Data Stream Classification using Active Learned Neural Networks

Paweł Ksieniewicz[a,*], Michał Woźniak[a], Bogusław Cyganek[b], Andrzej Kasprzak[a], Krzysztof Walkowiak[a]

[a]*Department of Systems and Computer Networks, Wrocław University of Science and Technology, Wyb. Wyspiańskiego 27, 50-370 Wrocław, Poland*
[b]*AGH University of Science and Technology Al. Mickiewicza 30, 30-059 Kraków, Poland*

Abstract

Due to variety of modern real-life tasks, where analyzed data is often not a static set, the data stream mining gained a substantial focus of machine learning community. Main property of such systems is the large amount of data arriving in a sequential manner, which creates an endless stream of objects. Taking into consideration the limited resources as memory and computational power, it is widely accepted that each instance can be processed up once and it is not remembered, making reevaluation impossible. In the following work, we will focus on the data stream classification task where parameters of a classification model may vary over time, so the model should be able to adapt to the changes. It requires a forgetting mechanism, ensuring that outdated samples will not impact a model. The most popular approaches base on so-called *windowing*, requiring storage of a batch of objects and when new examples arrive, the least relevant ones are forgotten. Objects in a new window are used to retrain the model, which is cumbersome especially for *online* learners and contradicts the principle of processing each object at most once. Therefore, this work employs inbuilt forgetting mechanism of neural networks. Additionally, to reduce a need of expensive (sometimes even impossible) object labeling, we are focusing on *active learning*, which asks for labels only for *interesting* examples, crucial for appropriate model upgrading. Characteristics of proposed methods were evaluated on the basis of the computer experiments, performed over diverse pool of data streams. Their results confirmed the convenience of proposed strategy.

Keywords: pattern classification, data stream, active learning, concept drift, forgetting

*Corresponding author
Email addresses: `pawel.ksieniewicz@pwr.edu.pl` (Paweł Ksieniewicz), `michal.wozniak@pwr.edu.pl` (Michał Woźniak), `cyganek@agh.edu.pl` (Bogusław Cyganek), `andrzej.kasprzak@pwr.edu.pl` (Andrzej Kasprzak), `krzysztof.walkowiak@pwr.edu.pl` (Krzysztof Walkowiak)

1. Introduction and related works

Classification is one of the most frequent decision task which aims at assigning an observed object to one of the predefined categories. There are plethora of solutions [1], nonetheless, there are still a few issues that need to be discussed for contemporary pattern recognition systems. We have to realize that modern machine learning algorithms should build the predictive models on the basis of huge, rapidly changing databases, i.e., we have to propose efficient procedures which can produce accurate classifiers on the basis of so-called data streams. This issue is still a focus of diligent studies, because many classifiers cannot upgrade their models using recent observations, and they usually do not take into consideration that the statistical dependencies between the attributes, describing incoming objects, and their correct classification may change over time.

1.1. Concept drift

Aforementioned phenomenon is called a *concept drift* [2] and its appearances have become a challenge for many practical tasks, such as medical diagnosis (surgery prediction) [3], computer systems security (SPAM filtering, IPS design) [4, 5], marketing (client profiling) [6], or banking (fraud detection) [7] to enumerate only a few.

There are several taxonomies of *concept drift*, based on its quickness (sudden shift and smooth ones, as incremental changes) or impact on the *posterior* probabilities [2, 8] (*real concept drift* or *virtual concept drift*). Considering classification task, the *real drift* is crucial, because, in opposite to *virtual*, it can significantly change the shape of the decision boundaries.

Changes may be discovered by monitoring the unlabeled data and observing novelties related to the presence of *outliers* or by monitoring classification accuracy [9, 8]. However, to properly detect the *real concept drift* we require the labels, because detectors based on unlabeled examples do not guarantee to sense the *real* drift, while the *virtual* one may be detected precisely [10].

1.2. Data stream classification

Development of the methods efficiently tackling phenomenon of *concept drift*, have become an important research challenge for machine learning community. Basically, we may consider the following approaches to deal with it:

- **Frequently retraining a classification model, when data comes.** It is costly and practically unviable. Moreover, in the case, when a drift does not appear, rebuilding is needless, but if a model shift occurs rapidly, the time laps of a new model may be unacceptable.

- **Detecting *concept drift* by monitoring incoming data.** Classifier is retrained on the basis of newly collected objects if the changes of accuracy is significant *enough*.

- **Constant classifier updating**. It uses incremental learning methods that allow to add new training observations during a classifier exploiting or by implementing forgetting mechanisms as dataset weighting or windowing [11].

Gama et al. [11] analyze adaptive algorithms that can react to changes in data streams. Authors enumerate two main concepts of memory and forgetting. Firstly, let's focus on the *online* learners[12], which have to meet the following requirements:

- The classifier learning algorithm has limited memory and processing time at its disposal.
- Every incoming learning object may be processed at most once over a training and it is not memorized, because of aforementioned limitations.
- The training can be paused at any time, but the accuracy of *online* trained classifier should be the same or higher than the quality of a model trained on the whole batch of data collected by then.

The *online* learners are seen as naturally adaptive methods, because they continuously update the predictive model on the basis of incoming observations and they behave pretty well especially for slow changing data streams. There are several propositions of such algorithms, which should be mentioned, as WINNOW [13] or VFDT [14]. Nevertheless, the main drawback of *online* learners is slow adaptation to sudden concept drift, therefore the several works are trying to combat with this problem by control the model updating on the basis of new incoming observations [15]. We have also mention the works where authors propose how to explicitly react to changes in data streams as STAGGER [16], DWM [17], or GT2FC [18] to enumerate only a few.

One may also mentions algorithms that incorporate the *forgetting mechanism* in the form of windowing or data weighting. This approach assumes that examples come in recent time are the most significant, because they are consistent with the present context. Nevertheless, their relevance decreases in the process of time. Therefore, taking into consideration only the latest incoming objects seems to be reasonable, because it helps to collect a dataset representing the present context. We may enumerate the following strategies:

- Selecting the learning examples by means of a sliding data window that cuts off older observations [2].
- Weighting learning instances according to their relevance [19].
- Applying bootstraping-based algorithms as *boosting*, that focus on incorrect classifier objects [20].

One may observe that two main forgetting mechanisms are usually employed. For typical windowing *abrupt forgetting* is frequently used, i.e., the outdated (old) observations are not used to update the model and they are also removed from memory, while *gradual forgetting* means that no observation is discarded from memory, but the observations are associated with weights related to the period of their staying in memory [11]. This approach requires implementation of a decay function, which ensures that old observations do not impact strongly to the recent model. There are several examples of decay functions, as linear one proposed by Koychev [21] or exponential decay function proposed by Klinkerberg [22]. Nevertheless, this approach has one significant drawbacks, because it requires that all observations have to be stored in the memory. This assumption does not fulfill the requirement that we have the limited memory at our disposal. Therefore, if the weight of a given object is less than a given threshold then the object is removed from the memory [23]. One has also mention the interesting researches on *Reservoir Sampling*, where a representative sample are obtained for the analyzed data stream [24].

On the other hand, the main problem of using a sliding window approach is to properly set the window size. The shorter window allows to concentrate on the emerging context, though data may not be representative for a longer lasting context. On the other hand, the bigger window may include instances representing different contexts [25]. Thus, we may find the propositions where the fixed window size is used, i.e., the fixed number of recent incoming examples are stored, or the number of the memorized examples may vary over time. Usually, the windows size depends on drift detector output, i.e., if drift is detected then the window size is decreased. It is worth mentioning FLORA2 [2], which is recognized as the first method, where adaptive windowing had been used, but its descendants as FLORA3 and FLORA4 should be instanced, because they are able to deal with recurring concept drift and noisy data, again several works propose to adjust the window size dynamically, as ADWIN2 [26], or [27] where multiple window approach is discussed.

In this work, we decided to benefit from the phenomenon called *catastrophic forgetting* [28], which is unnecessary for the multi-task learning, because neural networks exposed on the data relevant to new task have the tendency to abruptly forget the knowledge of the previously learned ones. This forgetting occurs specifically when the network is trained sequentially on multiple tasks, because the weights in the network that are important for the previous tasks are changed to meet the new objectives [29].

This adverse behavior for multi-task learning is highly desirable in classifier updating for the drifted data stream, because artificial neural networks should naturally forget outdated concept by rebuilding their internal knowledge representation to properly solve incoming problem.

1.3. Labeling

In this work we will mostly focus on the labeling cost. Majority of the data stream classifiers produce models on the basis of supervised learning algorithms. For some practical tasks we are able to obtain true labels with reasonable cost and time (weather prediction), but for the most of practical tasks, labeling requires human effort or access to the kind of an oracle (e.g., for medical diagnosis

– human expert should always verify the diagnosis), thus it is usually very expensive. Let us notice that labels are usually assigned by human experts and therefore they can not label all the new examples if they come too fast, e.g., for SPAM filtering – user should confirm the decision if incoming mail is legitimate or not, i.e., the continuous access to the human expert should be granted. Additionally, sometimes the proper classification is available with a long delay (e.g., for the true label for credit approval is available ca. 2 years after the decision).

Therefore methods of classifier design which could produce the recognition system on the basis of a partially labeled set of examples (especially if learner could point out the interesting example to be labeled) are still very desirable goal [30].

Žliobaitė et al. [31] discuss the theoretical framework for predictive model learning using active learning approach and discuss tree labeling strategies. Kurlej and Wozniak [32] propose the active learning approach for minimal distance classifier applied to drifted streams, where decision about labeling depends on the distance between a given example to the decision boundary. In their later works [25, 33] they analyze selected characteristics of such an approach.

Nguyen et al. [34] develop an incremental algorithm CSL-*stream*, that performs clustering and classification at the same time. Mohamad [35] propose similar data stream classifier, which combines uncertainty and density-based querying criteria. Korycki and Krawczyk [36] apply active learning strategy to classify data streams and combine it with self-labeling approach.

The windowing is very accurate for the models which does not require learning with *lazy learners* [37], but for the classifiers based on models, the data window shift requires to rebuild the model on the basis of data in a recent window. Our work will focus on a hybrid active learning approach which is combination of *online* learning approach and sliding windows method with forgetting. Because of the fact, that implementation of the forgetting mechanism requires additional memory to remember the data in window and additional computational effort to rebuild the model when the window is shifted, we have been looking for a classification model with inbuilt forgetting mechanism. It will make our method highly suitable to real-life data stream mining with a limited budget.

1.4. Contributions

In a nutshell, the contributions of this work are as follows:

- Proposition of the active learning strategy which makes a decision for each classification model on the basis of support function.
- Employing *catastrophic forgetting* phenomenon as forgetting mechanism for neural network classifiers.
- Experimental evaluation of the proposed approach on the basis of diverse benchmark datasets and a detailed comparison with the semi-supervised and fully supervised strategies.

2. Proposition of active learning neural network classifier for data stream

Firstly, let us formalize the classification model. Usually, a classifier Ψ makes a decision using so-called support functions which return support for each class [1]

$$\mathcal{F} = \{F_1, F_2, ..., F_M\} \tag{1}$$

To make a decision, Ψ usually uses the maximum rule

$$\Psi(x) = \max_{k \in \mathcal{M}} \left(F_k(x)\right), \tag{2}$$

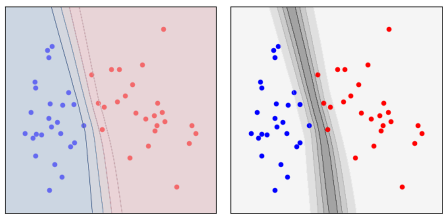
where x is the feature vector, k stands for the label from finite set of possible classes $\mathcal{M} = \{1, 2, ..., M\}$.

We assume that if a decision about an object is supported by the high value of the support, then the label is not so *"interesting"*, because it probably will not have a significant impact on the classification model improvement. If support functions have probabilistic interpretation, e.g., they are *posterior* probabilities (in the case of so-called 0-1 loss function) [1], then high value of the support function associated with a chosen class means that the probability of misclassification is very low. Otherwise, if the difference among the support function values is low, i.e., the decision is very uncertain.

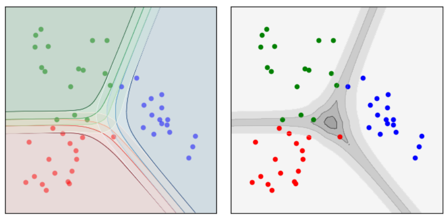
Let us propose RSFD (*Relative Support Function Difference*) function, which measures average difference among the highest support and support for the rest of the classes for a given observation x

$$RSFD(x) = \frac{\sum_{i=1}^{M} \left[\max_{k \in \mathcal{M}}(F_k(x)) - F_i(x)\right]}{M - 1} \tag{3}$$

The graphical interpretation of RSFD is presented in Fig. 1.



(a) binary problem



(b) multiclass problem

Figure 1: An example of the supports for each of classes (left) and RSFD (*Relative Support Function Difference*) (right) for 2 and 3 class problems.

This approach is similar to the *classification with reject option* [38, 39], where the trade-off between the error and rejection rate is considered. Basically, in this approach, the decision is made only in a case if the maximum support is high enough (or difference between maximum support and supports for the rest of classes is significant), otherwise the classifier rejects the decision, instead of a giving label, answering *"I do not know"*.

The proposed framework works as the block classifier [40], because it collects the data in the form of chunk, but for each chunk the *online* learner is used. To avoid the influence of object order on a decision about labeling, samples in each chunk are randomized before processing.

The decision about the object labeling depends on two parameters:

- *threshold* – responsible for choosing the *"interesting"* examples, i.e., if relative support function difference is lower than a given threshold the object seems to be interesting and the learning algorithm is asking for its label.
- *given_budget* – the label will be assigned only in the case if its budget related to a given chunk allows to pay for it. For each chunk only limited percentage of the objects could be labeled.

The idea of the algorithm is presented in Alg. 1.

3. Inbuilt forgetting mechanism of artificial neural networks

Because we would like to employ inbuilt forgetting mechanism of artificial neural networks, therefore let us shortly present their main properties when they are used as the data stream classifiers. For each simulations presented in sections 3 and 4, the Multilayer Perceptron (MLP) has been used with one hidden layer. In this section the chosen results of the simulation research are presented, but the additional results, as well as the used software may be found in article repository[1]. All the examples presented below have been prepared with data streams generated using *Radial Basis Function* for different concept drift types.

Figure 2 shows learning curves of three identical neural networks (MLP with 100 neurons in a hidden layer) classifying a stream (10k instances with two sudden drifts) using different size of data chunk.

As we can observe, a smaller chunk (red line) means a more dynamic learning curve (faster accuracy growth). This is most likely due to the increased frequency of repeating the learning procedure, even with smaller portions of information, leading faster to establishment of a generalization power. Simultaneously, a larger chunk (blue line) leads to a larger decrease of accuracy caused by a sudden drift. At the other side, the smaller chunk, with extreme of incremental learning when we run a learning procedure over every single object, also leads to a higher computational complexity.

Figure 3 shows the reaction of different neural networks (10, 50 and 500 neurons) to different types of concept drift (from sudden drift to stages of incremental drift).

[1] `https://github.com/w4k2/active_learning`

Algorithm 1 Active learning classifier for data stream

```
Require: input data stream,
    $n$ - data chunk size,
    $M$ - number of classes,
    $incremental\_training\_procedure()$ - classifier training procedure,
    $label()$ - function which returns label of a given example,
    $classifier()$ - classification model,
    $F_1(), F_2(), ..., F_M()$ - support functions using by $classifier()$,
    $m$ - number of examples required to initialize $classifier()$,
    $given\_budget$ - max. percent of labeled example in a chunk,
    $treshold$
1: $i \leftarrow 0$
2: for $j = 0$ to $m$ do
3:    ask for the label of the $j$th example $x_j$
4:    $classifier \leftarrow incremental\_training\_procedure(x_j, label(x_j))$
5: end for
6: repeat
7:    $budget \leftarrow floor(n * given\_budget)$
8:    collect new data chunk $DS_i = \{x_i^1, x_i^2, ..., x_i^n\}$
9:    set random order of collected examples in $DS_i$
10:   for $j = 0$ to $n$ do
11:      if $RFSD(x) < treshold$ then
12:         if $budget > 0$ then
13:            ask for the label of the $j$th example $x_j$
14:            $classifier() \leftarrow incremental\_training\_procedure(x_j, label(x_j))$
15:            $budget \leftarrow budget - 1$
16:         end if
17:      end if
18:   end for
19:   $i \leftarrow i + 1$
20: until end of the input data stream
```

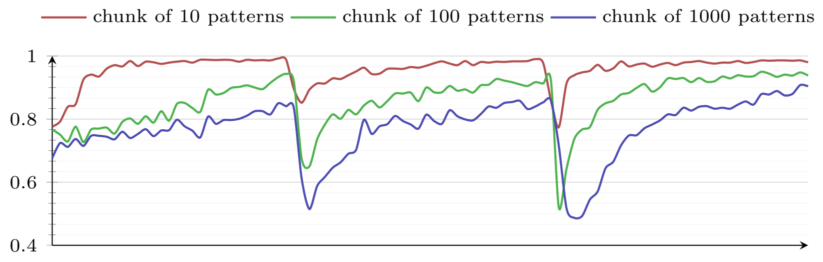


Figure 2: Influence of chunk size on a learning curve.

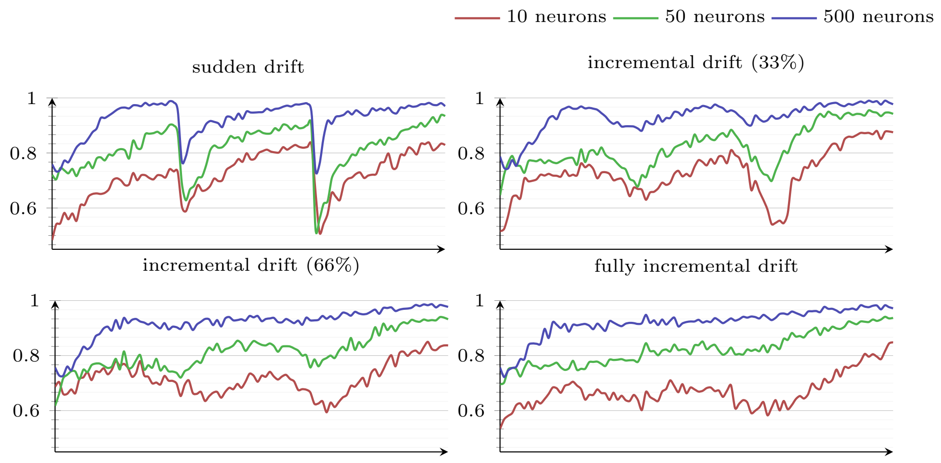


Figure 3: Reaction of different artificial neural network structures to different types of drift.

As we can see, in case of a sudden drift, a size of MLP hidden layer influences a learning curve in a similar way as a size of data chunk, but here the fewer number of artificial neurons decreases the learning abilities. On the other hand, more smooth drifts cause a lower drop of accuracy for, which means that in a fully incremental drift, for a structure of 50 and 500 neurons, we no longer observe a negative reaction to the occurrence of a drift. Natural mechanism of neural network forgetting allows (for a sufficiently large structure) to adapt smoothly to the changing concept.

In both previous examples, we observe that the change in learning parameters (chunk size) or network size (the number of neurons in the hidden layer) slows the rate of learning. In each case, the learning curves tend to go up in different dynamics. Figure 4 presents that thanks to a sufficiently large data stream, various network structures achieve maximum discriminatory power and then encounter an sudden drift.

As expected, the return to a full accuracy (restoration time) of classification

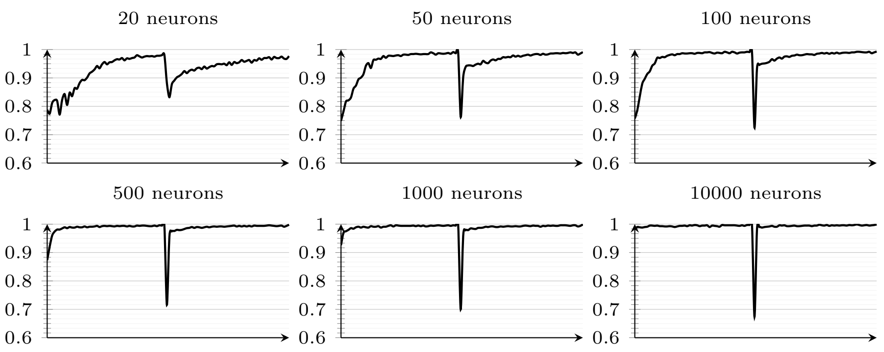


Figure 4: Reaction of different artificial neural network structures to a sudden drift.

takes more time for smaller structures. Interesting, however, is the difference in the accuracy achieved in the first moments after the concept drift. Enlarging the structure leads to a greater decrease from the maximum accuracy when the drift occurs. It can therefore be concluded that the more complex structures, despite the greater ability to rebuild knowledge, are also more severely affected by the effects of sudden drifts. That is why for a practical task, it is necessary to find such a structure of MLP which will be a kind of trade-off between the restoration time and the classification accuracy drop after a sudden concept drift appearance.

4. Experiments

4.1. Goals

The main goal of conducted experimental evaluation was to verify the impact of *given budget* and *treshold* values on overall classification accuracy, measured for each processed chunk, with the aim to calibrate parameters in a way that (a) reduces usage of labels (b) without classification accuracy deterioration.

4.2. Setup

4.2.1. Software environment

Most of the research devoted to *data streams* is currently conducted using the MOA environment [41], implemented in the *Java* programming language. It includes a collection of base classifiers, necessary experimental methods, measurement tools and, above all, data stream generators. Knowing a growing tendency to employ *scikit-learn* library [42], it was decided to use it in implementation of the method described in following paper. It is not widely used with data streams, however, the potentiality of such processing was confirmed by a short article with a *proof-of-concept* available in the on-line *scikit-learn* documentation[2].

It was necessary to create a new experimental flow for processing streams, enhanced with a class used to control a learning process. Current state of *scikit-learn* library allows to process data streams using classifiers having implemented the *partial fit* method, nevertheless as we mentioned above, we would like to employ inbuilt forgetting mechanism used by neural networks, therefore we used the implementation of MLP — Multi-layer Perceptron classifier, optimizing the log-loss function using LBFGS (*Limited memory Broyden-Fletcher-Goldfarb-Shanno*) algorithm [43].

The implementation of the active approach described in this paper as well as the workflow of learning from data streams using the *scikit-learn* library are part of the *stream-learn* module being developed by our research team[3]. Full code of experiments and presented examples, together with extended research results, can be found in the article repository[4].

4.3. Benchmark data streams

The pool of analyzed data consist of 12 streams:

- Three real streams:
  - *covtypeNorm* dataset [44] includes the observations which may be used to classify the cover type of a forest on the basis of cartographic variables. It contains 54 attributes and 7 class labels. The appearance of the concept drift is a result of the changes in geographical condition.
  - *clecNormNew* (electricity) dataset [45] includes the data which may be used to predict the rise or fall of the electricity price in New South Wales, Australia. It contains 6 attributes and 2 classes. Concept drift is caused by the changes in consumption habits, events, and seasons.
  - *poker-lsn* (poker hand) dataset [44] includes the data which may be used to poker hands. It contains 10 attributes and 10 classes. Concept drift is caused by card changing at hand.
- Nine computer generated streams:
  - Two streams with concept drift (*RBFBlips, RBFGradualRecurring*) and one without it (*RBFNoDrift*) generated with *Radial Basis Function*,
  - A stream with a sudden drift (*LED*) and without it (*LEDNoDrift*) generated with LED generator,

[2]`http://scikit-learn.org/stable/modules/scaling_strategies.html`
[3]`https://github.com/w4k2/stream-learn`
[4]`https://github.com/w4k2/active_learning`

– Two streams with sudden drifts with different dynamics (*SEASudden, SEASuddenFaster*) generated with *Streaming Ensemble Algorithm*[46],
– Two streams with gradual drifts with different dynamics (*HyperplaneFaster, HyperplaneSlow*) made by *Hyperplane* generator.

All the synthetic streams were generated by the MOA software.

4.3.1. Error evaluation

Every classifier uses a recent portion of data to train, but its evaluation (i.e., error estimation) is done on the following (unseen) data chunk. This type of performance evaluation is known as *test and train* or *block evaluation method* [41].

All the experiments were conducted with MLP classifier with 100 neurons in hidden layer, using chunks with size of 500 samples, evaluating the learning procedure after each percent of processed patterns.

4.4. Results

Experiments were conducted with 12 streams with three strategies:

- Measuring accuracy of incrementally training a model with all samples available in data stream.
- Measuring accuracy of incrementally trained model with randomized chunk subset, according to a given budget with five values in range 10—90%.
- Measuring accuracy of incrementally trained model with active learning approach described in Section 2 with five values in range 10—90% for both given budget and threshold.

Because conducted experiments produced many learning curves, only the part of them is included in the paper, but the detailed experimental results may be found in the article repository[5].

Figure 5 shows, with a blue line, a learning curve for three different budget values (30%, 50%, and 70%), while the black line is a curve for the classifier trained on the basis of all labeled examples (budget = 100%).

Decreasing a given budget leads, similarly to increasing the chunk or reducing a structure, to reduce the dynamics of learning. Therefore, it can not be a sufficient practice to reduce the cost of labeling. To maintain the accuracy of learning on a smaller training set, it is necessary to select the appropriately reduced set of samples, which we try to achieve by the proposed active learning method.

For the same stream, Figure 6 shows, with a continuous red line, a learning curve for three different threshold values (.1, .3, and .9) with a fixed budget of

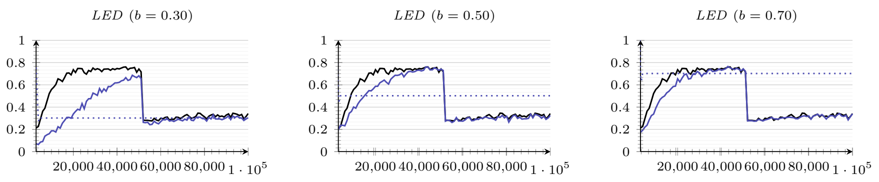


Figure 5: Budget and learning curve

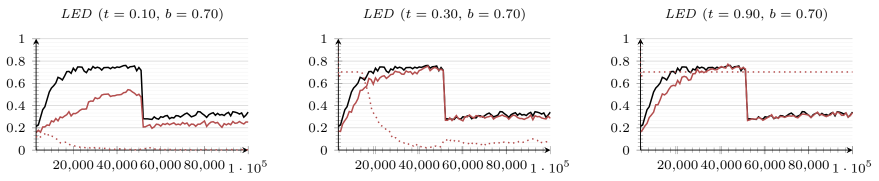


Figure 6: Treshold and learning curve

70%. The black line is a curve for learning with all samples labeled, while the red dotted line is the percentage of samples used to train the model.

As we can see, too restrictive value of the threshold ($t = .1$) reduces the dynamics of the learning curve similarly to a low budget. The case of overly lax threshold ($t = .9$) allows to obtain the correct learning curve, but it does not reduce the samples necessary for the labeling, so it is no different from using only the budget parameter. However, proper calibration of the threshold ($t = .3$) allows to obtain the optimal learning curve, gradually reducing the need for sample labeling while obtaining knowledge by the classifier.

Figure 7 shows the learning curves for selected, best parameters of a given budget and threshold for all tested data streams (continuous red lines). The black line is a curve for learning with all samples labeled. The red dotted line is the percentage of samples which were labeled during the model training.

Table 1 presents a summary of the results obtained for the best combinations. It contains, for each data set, selected parameters, average accuracy for the model learned on the full and reduced stream, the difference between the accuracies and the percentage use of the stream.

4.5. Analysis of the results

Most of the observations discussed around the Figures 5 and 6 are confirmed by curves available in the Figure 7. We need to reject the *elecNormNew* dataset, where even training with the fully labeled dataset led to a model with results similar to a random classifier. It is worth mentioning that Zliobaite [47] observes the problem of testing data stream classifiers on autocorrelated data and

[5]`https://github.com/w4k2/active_learning`

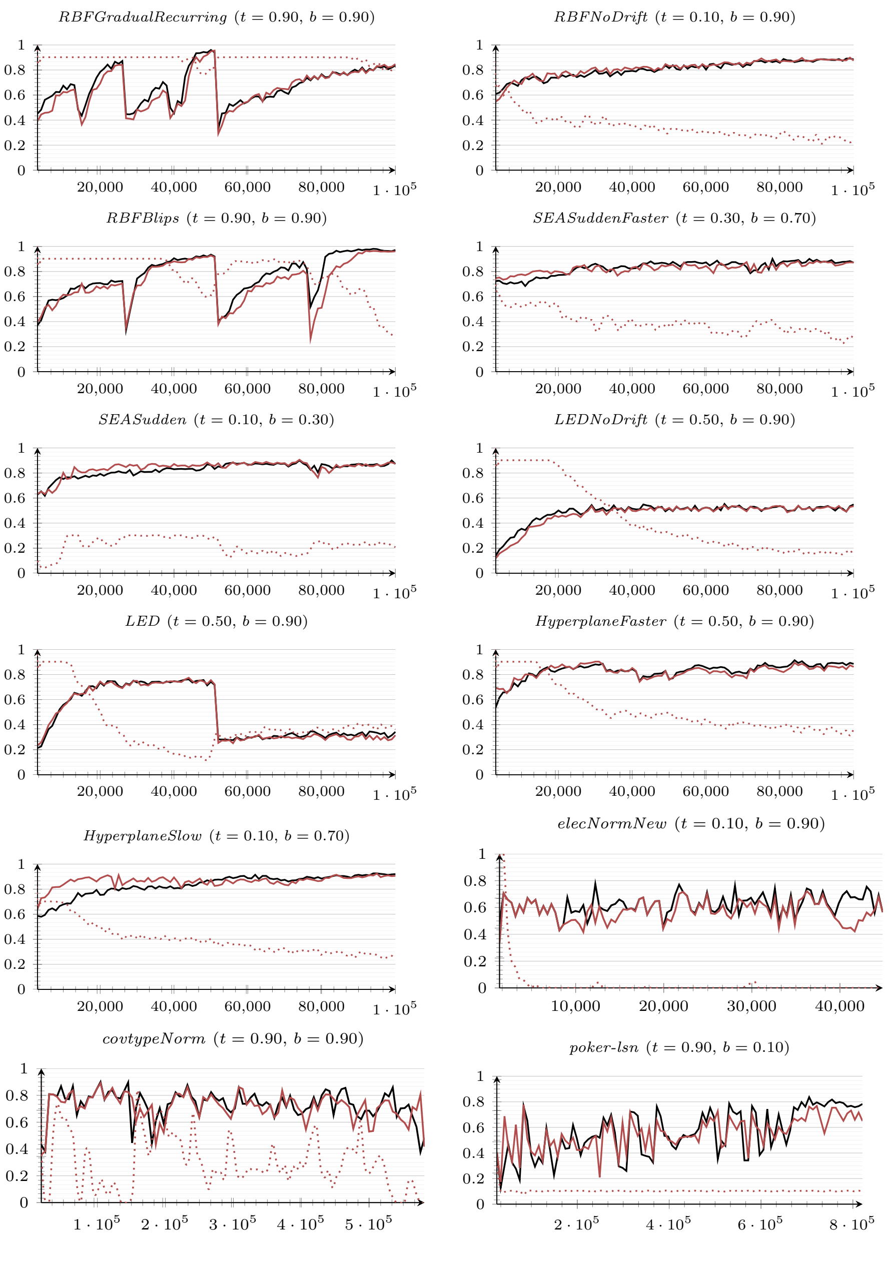


Figure 7: Learning curves and label usages for selected, best parameters of a given budget and threshold for all tested data streams

that getting high accuracy, especially on the *elecNormNew* dataset does not necessarily mean that the adaptation mechanism work well.

Table 1: A summary of the results obtained for the best combinations

Data stream			Accuracies			
	t	*b*	full	active	difference	usage
RBFGradualRecurring	.9	9	0.677	0.653	−0.025	88.8%
RBFBlips	.9	9	0.773	0.725	−0.048	79.2%
HyperplaneFaster	.5	9	0.834	0.826	−0.008	52.0%
LEDNoDrift	.5	9	0.491	0.477	−0.013	42.8%
HyperplaneSlow	.1	7	0.837	0.868	0.030	39.9%
LED	.5	9	0.490	0.482	−0.008	39.5%
SEASuddenFaster	.3	7	0.833	0.828	−0.005	39.2%
RBFNoDrift	.1	9	0.807	0.817	0.010	34.5%
covtypeNorm	.9	9	0.734	0.719	−0.015	29.2%
SEASudden	.1	3	0.829	0.842	0.013	22.4%
poker-lsn	.9	1	0.563	0.558	−0.005	10.3%
elecNormNew	.1	9	0.623	0.577	−0.046	2.3%

In every dataset, after this exclusion, a tendency to consequently reducing the chunk usage is observed. In the case streams with many concept drifts (*RBFGradualRecurring* and *RBFBlips*) the reduction of label use is relatively small, but it is caused by a slow accumulation of knowledge of the selected classifier, where 100 neurons in the hidden layer were not sufficient to achieve the maximum accuracy of classification before the next drift. We may also observe the intensive growth of the label demand, when a concept drift appears.

For data streams generated by *Streaming Ensemble Algorithm*, the reduction of the accuracy is not observed. In the case of two sets of data (*SEASudden* and *HyperplaneSlow*), the appropriate combination of a given budget and threshold allows not only to preserve the dynamics of learning, but also to accelerate it.

Even in the case of data streams where the selected neural network structure was insufficient to achieve full discriminative power before occurrence of the next drift, it was possible to reduce the number of samples necessary for labeling by 10–20%. In the case of the remaining sets, a reduction of 50–90% was achieved, without a negative impact on the average quality of the classification.

4.6. Lessons learned

Let us summarize the research findings and observations that could be drawn from the experimental analysis:

- Active learning approach may lead to a solution where a trained model keeps the classification accuracy of full-stream learning, with a slight reduction of used labels.
- There is no rule of correct setting the parameters related to a given budget and threshold, so it should be calibrated for a particular task.

- Active learning approach is able to detect a concept drift and react on it without negative impact on classification accuracy.
- Chunk usage decreases slowly in time, according to stabilization of a model on current concept.
- Acquiring the necessary knowledge by a model reduces the need for new samples.
- A *knowledge saturation* (degree of obtaining the maximum discriminative power) of a model can be successfully measured by RFSD.
- Chunk reduction has a positive effect by increasing the learning frequency but extends the learning time itself.
- The size of a neural network which is responsible for the concept memorization should be set carefully, for a given decision task, because on the one hand the smaller structure of the network the faster is its reaction to the concept drift, what is especially important in the case of sudden drift. Such fast model adaptation protect against the huge classification accuracy drop. On the other hand appropriate increasing of neural network size allows the immunization of the model to accuracy drop when the slow, incremental concept drift goes ahead.
- Limitation of training set by just a given budget with random subset of samples causes a proportional reduction in the learning dynamics.

5. Conclusions

The novel active learning strategy for neural network classifiers has been proposed, where to implement the forgetting mechanism, we employed the *catastrophic forgetting* phenomenon. The results of the experimental evaluation of the proposed algorithm carried out on the basis of diverse benchmark datasets and a detailed comparison with the semi-supervised and fully supervised strategies prove the usefulness of the proposed methods and show that it can better allocate the labeling budget.

In the near future we are going to:

- Develop the methods which will allow to control the speed of forgetting, especially adaptive forgetting is the desirable characteristic, i.e., possibility of changing the forgetting speed on the basis of concept drift appearance frequency and/or its severity.
- Evaluating the proposed algorithm behavior for more type of concept drift and maybe employ concept drift detector to establish the chunk size, threshold and budget dynamically.
- Applying the proposed approach to the classifier ensemble.

Acknowledgement

This work is supported by the Polish National Science Center under the Grant no. UMO-2015/19/B/ST6/01597 as well the statutory funds of the Department of Systems and Computer Networks, Faculty of Electronics, Wrocław University of Science and Technology.

References

[1] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, Wiley, New York, 2. edition, 2001.

[2] G. Widmer, M. Kubat, Learning in the presence of concept drift and hidden contexts, Mach. Learn. 23 (1996) 69–101.

[3] A. A. Beyene, T. Welemariam, M. Persson, N. Lavesson, Improved concept drift handling in surgery prediction and other applications, Knowledge and Information Systems 44 (2015) 177–196.

[4] T. Lane, C. E. Brodley, Approaches to online learning and concept drift for user identification in computer security, in: R. Agrawal, P. E. Stolorz, G. Piatetsky-Shapiro (Eds.), Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD-98), New York City, New York, USA, August 27-31, 1998, AAAI Press, 1998, pp. 259–263.

[5] J. R. Méndez, F. Fdez-Riverola, E. L. Iglesias, F. Díaz, J. M. Corchado, Tracking Concept Drift at Feature Selection Stage in SpamHunting: An Anti-spam Instance-Based Reasoning System, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 504–518.

[6] M. Black, R. Hickey, Classification of customer call data in the presence of concept drift and noise, Soft-Ware 2002: Computing in an Imperfect World (2002) 221–254.

[7] A. D. Pozzolo, G. Boracchi, O. Caelen, C. Alippi, G. Bontempi, Credit card fraud detection and concept-drift adaptation with delayed supervised information., in: IJCNN, IEEE, 2015, pp. 1–8.

[8] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. Bouchachia, A survey on concept drift adaptation, ACM Computing Surveys in press (2013).

[9] L. I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley-Interscience, 2004.

[10] P. Sobolewski, M. Wozniak, Concept drift detection and model selection with simulated recurrence and ensembles of statistical detectors, Journal of Universal Computer Science 19 (2013) 462–483.

[11] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. Bouchachia, A survey on concept drift adaptation, ACM Comput. Surv. 46 (2014) 44:1–44:37.

[12] P. Domingos, G. Hulten, A general framework for mining massive data streams., Journal of Computational and Graphical Statistics 12 (2003) 945–949.

[13] N. Littlestone, Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm, Machine Learning 2 (1988) 285–318.

[14] P. Domingos, G. Hulten, Mining high-speed data streams, in: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '00, ACM, New York, NY, USA, 2000, pp. 71–80.

[15] G. A. Carpenter, S. Grossberg, D. B. Rosen, Fuzzy art: Fast stable learning and categorization of analog patterns by an adaptive resonance system, Neural Netw. 4 (1991) 759–771.

[16] J. C. Schlimmer, R. H. Granger, Jr., Incremental learning from noisy data, Mach. Learn. 1 (1986) 317–354.

[17] J. Kolter, M. Maloof, Dynamic weighted majority: a new ensemble method for tracking concept drift, in: Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, pp. 123 – 130.

[18] A. Bouchachia, C. Vanaret, GT2FC: an online growing interval type-2 self-learning fuzzy classifier, IEEE Trans. Fuzzy Systems 22 (2014) 999–1018.

[19] B. Krawczyk, M. Wozniak, Incremental learning and forgetting in one-class classifiers for data streams, in: R. Burduk, K. Jackowski, M. Kurzynski, M. Wozniak, A. Zolnierek (Eds.), Proceedings of the 8th International Conference on Computer Recognition Systems CORES 2013, volume 226 of *Advances in Intelligent Systems and Computing*, Springer International Publishing, 2013, pp. 319–328.

[20] A. Bifet, G. Holmes, B. Pfahringer, R. Kirkby, R. Gavaldà, New ensemble methods for evolving data streams, in: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, KDD '09, ACM, New York, NY, USA, 2009, pp. 139–148.

[21] I. Koychev, Gradual forgetting for adaptation to concept drift, in: ECAI 2000 Workshop on Current Issues in Spatio-Temporal Reasoning, Berlin, Germany, pp. 101–106.

[22] R. Klinkenberg, Learning drifting concepts: Example selection vs. example weighting, Intell. Data Anal. 8 (2004) 281–300.

[23] M. Wozniak, A. Kasprzak, P. Cal, Application of combined classifiers to data stream classification, in: Proceedings of the 10th International Conference on Flexible Query Answering Systems FQAS 2013, LNCS, Springer-Verlag, Berlin, Heidelberg, 2013, p. in press.

[24] J. S. Vitter, Random sampling with a reservoir, ACM Trans. Math. Softw. 11 (1985) 37–57.

[25] B. Kurlej, M. Wozniak, Impact of window size in active learning of evolving data streams, in: Proceedings of the 45th International Conference on Modelling and Simulation of Systems MOSIS 2011, pp. 56–62.

[26] A. Bifet, R. Gavaldà, Learning from time-changing data with adaptive windowing, in: Proceedings of the Seventh SIAM International Conference on Data Mining, April 26-28, 2007, Minneapolis, Minnesota, USA, pp. 443–448.

[27] M. M. Lazarescu, S. Venkatesh, H. H. Bui, Using multiple windows to track concept drift, Intell. Data Anal. 8 (2004) 29–59.

[28] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, R. Hadsell, Overcoming catastrophic forgetting in neural networks, Proceedings of the National Academy of Sciences 114 (2017) 3521–3526.

[29] D. Kumaran, D. Hassabis, J. L. McClelland, What Learning Systems do Intelligent Agents Need? Complementary Learning Systems Theory Updated, Trends in Cognitive Sciences 20 (2016) 512–534.

[30] R. Greiner, A. J. Grove, D. Roth, Learning cost-sensitive active classifiers, Artif. Intell. 139 (2002) 137–174.

[31] I. Žliobaitė, A. Bifet, B. Pfahringer, G. Holmes, Active learning with drifting streaming data, IEEE Trans. Neural Netw. Learning Syst. 25 (2014) 27–39.

[32] B. Kurlej, M. Wozniak, Active learning approach to concept drift problem, Logic Journal of the IGPL 20 (2012) 550–559.

[33] B. Kurlej, M. Woniak, Learning curve in concept drift while using active learning paradigm, in: A. Bouchachia (Ed.), Adaptive and Intelligent Systems, volume 6943 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2011, pp. 98–106.

[34] H.-L. Nguyen, W.-K. Ng, Y.-K. Woon, Concurrent Semi-supervised Learning with Active Learning of Data Streams, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 113–136.

[35] S. Mohamad, A. Bouchachia, M. Sayed-Mouchaweh, A bi-criteria active learning algorithm for dynamic data streams, IEEE Transactions on Neural Networks and Learning Systems 29 (2018) 74–86.

[36] Ł. Korycki, B. Krawczyk, Combining Active Learning and Self-Labeling for Data Stream Mining, Springer International Publishing, Cham, pp. 481–490.

[37] D. W. Aha, D. Kibler, M. K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.

[38] C. Chow, On optimum error and reject trade-off, IEEE Transactions on Information Theory 16 (1970) 41–46.

[39] G. Fumera, F. Roli, G. Giacinto, Multiple Reject Thresholds for Improving Classification Reliability, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 863–871.

[40] B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, M. Wozniak, Ensemble learning for data stream analysis: A survey, Information Fusion 37 (2017) 132–156.

[41] A. Bifet, G. Holmes, R. Kirkby, B. Pfahringer, Moa: Massive online analysis, J. Mach. Learn. Res. 11 (2010) 1601–1604.

[42] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[43] A. Mokhtari, A. Ribeiro, Global convergence of online limited memory bfgs, Journal of Machine Learning Research 16 (2015) 3151–3181.

[44] A. Frank, A. Asuncion, UCI machine learning repository, http://archive.ics.uci.edu/ml, 2010.

[45] M. Harries, SPLICE-2 Comparative Evaluation: Electricity Pricing, Technical Report, The University of South Wales, 1999.

[46] W. N. Street, Y. Kim, A streaming ensemble algorithm (SEA) for large-scale classification, in: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '01, ACM, New York, NY, USA, 2001, pp. 377–382.

[47] I. Zliobaite, How good is the electricity benchmark for evaluating concept drift adaptation, CoRR abs/1301.3524 (2013).