Memory management for data streams subject to concept drift

Pierre-Xavier Loeffel, Christophe Marsala, Marcin Detyniecki

Memory management for data streams subject to concept drift

Pierre-Xavier Loeffel[1], Christophe Marsala[1] and Marcin Detyniecki[1 2]

1- Sorbonne Universités, UPMC Univ Paris 06 - CNRS, LIP6 UMR 7606
4 place Jussieu 75005 Paris - France

2- Polish Academy of Sciences - IBS PAN
Warsaw - Poland

Abstract. Learning on data streams subject to concept drifts is a challenging task. A successful algorithm must keep memory consumption constant regardless of the amount of data processed, and at the same time, retain good adaptation and prediction capabilities by effectively selecting which observations should be stored into memory. We claim that, instead of using a temporal window to discard observations with a time stamp criterion, it is better to retain observations that minimize the change in outputted prediction and rule learned with the full memory case. Experimental results for the Droplets algorithm, on 6 artificial and semi-artificial datasets reproducing various types of drifts back this claim.

1 Introduction

Increasingly, data arrive as streams. Financial prices or values from sensors are examples of streams that produce large amounts of data. Processing these data streams has proved to be a very challenging task. A first challenge is to design an algorithm able to learn on the potentially infinite dataset produced by a stream, under the constrain of limited available computer memory. Another issue is the potential changes that can arise in the data distribution over time, a phenomenon known as Concept Drift.

In a previous paper, we devised an algorithm able to successfully deal with several types of drifts and perform better than state of the art algorithms [2], but which was storing all the past observations into memory. In this paper we devise a Rule Preserving (RP) criterion that keeps memory consumption constant and efficiently choose which observations should be retained or discarded.

The paper is organized as follows: Section 2 lays down the framework, the challenges and the existing approaches. Section 3 explains the proposed solution, Section 4 describes the experiments and Section 5 concludes.

2 The issue of memory management

The problem considered is supervised learning classification for data stream subjects to concept drift. Observations $X \in \chi$, are endlessly emitted, one by one, from a data stream. Each observation is at first labeless and the goal is to learn a model $f : \chi \to Y$ that accurately predicts the label $y \in Y$. In this framework, the observations are not assumed to be i.i.d. Consequently, the hidden joint distribution $P(X, y)$, generating the data, can change over time (Concept Drift). We assume that the memory used by an algorithm is proportional to the number of observations stored into memory. Hence, constraining memory usage is here equivalent to constraining the Maximum Number of Observations (MNO) in memory.

In order to deal with the limited available memory on a computer, a first solution is to use on-line learning algorithms [5]. Their memory consumption is kept constant by updating the current rule with the latest observation and deleting it after update. Another solution is to maintain a rule, based on a temporal window which includes a set of the latest observations. This window can be either of fixed or adaptive length [1].

Dealing with concept drift requires adaptation which in turns leads to forget outdated observations. The issue of on-line algorithms is that they do not really forget past observations as they include no explicit forgetting mechanism. The observations are simply diluted over time as the rule learned changes [1]. In the case of the temporal window, the underlying assumption is that the latest observations accurately reflect the actual joint distribution $P(X, y)$. Unfortunately, this assumption doesn't always hold in reality (e.g. noisy observations).

Instead of keeping into memory observations based on their time stamp, we propose a RP criterion that aims at keeping memory consumption constrained and at yielding predictions and learned rule as close as possible to the predictions and learned rule that would have been obtained in the full memory case. We claim that this strategy leads to better performances than a simple time stamp criteria. In the next section, this idea is developed for the Droplets algorithm.

3 The Droplets algorithm with memory management

The Droplets algorithm [2] uses the values of the observation X as coordinates for the center of an hypersphere in an orthonormal set. Each hypersphere is assigned the class of the related observation and a default radius $R_{default}$. When two hyper-spheres that do not belong to the same class overlap each other, their radii are decreased to make them tangent (Figure 1), whereas hyper-spheres associated with the same class are allowed to overlap each other. If the coordinates of the new unlabeled observation are inside an existing hypersphere, the algorithm will predict the class associated with this hypersphere and abstain otherwise.

The idea of the RP criterion is to establish a ranking of the observations in memory according to the differences with the full memory setting that their deletion would produce. Observations at the top of the ranking would change the predictions and learned rule the most and thus, should be saved in priority.

The Droplets algorithm equipped with the RP[1] is shown on the left hand side of Figure 1. Before reception of a new observation, the first part of the algorithm (lines 3 to 8 in Algorithm 1) checks whether the MNO is reached. If this is the case, it creates the list of observations for which the ranking will be

[1]A video of the algorithm running can be found here: https://www.youtube.com/watch?v=_uLhRX9FXxc

updated, deletes the observation that has the lowest ranking from the memory (referred to map here) and update the ranking. After reception of the label of the observation, the second part of the algorithm (lines 12 to 18) also creates the list of observations for which the ranking will be updated and performs the update (line 20).

The first criteria of the ranking is the Volume Not Overlapped (VNO) of each hypersphere: the volume of the observation that is not covered by at least another observation. Because the prediction of the Droplets depends only on the area covered by each class and, because there is no assumption on the distribution of the data, the higher this value, the more the deletion of an observation with high VNO should create changes with the full memory setting. Hence the observation with the lowest VNO[2] is deleted in priority.

When there is a tie in the ranking (in particular, observations that have $VNO = 0$), the second deletion criteria minimizes the change in areas covered by each class *after* update (i.e. the rule learned). The new rule learned depends on the values of X_{t+1}. If observation $t{+}1$ is not overlapping at least one observation belonging to another class, the choice that minimizes the change in the learned rule is to delete observations according to their VNO. Otherwise, the update mechanism decreases the radii of all the observations in conflict with observation $t{+}1$, up to the point where observation *i,* that has the largest overlap with the observation $t{+}1$, becomes tangent with observation $t{+}1$ (Figure 1). For two observations *t+1* and i, $(i < t+1)$ with respective radii R_{t+1}, R_i, the overlap is given by: $\lambda = R_{t+1} + R_i - \|Center_{t+1} - Center_i\|$. λ increases as the radii of the observations increase and decrease as the distance between the 2 centers grows. As no assumption can be made on the values of the features at $t{+}1$ (and hence the distance of the centers), it is assumed that an observation with a big radius is more likely to have a larger overlap with observation $t{+}1$ than an observation with a small radius. This leads to pick the radius as a second criteria for deletion (in case of tie for VNO) and delete in priority small radii.

The temporal complexity of the RP is $O\left(M.n^2\right)$ with M the number of Monte Carlo simulations and n the maximum number of observations allowed in memory.

We now experimentally show that the RP is better at preserving the outputs of the full memory algorithm and achieves superior performances than a simple temporal window.

4 Experimental results

In order to compare the RP against a temporal window, we carried out two sets of experiments on 6 artificial and semi-artificial datasets[3] including 1000 observations and reproducing different types of drifts (the drifts are evenly distributed

[2]To compute the VNO, Monte Carlo simulations were used, randomly sampling M points (where M is a fixed parameter) in the hypersphere of interest and assessing the fraction of points that belong to at least another hypersphere. M depends on the precision one wants to achieve: When M increases by a factor of 100, the precision increases by a factor of 10.

[3]The datasets used can be downloaded here: http://webia.lip6.fr/~loeffel/ESANN2016/

```
Algorithm 1 The Droplets with the Rule Preserving criterion
Inputs: $MNO$, $map_0 \leftarrow \emptyset$, $Ranking_0 \leftarrow \emptyset$, $R_{default}$
Foreach new observation :$k \leftarrow 1, 2, \ldots$
  $n \leftarrow Number\ of\ observations\ in\ map_{k-1}$
  If $n = MNO$ Then
    $Obs_j \leftarrow$Observation at the bottom of $Ranking_{k-1}$
    $Indexes \leftarrow$Get list of circles overlapped by $Obs_j$
    $map_{k-1} \leftarrow Delete\,(Obs_j)$
    $Ranking_{k-1} \leftarrow Update\ Ranking(map_{k-1}, Ranking_{k-1}, Indexes)$
  End If
  $Obs_k \leftarrow Receive\ New\ Observation$
  $\hat{y_k} \leftarrow Predict\,(Obs_k)$
  $y_k \leftarrow Receive\ Class\ Label\ of\ New\ Observation$
  $Indexes_k \leftarrow$Get list of all circles overlapped by $Obs_k$
  Foreach $y_i \in Indexes_k$ //Get class associated with obs in list
    If $y_i \neq y_k$ Then
      $Indexes_i \leftarrow$Get list of circles overlapped by $Obs_i$
      $Indexes \leftarrow Indexes \cup Indexes_i$
    End If
  End Foreach
  $map_k \leftarrow Update\ map\,(map_{k-1}, Obs_k)$
  $Ranking_k \leftarrow Update\ Ranking(map_k, Ranking_{k-1}, Indexes)$
End Foreach
```

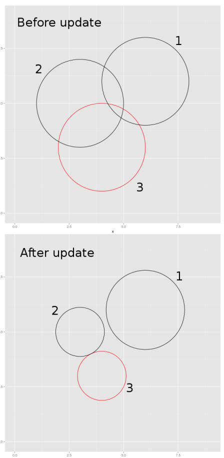

Figure 1: Left: The Droplets algorithm with Rule Preserving criterion. Right: An example of rule update (the classes are indicated by the colors of the circles)

on the datasets). The descriptions of the datasets RBF, Rotating Hyperplane and Temperatures can be found in [2], a description of SEA is given in [3]. The default radius for all the experiments was 0.1 whereas the number of Monte Carlo simulation was set to $M = 1000$. In each case, the first 100 observations were kept for initialization purpose and the performance was assessed on the remain observations. We implemented 2 versions of the Droplets that only differ by the way they manage memory. The first version is based on a Temporal Window (TW) that drops the oldest observation once the memory is full whereas the second version uses the RP.

In the first set of experiments, we compared the difference of performance of the 2 methods on the 6 datasets. The usual performance metrics of an algorithm K performing classification with a reject option are its percentage of good predictions (accuracy) $P_{acc}(K)$ and its percentage of unclassified observations $P_{unc}(K)$ [4]. For each dataset, the values displayed in Figure 2 are equal to: $P_{acc}(RP) - P_{acc}(TW)$ and $P_{unc}(TW) - P_{unc}(RP)$, on the upper and lower figure respectively. In both cases, a positive number indicates that the RP outperforms a simple window.

The Rule Preserving criterion outperforms a temporal window: The results indicate that globally the RP has a better accuracy and a lower percentage of unclassified observations than a simple window, regardless of the dataset, the type (or absence) of drift and the number of observations allowed in memory. The results exhibit a bell curve structure centered around 40-60 observations. The differences in performances can be as significant as +15% in correctly classified observations and -16% of unclassified observations for a MNO of 40. As the MNO increases, the coverage of both methods converge to a point where there is not much difference (less than 2%) but where the RP retains an hedge over

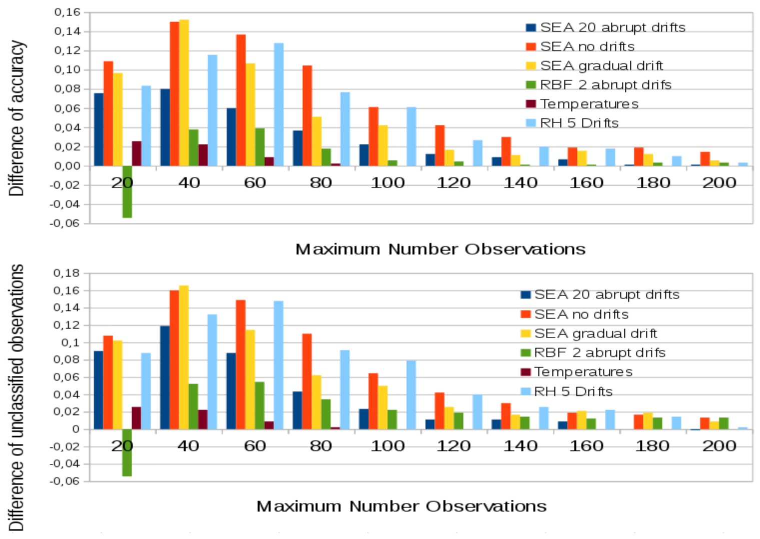

Figure 2: Difference in performances: Temporal Window vs Rule Preserving criterion

the window.

The second set of experiments aims at assessing whether the RP manages to minimize the change in output (a prediction or an unclassified observation) compared to the full memory setting. For each observation we compared the output of the full memory and RP and classified them into 4 categories: exactly the same output, prediction (for the full memory) to unclassified (for the RP), unclassified to prediction and two different predictions. For comparison purposes, we performed the same experiment with the TW. The results for the SEA dataset with 20 drifts are shown in Figure 3. For a given MNO, the left bar represents the RP whereas the right bar represents the TW. The results on the 5 remaining datasets are similar but couldn't be shown here for lack of space.

The RP replicates well the outputs of the Droplets with full memory: The results indicate that despite the constrain on memory usage, the majority of the outputs obtained by the RP are exactly the same as the ones obtained in the full memory setting (58% of the time in the worst case when MNO = 20) and that it consistently over performs the TW on this criteria. When the output was different, the majority of the formerly classified observations went to unclassified most of the time (35% of the observations went to unclassified against 0.5% for which the prediction completely changed when the MNO=20).

5 Conclusion and future directions

Learning on a data stream subject to concept drifts is a challenging task. A successful algorithm must be able to keep memory consumption constant and

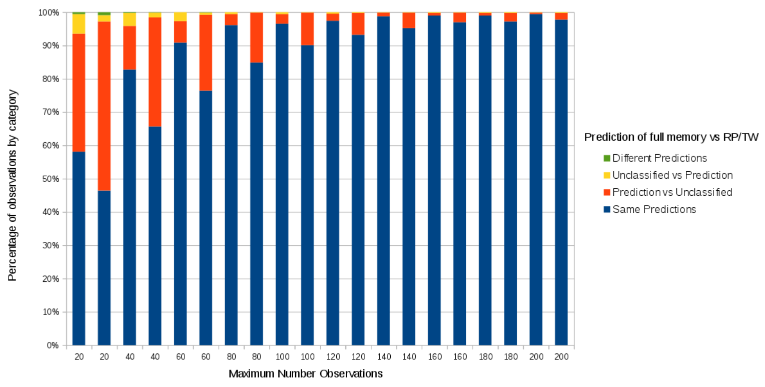

Figure 3: Outputs by categories of the RP (left) vs TW (right) on the SEA dataset with 20 abrupt drifts

retain good adaptation and predictions capabilities by effectively selecting which observations should be stored into memory. In this paper, we proposed a Rule Preserving criterion that retains observations which will result in prediction and rule learned similar to the full memory setting. An application of this criterion is devised for the Droplets algorithm. The results on several datasets reproducing evolving environments, indicate that this strategy leads to a larger percentage of correct predictions and a lower percentage of unclassified observations than discarding observations based on a temporal window.

Future work will investigate whether the Rule Preserving criterion can be used to improve the performances of other algorithms.

References

[1] Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M. & Bouchachia, A., A survey on concept drift adaptation. ACM Computing Surveys, 45(4), 44:1-44:37 (2014).

[2] Loeffel, P-X, Marsala, C., and Detyniecki, M., Classification with a reject option under Concept Drift: the Droplets Algorithm, IEEE/ACM International Conference on Data Science and Advanced Analytics (DSAA'2015), 1-9

[3] W. Street, Y. Kim, A streaming ensemble algorithm (SEA) for large-scale classification, in: KDD'01, 7th International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, August 2001, pp. 377-382.

[4] El-Yaniv, R., & Wiener, Y. (2010). On the Foundations of Noise-free Selective Classification. Journal of Machine Learning Research, 11, 1605–1641.

[5] N. Littlestone. 1987. Learning Quickly When Irrelevant Attributes Abound: A New Linear-threshold Algorithm. Machine Learning 2, 4 (1987), 285–318.