evaluating stream learning algorithms. *Machine Learning*, 90(3):317–346, 2013.

[23] M. Harries. Splice-2 comparative evaluation: Electricity pricing. Technical report, University of New South Wales, 1999.

[24] S. Huang and Y. Dong. An active learning system for mining time changing data streams. *Intelligent Data Analysis*, 11(4):401–419, 2007.

[25] H. Iba and C. C. Aranha. *Practical applications of evolutionary computation to financial engineering*, volume 11 of *Adaptation, Learning, and Optimization*. Springer, 2012.

[26] D. Kifer, S. Ben-David, and J. Gehrke. Detecting change in data streams. In *Proceedings of the International Conference on Very Large Data Bases*, pages 180–191. Morgan Kaufmann, 2004.

[27] C. Lanquillon. Information filtering in changing domains. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 41–48, 1999.

[28] P. Lichodzijewski and M. I. Heywood. Managing team-based problem solving with Symbiotic Bid-based Genetic Programming. In *ACM Proceedings of the Genetic and Evolutionary Computation Conference*, pages 363–370, 2008.

[29] P. Lichodzijewski and M. I. Heywood. Symbiosis, complexification and simplicity under GP. In *ACM Proceedings of the Genetic and Evolutionary Computation Conference*, pages 853–860, 2010.

[30] P. Lindstrom, B. MacNamee, and S. J. Delany. Handling concept drift in a text data stream constrained by high labelling cost. In *Proceedings of the International Florida Artificial Intelligence Research Society Conference*. AAAI, 2010.

[31] P. Lindstrom, B. MacNamee, and S. J. Delany. Drift detection using uncertainty distribution divergence. *Evolutionary Intelligence*, 4(1):13–25, 2013.

[32] Alexander Loginov and Malcolm I. Heywood. On the impact of streaming interface heuristics on GP trading agents: an FX benchmarking study. In *Proceeding of the ACM Genetic and Evolutionary Computation Conference*, pages 1341–1348, 2013.

[33] J. Quinonero-Candela, M. Sugiyama, A. Schwaighofer, and N. D. Lawrence, editors. *Dataset shift in machine learning*. MIT Press, 2009.

[34] S. Rahimi, A. R. McIntyre, M. I. Heywood, and N. Zincir-Heywood. Label free change detection on streaming data with cooperative multi-objective genetic programming. In *ACM Genetic and Evolutionary Computation Conference*, pages 159–160, 2013.

[35] R. Sebastio and J. Gama. Change detection in learning histograms from data streams. In *Proceedings of the Portuguese Conference on Artificial Intelligence*, volume 4874 of *LNCS*, pages 112–123. Springer, 2007.

[36] I. Žliobaitė, A. Bifet, B. Pfahringer, and G. Holmes. Active learning with evolving streaming data. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 597–612. Springer, 2011.

[37] I. Žliobaitė, A. Bifet, B. Pfahringer, and G. Holmes. Active learning with drifting streaming data. *IEEE Transactions on Neural Networks and Learning Systems*, 25(1):27–54, 2014.

[38] P. Vorburger and A. Bernstein. Entropy-based concept shift detection. In *Proceedings of the Sixth International Conference on Data Mining*, pages 1113–1118, 2006.

[39] G. Wilson and W. Banzhaf. Interday and intraday stock trading using PAM GP and linear GP. In A. Brabazon, O'Neill, and D. G. Maringer, editors, *Natural Computing in Computational Finance 3*, volume 293 of *SCI*, pages 191–212. Springer, 2010.

[40] X. Wu, K. Yu, W. Ding, H. Wang, and X. Zhu. Online feature selection with streaming features. *IEEE Transactions on Pattern Analysis and Machine Learning*, 35(5):1178–1182, 2013.

[41] X. Zhu, P. Zhang, X. Lin, and Y. Shi. Active learning from stream data using optimal weight classifier ensemble. *IEEE Transactions on Systems, Man, and Cybernetics – Part B*, 40(6):1607–1621, 2010.