СОФИЙСКИ УНИВЕРСИТЕТ „СВ. КЛИМЕНТ ОХРИДСКИ”
UNIVERSITY OF SOFIA “ST. KLIMENT OFRIDSKI”,
ФАКУЛТЕТ ПО МАТЕМАТИКА И ИНФОРМАТИКА
FACULTY OF MATHEMATICS AND INFORMATICS,
КАТЕДРА СОФТУЕРНИ ТЕХНОЛОГИИ
DEPARTMENT OF SOFTWARE ENGINEERING

Approaches for Learning Classifiers of Drifting Concepts

Подходи за учене на класификатори на променящи се понятия

IVAN KOYCHEV
ИВАН КОЙЧЕВ

HABILITATION THESIS
ХАБИЛИТАЦИОНЕН ТРУД

NOVEMBER, 2014.

Table of Contents

TABLE OF FIGURES

TABLE OF TABLES

SUMMARY

Recently, it is become easier than ever to collect data. Usually this data come as a steam, i.e. time ordered. One of the most usual task is for classify the items described by this data in two or more classes. For example, numerous activities on the Web have to distinguish between related and non-related items. In this case Machine Learning algorithms can be employed to construct classifiers from examples. Machine Learning applications often face the problem that real-life concepts tend to change over time and some of the learned classifiers from old observations become out-of-date. This problem is known as concept drift. This work gives a brief overview of the approaches that aim to deal with drifting concepts. Further it describes, in more detail, three mechanisms for dealing with drifting concepts, which are able to adapt dynamically to changes by forgetting irrelevant data and models. Moreover a fourth mechanism that able not only to forget, but also to remember relevant forgotten data is also proposed. The presented mechanisms are general in nature and can be an add-on to any concept learning algorithm. Results from experiments that give evidences for the effectiveness of the presented approaches are reported and discussed.

SUMMARY IN BULGARIAN

РЕЗЮМЕ

Първоначалният ми интереса ми към тази тематика беше мотивиран от работата ми по проекта *LaboUr (Learning about User)*[1] в Института за Приложни Информационни технологии към Германския национален изследователски център за информационни технологии (GMD), който целеше да се изследва приложението на методите за машинно обучение за автоматично извличане на модели на потребителя. (вж. публикация [44]). За да се моделира феноменът на променливия потребителски интерес, разработих подходи, които учат само-адаптиращи се модели към *промените в интересите на потребителя.* Задачата за актуализиране на модели на променящи се потребителските интереси се свежда до по-абстрактната задачата за учене на променящи се понятия, от областта на обучението на машинни (*learning drift concept*). В тази област бяха разработени четири подхода:

- Първият механизъм предлага *постепенно забравяне* на обучаващи примери, чрез подходяща тегловна схема с тегла според появата им във времето. Чрез теглата се задава по-голяма тежест за последните примери, като системата постепенно "забравя" най-старите примери. Определен са изисквания към забравящата функция, така че да може най-лесно да се интегрира в алгоритмите за учене на понятия (класификатори). Предложена е линейна забравяща функция. Този подход е представен в [24, 25]. Подходът е използван в уеб-базирана препоръчваща система, като опита показва че много добре се интегрира с подходите за избор на определящи характеристики [43].
- *Алгоритъм, който адаптира научените понятия към променливи и повтарящи се понятия,* е представен в публикациите [26]. Алгоритъмът "учи" на две стъпки: на първата (предварителна) стъпка "учи" за текущия контекст. След това се извършва тестване на предходните наблюдения като се търсят епизоди, които са подобни на текущия контекст. Така на тази стъпка, се отделят епизодите, където понятията са същите или много близки на текущият и се "забравят" тези, в които са различни. Най-накрая алгоритъмът се обучава само

[1] Ръководител на проекта Prof Alfred Kobsa

въз основа на подбраните релевантни примери. Проведените експерименти с реални данни за планиране на срещи показват, че разработеният алгоритъм е способен да учи планиращи правила, които съществено подобрява точността на предсказване в сравнение с други подобни подходи.

- Разработен и подход, който предлага само-адаптивен оптимизационен алгоритъм за учене на променящи се понятия. подхода предполага използването на статистически механизъм за *откриване на промяна на понятията* и ако бъде открита такава, алгоритъма *оптимизира времевият прозорец (т.е."скоростта на забравянето"),* за да бъде научен осъвременен модел, възможно най-точното описва на текущото понятие. Направените експерименти с 4 различни множества от данни показва, че разработеният подход съществено подобрява точността на предсказване в сравнение с класическите подходи [22].

Разработените подходи за "забравяне" при учене на променящи се понятия са от общ характер и могат да се използват при произволни алгоритми за обучение, а също така и в различни приложни области.

В [24] се описва експерименти, които целят да изучат и сравнят механизма за постепенно забравяне и този с оптимизация на времевия прозорец (първият и третият от описаните по-горе). Експериментално се изследва и възможността двата подхода се използват едновременно, като взаимно се допълват. Експериментите са проведени с три алгоритъма за машинно учене (ID3, NBC, IBL) и са използвани четири различни по тип множества от данни. Резултатите от експериментите показват, че:

- Прилагане на постепенно забравяйки вътре с фиксиран размер времеви прозорец обикновено води до значително подобряване на точността класификация на плаващи понятия;
- когато се сравняват двата забравящи механизъма предимството е на страната на алгоритъм със самос-адаптиращ се времеви прозорец;
- най-накрая, когато се прилагат и двата механизма заедно обикновено не води до допълнително подобрение в точността на класификация.

ACKNOWLEDGMENTS

The author would like to thank a number of people and organizations for their support and help throughout the pursuit of this research.

Different parts of this research were conducted during my employment in the following institutions:

- Faculty of Mathematics and Informatics at the University of Sofia,
- Institute of Mathematics and Informatics at the Bulgarian Academy of Science, Sofia.
- School of Computing at the Robert Gordon University, Aberdeen, UK, and
- Institute of Applied Informatics at GMD, St. Augustin, Germany, where I got the inspiration and done my early research on this topic.

Thank you to my colleagues from those institutions for help and support they have offered me. I am lucky being a member of such friendly and inspiring teams.

And last but not least, thank you to my family for their love and support.

STATEMENT OF ORIGINALITY AND DECLARATION OF PREVIOUS PUBLICATION

All original work in this thesis was undertaken by the author, some of them in collaboration with other researchers, which contribution is specified below. In addition, all novel contributions presented here have previously appeared in peer-reviewed publications. Most of the publications are well acknowledged by other researches via citations.

In particular, the author would like to address the nature of the collaboration between himself and:

- Ingo Schwab, how was the first one with him I discussed the idea of gradual forgetting mechanism. He integrated this mechanism in a web-based information system and conducted the experiment with logs from system's usage.
- Robert Lothian, who contributed by improving the presentation of the time window optimization method and results from the experiments.

What follows is a statement outlining the original contributions of this thesis, with references to previous publications:

The **Gradually forgetting** method is originally described in:

- Koychev, I. Gradual Forgetting for Adaptation to Concept Drift. Proceedings of ECAI 2000 Workshop on Current Issues in Spatio-Temporal Reasoning, Berlin, (2000) 101-107.
- Koychev, I. and Schwab, I. Adaptation to Drifting User's Interests. In proc. of ECML2000 Workshop: Machine Learning in New Information Age, Barcelona, Spain (2000), p. 39-46.

The **two levels learning algorithm that "forgets or remember" according to the current context** is first described in:

- Koychev, I. (2001). Learning about User in the Presence of Hidden Context. Proceedings of UM-2001 Workshop: Machine Learning for User Modeling, pp. 49-58.

An improved presentation and more results from experiments is published:

- Koychev, I. Tracking Changing User Interests through Prior-Learning of Context. In: P. de Bra, P. Brusilovsky and R. Conejo (eds.): Adaptive Hypermedia and Adaptive Web Based Systems. *Lecture Notes in Computer Science*, Vol. **2347**, Springer-Verlag (2002), 223-232.

The **Time Window Optimisation** method is first published in:

- Koychev, I. and Lothian, R. Tracking Drifting Concepts by Time Window Optimisation. In: *Research and Development in Intelligent Systems* **XXII** Proc. of AI-2005, the 25-th SGAI Int. Conference on Innovative Techniques and Applications of Artificial Intelligence. Bramer, Max; Coenen, Frans; Allen, Tony (Eds.), Springer-Verlag, London (2006), p.46-59.

Experiments that **aim to compare and study the possibilities of combination of different forgetting mechanisms i.e. creation of new hybrid methods** hoping that they will better in a performance measure for at least a particular class of tasks, is published in:

- Koychev, I. Experiments with Two Approaches for Tracking Drifting Concepts. *Serdica Journal of Computing* **1** (2007), 27-44.

An invited book chapter that describes the first and third of the developed mechanism and report some experimental results is published in:

- Koychev, I. Approaches for Learning Classifiers of Drifting Concepts. NATO Science for Peace and Security Series - D: Information and Communication Security Volume 15, 2008 Security Informatics and Terrorism: Patrolling the Web. Eds.: Cecilia S. Gal, Paul B. Kantor, Bracha Shapira, IOS Press. http://ebooks.iospress.nl/volume/security-informatics-and-terrorism-patrolling-the-web

1 Introduction

In last two decades almost everything become digital. There many sensors, cameras and active user connected to the internet producing enormous amount of data. This data are potentially valuable resource, but need to be processed and analyzed to be discovered interesting trends and models. The research area that deals with this problem is known as Data Mining, which has its roots in areas of Machine learning and statistics. In this paper, we will focus on of the class of problems, which requires automated classification of items (e.g. documents, user's profiles etc.) [1]. For this purpose a classifier (model) needs to be learned from the observations (data). Usually these systems emuser's interests profiles. These systems face the problem that users' interests are inclined to change over time, or when the context is shifting [37, 33, 23 26, 11]. The ability to deal with concept drift is becoming more important in the Web era, because many of the concepts/classes/groups/categories in which the new items need to be classified are dynamic.

A number of context-aware "forgetting" mechanisms have been developed to cope with this problem. They can be divided into two major types depending on whether they are able to *forget only* or whether they are also able to keep old data/knowledge and *recall* them if they seem to be useful again.

The first type of mechanism has the advantage of being simpler and faster and in many cases provides good enough solutions. A typical example is the fixed *time window* approach. It performs well with concepts that do not change often and change gradually rather than abruptly, but it performs very poorly on frequently changing and recurring concepts. An improvement of this approach is by adding a mechanism that abele to detect the changes in underling concepts and dynamically adapt the size of the time window according current concept changes.

The second type of mechanisms has a clear advantage in the case of *recurring concepts* (e.g., caused by seasonal changes or other transformations in context), where the relevant old data or acquired knowledge can be very helpful. However, it requires more storage space for the old information, extra time for retrieving and adapting/weighting of the old relevant episodes of data or knowledge.

This paper describes two forgetting mechanisms which belong to the forget only type of forgetting mechanisms: the first approach is the time window with weights, which decrease over time [23], thus it simulates a

forgetting that is gradual, which seems to be more natural. The second approach dynamically adapts the size of the time window according to the current concept behaviour [22].

The paper also presents a third approach that belongs to the second type of mechanisms – i.e. the one that can "remember" old examples if it seems they describe the same concepts.

This paper reports the results from experiments with these three forgetting mechanisms. The objectives of the experiments are to test and compare the approaches, as well as to explore the possibility of combining some of them for further improvement of the model's performance.

The next section gives a short overview of related work. The forgetting mechanisms are described in section 3. The experimental design is introduced in section 4. The results from the experiments are reported and discussed in section 5.

2 Related Work

Comprehensive overviews of the related works, for each of methods described in the next section, are done at the time each of it was published. In this section we will acknowledge mostly the original works that provides the basic ideas and methods which comes from area of machine learning and user modelling, which was, in my opinion, probably the key application area that inspires the development of methods for learning drifting concepts. Even nowadays many of the research are inspired and done in this area (e.g. [50], [41], [39] and etc.). There is a significant grow of research in this are lately, mainly tailored to particular type of problem or application area, therefore for the state of the art in the area we will refer some recent surveys in the area.

2.1 Learning drifting concepts

Different approaches have been developed to track changing (also known as shifting, drifting or evolving) concepts. Typically it is assumed that if the concept changes, then the old examples become irrelevant to the current period. The concept descriptions are learned from a collection of the most recent examples called a *time window*. For example, Mitchell et al. [37] developed a software assistant for scheduling meetings, which employs machine learning to acquire assumptions about individual habits of arranging meetings and uses a time window to adapt faster to the changing preferences of the user. Widmer and Kubat [44] developed the first approach that uses adaptive window size. The algorithm monitors the learning process and if the performance drops under a predefined threshold it uses heuristics to adapt the time window size dynamically. Maloof and Michalski [33] have developed a method for selecting training examples for a partial memory learning system. The method uses a time-based function to provide each instance with an age. Examples that are older than a certain age are removed from the partial memory. Delany et al. [9] employ a case base editing approach that removes noise cases (i.e., the cases that contribute to the classification incorrectly are removed) to deal with concept drift in a spam filtering system. The approach is very promising, but seems applicable to lazy learning algorithms only. To manage the problems with gradual concept drift and noisy data, the approach suggests the use of three windows: a small (with fixed size), a medium and a large (dynamically adapted by simple heuristics) [32].

The pure time window approaches totally forget the examples that are outside the given window, or older than a certain age. The examples which remain in the partial memory are equally important for the learning algorithms. This is an abrupt forgetting of old examples, which probably does not reflect their rather gradual aging. To deal with this problem it was suggested to weight training examples in the time window according to their appearance over time [6]. These weights make recent examples more important than older ones, essentially causing the system to gradually forget old examples. This approach has been explored further in [25], [29] and [20].

Some systems use different approaches to avoid loss of useful knowledge learned from old examples. The CAP system [37] keeps old rules as long as they are competitive with the new ones. The FLORA system [44], also maintains a store of concept descriptions relevant to previous contexts. When the learner suspects a context change, it will examine the potential of previously stored descriptions to provide better classification. The COPL approach [26] employs a two-level schema to deal with changing and recurring concepts. On the first level the system learns a classifier from a small set of the most recent examples. The learned classifier is accurate enough to be able to distinguish the past episodes that are relevant to the current context. Then the algorithm constructs a new training set, "remembers" relevant examples and "forgets" irrelevant ones. As explained in the introduction, the approach explored in this paper, does not assume that old examples or models can be retrieved.

Widmer [49] assumes that the domain provides explicit clues to the current context (e.g. attributes with characteristic values). A two-level learning algorithm is presented that effectively adjusts to changing contexts by trying to detect (via meta-learning) contextual clues and uses this information to focus the learning process. Another two-level learning algorithm assumes that concepts are likely to be stable for some period of time [17]. This approach uses contextual clustering to detect stable concepts and to extract hidden context. The mechanisms studied in this paper do not assume that the domain provides some clues that can be discovered using a meta-learning level. They rather aim to get the best performance using one-level learning.

An adaptive boosting method based on dynamic sample-weighting is presented in Chu and Zaniolo [6]. This approach uses statistical hypothesis testing to detect concept changes. Gama et al., [14] also use a hypothesis testing procedure, similar to that used in control charts, to detect concept drift, calculated on all of the data so far. The mechanism gives a warning at

2 standard deviations (approximately 95%) and then takes action at 3 standard deviations (approximately 99.7%). If the action level is reached, then the start of the window is reset to the point at which the warning level was reached. However, for this mechanism, it can take quite a long time to react to changes and the examples that belong to the old concept are not always completely useless, especially when the concept drift is rather gradual.

Baron and Spiliopoulou [5] suggested a pattern monitoring mechanism that observes pattern evolution across the timeline aiming to detect interesting changes. It uses devise heuristics that detect specific types of change. The objective is the identification and categorization of changes according to an "interestingness" model, so that the mining expert can be notified accordingly.

To adapt the size of the window according to current changes in the concept, the approach presented in [20] uses a naïve optimization approach, which tries all possible window sizes and selects the one with the smallest error rate. This work provides interesting results from experiments with different forgetting mechanisms used with a support vector machines classifier, using a textual data corpus.

To detect concept changes the TWO (Time Window Optimisation) approach [22] uses a statistical test on a selected population from the time window, which excludes its beginning and end. If a concept drift is detected an efficient optimization algorithm is employed to detect the optimal window size. This approach is explained in more detail in the next section and studied in the experiments.

2.2 Tracking Drifting User Interests

Traditionally, users' selections over a period of time from a menu of items are regarded as relatively independent tries. However, often users' interests drift with time (Allan, 1996 [3]; Grabtree and Soltysiak, 1998 [16]). This normally means that the most recent observations represent the current user's interests better than older ones. From a more general perspective, the task of catering to drifting user interests corresponds to learning drifting (aka changing, evolving or shifting) concepts discussed in the above subsection.

The most frequently used approach to deal with this problem is to learn the description of users' interests from the newest observations only. The training examples are selected from a so-called time window, i.e. only the

last *l* examples are used for training (Mitchell et. al., 1994 [37]; Grabtree and Soltysiak, 1998 [16]). An improvement of this approach is the use of heuristics to adjust the size of the window according to the current predictive accuracy of the learning algorithm (Widmer and Kubat, 1996 [44]). In a similar way, the time-based forgetting mechanism in Maloof and Michalski (2000) [33] uses a time function for aging the examples. Instances that are older than a certain age are deleted from the partial memory. However, this approach totally "forgets" the observations that are outside the given window or older than a certain age. These observations may however still be valuable. To avoid loss of useful knowledge learned from old examples, some of these systems keep old rules as long as they are competitive to new ones (Mitchell et. al. 1994 [37]). Another approach is to use a dual user model (Chiu and Webb, 1998 [9]; Billsus and Pazzani, 1999 [6]). The approach pursued in Billsus and Pazzani, 1999 [6] uses a hybrid user model consisting of both a short-term and a long-term model of the user's interests. The method employs the short-term model first, which is based on the most recent observations. If a story cannot be classified with the short-term model, the long-term model is used. This hybrid user model is useful in domains where the long-term user's interests are quite broad and short-term interests change fast, as is the case for news stories.

2.3 Recent development

The boom of research in this field makes the task of creation a good survey very hard. In this section we will refer only some of the most significant reviews in the area.

The recent development of research on concept drift, being very widespread motivated by easy gathered steams of data. However, it was dispersed among various application areas. Therefore making a comprehensive summary of the current state of the art of research is a hard and demanding work. However it is important such surveys to be done regularly to align or unify the terminology among the researchers and to survey the state-of-the-art methodologies and techniques. A few reviews related to drift-aware learning are available. However, often they relate to specific topics of adaptive learning, thus they are fragmented or outdated.

The earlier survey on concept drift was published in 2004 by Tsymbal [45]. Even it is not quite full, it is still the most cited one.

Next overviews related to the topic of concept drift are:

- Kuncheva 2008 [30] - focused on ensemble techniques;

- Zliobaite 2009 [51] - focused mainly on non-incremental learning techniques
- Maloof 2010 [34] - inductive rule learning algorithms that can use computational resources unrestrictedly.

Thus reviews were limited in scope of particular learning algorithms.

The data streams research (e.g. Gaber et al. 2005 [12] and Gama 2010; [13]) covers learning drifting concept only to some extent, though the main focus remains on making learning algorithms incremental and optimizing the balance of computational resources and the predictive accuracy. The methods presented in this work are considering the problem in a more general level and in this sort overview we will not cover this area of research.

Some recent reviews on the concept drift problem that are limited to specific application domain are:

- Kadlec et al. 2011 [19] - surveys adaptation mechanisms that have been used for soft sensors.
- Moreno-Torres et al. 2012 [38] - focuses on describing various ways how data distribution can change over time and only briefly covers adaptation techniques, leaving out many works on concept drift.
- Alberg et al. 2012 [3] - a recent review focuses on dealing with concept drift in case of decision trees.

Currently the most comprehensive and counterparty survey is from Gama, J. et al. [15]. It provides an integrated view on handling concept drift, by surveying adaptive learning methods, presenting evaluation methodologies and discussing illustrative applications. It focuses on online supervised learning when the relation between the dataset attributes and the target concept changes over time. This survey presents a comprehensive taxonomy of methods for adaptive learning and discusses the experimental settings and evaluation methodologies of adaptive learning algorithms.

Change detection in streaming data is very important task for each self-adapting mechanist that tracks changing concepts. Most of them rely on a fast estimation of the probability that the data in two consecutive windows come from different distributions. This problem is recently considered in Kuncheva 2013 [31], two well-known criteria for detecting change in streaming multidimensional data: Kullback-Leibler distance and Hotelling's T-square test for equal means are studied and a new one is propose - semiparametric log-likelihood criterion for change detection. Conducted experiment confirmed that the suggested new criteria can perform better in come cases.

3 Approaches for Learning Drifting Concepts

Let us consider a sequence of examples. Each example is classified according to underlying criteria into one of a set of classes, which forms the training set. The task is to "learn" a classifier that can be used to classify the next examples in the sequence. However, the underlying criteria can subsequently change and the same example can be classified differently according to the time of its appearance, i.e., a concept drift takes place. As we discussed above, to deal with this problem machine learning systems often use a time window i.e., the classifier is not learned on all examples, but only on a subset of recent examples. The next section describes forgetting mechanisms that further develop this idea.

3.1 Gradual Forgetting

This section describes a gradual forgetting mechanism introduced earlier in 23 and 25. Just like the time window approach it assumes that when a concept tends to drift the newest observations better represent the current concept. Additionally, it aims to make the forgetting of old examples gradual. Let us define a gradual forgetting function $w = f(t)$, which provides weights for each instance according to its location in the course of time. The weights assign a higher importance value to the recent examples. Earlier, Widmer [49] suggests an "exemplar weighting" mechanism, which is used for the IBL algorithm in METAL(IB), however it is not exploited for NBC in METAL(B). The researcher in the area of boosting also saw the need for weighting examples. There are two ways in which boosting employs the weights. The first one is known as boosting by sampling - the examples are drawn with replacement from the training set with a probability proportional to their weights. However, this approach requires a pre-processing stage where a new set of training examples should be generated. The better the sampling approximates the weights, the larger the new training set becomes. The second method, boosting by weighting, is used with learning algorithms that accept weighted training examples directly. In this case the weight is constrained as follows:

$$w_i' \geq 0 \text{ and } \sum_{i=1}^{n} w'_i = 1 \qquad (1)$$

where, n is the size of the training set. Most of the basic learning algorithms are designed to treat all examples as equally important. For kNN it can be

easily implemented by multiplying the calculated distances with the weights of the examples. For other algorithms it is not so straightforward. When there are no weights (i.e., all weights are the same) the formula (1) will provide weights $\forall w_i' = \frac{1}{n}$. However, as the weights are multiplied, it seems to be better if we have $\forall w_i = 1$ in this boundary case. If we substitute $w_i = nw'_i$ then the constraints in equation (1) will be transformed as follows:

$$w_i \geq 0 \text{ and } \frac{\sum_{i=1}^{n} w_i}{n} = 1 \tag{2}$$

Weights that obey the constraints (2), can be easily used in almost any learning algorithm requiring minor changes in the code i.e., every time the algorithm counts an example it should be multiplied by its weight.

Various functions that model the process of forgetting and satisfy constraints (2) can be defined. For example, the following linear gradual forgetting function has been defined:

$$w_i = -\frac{2k}{n-1}(i-1) + 1 + k \tag{3}$$

where: i is a counter of observations starting from the most recent observation and it goes back in time (as the function is forgetting $w_i \geq w_{i+1}$); $k \in [0,1]$ is a parameter that controls the relative weight of the most recent and oldest observation in comparison to the average (see **Figure 1**). By varying *k,* the slope of the forgetting function can be adjusted to reach better predictive accuracy.

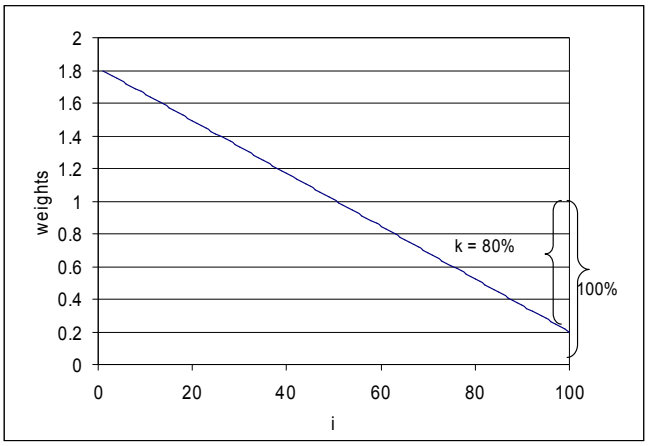

Figure 1. A linear gradual forgetting function. Where: n=100, k=80%.

The presented gradual forgetting mechanism is naturally integrated into a time window, by weighting the examples in it (i.e., $n = l$, where l is the size of the time window). The system will forget gradually inside the window. When $k = 0$ then all weights will be equal to 1, which means that we will have a "standard" time window. When k is approaching 1 then the weights of the examples in the end on the window approach 0 and there is not an abrupt forgetting after the window's end.

3.2 Tracking Changes through Prior-Leaning of Context

When the concept drifts and possibly recurs, we can use time window based forgetting mechanisms. However, the recent examples that represent the current context can be insufficient for learning accurate descriptions. Therefore, if the context recurs, then remembering the 'old' examples that are relevant to the current context should enlarge the size of the training set and thus improve the predictive accuracy. However, the context is frequently hidden and explicit indicators about its changes and recurrences cannot be discovered easily. Hence, in such cases the aim should be to learn more about the current context and then to search for old observations that were made in a similar context. An algorithm that makes use of this idea consists of the following three steps:

1. *Learning about current context.* A relatively small time window is used to learn a description of the current context (e.g. learning a description of the user interests based on the recent observations about the user).

2. *Remembering relevant past episodes.* The learned description in step 1. is tested against the rest of the training set. The episodes that show a predictive accuracy that is greater than a predefined threshold are selected (i.e. selecting the episodes that are relevant to the current context).
3. *Learning from context-related examples.* The new data set selected in step 2. is used for learning a new description of the current user interests, which is expected to be more accurate.

Let's call this algorithm COPL (COntext Prior Learning algorithm). The COPL algorithm requires a predefinition of the following settings:

- *The size of the time window* used in step 1. This time window should be long enough to allow a sufficiently accurate description of the current context to be learned, as well as short enough to be able to track fast changing user interests. Some enhancements like adaptive time window can be employed aiming at improving predictive accuracy.
- *The episode selection criterion* for step 2. This criterion should be able to distinguish the episodes that are relevant to the learned context in step 1. The criterion should be resistant to noise in the sequence of examples.
- *The threshold for the episode-selecting criterion* in step 2. After the episode selection criterion has been established, a suitable threshold should be defined, which should assure as much as possible that only the relevant old examples be selected.
- *The learning algorithms* used in steps 1. and 3. The same or different learning algorithms can be used in those steps.

Those settings should be defined empirically and based on preliminary investigation of the application domain. The implementation of the algorithm described in the next section gives an example of such definitions.

The next section presents the results from experiments that compare the designed algorithm where the main idea is to extend the set of examples by recovering relevant past examples as opposite to the CAP and FLORA approaches where the model was extended by past rules.

3.3 Time Window Optimization

This section presents an approach that learns an up-to-date classifier on drifting concepts by dynamic optimization of the time window size to maximize accuracy of prediction [22].

In the case of dynamical adaptation of window size there are two important questions that have to be addressed in the case of concept drift. The first one is how to detect a change in the concept? We are assuming that there is no background information about the process and we use the decrease in predictive accuracy of the learned classifier as an indicator of changes in the concept. Usually, the detection mechanism uses a predefined threshold tailored for the particular dataset. However the underlying concept can change with different speeds to a larger or smaller extent. To detect the changes the approach uses a statistical hypothesis test, which adapts the thresholds according to the recently observed concept deviation. The second important question is how to adapt if a change is detected? Other approaches use heuristics to decrease the size of the time window when changes in the concept are detected (e.g., [44] and [14]). This approach employs a fast 1-D optimization to get the optimal size of the time window if a concept drift is detected. The next two subsections describe the algorithm in more detail.

3.3.1 Detecting the Concept Drift

To detect concept changes the approach monitors the performance of the algorithm. For the presentation below we choose to observe the classification accuracy as a measure of the algorithm performance, but other measures, such as error rate or precision could have been used. The approach processes the sequence of examples on small episodes (a.k.a. batches). On each step, a new classifier is learned from the current time window then the average accuracy of this classifier is calculated on the next batch of examples [37]. Then the approach suggests using a statistical test to check whether the accuracy of classification has significantly decreased compared with its historical level. If the average prediction accuracy of the last batch is significantly less than the average accuracy for the population of batches defined on the current time window, then a concept drift can be assumed.

The significance test is done using a population from the time window that does not include the first one, or a few batches from the beginning of the time window, because the predictive accuracy could be low due to a previous concept drift. Anywhere from one to a few most recent batches are not included in the test window, because if the drift is gradual the accuracy

will drop slowly. A test that uses a test population from the core of the time window works well for both abrupt and gradual drift.

The confidence level for the significance test should be sensitive enough to discover concept drift as soon as possible, but not to mistake noise for changes in the concept. The experience from the experiments shows that the "standard" confidence level of 95% works very well in all experiments. This drift detection level is rather sensitive and it assists the algorithm to detect the drift earlier. If a false concept drift alarm is triggered, it will activate the window optimizing mechanism, but in practice, this only results in an insignificant decrease in the time window size.

This mechanism works as follows: *If concept drift is detected then the optimization of the size of the time window is performed (see the next section) otherwise, the time window size is increased to include the new examples.*

3.3.2 Optimising the Time Window Size

In general, if the concept is stable, the bigger the training set is (the time window), the more accurately the classifier can be learned. However, when the concept is changing, a big window will probably contain a lot of old examples, which will result in a decrease of classification accuracy. Hence, the window size should be decreased to exclude the out-of-date examples and in this way the algorithm can learn a more accurate classifier. However, if the size of the window becomes too small, this will lead to a decrease in accuracy. The shape of the curve that demonstrates the relationship between the size of the time window and the accuracy of the classification is shown in **Figure 2**.

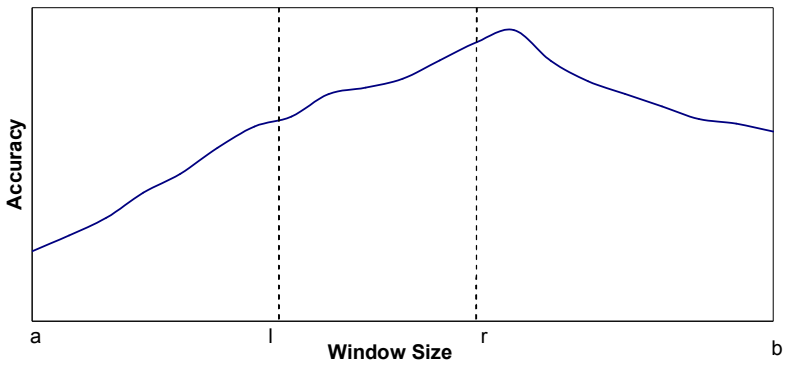

Figure 2. A sample shape of the correlation between the window size and accuracy of the classifier.

To adapt the size of the window according to current changes in the concept, the presented mechanism uses the golden section algorithm for one-dimensional optimization. The algorithm looks for an optimal solution in a closed and bounded interval $[a,b]$ - in our case the possible window sizes $X=[x_{\min}, x_c]$, where $x_{\min}$ is a predefined minimum size of the window and x_c is the current size of the time window. It assumes that the function $f(x)$ is *unimodal* on *X*, i.e., there is only one max x^* and it is strictly increasing on $(x_{\min}, x^*)$ and strictly decreasing on (x^*, x_c), which is the shape that can be seen on **Figure 2**. In our case the function $f(x)$ calculates the classification accuracy of the learned model using a time window with size *x*.

The basic idea of this algorithm is to minimize the number of function evaluations by trapping the optimum solution in a set of nested intervals. On each step the algorithm uses the golden section ratio $\tau=\phi\approx 0.618$ to split the interval into three subintervals, as shown in Figure 1, where $l=b-\tau(b-a)$ and $r=a+\tau(b-a)$. If $f(l)>f(r)$ then the new interval chosen for the next step is $[a,r]$ else $[l,b]$. The length of the interval for the next iteration is $\tau(b-a)$. Those iterations continue until the interval containing the maximum reaches a predefined minimum size, x^* is taken to lie at the centre of the final interval.

The golden section optimization algorithm is a very efficient way to trap the x^* that optimizes the function $f(x)$. After *n* iterations, the interval is reduced to 0.618^n times its original size. For example if $n=10$, less than 1% of the original interval remains. Note that, due to the properties of the golden section (e.g. $1/\phi=\phi+1$), each iteration requires only one new function evaluation.

In conclusion, if we can assume that the classification accuracy in relation to the time window is a unimodal function then the golden section algorithm can be used as an efficient way to find the optimal size of the time window. It is possible to find datasets for which the unimodal assumption is not true (e.g., when the concept changes very often and abruptly). In such cases, we can use other optimization methods that do not assume a unimodal distribution, however they take much longer. The trade-off that we have to make is that we can occasionally be trapped in a local maximum, but have a fast optimization; or find a global maximum, but have significantly slower optimization.

4 Experimental Design

All experiments were designed to run iteratively, in this way simulating the process of the mechanism's utilisation. For this reason the data streams were chunked on episodes/batched. **Figure 3** tries to illustrate how the experiments were conducted: on each iteration, a concept description is learned from the examples in the current time window; then the learned classifier is tested on the next batch.

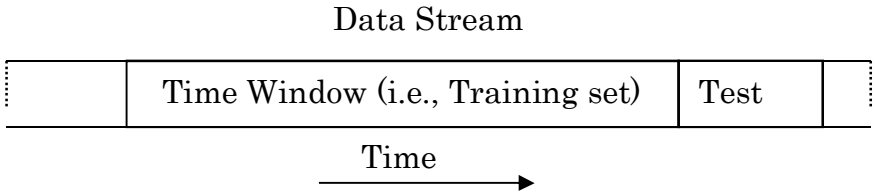

Figure 3. Iterations' cross-validation design.

To facilitate the presentation below the basic forgetting mechanisms are abbreviated as follows:

FTW - Fixed-size Time Window

GF - Gradual Forgetting

TWO - Time Window Optimization

COPL - COntext Prior Learning algorithm

For each dataset, the window size for the FTW was chosen to approximate the average time window size obtained in the experiments with the TWO mechanism on the same set. This functions as extra help for the FTW that would not be available in a real situation where the forthcoming sequence of events is unknown. The aim here is to allow the FTW approach to achieve its best performance.

Each hypothesis was tested on four experimental datasets; using three basic machine learning algorithms. The results from the experiments are reported and discussed in the next subsection.

4.1 Datasets

This subsection explains how the data streams used in the experiments were generated.

4.1.1 STAGGER Dataset

The first experiments were conducted with an artificial learning problem that was defined and used by Schlimmer and Granger 42 for testing STAGGER, probably the first concept drift tracking system. Much of the subsequent work dedicated to this problem used this dataset for testing purposes (e.g., [33], [6], [26], [22], [44], [25], [49], [17], [33], [24] etc.).

The STAGGER problem is defined as follows: The instance space is described by three attributes: ***size*** = *{small, medium, large}*, ***color*** = *{red, green, blue}*, and ***shape*** = *{square, circular, triangular}*. There is a sequence of three target concepts: (1) ***size*** = *small and* ***color*** = *red*; (2) ***color*** = *green or* ***shape*** = *circular*; and (3) ***size*** = *(medium or large),* 120 training instances are generated randomly and classified according to the current concept. The underlying concept is forced to change after every 40 training examples in the sequence: (1)-(2)-(3). The setup of the experiments with the STAGGER dataset was done exactly in the same way as in other similar experiments. The retraining step is 1, however there is a test set with 100 instances, generated randomly and classified according to the current concept. This differs from the experiments with the other datasets where the retraining step and the test set are the same - one batch. The size of the FTW is set up to 25, which approximates the average size of the optimized windows.

4.1.2 Calendar Apprentice Dataset.

Mitchell et al. [37] have developed a software assistant that helps schedule a particular user's calendar: a calendar manager called CAP (Calendar APprentice). CAP learns the users' scheduling preferences through routine use, enabling it to give customized scheduling advice to each user. It can be considered as an analogy to a human secretary who might assist someone in managing a calendar. CAP employs induction on decision tree to acquire assumptions about individual habits of arranging meetings. The learning method uses a time window to adapt faster to the changing preferences of the user. The newly generated rules are merged with old ones. The rules that perform poorly on the test set drop out of the list.

The user's scheduling preferences depend very much on a hidden context. Some of this context can be assumed and explicitly presented and used for improving predictive accuracy (e.g. academic semesters, etc.). However, there are many other events and conditions that can influence the meeting schedule and which cannot be explicitly represented by an attribute space (e.g. room availability, the schedule preferences of other participants of a

meeting and many others). Under this condition, the predictive accuracy of the system can oscillate with very high amplitude. A more comprehensive investigation and analysis of the specifics of the domain can be found in Mitchell et al. [37].

Third-most-common-time-last-60-days-this-meeting-type
Third-most-common-time-last-60-days
Second-most-common-time-last-60-days-this-meeting-type
Second-most-common-time-last-60-days
Most-common-time-these-attendees-last-60-days
Most-common-time-these-attendees
Most-common-time-last-60-days-this-meeting-type
Most-common-time-last-60-days
Most-common-day-these-attendees-last-60-days
Most-common-day-these-attendees
Duration-of-next-meeting-with-these-attendees
Duration-of-last-meeting-with-these-attendees
Day-of-week-of-next-meeting-with-these-attendees
Day-of-week-of-last-meeting-with-these-attendees
Required-seminar-type
Required-course-name
Required-speakers
Single-person?
Action
CMU-attendees?
Group-attendees?
Position-attendees
Department-attendees
Sponsor-attendees
Known-attendees?
Duration
Day-of-week
Location
Start-time

Table 1. The list of features that are used for describing calendar events.

The section below presents the results from experiments conducted with the CAP data set[2]. The attributes used for describing the calendar events in the current experiments are listed in Table 1. The task is to predict the following meeting characteristics:

[2] http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-5/www/cap-data.html

- *Duration* - the duration of the meeting in minutes e.g. 30, 60, 90, etc. (number of values legal - 13);
- *Day-of-week* - the day of the week of this meeting; e.g. Monday, Thursday, etc. (number of legal values - 6);
- *Location* – the place where the meeting is held; e.g. weh5409 (number of legal values - 142);
- *Start-time* - the time at which the meeting begins, in military time; e.g. 930 (9:30am), 1400 (2pm), etc.) (number of legal values - 21);

4.1.3 German Credit Dataset

Experiments were conducted with the German credit dataset, from the UCI Machine Learning Repository[3]. The dataset contains 1000 instances of bank credit data which are described by 20 attributes. The examples are classified in two classes as either *"good"* or *"bad".* To simulate hidden changes in the context the dataset was sorted by an attribute then this attribute was removed from the dataset for the experiments. Using an attribute to sort the dataset and in this way simulate changing context is a commonly used approach to set up experiments to study concept drift. Two sorted datasets were created using the German credit dataset: The first one was sorted by a continuous attribute: *"age"*, which would produce a gradual drift of the "class" concept. The second one was sorted by the attribute *"checking_status"*, which has three discrete values. In this way we aimed to create abrupt changes of the "class" concept. Thus using this source dataset, two data streams are generated for the experiments. The datasets were divided into a sequence of batches, each of them containing 10 examples. The size of the FTW is set to 200, which approximates the average size of the optimized windows.

4.1.4 Spam Dataset

Experiments have also been conducted with the Spam dataset from the UCI Machine Learning Repository. Spam is an unsolicited email message. The dataset consists of 4601 instances, 1813 (39.4%) of which are spam messages. The dataset is represented by 54 attributes that represent the occurrence of a pre-selected set of words in each of the documents plus three attributes representing the number of capital letters in the e-mail. To simulate the changing hidden context the examples in the dataset are sorted according to the *"capital_run_length_total"*, which is the total number of capital letters

[3] http://www.ics.uci.edu/~mlearn/MLRepository.html

in the e-mail. This attribute and the related two attributes *"capital_run_length_average"* and *"capital_run_length_longest"* are removed from the dataset, because they can provide explicit clues for the concept changes. The sorted dataset was divided into a sequence of batches with a length of 10 examples each. For this dataset the fixed window size was set to 400 - an approximation of the average window size for this dataset used by the TWO mechanism.

4.2 Used Machine Learning Algorithms

The experiments were conducted using three basic machine learning algorithms:

- k Nearest Neighbours (kNN), also known as Instance Based Learning (IBL) [2]. *k=3* was the default setting for the experiments reported below except for experiments with STAGGER dataset, where *k=1* was chosen, because it produces a more accurate classification than *k=3;*
- Induction of Decision Trees (ID3) [40] (using an attribute selection criteria based on the χ^2 statistic [35]);
- Naïve Bayesian Classifier (NBC) [36]

5 Conducted Experiments and Results

In this section, results from numerous experiments that aim to study the developed forgetting mechanisms on different benchmarking and real problems datasets are presented.

5.1 Study of the Gradula Forgeting mechanishm

5.1.1 Experiments with the ID3 Algorithm

Top Down Induction on Decision Tree (TDIDT) is one of the most widely applied learning algorithm. It is well known as ID3 [40]. The contingence tables are often used in the base of the algorithm [35]. An element $x^i_{j,k}$ in a contingence table presents the appearance of the attribute value $a_i = v^i_j$ in the subset of examples that belong to class c_k. The presented approach utilizes the weights of examples for calculating the confidence tables elements as follows: $x^i_{j,k} = \sum_{l=1}^{m} w_l b_l^{i,j}$, where: $b_l^{i,j} \in \{0,1\}$; $b_l^{i,j} = 1$ when for the example $e_l \in c_k$ the attribute value is $a_i = v^i_j$ and 0 otherwise; w_l is the weight for the example e_l calculated by a forgetting function. The different measures for goodness of split use the contingence tables in their calculations. Hence, the application of forgetting weights in creating of contingence tables will influence the whole process of building the decision tree.

Figure 4. The STAGGER problem: The improvement in predictive accuracy of ID3 when the gradual forgetting is utilized in a time window.

Figure 4 shows the results from first experiment (full memory learning). The utilization of the forgetting function (1) improves the average prediction accuracy from 69% to 77%.

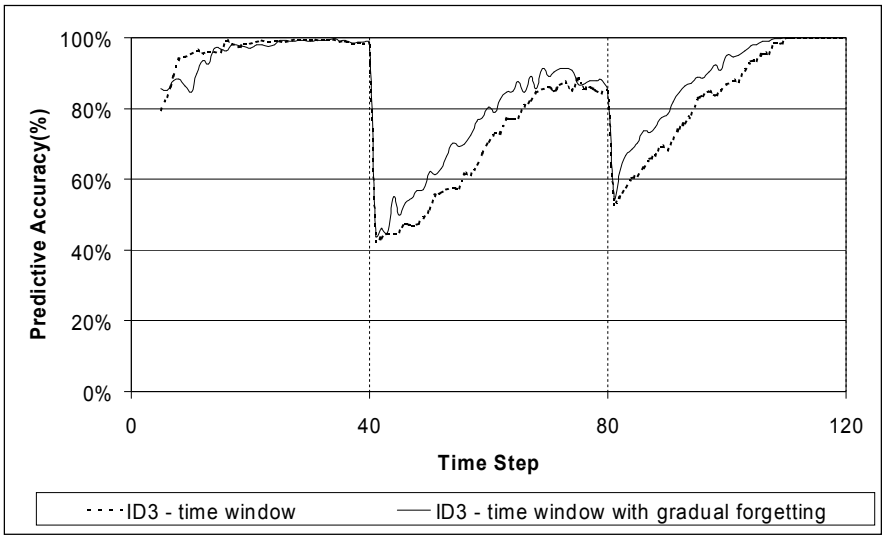

Figure 5. The STAGGER problem: The improvement of predictive accuracy of ID3 when the gradual forgetting is utilized.

Figure 5 present, the result from second experiment, which demonstrate that the presented approach can be an augmentation to the time window

approach (i.e. the examples in the time window also can be weighted according to their appearance over the time). The simple time window improves the average predictive accuracy from 69% to 83%. The usage of a gradually forgetting function in it additionally improves the average predictive accuracy to 87%. The dotted vertical lines indicate where the underlying concept changes. It can be seen that the dramatic concept shifts lead to a sharp decrease of the predictive accuracy. After such falls, the utilization of the gradual forgetting speedup the adaptation to concept changes.

5.1.2 Experiments with the NBC Algorithm

The Naïve Bayes Classifier (NBC) is a fast learning algorithm, which is also broadly used. The forgetting weights are utilized for NBC in calculating the probabilities. In order to calculate a probability it is necessary to count how many times an attribute value appears in a subset of examples. In current consideration the appearances of an attribute value in different examples are not equal important. Hence the counting of the appearance of a value in an example e_i should be multiplied with its weight w_i (1).

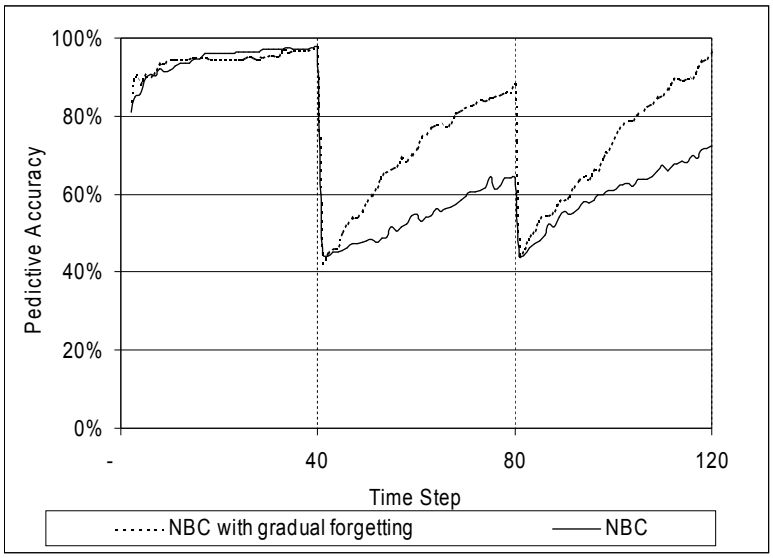

Figure 6. The STAGGER problem: The improvement of predictive accuracy of NBC when the gradual forgetting is utilized.

Figure 6 shows the results from the experiment using full memory. The utilization of the forgetting increases the average predictive accuracy from 69% to 79%.

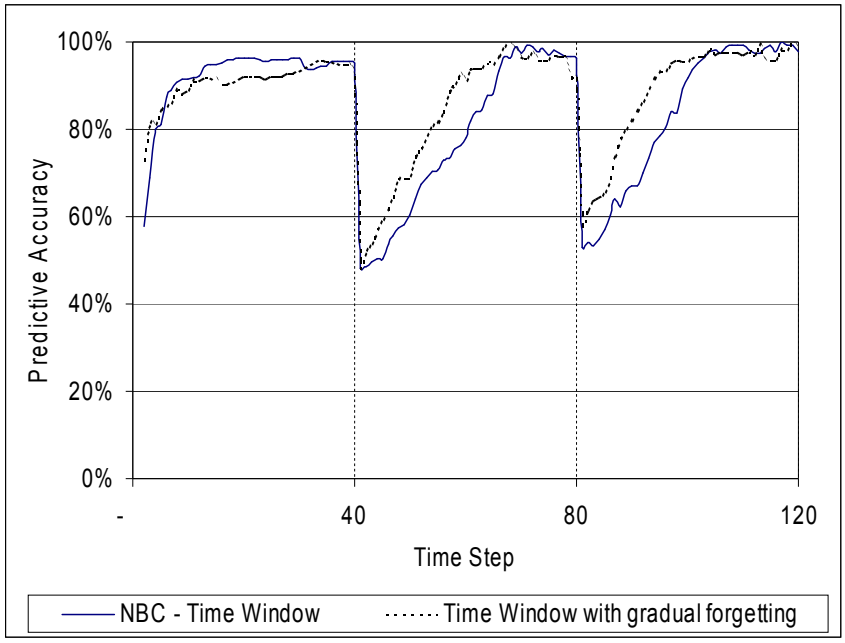

Figure 7. The STAGGER problem: The improvement in predictive accuracy of NBC when the gradual forgetting is utilized in a time window.

Figure 7 shows the results from application of gradual forgetting in a time window. The employment of a time window improves the average predictive accuracy from 69% to 84%. The gradual forgetting in the time window additionally improves the average predictive accuracy to 88%. The concept shifts lead to a sudden decrease of the predictive accuracy of the learning algorithm. The experiments show again that the algorithms utilizing gradual forgetting are able to adapt faster when the concept changes appear.

5.1.3 Conclusion

The presented approach introduces the gradual forgetting by weighting the training examples according to their appearance over time. The conducted experiments show that the presented method is applicable to different learning algorithms and is able to improve its adaptability to drifting concepts. The experiments show also that the presented forgetting method can work in cooperation with other forgetting mechanisms for partial memory learning (e.g. time window). The "speed of forgetting" can be adjusted by varying k and it can be dynamically adapted using heuristics like those used by Widmer and Kubat for adapting the size of the time [48]. Furthermore other types of forgetting functions can be defined (e.g. logarithmic, exponential etc).

5.2 Gradual Forgeting for Adaptation to Drifting User's Interests

A content-based recommender agent was developed [44]. In order to be unobtrusive, it learns individual interest profiles based on passive observations only. The central source of information about interests of a user is the sequence of objects he selected. A feature selection is employed to select those features that are extraordinarily important to the user for identifying relevant objects - i.e. an explicit user's profile. A probabilistic approach and an instance-based learning approach have been used for recommending. Both approaches have been modified to deal with a single class only. The feature selection additionally helps to improve the distance measure for instance-based learning. Moreover the feature selection is employed to focus the learning task. The developed algorithms have been implemented and evaluated in a real-world application - a WWW-based information system. A recent development for Internet browsing is presented in [43].

The presented forgetting mechanism is used to cope with drifting user's interests [23]. Since the feature selection plays a basic role in the developed methodology [44], the utilization of the forgetting for the feature selection should affect both, the explicit users' profiles and system's recommendations. In this case the forgetting weights are employed in counting the appearance of the features in user's selections by multiplying each its occurrence with the weight w_i of the observation. Consequently, this reflects the estimation of features' significance. In this way the appearance of a feature in the resent observations become more valuable.

Therefore, the problem of defining a suitable example-weighting schema for more complicated learning algorithms like TDIDT, NBC and our analysis of significance remains open. We employ the Gradual Forgetting mechanism, described above. It obeys all this requirements.

In the LaboUr project [44], feature selection is used for creating the explicit user profiles. Hence, applying gradual forgetting to feature selection will affect the explicit user profiles and consequently the generated recommendations. In particular the forgetting weights are utilized in counting the appearance of a feature j in the user's selections by multiplying each occurrence of the feature in a observation by its weight (i.e $c^j = \sum_{i=1}^{n} w_i v_i^j$, where $v_i^j \in \{0,1\}$ are the feature values in the Boolean vectors that represent the observed user's selections; w_i are the weights calculated by the forgetting function for the observations).

Observation no. / Feature	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
Control engineering	0.93	0.92	0.91	0.90	0.89	0.88	0.87	0.86	0.85	0.84	0.83	0.82	0.81	0.80	0.79
	0.67	0.65	0.63	0.61	0.59	0.58	0.56	0.55	0.53	0.52	0.51	0.49			
Mechanical engineering	0.94	0.93	0.92	0.91	0.90	0.89	0.88	0.87	0.86	0.85	0.84	0.83	0.82	0.81	0.81
	0.82	0.80	0.78	0.76	0.75	0.73									
Multimedia									0.48	0.52	0.56	0.59	0.62	0.65	0.68
				0.50	0.55	0.58	0.62	0.70	0.73	0.75	0.77	0.78	0.83	0.84	0.85
Developing countries											0.58	0.57	0.56	0.55	0.53
											0.88	0.87	0.87	0.86	0.85
Electrical engineering	0.83	0.81	0.79	0.78	0.76	0.75	0.73	0.72	0.71	0.69	0.68	0.67	0.66	0.65	0.64
	0.83	0.81	0.79	0.78	0.76	0.75	0.73	0.72	0.71	0.69	0.68	0.67	0.66	0.65	0.64
Europe	0.62	0.59	0.57	0.55	0.54	0.52	0.50	0.49	0.48						
	0.62														
Numerics											0.51	0.49	0.48	0.47	
										0.83	0.82	0.81	0.80	0.79	
Mathematics							0.58	0.57	0.55	0.53	0.52	0.51	0.49	0.48	
			0.66	0.64	0.62		0.78	0.77	0.75	0.74	0.52	0.51	0.49	0.48	

Table 2. The influence of the gradual forgetting mechanism on generated user profile. The gray lines represents the weights generated by pure feature selection. The white lines present the changes in the user profile using forgetting function (2), with k=80%. The confidence interval for the feature selection is set to 99%.

To illustrate how the utilization of the proposed forgetting mechanism affects the user profiles let us study in more detail the changes in a real user profile. Table 1 compares an excerpt of two profiles of a user, one generated without and one with a forgetting function. The rows represent some of the features that have been selected as the user's interests. The gray lines represent the changes in the user profile generated by pure feature selection. The white lines represent the changes in the user profile when the forgetting function (3) is utilized (k=80%). Each column presents a subset of features selected for the user profile generated after each selection. The period presented in the table ranges from selection 11 to 25. The numbers in the cells represent the calculated normalized weights for each selected feature. If the cell is empty, the feature was not selected as interesting at this stage. The confidence interval for feature selection was 99%, which leads to the selection of the most significant user interests only. It can be seen that user interests change over time. It appears that the user profile generated using the forgetting function is more sensitive and responds faster to changes in the user's interests. Interests that do not occur in recent observations are forgotten faster (e.g., the features "control engineering" and "mechanical engineering"). Apparently, these features drop out from the current user interests faster if the forgetting mechanism is employed. In a similar way, new interests are recognized earlier (e.g., the algorithm that employs the presented forgetting mechanism is able to recognize feature "multimedia" as a user interest five steps earlier than the approach without forgetting). Since the presented aging mechanism does not totally forget old observations, a reoccurring user interest will also be recognized more easily. Moreover, features that appear in the user's selection regularly, i.e. represent stable user interests, are left almost unchanged (e.g. the feature "electrical engineering"). And finally, when the method is applied to a larger set of training examples, it becomes more noise resistant without losing its sensitivity to real changes in interest. We can conclude that the user profile created using this forgetting mechanism includes mostly those features which mainly represent the interests that occurred in recent observations and those which are stable over time.

5.2.1 Experiments

In the first experiment our goal was to investigate how the weights for gradual forgetting are able to influence the predictive accuracy of recommendations.

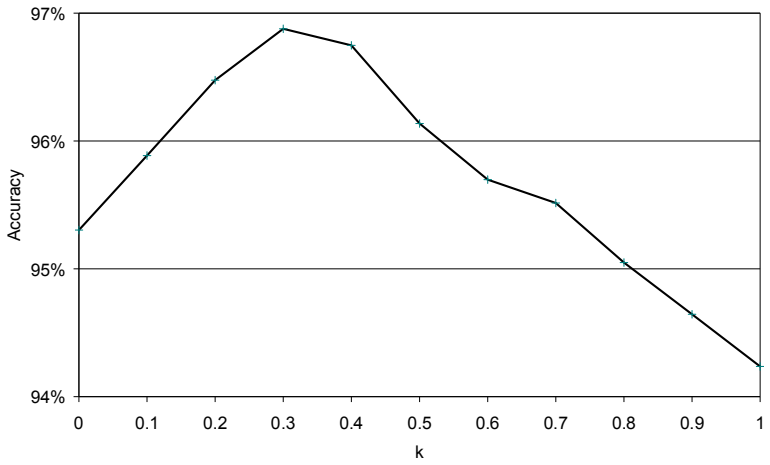

Figure 8. The correlation between the "forgetting value" *k* and the quality of prediction for groups 2-5. *k=0* means no forgetting.

The experiments were conducted using the same approach as reported in Section 3.3.4, and the users were divided into the same 5 groups depending on their total number of document selections. We omitted Group 1, however, since we did not expect to be able to detect interest drifts in the first five user selections (this was also confirmed in informal trials). The first experiment explores the relationship between the value of k and the prediction accuracy of the recommendations. For each of the groups 2-5 there exists a value of k, named optimal group average, for which the average prediction accuracy is greater than the one for k=0. The optimal group average value of k tends to increase slowly with the size of the training set as follows: Group 2: k=0.2; Groups 3 and 4: k=0.3; Group 5: k=0.4. k=0.3 is an "optimal average" for those groups of users (see **Figure 8**). To be even more precise one can use an empirical function $k = f_k(n)$, where n is the position of the observation.

Figure 9 compares the optimal learning algorithm for LaboUr system (namely instance-based learning *cum* weighted distance measure and feature selection) with supplementary interest drift monitoring using an optimal average k and an optimal group average. The improvement of the predictive accuracy for the optimal average (*k=0.3*) is significant for $\alpha = 0.05$, but it is very close to the critical value. When we use the optimal group average, the improvement becomes significant. The improvement in average accuracy largely depends on the behavior of the users. When the user's interests change, the prediction accuracy can decrease very much. After such shifts, the presented approach is able to adapt faster to the new user interests. Therefore the current approach can be improved by dynamically adapting the value of *k* to each user. This is however left to future research.

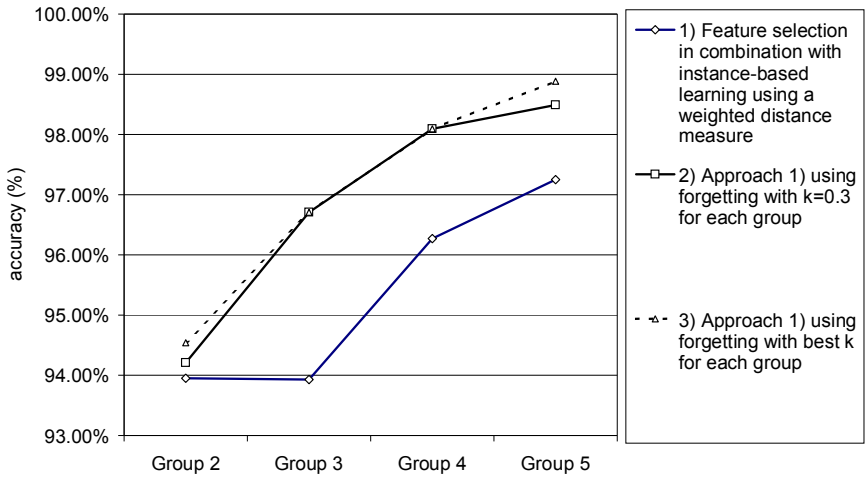

Figure 9. The improvement in accuracy of recommendations when gradual forgetting is used.

The above experiments show that the presented forgetting mechanism can be successfully used for learning drifting user interests. Experiments conducted with a data set from an artificial learning problem reported above demonstrate that the presented approach is able to significantly improve the predictive accuracy of inductive learning algorithms (TDIDT, NBC and IBL) on drifting concepts can work in cooperation with other forgetting mechanisms like time windows, by weighting the observation inside the window. Since ELFI users are relatively short time users and since the average prediction accuracy increases with the number of selections, we do not consider combining gradual forgetting with a time window for this application however.

5.2.2 Conclusions

The experiment shows that the employment of gradual forgetting is able to influence both the generated explicit user profiles and the recommendations. The effect on the user profiles is that they include mainly the features that represent recent observations and those, which characterize interests that are stable over time. The average predictive accuracy of the recommendations improves about 2%. This improvement may not look very large, but we should take into account that average predictive accuracy approaches 90% and in about 40% of the cases the system accuracy approaches 100%. In such cases the interests are stable and any improvement is actually impossible. When the user's interests are changed the predicting accuracy can drop down dramatically. After such

changes, the utilization of presented forgetting mechanism results in faster adaptation to the new user's interests.

5.3 Experiments with COPL mechanism

This section present results from experiments with the COPL algorithm. Two data sets are used in the experiments. The first one contains data from a real use of a calendar manager tool aiming at helping the user to scheduling meetings Mitchell et al. [37]. This dataset is described above in section 4.1.2. The second one is the an artificial data set that is used in many papers in the area of Machine Learning dedicated to concept drift – i.e. STAGGER. This dataset is described above in section 4.1.1.

The settings of the COPL algorithm listed in the previous section are defined for the conducted experiments with CAP dataset (see section 4.1.2 as follows:

- The size of the time window: Preliminary experiments show that for different prediction tasks the size of the window that produces best predictive accuracy can be quite different. For the given data set the best accuracy is reached for the window of the following size: Location - 200; Duration - 350; Start-time - 350; Day-of-week - 400.
- The episode selection criterion for step 2. The criterion used in this implementation selects the examples e_j for the new data set taking into account the average predictive accuracy in its neighborhood. In particular, a small episode around the example which includes the previous two and next two examples is used. An event will be selected for the new training set $e_j \in S_{new}$ if the average predictive accuracy for this episode is greater than or equal to a predefined threshold τ .
- The threshold for the episode-selecting criterion in step 2. is set up to $\tau = 0.6$ in all experiments.
- The learning algorithm used in steps 1. and 3. is Induction on Decision Tree (aka ID3) [40]. This algorithm was used in CAP, which makes the comparison between different approaches clearer. This algorithm produces an explicit user profile (e.g. set of rules) that is understandable for the user. This is an important advantage from the viewpoint of user modeling.

Prediction task	*CAP*	*ID3-FM*	*COPL (ID3)*
Location	64%	58%	67%
Duration	63%	71%	79%
Start-time	34%	39%	48%
Day-of-week	50%	52%	66%
Average	53%	55%	65%

Table 3. Comparison of predictive accuracy for the User1.

Table 3 presents the results from experiments with data for User 1. In this experiment a new description of user preferences is learned after each 10 meetings. The learned description at each step is tested on the next 10 meetings. The line in the table presents the accuracy of prediction for different learning tasks. The results are compared with the CAP. The average predictive accuracy of the ID3 with full memory (ID3-FM) to some extent outperforms the CAP. This is slightly surprising, because CAP is designed to track changing user preferences better than a simple learning algorithm. An explanation of this phenomenon is that some implementation details like attribute selection criteria and used pruning method can change the outcome of the algorithm. The use of one level time window, even with an adaptive size, does not improve the predictive accuracy because the user preferences alternate very often and with high amplitude. The comparison between full-memory learning algorithm (ID3-FM) and the presented two-level learning algorithm is fully compatible because the same implementation of the basic learning algorithm is used. The results from the experiments show that the context-learning algorithm is able to improve the average predictive accuracy for each feature. All those improvements are significant (using t-test with $\alpha = 0.01$).

Figure 10, Figure 11, Figure 12 and Figure 13 show the results from experiments for the predicted features. It can be seen that the user's preferences can change abruptly, which leads to a dramatic decrease of the predictive accuracy. The presented two-level algorithm tracks changes better than the basic algorithm and produces a significantly improved average accuracy.

Experiments with this data set, which use the Winnow and Weighted-Majority algorithms, were reported in Blum [7]. The Winnow with a large feature set reaches the best average accuracy, which is equal to that reached by the algorithm in the presented experiments. However, these algorithms

are not suitable for producing explicit user profiles, which is considered to be important in the area of user modeling.

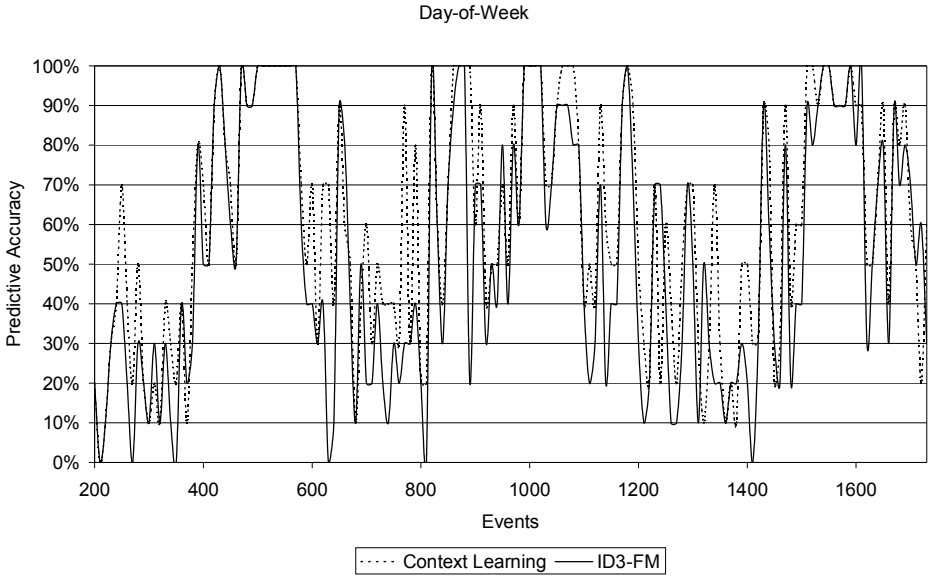

Figure 10 The improvement in predictive accuracy for the feature *Day-of-week*

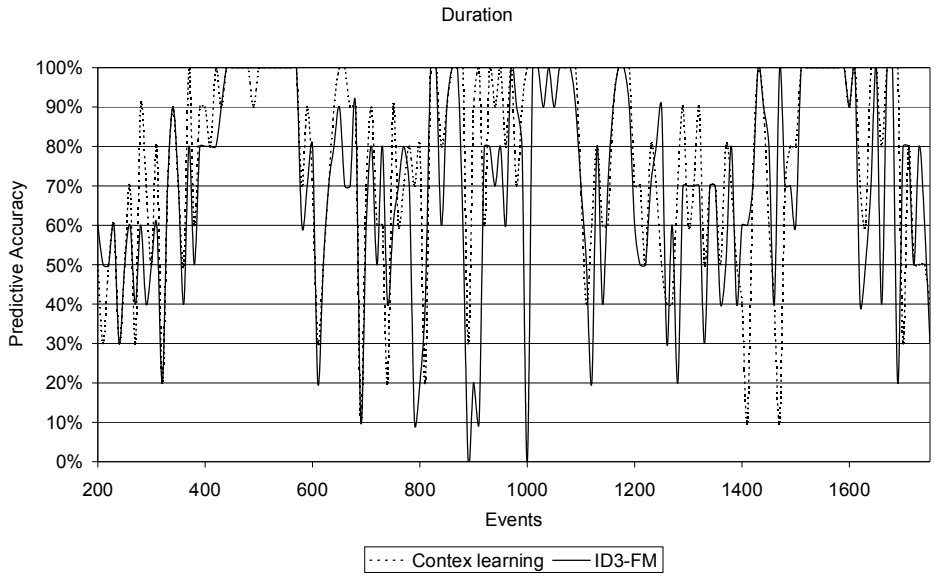

Figure 11 The improvement in predictive accuracy for the feature *duration*

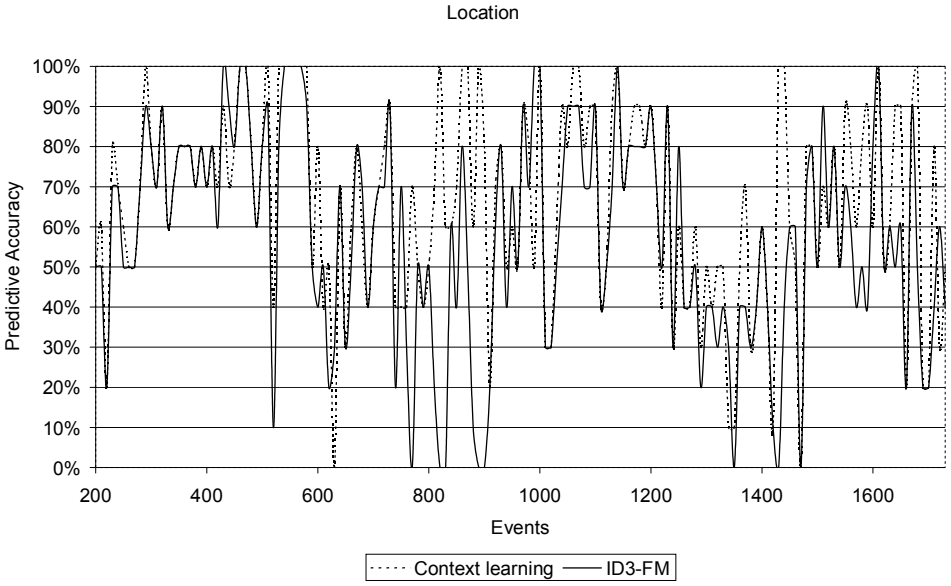

Figure 12 The improvement in predictive accuracy for the feature *location*

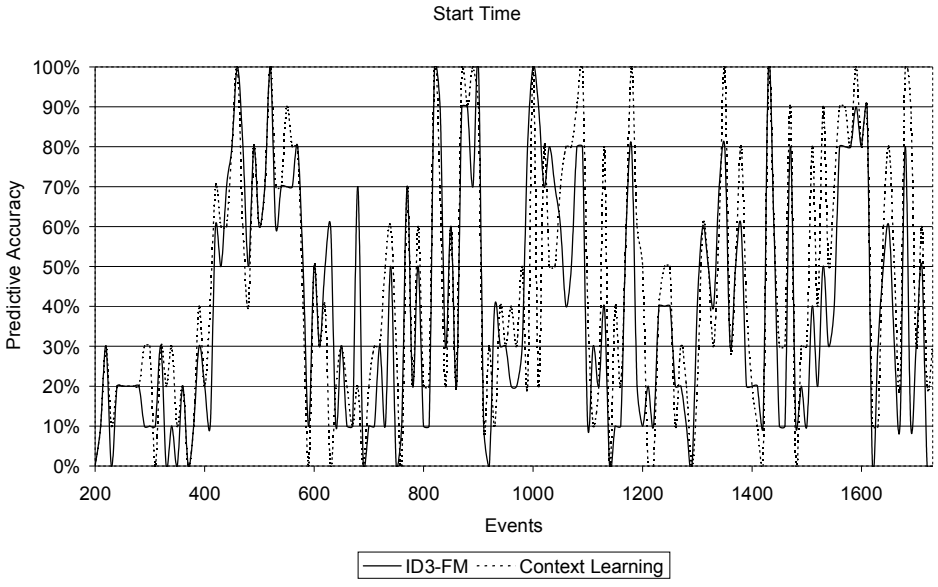

Figure 13 The improvement in predictive accuracy for the feature *start time*

To compare the presented approach with FLORA3, which is able to recover 'old' rules learned in a similar context [49], experiments were conducted also with STAGGER dataset described in section 4.1.1.

The parameters for the COPL algorithm in this experiment are set up as follows: the size of the time window used at step 1 is 18; the episode selection criteria and the related threshold remain the same as above; and the used learning algorithms at step 1 and 3 is Naïve Bayes Classifier

(NBC) to demonstrate the ability of the presented two-level algorithm to work with other learning algorithms.

Algorithm\Examples:	*2-120*	*121-240*	*241-360*
FLORA 3 context	85.9%	85.4%	83.5%
COPL (NBC)	85.3%	85.6%	87.1%

Table 4. Comparison between FLORA3 and COPL (NBC) on recurring context.

Table 4 compares the presented algorithm with FLORA3 [49]. On the basic data set (1-120) the FLORA3 produces a slightly better accuracy (i.e. non-significant difference). On recurring concepts (i.e. examples 121-360) both algorithms perform better than the ones that do not recover the context (e.g. FLORA2 [48]- 81.5%). The COPL (NBC) algorithm benefits from the recurrence of context better than FLORA3 (see columns *121-240* and *241-360* of Table 3). Moreover, the predictive accuracy of the presented algorithm increases when context recurs, which shows that it really takes advantage of context recurrence. For example, on second recurrence of the concept (see column *241-360* of Table 3) the COPL algorithm produces a significantly better (using t-test with $\alpha = 0.01$) average accuracy than FLORA3.

5.3.1 Conclusion

The paper describes a two-level learning algorithm that is able to track changing user interests and preferences through prior-learning of context. The algorithm benefits from the recurrence of the context by remembering the relevant observations and forgetting the irrelevant ones. The presented approach provides a general framework for dealing with changing and recurring user interests that can be used with different machine learning algorithms. Conducted experiments with recommendations about calendar scheduling demonstrate that the approach is able to improve the predictive accuracy significantly. Additional experiments conducted with an artificial data set demonstrate that the presented algorithm really makes use of context recurrence and increases the predictive accuracy when the context recurs. Further investigations of the episode selection criterion and designing a mechanism for its threshold detection are expected to improve the predictive accuracy of the algorithm additionally.

5.4 Experiments with Time Window Optimisation Mechanism

The aim of the experiments reported in this section is to explore whether the present forgetting mechanism is able to improve the performance of different learning algorithms on drifting concepts.

All experiments were designed to follow the natural scenario of using such mechanisms [37]. For this reason the data streams were chunked on episodes/batched. The algorithm was run on this data set iteratively - on each iteration, a concept description is learned from the examples in the current time window. Then the learned classifier is tested on the next batch.

The experiments were conducted with three popular learning algorithms:

- k Nearest Neighbours (kNN) - also known as Instance Based Learning (IBL) [2]. k=3 was the default setting for the experiments reported below except for experiments with STAGGER dataset, where k=1 was chosen, because it produces a more accurate classification than k=3;
- Induction of Decision Trees (ID3) [63] (using an attribute selection criteria based on the χ^2 statistics);
- Naïve Bayesian Classifier (NBC) [63].

The first experiments were conducted with the STAGGER dataset, described above in section 4.1.1. Those results are presented in the next subsection. Experiments also were conducted with two datasets from the UCI machine learning repository[4], which are presented in subsections below.

The results from the conducted experiments are presented in Tables 5, 6, 7 and 9 below. In all these tables, rows present the used learning algorithms: kNN, ID3 and NBC. The first column shows the predictive accuracy of the algorithms using Full Memory (FM) learning – all data available up to the current moment are used for learning the concept. The second column presents the results from the experiments with Fixed-size Time Window (FTW). The third column shows the results from the experiments with the algorithms using this paper's Time Window Optimisation (TWO) mechanism. For each data set, the window size for the FTW was chosen to approximate the average time window size obtained in the experiments with the TWO mechanism on the same dataset. This is extra help for the FTW that would not be available in a real situation where

[4] http://www.ics.uci.edu/~mlearn/MLRepository.html

the forthcoming sequence of events is unknown. The aim here is to allow the FTW approach to show its best performance.

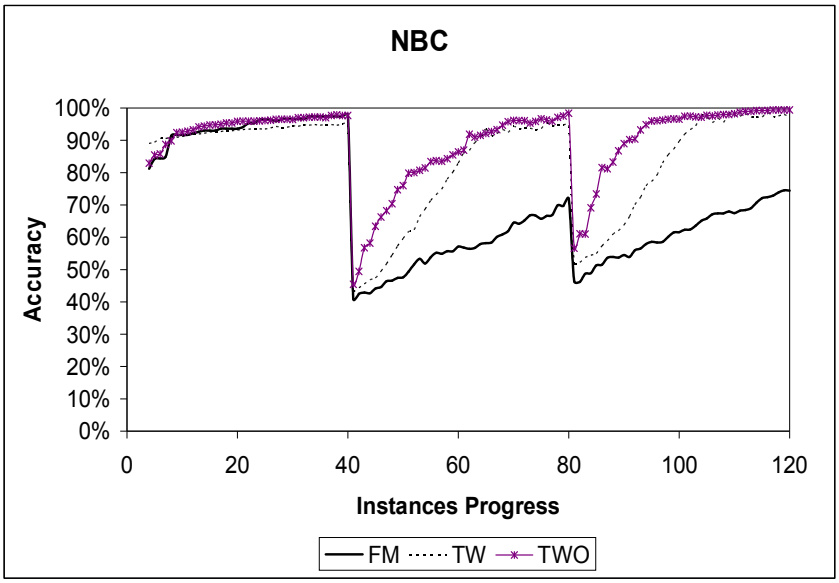

Figure 14 Classification accuracy of the NBC using Full Memory, Time Window and Time Window Optimisation.

We used the paired t-tests with 95% confidence level to see whether the presented approach significantly changes the accuracy of learned classifiers. The pairs are formed by comparing the algorithms' accuracies on the same iteration. In the tables below, reporting the results from the experiments, the sign * denotes that the TWO approach achieves a significantly better classification accuracy than the FM and the sign ^ - that TWO is significantly better than the FTW approach.

5.4.1 STAGGER dataset

The setup of the experiments with the STAGGER dataset was done exactly in the same way us in other similar works. The retraining step is 1, however there is a test set with size 100, generated randomly and classified according to the current concept. This differs from the other experiments where the retraining step and the test set are the same - a batch. The size of the FTM is set up to 25, which approximates the average size of the optimised windows.

Memory: / Algorithm:	FM	FTW	TWO
kNN	**63.03**	**80.47**	**86.56*^**
ID3	**69.05**	**83.73**	**89.03*^**
NBC	**69.97**	**82.99**	**90.26*^**

Table 5. The improvement of the classification accuracy when the TWO mechanism is applied to the STAGGER dataset

Table 5 shows the results from the experiments with this dataset. In this dataset we have two abrupt changes in the underlying concept and the fixed size time window is able to improve the classification accuracy significantly compared with the full memory. The TWO mechanism additionally improves the classification accuracy significantly compared to FTW.

Figure 14 shows a plot of the results from the experiments with NBC, which illustrate the behaviour of the three mechanisms. It can be seen from the chart that when the algorithm uses the TWO mechanism it adapts faster to the changes.

Figure 15 shows the average size of the time window for each step in one of the experiments with NBC-TWO. The experiments with other algorithms produce very similar graphs.

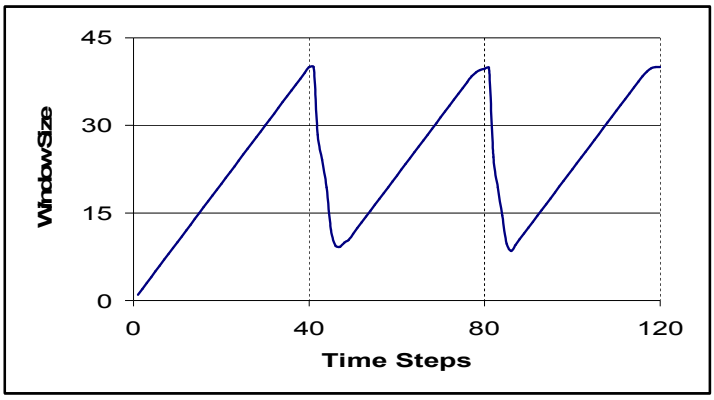

Figure 15 The average window size for each step in the experiment with NBC-TWO

As we mentioned above the STAGGER problem was used by a number of other systems that deal with concept drift. To compare the present approach

with the previous approaches, we summarised all available results from experiments using this dataset in Table 6. The first ten rows of this table present the results from experiments with other similar approaches. The last three rows present the results from the experiments with the algorithms using the Time Window Optimisation (TWO) mechanism. The results are shown in more detail to facilitate a better comparison of the systems' performance on different concepts: (1), (2) and (3), shown in separate columns. The last column shows the average performance on the whole dataset. We will look more closely at the results from the second and third concepts, where actually the systems adapt to changes in the underlying concept. The performance of the systems on this dataset depends on the basic learning algorithms, which seem to perform differently on this dataset. Therefore we will mainly be interested in comparing the systems that use the same basic learning algorithm.

The algorithms that use the present TWO mechanism (rows 11-13) significantly outperform the first three systems (rows 1 to 3). The results presented in row 4, are from the experiments with the COPL mechanism, using NBC as the basic learning algorithm. Therefore, we compared it with NBC-TWO, which achieves a significantly better performance for this dataset.

The FLORA systems reach significantly higher accuracy, than all other approaches on the first concept, which is actually a stable one. However, NBC-TWO and ID3-TWO significantly outperform all FLORA systems on the concepts (2)-(3), i.e. after concept drift has occurred. In general, kNN does not perform very well on this dataset, despite this kNN-TWO significantly outperforms the FLORA2 and FLORA3 algorithms on the concepts (2)-(3).

Concept: / Algorithm:	(1)	(2)	(3)	(1)-(2) -(3)
1. IB2 [1]	**80.4**	**51.9**	**55.6**	**63.9**
2. AQBL [33]	**89.6**	**57.2**	**55.7**	**67.0**
3. AQPM [33]	**89.7**	**70.5**	**75.1**	**79.9**
4. COPL (NBC) [26]	**91.2**	**78.9**	**85.9**	**85.3**
5. FLORA 2 [44]	**98.8**	**80.4**	**80.3**	**86.5**
6. FLORA 3 [44]	**98.7**	**80.7**	**79.5**	**86.0**
7. FLORA 4 [44]	**98.0**	**82.5**	**82.7**	**87.7**
8. kNN – GF [25]	**91.8**	**79.5**	**83.2**	**84.8**
9. ID3 – GF [25]	**93.9**	**78.3**	**89.0**	**87.07**
10. NBC – GF [25]	**92.4**	**83.9**	**88.9**	**88.4**
11. kNN – TWO	***91.9***	***81.4***	***86.3***	***86.5***
12. ID3 – TWO	***93.1***	***84.2***	***89.8***	***89.0***
13. NBC – TWO	***93.9***	***85.4***	***91.5***	***90.3***

Table 6. Average classification accuracy of systems with embedded concept drift tracking mechanisms on the STAGGER dataset

The results reported in rows 8 to 11 are from algorithms that use the Gradual Forgetting (GF) mechanism [25]. For this dataset the GF mechanism uses a fixed-size (*l=30*) time window in which the examples are assigned gradually decreasing weights using a linear forgetting function. We are comparing TWO and GF mechanisms on pairs that use the same learning algorithm (e.g. rows 9 and 12). We can see that there is no difference in average accuracy on the first concept. There is a small, but significant improvement of the accuracy obtained by TWO for the changed concepts (2)-(3).

5.4.2 German Credit Dataset

This subsection presents the results from the experiments conducted with the German credit dataset, described in section 4.1.3. Using an attribute to sort the data set and in this way simulate changing context is a commonly used approach to set up experiments to study concept drift. Two sorted datasets were created using the German credit dataset: The first one was

sorted by a continuous attribute: "age", which would produce a gradual drift of the "class" concept. The second one was sorted by the attribute "checking_status", which has three discrete values. We aimed in this way to create abrupt changes of the "class" concept. The dataset was divided into a sequence of batches, each of them containing 10 examples. The size of the FTM is set to 200, which approximates the average size of the optimised windows.

Memory: / Algorithm:	FM	FTW	TWO
kNN	**68.25**	**72.37**	**77.75*^**
ID3	**77.00**	**75.50**	**79.00*^**
NBC	**78.37**	**75.87**	**78.63 ^**

Table 7. The improvement of the classification accuracy when the TWO mechanism is applied to the Credit dataset (sorted by *"age"* attribute).

Table 7 shows the results from the experiments with the Credit dataset (sorted by *"age"* attribute). With all algorithms an improvement in classification accuracy was achieved when the algorithms using TWO mechanism were applied. All these improvements are significant except the comparison with NBC-FM.

Table 8 shows the results from the experiments with the Credit dataset (sorted by "*checking_status"* attribute). The results show that the TWO mechanism improves the classification accuracy of the algorithms and these improvements are significant for all algorithms.

Memory: / Algorithm:	FM	FTW	TWO
kNN	**64.00**	**71.75**	**77.13*^**
ID3	**64.87**	**71.75**	**75.25*^**
NBC	**74.37**	**74.14**	**77.76*^**

Table 8. The improvement of the classification accuracy when the TWO mechanism is applied to the Credit dataset (sorted by "*checking_status"* attribute).

The results also show that a fixed time window does not always provide an improvement of the accuracy and can even be destructive compared to the full memory learning algorithm. The problem with it is that we do not know in advance how the concept will change and what will be the best size in the future. Even with some "cheating", by using a time window approximating the average optimal window size in the experiments with the TWO mechanism, an improvement is achieved in only half of the cases with this dataset.

5.4.3 Spam dataset

Experiments have also been conducted with the Spam dataset from the UCI machine learning Repository, described above in section 4.1.4. To simulate the changing hidden context the examples in the dataset are sorted according to the *"capital_run_length_total",* which is the total number of capital letters in the e-mail. This attribute and the related two attributes "*capital_run_length_average"* and "*capital_run_length_longest"* are removed from the dataset, because they can provide explicit clues for the concept changes. The sorted dataset was divided into a sequence of batches with a length of 10 examples each.

Table 9 presents the results from the experiments with the Spam dataset comparing full memory learning, time window with fixed size and the time window with optimized size. For this dataset the fixed window size was set to 400 - an approximation of the average window size for this dataset used by the TWO mechanism. For this dataset for two of the algorithms (kNN and NBC) the fixed time window improves the classification accuracy, but not significantly in either case. For ID3 we can even see a slight decrease in the accuracy. An improvement of the classification accuracy for all

algorithms was achieved when the TWO mechanisms were applied and all those improvements are significant compared to FM and FTW as well.

Memory: / Algorithm:	FM	FTW	TWO
kNN	**90.12**	**90.10**	**92.48*^**
ID3	**87.08**	**86.56**	**89.51*^**
NBC	**90.61**	**90.78**	**91.56*^**

Table 9. The improvement of the classification accuracy when the TWO mechanism is applied to the Spam dataset, sorted by *"capital_run_length_total"* attribute.

To explore how the TWO mechanisms behave, according to the nature of the changes in the concept, we draw on the same diagram on Figure 16 the classification accuracy in one of the experiments with the TWO mechanism (the thin line) and the size of the optimized time window (the thick line) on each step. We see that a drop in the accuracy normally leads to a decrease in the time widow size. However, a sudden decrease in the classification accuracy does not always indicate a concept drift, it can be caused by noise in the data stream. It can be seen that the TWO mechanism is very robust to noise, merely decreasing the window size insignificantly, e.g., see the *arrow 1* on the Figure. However, it remains sensitive enough to detect genuine concept drifts that decrease the accuracy by a relatively small value – e.g., see the *arrow 2* on the Figure. The detection mechanism flags both real and false concept drifts, but the window size optimizer responds very differently to the two possibilities.

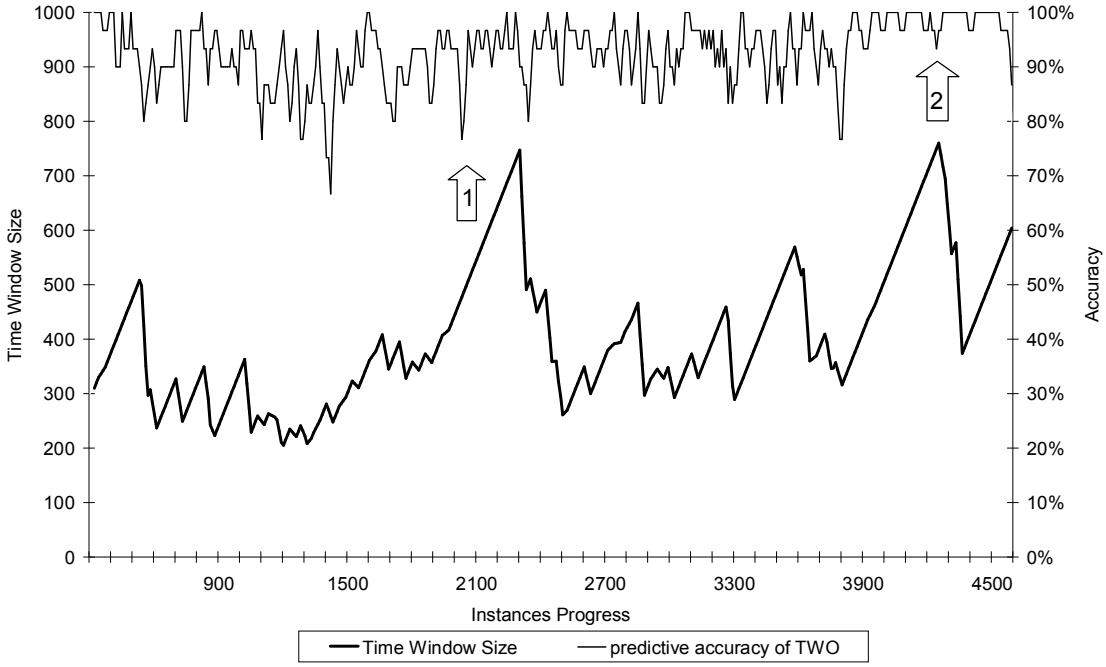

Figure 16 The relationship between the accuracy of prediction measured on the test set and the time window size.

5.4.4 Conclusion

The paper presents a mechanism for dealing with the concept drift problem, which uses a statistical test to detect whether the current concept is changing. If a concept drift is detected, then the mechanism optimizes the time window size to achieve maximum accuracy of prediction. The algorithm is self-adapting and it can be used in many datasets without any predefined domain-dependent heuristics or parameter. Moreover, the developed mechanism is not attached to a particular learning algorithm. It is general in nature and can be added to any relevant algorithm.

The results from experiments with three learning algorithms using three datasets provide strong evidence that the mechanism is able significantly to improve the classification accuracy on drifting concepts.

5.5 Comparative Study of Gradual Forgetting and Time Window Optimization

In this section we will first raise hypothesis, we will conduct experiment to test them, using statistical test for significance of the differences of performance measured in accuracy.

5.5.1 Hypotheses

The following four hypotheses were tested:

H1: *Fixed-size Time Window with Gradual Forgetting (FTW+GF) is better than pure Fixed-size Time Window (FTW), measured in classification accuracy.*

H2: *Time Window Optimization (TWO) is better than Fixed-size Time Window (FTW) measured in classification accuracy.*

H3: *Time Window Optimization (TWO) is better than Fixed-size Time Window with Gradual Forgetting (FTW+GF) measured in classification accuracy.*

H4: *Time Window Optimization with Gradual Forgetting (TWO+GF) is better than Time Window Optimization (TWO) measured in classification accuracy.*

5.5.2 Experimental Results and Discussion

This section presents the results from the experiments. Parts of these results were introduced in 22 and 24. The current paper summarizes the findings.

Tables 10, 11, 12 and 13 below summarise the results in the form: the rows show the learning algorithms used: kNN, ID3 and NBC; the columns show different datasets.

For a more compact presentation in the tables the datasets are referred to with numbers as follows:

1 – STAGGER dataset

2 – German dataset - sorted by the attribute *"checking_status"*

3 – German dataset - sorted by the attribute *"age"*

4 – Spam dataset - sorted by the attribute "*capital_run_length_total"*

We used paired t-tests with a 95% confidence level to see whether there is a significant difference in the accuracy of learned classifiers. The pairs are formed by comparing the algorithms' accuracies on the same iteration. In the tables below, we are reporting the results from the experiments testing the formulated hypothesis. The cells represent the results from the significance test comparing the results from the runs using the different forgetting mechanisms. The basic learner used is in the row and the dataset used is in the column. The notation is as follows:

✓ - the second algorithm performs significantly better then the first one;

× - the first algorithm performs significantly better then the second one;

÷ - there is no significant difference in the performance of the compared algorithms.

For example: the sign ✓ in the first column and first row in **Table 10** denotes that the experiments conducted using kNN as a basic algorithm and using the STAGGER dataset (corresponds to column 1), shows that FTW with GF is significantly better than the plain FTW, measured in accuracy of classification.

Table 10. Comparison of FTW versus FTW+GF forgetting mechanisms.

Algorithm	dataset	1	2	3	4
kNN		✓	✓	✓	×
ID3		✓	✓	✓	÷
NBC		✓	✓	✓	÷

The results from the experiments testing hypothesis **H1** are presented in **Table 10**. For three of the four datasets, for any of the gradual forgetting is able to significantly improve the average classification accuracy in comparison with Fixed Time Window, however gradual forgetting fails to improve the accuracy on the fourth dataset. In comparison with other datasets, the main difference in this dataset is that its features represent words' frequency in e-mail messages. As a result, in most cases, the features' value is equal to zero i.e., the dataset is sparse. Probably this can be the reason for the lack of improvement. Perhaps, using features selection and adapted for text similarity measures for kNN will diminish this problem. Further studies are needed to find the correct answer to these questions.

Table 11. Comparison of FTW versus TWO forgetting mechanisms.

Algorithm	dataset	1	2	3	4
kNN		✓	✓	✓	✓
ID3		✓	✓	✓	✓
NBC		✓	✓	✓	✓

The experimental results presented in **Table 11** provided very strong evidence that the Time Window Optimization mechanism is able to improve

the prediction accuracy significantly in comparison with Fixed-size Time Window, i.e., the **H2** hypothesis is confirmed.

Table 12. Comparison of FTW+GF versus TWO forgetting mechanisms.

Algorithm	dataset	1	2	3	4
kNN		✓	÷	✓	✓
ID3		✓	✓	✓	✓
NBC		✓	✓	✓	✓

Table 12 presents the results from experiments testing the third hypostasis **H3.** It can be seen that in all experiments the Time Window Optimization mechanisms achieve significantly better classification accuracy than Gradual Forgetting, except one instance where there is no significant difference in performance.

Table 13. Comparison of TWO versus TWO+GF forgetting mechanisms.

Algorithm	dataset	1	2	3	4
kNN		÷	✓	✓	×
ID3		✓	✓	÷	×
NBC		×	÷	✓	÷

Table 13 shows the results from the experiment testing the hypothesis that applying Gradual Forgetting together with TWO can improve the prediction accuracy in comparison to pure TWO, i.e., the two mechanisms are having a synergetic effect. The results from the experiments do not provide enough evidence in support the **H4** hypothesis.

Further the presented forgetting mechanisms (GF and TWO) were compared with similar approaches (i.e. 33, 26, 44 and 33) on the STAGGER dataset. A detailed discussion of this comparison can be found in 22. In summary the results show:

- *GF mechanism is a simple approach that usually performs comparably to other approaches, in some cases outperforms them significantly when concepts are drifting;*
- *TWO mechanism significantly outperforms the other forgetting mechanisms.*

5.5.3 Conclusion

This work presents two forgetting mechanism for dealing with the concept drift problem and an experimental study. Both are general in nature and can be added to any relevant algorithm. Experiments were conducted with three learning algorithms using four datasets. The results from the experiments show that: applying Gradual Forgetting inside a fixed-size time window usually leads to a significant improvement of the classification accuracy on drifting concepts; when comparing the two forgetting mechanisms the advantage clearly is on the side of the self-adapting algorithm - Time Window Optimization; finally, applying both mechanisms together usually does not lead to an additional improvement in the classification accuracy.

Further controlled experiments using common patterns from the space of different type of drift (i.e., frequent – rare; abrupt - gradual; slow – fast; permanent – temporary; etc.) will provide clearer evidence for which pattern, and what kind of forgetting mechanism is most appropriate.

The importance of taking into account the idea of concept drift in cases of counter-terrorists mining of the Web springs from the nature of the typical task: a limited data history and often changing concepts. The significance of this problem was also raised by other participants in their formal talks and informal discussion.

6 Conclusion

The results presented in this thesis are motivated by work in area of User Modelling [44]. To model the phenomenon of changing user's interest, I developed approaches that learn self-adaptive models to changes in the interests of the consumer. The task of updating the models of changing user's interests is reduced to more abstract task of learning changing concepts in the field of machine learning aka learning drift concept. In this area were developed tree new methods for learning classifiers of changing concept:

- The first mechanism proposes a gradual forgetfulness of training examples through appropriate weighting scheme with weights according to their appearance in time. By weighting schema assigns more weight to recent examples, thus the system gradually "forgets" the oldest examples. Furthermore as the classical weighting schema (i.e. sum of weights equal 1) does not easy fit to many machine learning algorithms for learning concepts (classifiers), a new weighting scheme is proposed (see section 3.1). It requires the average of weights to equal to 1, so that it can easily be integrated into the learning algorithms. Proposed is a linear function forgetful (see section 3.1 for details of how it is utilised within the learning algorithms). This approach is presented in [23, 24]. The approach is implemented in a web-based recommending system [23, 43]. Experiment with it shows that this approach is very well integrating with a method for feature selection and it is able to improve the accuracy of prediction (see section 5.2).
- An algorithm that adapts to changing concepts that can reoccur presented in section 3.2 and publication [28, 26]. Algorithm "learns" in two steps: the first (preliminary) step "learn" the current context. Then, perform a test on the previous observations by looking episodes that are similar to the current context. So this step, separate episodes, where the concepts are the same or very close to the current and "forget" those which are different. Finally, the algorithm learns a predictive model only on selected relevant examples. The experiments with real data for planning of meetings show that the developed algorithm is able to study planning rules, which significantly improves the accuracy of prediction compared with similar approaches.

- The most resent approach that was developed is a self-adaptive optimization algorithm for learning changing concepts (described in section 3.3. This approach involves the use of statistical mechanism to detect changes in concepts and if it is found such ones, the algorithm optimizes the size of the time window (i.e. the "speed of forgetting") to make the learned model accurately as possible to describe the current concepts. Conducted experiments with four different sets of data shows that the developed approach significantly improves the accuracy of prediction than classical approaches [22].

The developed "forgetting" methods for learning of changing concepts are general in nature and can be used with any concept learning algorithms, also in different application areas.

In section 5.4 and [24] are described experiments aimed to study and compare the two of developed mechanism: the gradually forgetting and the optimization of the time window (i.e. the first and third of the above described). Furthermore the experimentally investigates the possibility those two approaches to be used together, complementing each other in this way to provide a new hybrid method. Four hypotheses are defined and tested. Experiments were conducted with three algorithms for machine learning (ID3, NBC, IBL) and using four different types of datasets. Results of the experiments show that:

- Implementing gradually forgetting within a fixed time window size usually leads to a significant improvement in the accuracy of learned classifiers of changing concepts;
- When comparing the two considered mechanism advantage is on the side of time window optimisation algorithm;
- Finally, there are not enough evidences to claim that when the both mechanisms are combined together will lead to further improvement in the accuracy of prediction of the learned classifier. Thus further experiments, with larger and different type of datasets, are needed to study this hybrid approach.

Some of the possible future works of this area are mentioned in the discussions and conclusions for the sections in this manuscript, in an appropriated context. Some more general ideas for future investigation that can be listed here are: further study of the possibilities to combine the developed approaches and development of new mechanism that are more specific for a particular class of learning algorithms and/or class of problems.

References

1. Abbasi and Chen H. Applying Authorship Analysis to Extremist-Group Web Forum Messages, *IEEE Computer Society*, september/october (2005), 67-75.
2. Aha, D., Kibler, D. and Albert, M. Instance-Based Learning Algorithms. *Machine Learning* **6** (1991), 37-66.
3. ALBERG, D., LAST, M., AND KANDEL, A. 2012. Knowledge discovery in data streams with regression treemethods. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 2, 69 – 78.
4. Allan J. Incremental Relevance Feedback for Information Retrieval. Proceedings of SIGIR'96 Zurich, Switzerland, ACM Press, (1996) pp. 270-278.
5. Baron, S. and Spiliopoulou, M. Temporal Evolution and Local Patterns. In Morik, K., Boulicaut, JF., and Siebes A. (Eds.): Local Pattern Detection, International Seminar, Dagstuhl Castle, Germany, April 12-16, 2004, Revised Selected Papers. *Lecture Notes in Computer Science* **3539**, Springer (2005).
6. Billsus, D., and Pazzani, M. J. (1999). A Hybrid User Model for News Classification. In *Kay J. (ed.), UM99 User Modeling Proceedings of the Seventh International Conference*, Springer-Verlag, Wien, New York, pp. 99-108.
7. Blum, Avrim. 1997. Empirical Support for Winnow and Weighted-MajorityAlgorithms: Results on a Calendar Scheduling Domain. *Mach. Learn.* 26, 1 (January 1997), 5-23.
8. Chu, F. and Zaniolo, C. Fast and Light Boosting for Adaptive Mining of Data Streams. In: Proc. of the 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining. *Lecture Notes in Computer Science*, Vol.**3056**, Springer-Verlag, (2004), 282-292.
9. Chiu B. C. and Webb G. I. (1998). Using Decision Trees For Agent Modelling: Improving Prediction Performance. *User Modeling and User-Adapted Interaction.* 8(1-2): 131-152. Netherlands: Springer.
10. Delany, SJ. and Cunningham, P., Tsymbal, A. and Coyle, L. A Case-Based Technique for Tracking Concept Drift in Spam Filtering. In: Macintosh, A., Ellis, R. & Allen T. (eds.) *Applications and Innovations in Intelligent Systems XII*, Proceedings of AI2004, Springer (2004), 3-16.

11. Ducatel, G. and Nürnberger, A. iArchive: An Assistant To Help Users Personalise Search Engines. In Chapter 2. User Profiles in Information Retrieva, Enhancing the Power of the Internet, Series: *Studies in Fuzziness and Soft Computing* , Vol. **139**, Nikravesh, M.; Azvine, B.; Yager, R.; Zadeh, L.A. (Eds.), VIII, Stinger-Verlag, (2004).
12. GABER, M. M., ZASLAVSKY, A., AND KRISHNASWAMY, S. 2005. Mining data streams: a review. SIGMOD Rec. 34, 18–26.
13. GAMA, J. 2010. Knowledge Discovery from Data Streams. Chapman & Hall/CRC.
14. Gama, J., Medas, P., Castillo, G. and Rodrigues, P. Learning with Drift Detection. In: Ana, C., Bazzan, S. and Labidi (Eds.): Proceedings of the 17th Brazilian Symposium on Artificial Intelligence. *Lecture Notes in Computer Science*, Vol. **3171**, Springer, (2004) 286-295.
15. Gama, João, et al. "A survey on concept drift adaptation." *ACM Computing Surveys (CSUR)* 46.4 (2014): 44.
16. Grabtree I. Soltysiak S. Identifying and Tracking Changing Interests. International Journal of *Digital Libraries*, Springer Verlag, vol. 2, (1998). 38-53.
17. Harries, M. and Sammut, C. Extracting Hidden Context. *Machine Learning* **32** (1998), 101-126.
18. Harries, M. Splice-2 comparative evaluation: Electricity pricing. Technical report, The University of South Wales, 1999.
19. KADLEC, P., GRBIC, R., AND GABRYS, B. 2011. Review of adaptation mechanisms for data-driven soft sensors. Computers & Chemical Engineering 35, 1, 1–24.
20. Klingenberg, R. and Renz, I. (1998): Adaptive information filtering: learning in the presence of concept drift. AAAI/ICML-98 Workshop on Learning for Text Categorization, TR WS-98-05, Madison, WI.
21. Klinkenberg, R. Learning Drifting Concepts: Example Selection vs. Example Weighting. In *Intelligent Data Analysis,* Special Issue on Incremental Learning Systems Capable of Dealing with Concept Drift, Vol. **8**, No. 3 (2004), 281-300.
22. Koychev, I. and Lothian, R. Tracking Drifting Concepts by Time Window Optimisation. In: *Research and Development in Intelligent Systems* **XXII** Proc. of AI-2005, the 25-th SGAI Int. Conference on Innovative Techniques and Applications of Artificial Intelligence. Bramer, Max; Coenen, Frans; Allen, Tony (Eds.), Springer-Verlag, London (2006), p.46-59.

23. Koychev, I. and Schwab, I. Adaptation to Drifting User's Interests. In proc. of ECML2000 Workshop: Machine Learning in New Information Age, Barcelona, Spain (2000), p. 39-46.

24. Koychev, I. Experiments with Two Approaches for Tracking Drifting Concepts. *Serdica Journal of Computing* **1** (2007), 27-44.

25. Koychev, I. Gradual Forgetting for Adaptation to Concept Drift. Proceedings of ECAI 2000 Workshop on Current Issues in Spatio-Temporal Reasoning, Berlin, (2000) 101-107.

26. Koychev, I. Tracking Changing User Interests through Prior-Learning of Context. In: P. de Bra, P. Brusilovsky and R. Conejo (eds.): Adaptive Hypermedia and Adaptive Web Based Systems. *Lecture Notes in Computer Science*, Vol. **2347**, Springer-Verlag (2002), 223-232.

27. Koychev, I. Approaches for Learning Classifiers of Drifting Concepts. NATO Science for Peace and Security Series - D: Information and Communication Security Volume 15, 2008 Security Informatics and Terrorism: Patrolling the Web. Eds.: Cecilia S. Gal, Paul B. Kantor, Bracha Shapira, IOS Press.

28. Koychev, I. (2001). Learning about User in the Presence of Hidden Context. Proceedings of UM-2001 Workshop: Machine Learning for User Modeling, pp. 49-58.

29. Kukar, M. Drifting Concepts as Hidden Factors in Clinical Studies. In Dojat, D., Elpida T. Keravnou, Pedro Barahona (Eds.): Proceedings of 9th Conference on Artificial Intelligence in Medicine in Europe, AIME 2003, Protaras, Cyprus, October 18-22, 2003, *Lecture Notes in Computer Science*, Vol. **2780**, Springer-Verlag (2003) 355-364.

30. KUNCHEVA, L. 2008. Classifier ensembles for detecting concept change in streaming data: Overview and perspectives. In Proc. of the 2nd Workshop SUEMA 2008.

31. Kuncheva, Ludmila I. "Change detection in streaming multivariate data using likelihood detectors." Knowledge and Data Engineering, IEEE Transactions on 25.5 (2013): 1175-1180.

32. Lazarescu, M., Venkatesh, S. and Bui, H. Using Multiple Windows to Track Concept Drift. In the *Intelligent Data Analysis Journal*, Vol **8** (1), (2004) 29-59.

33. Maloof, M. and Michalski, R. Selecting Examples for Partial Memory Learning, *Machine Learning* **41** (2000), 27-52.

34. MALOOF, M. A. 2010. The AQ methods for concept drift. In Advances in Machine Learning I: Dedicated to the Memory of Professor Ryszard S. Michalski. 23–47.

35. Mingers J. An Empirical Comparison of Selected Measures for Decision Tree Induction, Machine Learning 3, 319-142, Kluwer Academic Publishers, (1989).

36. Mitchell, T.M. Machine Learning. McGraw-Hill, 1997.

37. Mitchell, T., Caruana, R., Freitag, D., McDermott J. and Zabowski, D. Experience with a Learning Personal Assistant. *Communications of the ACM* **37(7)** (1994), 81-91.

38. MORENO-TORRES, J. G., RAEDER, T., ALAIZ-RODRIGUEZ, R., CHAWLA, N. V., AND HERRERA, F. 2012. Aunifying view on dataset shift in classification. Pattern Recognition 45, 1, 521 – 530.

39. Oommen, B. J., Yazidi, A., & Granmo, O. C. (2012). An Adaptive Approach to Learning the Preferences of Users in a Social Network Using Weak Estimators. Journal of Information Processing Systems, 8(2).

40. Quinlan, R. Induction of Decision Trees. *Machine Learning* **1** (1986), 81-106.

41. Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In Recommender systems handbook (pp. 257-297). Springer US.

42. Schlimmer, J. and Granger, R. Incremental Learning from Noisy Data. *Machine Learning* **3** (1986), 317-357.

43. Schwab, I., Kobsa, A. and Koychev, I. (2000). Learning about Users from Observation. Proceedings from the AAAI Spring Symposium: Adaptive User Interfaces, Menlo Park CA: AAAI Press.

44. Schwab I., Pohl W. and Koychev, I. (2000). Learning to Recommend from Positive Evidence, *Proceedings of Intelligent User Interfaces* 2000, ACM Press, pp.241-247.

45. TSYMBAL, A. 2004. The problem of concept drift: Definitions and related work. Technical report, Department of Computer Science, Trinity College: Dublin.

46. Webb, G. and Kuzmycz, M. (1996): Feature-based modelling: a methodology for producing coherent, consistent, dynamically changing models of agents' competencies. *User Modeling and User-Adapted Interaction* 5(2), 117-150.

47. Webb G. and Kuzmycz M. (1998). Evaluation Of Data Aging: A Technique For Discounting Old Data During Student Modeling. In B.P. Goettl, H. M. Halff, C. Redfield and V. Shute (Eds.), *Lecture Notes in Computer Science Vol. 1452: Proceedings of the Fourth International Conference on Intelligent Tutoring Systems (ITS '98)* San Antonio, Texas. Berlin: Springer-Verlag, pp. 384-393.

48. Widmer, G. and Kubat, M. Learning in the Presence of Concept Drift and Hidden Contexts, *Machine Learning* **23** (1996), 69-101.

49. Widmer, G. Tracking Changes through Meta-Learning. Machine Learning 27 (1997) 256-286.

50. Yampolskiy, R. V., & Govindaraju, V. (2008). Behavioural biometrics: a survey and classification. International Journal of Biometrics, 1(1), 81-113.

51. ZLIOBAITE, I. 2009. Learning under concept drift: an overview. Technical report, Vilnius University.