Open Challenges for Data Stream Mining Research

Georg Krempl
University Magdeburg, Germany
georg.krempl@iti.cs.uni-magdeburg.de

Indre Žliobaite
Aalto University and HIIT, Finland
indre.zliobaite@aalto.fi

Dariusz Brzeziński
Poznan U. of Technology, Poland
dariusz.brzezinski@cs.put.poznan.pl

Eyke Hüllermeier
University of Paderborn, Germany
eyke@upb.de

Mark Last
Ben-Gurion U. of the Negev, Israel
mlast@bgu.ac.il

Vincent Lemaire
Orange Labs, France
vincent.lemaire@orange.com

Tino Noack
TU Cottbus, Germany
noacktin@tu-cottbus.de

Ammar Shaker
University of Paderborn, Germany
ammar.shaker@upb.de

Sonja Sievi
Astrium Space Transportation, Germany
sonja.sievi@astrium.eads.net

Myra Spiliopoulou
University Magdeburg, Germany
myra@iti.cs.uni-magdeburg.de

Jerzy Stefanowski
Poznan U. of Technology, Poland
jerzy.stefanowski@cs.put.poznan.pl

ABSTRACT

Every day, huge volumes of sensory, transactional, and web data are continuously generated as streams, which need to be analyzed online as they arrive. Streaming data can be considered as one of the main sources of what is called big data. While predictive modeling for data streams and big data have received a lot of attention over the last decade, many research approaches are typically designed for well-behaved controlled problem settings, overlooking important challenges imposed by real-world applications. This article presents a discussion on eight open challenges for data stream mining. Our goal is to identify gaps between current research and meaningful applications, highlight open problems, and define new application-relevant research directions for data stream mining. The identified challenges cover the full cycle of knowledge discovery and involve such problems as: protecting data privacy, dealing with legacy systems, handling incomplete and delayed information, analysis of complex data, and evaluation of stream mining algorithms. The resulting analysis is illustrated by practical applications and provides general suggestions concerning lines of future research in data stream mining.

1. INTRODUCTION

The volumes of automatically generated data are constantly increasing. According to the Digital Universe Study [18], over 2.8ZB of data were created and processed in 2012, with a projected increase of 15 times by 2020. This growth in the production of digital data results from our surrounding environment being equipped with more and more sensors. People carrying smart phones produce data, database transactions are being counted and stored, streams of data are extracted from virtual environments in the form of logs or user generated content. A significant part of such data is volatile, which means it needs to be analyzed in real time as it arrives. Data stream mining is a research field that studies methods and algorithms for extracting knowledge from volatile streaming data [14; 5; 1]. Although data streams, online learning, big data, and adaptation to concept drift have become important research topics during the last decade, truly autonomous, self-maintaining, adaptive data mining systems are rarely reported. This paper identifies real-world challenges for data stream research that are important but yet unsolved. Our objective is to present to the community a position paper that could inspire and guide future research in data streams. This article builds upon discussions at the International Workshop on Real-World Challenges for Data Stream Mining (RealStream)[1] in September 2013, in Prague, Czech Republic.

Several related position papers are available. Dietterich [10] presents a discussion focused on predictive modeling techniques, that are applicable to streaming and non-streaming data. Fan and Bifet [12] concentrate on challenges presented by large volumes of data. Zliobaite et al. [48] focus on concept drift and adaptation of systems during online operation. Gaber et al. [13] discuss ubiquitous data mining with attention to collaborative data stream mining. In this paper, we focus on research challenges for streaming data inspired and required by real-world applications. In contrast to existing position papers, we raise issues connected not only with large volumes of data and concept drift, but also such practical problems as privacy constraints, availability of information, and dealing with legacy systems.

The scope of this paper is not restricted to algorithmic challenges, it aims at covering the full cycle of knowledge discovery from data (CRISP [40]), from understanding the context of the task, to data preparation, modeling, evaluation, and deployment. We discuss eight challenges: making models simpler, protecting privacy and confidentiality, dealing with legacy systems, stream preprocessing, timing and availability of information, relational stream mining, analyzing event data, and evaluation of stream mining algorithms. Figure 1 illustrates the positioning of these challenges in the CRISP cycle. Some of these apply to traditional (non-streaming) data mining as well, but they are critical in streaming environments. Along with further discussion of these challenges, we present our position where the forthcoming focus of research and development efforts should be directed to address these challenges.

In the remainder of the article, section 2 gives a brief introduction to data stream mining, sections 3–7 discuss each identified challenge, and section 8 highlights action points for future research.

[1] http://sites.google.com/site/realstream2013

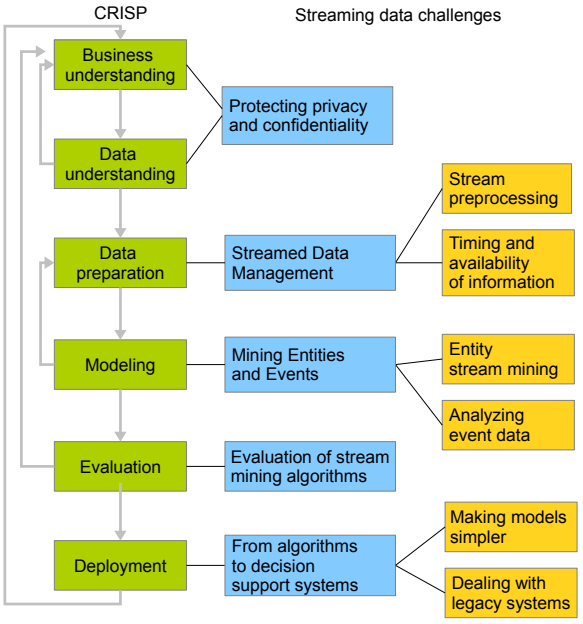

Figure 1: CRISP cycle with data stream research challenges.

2. DATA STREAM MINING

Mining big data streams faces three principal challenges: *volume*, *velocity*, and *volatility*. Volume and velocity require a high volume of data to be processed in limited time. Starting from the first arriving instance, the amount of available data constantly increases from zero to potentially infinity. This requires incremental approaches that incorporate information as it becomes available, and online processing if not all data can be kept [15]. Volatility, on the other hand, corresponds to a dynamic environment with ever-changing patterns. Here, old data is of limited use, even if it could be saved and processed again later. This is due to change, that can affect the induced data mining models in multiple ways: *change of the target variable*, *change in the available feature information*, and *drift*.

Changes of the target variable occur for example in credit scoring, when the definition of the classification target "default" versus "non-default" changes due to business or regulatory requirements. Changes in the available feature information arise when new features become available, e.g. due to a new sensor or instrument. Similarly, existing features might need to be excluded due to regulatory requirements, or a feature might change in its scale, if data from a more precise instrument becomes available. Finally, drift is a phenomenon that occurs when the distributions of features x and target variables y change in time. The challenge posed by drift has been subject to extensive research, thus we provide here solely a brief categorization and refer to recent surveys like [17].

In supervised learning, drift can affect the posterior $P(y|x)$, the conditional feature $P(x|y)$, the feature $P(x)$ and the class prior $P(y)$ distribution. The distinction based on which distribution is assumed to be affected, and which is assumed to be static, serves to assess the suitability of an approach for a particular task. It is worth noting, that the problem of changing distributions is also present in unsupervised learning from data streams.

A further categorization of drift can be made by:

- **smoothness of concept transition**: Transitions between concepts can be sudden or gradual. The former is sometimes also denoted in literature as shift or abrupt drift.
- **singular or recurring contexts**: In the former case, a model becomes obsolete once and for all when its context is replaced by a novel context. In the latter case, a model's context might reoccur at a later moment in time, for example due to a business cycle or seasonality, therefore, obsolete models might still regain value.
- **systematic or unsystematic**: In the former case, there are patterns in the way the distributions change that can be exploited to predict change and perform faster model adaptation. Examples are subpopulations that can be identified and show distinct, trackable evolutionary patterns. In the latter case, no such patterns exist and drift occurs seemingly at random. An example for the latter is fickle concept drift.
- **real or virtual**: While the former requires model adaptation, the latter corresponds to observing outliers or noise, which should not be incorporated into a model.

Stream mining approaches in general address the challenges posed by volume, velocity and volatility of data. However, in real-world applications these three challenges often coincide with other, to date insufficiently considered ones.

The next sections discuss eight identified challenges for data stream mining, providing illustrations with real world application examples, and formulating suggestions for forthcoming research.

3. PROTECTING PRIVACY AND CONFIDENTIALITY

Data streams present new challenges and opportunities with respect to protecting privacy and confidentiality in data mining. Privacy preserving data mining has been studied for over a decade (see. e.g. [3]). The main objective is to develop such data mining techniques that would not uncover information or patterns which compromise confidentiality and privacy obligations. Modeling can be done on original or anonymized data, but when the model is released, it should not contain information that may violate privacy or confidentiality. This is typically achieved by controlled distortion of sensitive data by modifying the values or adding noise.

Ensuring privacy and confidentiality is important for gaining trust of the users and the society in autonomous, stream data mining systems. While in offline data mining a human analyst working with the data can do a sanity check before releasing the model, in data stream mining privacy preservation needs to be done online. Several existing works relate to privacy preservation in publishing streaming data (e.g. [46]), but no systematic research in relation to broader data stream challenges exists.

We identify two main challenges for privacy preservation in mining data streams. The first challenge is incompleteness of information. Data arrives in portions and the model is updated online. Therefore, the model is never final and it is difficult to judge privacy preservation before seeing all the data. For example, suppose GPS traces of individuals are being collected for modeling traffic situation. Suppose person A at current time travels from the campus to the airport. The privacy of a person will be compromised, if there are no similar trips by other persons in the very near future. However, near future trips are unknown at the current time, when the model needs to be updated.

On the other hand, data stream mining algorithms may have some inherent privacy preservation properties due to the fact that they do not need to see all the modeling data at once, and can be incrementally updated with portions of data. Investigating privacy preservation properties of existing data stream algorithms makes another interesting direction for future research.

The second important challenge for privacy preservation is concept drift. As data may evolve over time, fixed privacy preservation rules may no longer hold. For example, suppose winter comes, snow falls, and much less people commute by bike. By knowing that a person comes to work by bike and having a set of GPS traces, it may not be possible to identify this person uniquely in summer, when there are many cyclists, but possible in winter. Hence, an important direction for future research is to develop adaptive privacy preservation mechanisms, that would diagnose such a situation and adapt themselves to preserve privacy in the new circumstances.

4. STREAMED DATA MANAGEMENT

Most of the data stream research concentrates on developing predictive models that address a simplified scenario, in which data is *already pre-processed*, *completely* and *immediately available* for *free*. However, successful business implementations depend strongly on the alignment of the used machine learning algorithms with both, the business objectives, and the available data. This section discusses often omitted challenges connected with streaming data.

4.1 Streamed Preprocessing

Data preprocessing is an important step in all real world data analysis applications, since data comes from complex environments, may be noisy, redundant, contain outliers and missing values. Many standard procedures for preprocessing offline data are available and well established, see e.g. [33]; however, the data stream setting introduces new challenges that have not received sufficient research attention yet.

While in traditional offline analysis data preprocessing is a once-off procedure, usually done by a human expert prior to modeling, in the streaming scenario manual processing is not feasible, as new data continuously arrives. Streaming data needs fully automated preprocessing methods, that can optimize the parameters and operate autonomously. Moreover, preprocessing models need to be able to update themselves automatically along with evolving data, in a similar way as predictive models for streaming data do. Furthermore, all updates of preprocessing procedures need to be synchronized with the subsequent predictive models, otherwise after an update in preprocessing the data representation may change and, as a result, the previously used predictive model may become useless.

Except for some studies, mainly focusing on feature construction over data streams, e.g. [49; 4], no systematic methodology for data stream preprocessing is currently available.

As an illustrative example for challenges related to data preprocessing, consider predicting traffic jams based on mobile sensing data. People using navigation services on mobile devices can opt to send anonymized data to the service provider. Service providers, such as Google, Yandex or Nokia, provide estimations and predictions of traffic jams based on this data. First, the data of each user is mapped to the road network, the speed of each user on each road segment of the trip is computed, data from multiple users is aggregated, and finally the current speed of the traffic is estimated.

There are a lot of data preprocessing challenges associated with this task. First, *noisiness* of GPS data might *vary* depending on location and load of the telecommunication network. There may be *outliers*, for instance, if somebody stopped in the middle of a segment to wait for a passenger, or a car broke. The number of pedestrians using mobile navigation may vary, and require *adaptive instance selection*. Moreover, road networks may change over time, leading to changes in average speeds, in the number of cars and even car types (e.g. heavy trucks might be banned, new optimal routes emerge). All these issues require automated preprocessing actions before feeding the newest data to the predictive models.

The problem of preprocessing for data streams is challenging due to the challenging nature of the data (continuously arriving and evolving). An analyst cannot know for sure, what kind of data to expect in the future, and cannot deterministically enumerate possible actions. Therefore, not only models, but also the procedure itself needs to be fully automated.

This research problem can be approached from several angles. One way is to look at existing predictive models for data streams, and try to integrate them with selected data preprocessing methods (e.g. feature selection, outlier definition and removal).

Another way is to systematically characterize the existing offline data preprocessing approaches, try to find a mapping between those approaches and problem settings in data streams, and extend preprocessing approaches for data streams in such a way as traditional predictive models have been extended for data stream settings.

In either case, developing individual methods and methodology for preprocessing of data streams would bridge an important gap in the practical applications of data stream mining.

4.2 Timing and Availability of Information

Most algorithms developed for evolving data streams make simplifying assumptions on the timing and availability of information. In particular, they assume that information is *complete*, *immediately available*, and *received passively and for free*. These assumptions often do not hold in real-world applications, e.g., patient monitoring, robot vision, or marketing [43]. This section is dedicated to the discussion of these assumptions and the challenges resulting from their absence. For some of these challenges, corresponding situations in offline, static data mining have already been addressed in literature. We will briefly point out where a mapping of such known solutions to the online, evolving stream setting is easily feasible, for example by applying windowing techniques. However, we will focus on problems for which no such simple mapping exists and which are therefore open challenges in stream mining.

4.2.1 Handling Incomplete Information

Completeness of information assumes that the true values of all variables, that is of features and of the target, are revealed eventually to the mining algorithm.

The problem of missing values, which corresponds to incompleteness of features, has been discussed extensively for the offline, static settings. A recent survey is given in [45]. However, only few works address data streams, and in particular *evolving* data streams. Thus several open challenges remain, some are pointed out in the review by [29]: how to address the problem that the frequency in which missing values occur is unpredictable, but largely affects the quality of imputations? How to (automatically) select the best imputation technique? How to proceed in the trade-off between speed and statistical accuracy?

Another problem is that of missing values of the target variable. It has been studied extensively in the static setting as semi-supervised learning (SSL, see [11]). A requirement for applying SSL techniques to streams is the availability of at least some labeled data from the most recent distribution. While first attempts to this problem have been made, e.g. the online manifold regularization approach in [19] and the ensembles-based approach suggested by [11], improvements in speed and the provision of performance guarantees remain open challenges. A special case of incomplete information is "censored data" in Event History Analysis (EHA), which is described in section 5.2. A related problem discussed below is active learning (AL, see [38]).

4.2.2 Dealing with Skewed Distributions

Class imbalance, where the class prior probability of the minority class is small compared to that of the majority class, is a frequent problem in real-world applications like fraud detection or credit scoring. This problem has been well studied in the offline setting (see e.g. [22] for a recent book on that subject), and has also been studied to some extent in the online, stream-based setting (see [23] for a recent survey). However, among the few existing stream-based approaches, most do not pay attention to drift of the minority class, and as [23] pointed out, a more rigorous evaluation of these algorithms on real-world data needs yet to be done.

4.2.3 Handling Delayed Information

Latency means information becomes available with significant delay. For example, in the case of so-called verification latency, the value of the preceding instance's target variable is not available before the subsequent instance has to be predicted. On *evolving* data streams, this is more than a mere problem of streaming data integration between feature and target streams, as due to concept drift patterns show temporal locality [2]. It means that feedback on the current prediction is not available to improve the subsequent predictions, but only eventually will become available for much later predictions. Thus, there is *no recent sample of labeled data at all* that would correspond to the most-recent unlabeled data, and semi-supervised learning approaches are not directly applicable.

A related problem in static, offline data mining is that addressed by unsupervised transductive transfer learning (or unsupervised domain adaptation): given labeled data from a source domain, a predictive model is sought for a related target domain in which no labeled data is available. In principle, ideas from transfer learning could be used to address latency in evolving data streams, for example by employing them in a chunk-based approach, as suggested in [43]. However, adapting them for use in evolving data streams has not been tried yet and constitutes a non-trivial, open task, as adaptation in streams must be fast and fully automated and thus cannot rely on iterated careful tuning by human experts.

Furthermore, consecutive chunks constitute several domains, thus the transitions between several subsequent chunks might provide exploitable patterns of systematic drift. This idea has been introduced in [27], and a few so-called *drift-mining* algorithms that identify and exploit such patterns have been proposed since then. However, the existing approaches cover only a very limited set of possible drift patterns and scenarios.

4.2.4 Active Selection from Costly Information

The challenge of intelligently selecting among costly pieces of information is the subject of active learning research. Active stream-based selective sampling [38] describes a scenario, in which instances arrive one-by-one. While the instances' feature vectors are provided for free, obtaining their true target values is costly, and the definitive decision whether or not to request this target value must be taken before proceeding to the next instance. This corresponds to a data stream, but *not necessarily* to an *evolving* one. As a result, only a small subset of stream-based selective sampling algorithms is suited for non-stationary environments. To make things worse, many contributions do not state explicitly whether they were designed for drift, neither do they provide experimental evaluations on such evolving data streams, thus leaving the reader the arduous task to assess their suitability for evolving streams. A first, recent attempt to provide an overview on the existing active learning strategies for evolving data streams is given in [43]. The challenges for active learning posed by evolving data streams are:

- **uncertainty regarding convergence**: in contrast to learning in static contexts, due to drift there is no guarantee that with additional labels the difference between model and reality narrows down. This leaves the formulation of suitable stop criteria a challenging open issue.
- **necessity of perpetual validation**: even if there has been convergence due to some temporary stability, the learned hypotheses can get invalidated at any time by subsequent drift. This can affect any part of the feature space and is not necessarily detectable from unlabeled data. Thus, without perpetual validation the mining algorithm might lock itself to a wrong hypothesis without ever noticing.
- **temporal budget allocation**: the necessity of perpetual validation raises the question of optimally allocating the labeling budget over time.
- **performance bounds**: in the case of drifting posteriors, no theoretical work exists that provides bounds for errors and label requests. However, deriving such bounds will also require assuming some type of systematic drift.

The task of active feature acquisition, where one has to actively select among costly features, constitutes another open challenge on evolving data streams: in contrast to the static, offline setting, the value of a feature is likely to change with its drifting distribution.

5. MINING ENTITIES AND EVENTS

Conventional stream mining algorithms learn over a single stream of arriving entities. In subsection 5.1, we introduce the paradigm of *entity stream mining*, where the entities constituting the stream are linked to instances (structured pieces of information) from further streams. Model learning in this paradigm involves the incorporation of the streaming information into the stream of entities; learning tasks include cluster evolution, migration of entities from one state to another, classifier adaptation as entities re-appear with another label than before.

Then, in subsection 5.2, we investigate the special case where entities are associated with the occurrence of events. Model learning then implies identifying the moment of occurrence of an event on an entity. This scenario might be seen as a special case of entity stream mining, since an event can be seen as a degenerate instance consisting of a single value (the event's occurrence).

5.1 Entity Stream Mining

Let T be a stream of entities, e.g. customers of a company or patients of a hospital. We observe entities over time, e.g. on a company's website or at a hospital admission vicinity: an entity appears and re-appears at discrete time points, new entities show up. At a time point t, an entity $e \in T$ is linked with different pieces of information - the purchases and ratings performed by a customer, the anamnesis, the medical tests and the diagnosis recorded for the patient. Each of these information pieces $i_j(t)$ is a structured record or an unstructured text from a stream T_j, linked to e via the foreign key relation. Thus, the entities in T are in 1-to-1 or 1-to-n relation with entities from further streams $T_1, \ldots, T_m$ (stream of purchases, stream of ratings, stream of complaints etc). The schema describing the streams $T, T_1, \ldots, T_m$ can be perceived as a conventional relational schema, except that it describes streams instead of static sets.

In this relational setting, the *entity stream mining* task corresponds to learning a model ζ_T over T, thereby incorporating information from the adjoint streams $T_1, \ldots, T_m$ that "feed" the entities in T.

Albeit the members of each stream are entities, we use the term "entity" only for stream T – the target of learning, while we denote the entities in the other streams as "instances". In the unsupervised setting, entity stream clustering encompasses learning and adapting clusters over T, taking account the other streams that arrive at different speeds. In the supervised setting, entity stream classification involves learning and adapting a classifier, notwithstanding the fact that an entity's label may change from one time point to the next, as new instances referencing it arrive.

5.1.1 Challenges of Aggregation

The first challenge of entity stream mining task concerns information summarization: how to aggregate into each entity e at each time point t the information available on it from the other streams? What information should be stored for each entity? How to deal with differences in the speeds of the individual streams? How to learn over the streams efficiently? Answering these questions in a seamless way would allow us to deploy conventional stream mining methods for entity stream mining after aggregation.

The information referencing a relational entity cannot be held perpetually for learning, hence aggregation of the arriving streams is necessary. Information aggregation over time-stamped data is traditionally practiced in document stream mining, where the objective is to derive and adapt content summaries on learned topics. Content summarization on entities, which are referenced in the document stream, is studied by Kotov et al., who maintain for each entity the number of times it is mentioned in the news [26].

In such studies, summarization is a task by itself. Aggregation of information for subsequent learning is a bit more challenging, because summarization implies information loss - notably information about the evolution of an entity. Hassani and Seidl monitor health parameters of patients, modeling the stream of recordings on a patient as a sequence of events [21]: the learning task is then to predict forthcoming values. Aggregation with selective forgetting of past information is proposed in [25; 42] in the classification context: the former method [25] slides a window over the stream, while the latter [42] forgets entities that have not appeared for a while, and summarizes the information in frequent itemsets, which are then used as new features for learning.

5.1.2 Challenges of Learning

Even if information aggregation over the streams $T_1, \ldots, T_m$ is performed intelligently, entity stream mining still calls for more than conventional stream mining methods. The reason is that entities of stream T re-appear in the stream and evolve. In particular, in the unsupervised setting, an entity may be linked to conceptually different instances at each time point, e.g. reflecting a customer's change in preferences. In the supervised setting, an entity may change its label; for example, a customer's affinity to risk may change in response to market changes or to changes in family status. This corresponds to *entity drift*, i.e. a new type of drift beyond the conventional concept drift pertaining to model ζ_T. Hence, how should entity drift be traced, and how should the interplay between entity drift and model drift be captured?

In the unsupervised setting, Oliveira and Gama learn and monitor clusters as *states* of evolution [32], while [41] extend that work to learn Markov chains that mark the entities' evolution. As pointed out in [32], these states are not necessarily predefined – they must be subject of learning. In [43], we report on further solutions to the entity evolution problem and to the problem of learning with forgetting over multiple streams and over the entities referenced by them.

Conventional concept drift also occurs when learning a model over entities, thus the challenges pertinent to stream mining also apply here. One of these challenges, and one much discussed in the context of big data, is *volatility*. In relational stream mining, volatility refers to the entity itself, not only to the stream of instances that reference the entities. Finally, an entity is ultimately *big data* by itself, since it is described by multiple streams. Hence, next to the problem of dealing with new forms of learning and new aspects of drift, the subject of efficient learning and adaption in the Big Data context becomes paramount.

5.2 Analyzing Event Data

Events are an example for data that occurs often yet is rarely analyzed in the stream setting. In static environments, events are usually studied through *event history analysis* (EHA), a statistical method for modeling and analyzing the temporal distribution of events related to specific objects in the course of their lifetime [9]. More specifically, EHA is interested in the duration before the occurrence of an event or, in the *recurrent* case (where the same event can occur repeatedly), the duration between two events. The notion of an *event* is completely generic and may indicate, for example, the failure of an electrical device. The method is perhaps even better known as *survival analysis*, a term that originates from applications in medicine, in which an event is the death of a patient and *survival time* is the time period between the beginning of the study and the occurrence of this event. EHA can also be considered as a special case of entity stream mining described in section 5.1, because the basic statistical entities in EHA are monitored objects (or subjects), typically described in terms of feature vectors $\boldsymbol{x} \in \mathbb{R}^n$, together with their survival time s. Then, the goal is to model the dependence of s on $\boldsymbol{x}$. A corresponding model provides hints at possible cause-effect relationships (e.g., what properties tend to increase a patient's survival time) and, moreover, can be used for predictive purposes (e.g., what is the expected survival time of a patient).

Although one might be tempted to approach this modeling task as a standard regression problem with input (regressor) $\boldsymbol{x}$ and output (response) s, it is important to notice that the survival time s is normally not observed for all objects. Indeed, the problem of *censoring* plays an important role in EHA and occurs in different facets. In particular, it may happen that some of the objects survived till the end of the study at time t_{end} (also called the *cut-off point*). They are censored or, more specifically, *right censored*, since t_{event} has not been observed for them; instead, it is only known that $t_{event} > t_{end}$. In *snapshot monitoring* [28], the data stream may be sampled multiple times, resulting in a new cut-off point for each snapshot. Unlike standard regression analysis, EHA is specifically tailored for analyzing event data of that kind. It is built upon the *hazard function* as a basic mathematical tool.

5.2.1 Survival function and hazard rate

Suppose the time of occurrence of the next event (since the start or the last event) for an object $\boldsymbol{x}$ is modeled as a real-valued random variable T with probability density function $f(\cdot \,|\, \boldsymbol{x})$. The hazard function or hazard rate $h(\cdot \,|\, \boldsymbol{x})$ models the *propensity* of the occurrence of an event, that is, the marginal probability of an event to occur at time t, given that no event has occurred so far:

$$h(t \,|\, \boldsymbol{x}) = \frac{f(t \,|\, \boldsymbol{x})}{S(t \,|\, \boldsymbol{x})} = \frac{f(t \,|\, \boldsymbol{x})}{1 - F(t \,|\, \boldsymbol{x})} \ ,$$

where $S(\cdot \,|\, \boldsymbol{x})$ is the survival function and $F(\cdot \,|\, \boldsymbol{x})$ the cumulative distribution of $f(\cdot \,|\, \boldsymbol{x})$. Thus,

$$F(t \,|\, \boldsymbol{x}) \ = \ \mathbf{P}(T \leq t) = \int_0^t f(u \,|\, \boldsymbol{x}) \, du$$

is the probability of an event to occur before time t. Correspondingly, $S(t \mid \boldsymbol{x}) = 1 - F(t \mid \boldsymbol{x})$ is the probability that the event did not occur until time t (the survival probability). It can hence be used to model the probability of the right-censoring of the time for an event to occur.

A simple example is the Cox proportional hazard model [9], in which the hazard rate is constant over time; thus, it does depend on the feature vector $\boldsymbol{x} = (x_1, \ldots, x_n)$ but not on time t. More specifically, the hazard rate is modeled as a log-linear function of the features x_i:

$$h(t \mid \boldsymbol{x}) = \lambda(\boldsymbol{x}) = \exp\left(\boldsymbol{x}^\top \beta\right)$$

The model is proportional in the sense that increasing x_i by one unit increases the hazard rate $\lambda(\boldsymbol{x})$ by a factor of $\alpha_i = \exp(\beta_i)$. For this model, one easily derives the survival function $S(t \mid \boldsymbol{x}) = 1 - \exp(-\lambda(\boldsymbol{x}) \cdot t)$ and an expected survival time of $1/\lambda(\boldsymbol{x})$.

5.2.2 EHA on data streams

Although the temporal nature of event data naturally fits the data stream model and, moreover, event data is naturally produced by many data sources, EHA has been considered in the data stream scenario only very recently. In [39], the authors propose a method for analyzing earthquake and Twitter data, namely an extension of the above Cox model based on a sliding window approach. The authors of [28] modify standard classification algorithms, such as decision trees, so that they can be trained on a snapshot stream of both censored and non-censored data.

Like in the case of clustering [35], where one distinguishes between clustering observations and clustering data sources, two different settings can be envisioned for EHA on data streams:

1. In the first setting, events are generated by multiple data sources (representing monitored objects), and the features pertain to these sources; thus, each data source is characterized by a feature vector $\boldsymbol{x}$ and produces a stream of (recurrent) events. For example, data sources could be users in a computer network, and an event occurs whenever a user sends an email.

2. In the second setting, events are produced by a single data source, but now the events themselves are characterized by features. For example, events might be emails sent by an email server, and each email is represented by a certain set of properties.

Statistical event models on data streams can be used in much the same way as in the case of static data. For example, they can serve predictive purposes, i.e., to answer questions such as "How much time will elapse before the next email arrives?" or "What is the probability to receive more than 100 emails within the next hour?". What is specifically interesting, however, and indeed distinguishes the data stream setting from the static case, is the fact that the model may change over time. This is a subtle aspect, because the hazard model $h(t \mid \boldsymbol{x})$ itself may already be time-dependent; here, however, t is not the absolute time but the duration time, i.e., the time elapsed since the last event. A change of the model is comparable to concept drift in classification, and means that the way in which the hazard rate depends on time t and on the features x_i changes over time. For example, consider the event "increase of a stock rate" and suppose that $\beta_i = \log(2)$ for the binary feature $x_i = \texttt{energy sector}$ in the above Cox model (which, as already mentioned, does not depend on t). Thus, this feature doubles the hazard rate and hence halves the expected duration between two events. Needless to say, however, this influence may change over time, depending on how well the energy sector is doing.

Dealing with model changes of that kind is clearly an important challenge for event analysis on data streams. Although the problem is to some extent addressed by the works mentioned above, there is certainly scope for further improvement, and for using these approaches to derive predictive models from censored data. Besides, there are many other directions for future work. For example, since the *detection* of events is a main prerequisite for analyzing them, the combination of EHA with methods for *event detection* [36] is an important challenge. Indeed, this problem is often far from trivial, and in many cases, events (such as frauds, for example) can only be detected with a certain time delay; dealing with delayed events is therefore another important topic, which was also discussed in section 4.2.

6. EVALUATION OF DATA STREAM ALGORITHMS

All of the aforementioned challenges are milestones on the road to better algorithms for real-world data stream mining systems. To verify if these challenges are met, practitioners need tools capable of evaluating newly proposed solutions. Although in the field of static classification such tools exist, they are insufficient in data stream environments due to such problems as: concept drift, limited processing time, verification latency, multiple stream structures, evolving class skew, censored data, and changing misclassification costs. In fact, the myriad of additional complexities posed by data streams makes algorithm evaluation a highly multi-criterial task, in which optimal trade-offs may change over time.

Recent developments in applied machine learning [6] emphasize the importance of understanding the data one is working with and using evaluation metrics which reflect its difficulties. As mentioned before, data streams set new requirements compared to traditional data mining and researchers are beginning to acknowledge the shortcomings of existing evaluation metrics. For example, Gama et al. [16] proposed a way of calculating classification accuracy using only the most recent stream examples, therefore allowing for time-oriented evaluation and aiding concept drift detection. Methods which test the classifier's robustness to drifts and noise on a practical, experimental level are also starting to arise [34; 47]. However, all these evaluation techniques focus on single criteria such as prediction accuracy or robustness to drifts, even though data streams make evaluation a constant trade-off between several criteria [7]. Moreover, in data stream environments there is a need for more advanced tools for visualizing changes in algorithm predictions with time.

The problem of creating complex evaluation methods for stream mining algorithms lies mainly in the size and evolving nature of data streams. It is much more difficult to estimate and visualize, for example, prediction accuracy if evaluation must be done online, using limited resources, and the classification task changes with time. In fact, the algorithm's ability to adapt is another aspect which needs to be evaluated, although information needed to perform such evaluation is not always available. Concept drifts are known in advance mainly when using synthetic or benchmark data, while in more practical scenarios occurrences and types of concepts are not directly known and only the label of each arriving instance is known. Moreover, in many cases the task is more complicated, as labeling information is not instantly available. Other difficulties in evaluation include processing complex relational streams and coping with class imbalance when class distributions evolve with time. Finally, not only do we need measures for evaluating single aspects of stream mining algorithms, but also ways of combining several of these aspects into global evaluation models, which would take into

account expert knowledge and user preferences.

Clearly, evaluation of data stream algorithms is a fertile ground for novel theoretical and algorithmic solutions. In terms of prediction measures, data stream mining still requires evaluation tools that would be immune to class imbalance and robust to noise. In our opinion, solutions to this problem should involve not only metrics based on relative performance to baseline (chance) classifiers, but also graphical measures similar to PR-curves or cost curves. Furthermore, there is a need for integrating information about concept drifts in the evaluation process. As mentioned earlier, possible ways of considering concept drifts will depend on the information that is available. If true concepts are known, algorithms could be evaluated based on: how often they detect drift, how early they detect it, how they react to it, and how quickly they recover from it. Moreover, in this scenario, evaluation of an algorithm should be dependent on whether it takes place during drift or during times of concept stability. A possible way of tackling this problem would be the proposal of graphical methods, similar to ROC analysis, which would work online and visualize concept drift measures alongside prediction measures. Additionally, these graphical measures could take into account the state of the stream, for example, its speed, number of missing values, or class distribution. Similar methods could be proposed for scenarios where concepts are not known in advance, however, in these cases measures should be based on drift detectors or label-independent stream statistics. Above all, due to the number of aspects which need to be measured, we believe that the evaluation of data stream algorithms requires a multi-criterial view. This could be done by using inspirations from multiple criteria decision analysis, where trade-offs between criteria are achieved using user-feedback. In particular, a user could showcase his/her criteria preferences (for example, between memory consumption, accuracy, reactivity, self-tuning, and adaptability) by deciding between alternative algorithms for a given data stream. It is worth noticing that such a multi-criterial view on evaluation is difficult to encapsulate in a single number, as it is usually done in traditional offline learning. This might suggest that researchers in this area should turn towards semi-qualitative and semi-quantitative evaluation, for which systematic methodologies should be developed.

Finally, a separate research direction involves rethinking the way we test data stream mining algorithms. The traditional train, cross-validate, test workflow in classification is not applicable for sequential data, which makes, for instance, parameter tuning much more difficult. Similarly, ground truth verification in unsupervised learning is practically impossible in data stream environments. With these problems in mind, it is worth stating that there is still a shortage of real and synthetic benchmark datasets. Such a situation might be a result of non-uniform standards for testing algorithms on streaming data. As community, we should decide on such matters as: What characteristics should benchmark datasets have? Should they have prediction tasks attached? Should we move towards online evaluation tools rather than datasets? These questions should be answered in order to solve evaluation issues in controlled environments before we create measures for real-world scenarios.

7. FROM ALGORITHMS TO DECISION SUPPORT SYSTEMS

While a lot of algorithmic methods for data streams are already available, their deployment in real applications with real streaming data presents a new dimension of challenges. This section points out two such challenges: making models simpler and dealing with legacy systems.

7.1 Making models simpler, more reactive, and more specialized

In this subsection, we discuss aspects like the simplicity of a model, its proper combination of offline and online components, and its customization to the requirements of the application domain. As an application example, consider the French Orange Portal[2], which registers millions of visits daily. Most of these visitors are only known through anonymous cookie IDs. For all of these visitors, the portal has the ambition to provide specific and relevant contents as well as printing ads for targeted audiences. Using information about visits on the portal the questions are: what part of the portal does each cookie visit, and when and which contents did it consult, what advertisement was sent, when (if) was it clicked. All this information generates hundreds of gigabytes of data each week. A user profiling system needs to have a back end part to preprocess the information required at the input of a front end part, which will compute appetency to advertising (for example) using stream mining techniques (in this case a supervised classifier). Since the ads to print change regularly, based on marketing campaigns, the extensive parameter tuning is infeasible as one has to react quickly to change. Currently, these tasks are either solved using bandit methods from game theory [8], which impairs adaptation to drift, or done offline in big data systems, resulting in slow reactivity.

7.1.1 Minimizing parameter dependence

Adaptive predictive systems are intrinsically parametrized. In most of the cases, setting these parameters, or tuning them is a difficult task, which in turn negatively affects the usability of these systems. Therefore, it is strongly desired for the system to have as few user adjustable parameters as possible. Unfortunately, the state of the art does not produce methods with trustworthy or easily adjustable parameters. Moreover, many predictive modeling methods use a lot of parameters, rendering them particularly impractical for data stream applications, where models are allowed to evolve over time, and input parameters often need to evolve as well.

The process of predictive modeling encompasses fitting of parameters on a training dataset and subsequently selecting the best model, either by heuristics or principled methods. Recently, model selection methods have been proposed that do not require internal cross-validation, but rather use the Bayesian machinery to design regularizers with data dependent priors [20]. However, they are not yet applicable in data streams, as their computational time complexity is too high and they require all examples to be kept in memory.

7.1.2 Combining offline and online models

Online and offline learning are mostly considered as mutually exclusive, but it is their combination that might enhance the value of data the most. Online learning, which processes instances one-by-one and builds models incrementally, has the virtue of being fast, both in the processing of data and in the adaptation of models. Offline (or batch) learning has the advantage of allowing the use of more sophisticated mining techniques, which might be more time-consuming or require a human expert. While the first allows the processing of "*fast* data" that requires real-time processing and adaptivity, the second allows processing of "*big* data" that requires longer processing time and larger abstraction.

Their combination can take place in many steps of the mining process, such as the data preparation and the preprocessing steps. For example, offline learning on big data could extract fundamental and sustainable trends from data using batch processing and massive parallelism. Online learning could then take real-time decisions

[2] www.orange.fr

from online events to optimize an immediate pay-off. In the online advertisement application mentioned above, the user-click prediction is done within a context, defined for example by the currently viewed page and the profile of the cookie. The decision which banner to display is done online, but the context can be preprocessed offline. By deriving meta-information such as "the profile is a young male, the page is from the sport cluster", the offline component can ease the online decision task.

7.1.3 Solving the right problem

Domain knowledge may help to solve many issues raised in this paper, by systematically exploiting particularities of application domains. However, this is seldom considered, as typical data stream methods are created to deal with a large variety of domains. For instance, in some domains the learning algorithm receives only partial feedback upon its prediction, i.e. a single bit of right-or-wrong, rather than the true label. In the user-click prediction example, if a user does not click on a banner, we do not know which one would have been correct, but solely that the displayed one was wrong. This is related to the issues on timing and availability of information discussed in section 4.2.

However, building predictive models that systematically incorporate domain knowledge or domain specific information requires to choose the right optimization criteria. As mentioned in section 6, the data stream setting requires optimizing multiple criteria simultaneously, as optimizing only predictive performance is not sufficient. We need to develop learning algorithms, which minimize an objective function including intrinsically and simultaneously: memory consumption, predictive performance, reactivity, self monitoring and tuning, and (explainable) auto-adaptivity. Data streams research is lacking methodologies for forming and optimizing such criteria.

Therefore, models should be simple so that they do not depend on a set of carefully tuned parameters. Additionally, they should combine offline and online techniques to address challenges of big and fast data, and they should solve the right problem, which might consist in solving a multi-criteria optimization task. Finally, they have to be able to learn from a small amount of data and with low variance [37], to react quickly to drift.

7.2 Dealing with Legacy Systems

In many application environments, such as financial services or health care systems, business critical applications are in operation for decades. Since these applications produce massive amounts of data, it becomes very promising to process these amounts of data by real-time stream mining approaches. However, it is often impossible to change existing infrastructures in order to introduce fully fledged stream mining systems. Rather than changing existing infrastructures, approaches are required that integrate stream mining techniques into legacy systems. In general, problems concerning legacy systems are domain-specific and encompass both technical and procedural issues. In this section, we analyze challenges posed by a specific real-world application with legacy issues — the ISS Columbus spacecraft module.

7.2.1 ISS Columbus

Spacecrafts are very complex systems, exposed to very different physical environments (e.g. space), and associated to ground stations. These systems are under constant and remote monitoring by means of telemetry and commands. The ISS Columbus module has been in operation for more than 5 years. For some time, it is pointed out that the monitoring process is not as efficient as previously expected [30]. However, we assume that data stream mining can make a decisive contribution to enhance and facilitate the required monitoring tasks. Recently, we are planning to use the ISS Columbus module as a technology demonstrator for integrating data stream processing and mining into the existing monitoring processes [31]. Figure 2 exemplifies the failure management system (FMS) of the ISS Columbus module. While it is impossible to simply redesign the FMS from scratch, we can outline the following challenges.

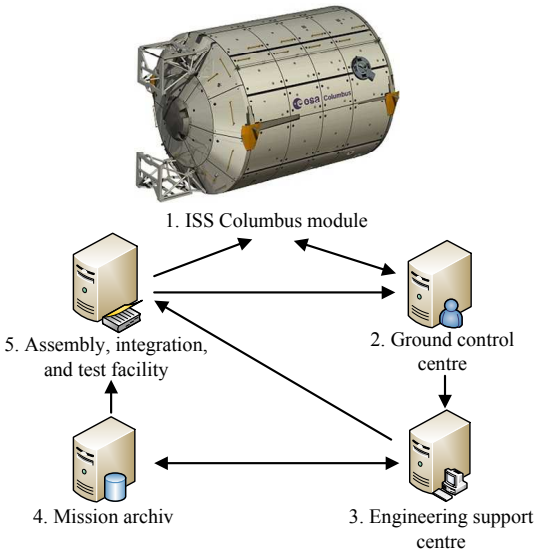

Figure 2: ISS Columbus FMS

7.2.2 Complexity

Even though spacecraft monitoring is very challenging by itself, it becomes increasingly difficult and complex due to the integration of data stream mining into such legacy systems. However, it was assumed to enhance and facilitate current monitoring processes. Thus, appropriate mechanism are required to integrate data stream mining into the current processes to decrease complexity.

7.2.3 Interlocking

As depicted in Figure 2, the ISS Columbus module is connected to ground instances. Real-time monitoring must be applied aboard where computational resources are restricted (e.g. processor speed and memory or power consumption). Near real-time monitoring or long-term analysis must be applied on-ground where the downlink suffers from latencies because of a long transmission distance, is subject to bandwidth limitations, and continuously interrupted due to loss of signal. Consequently, new data stream mining mechanisms are necessary which ensure a smooth interlocking functionality of aboard and ground instances.

7.2.4 Reliability and Balance

The reliability of spacecrafts is indispensable for astronauts' health and mission success. Accordingly, spacecrafts pass very long and expensive planning and testing phases. Hence, potential data stream mining algorithms must ensure reliability and the integration of such algorithms into legacy systems must not cause critical side effects. Furthermore, data stream mining is an automatic process which neglects interactions with human experts, while spacecraft monitoring is a semi-automatic process and human experts (e.g.

the flight control team) are responsible for decisions and consequent actions. This problem poses the following question: How to integrate data stream mining into legacy systems when automation needs to be increased but the human expert needs to be maintained in the loop? Abstract discussions on this topic are provided by expert systems [44] and the MAPE-K reference model [24]. Expert systems aim to combine human expertise with artificial expertise and the MAPE-K reference model aims to provide an autonomic control loop. A balance must be struck which considers both aforementioned aspects appropriately.

Overall, the Columbus study has shown that extending legacy systems with real time data stream mining technologies is feasible and it is an important area for further stream-mining research.

8. CONCLUDING REMARKS

In this paper, we discussed research challenges for data streams, originating from real-world applications. We analyzed issues concerning privacy, availability of information, relational and event streams, preprocessing, model complexity, evaluation, and legacy systems. The discussed issues were illustrated by practical applications including GPS systems, Twitter analysis, earthquake predictions, customer profiling, and spacecraft monitoring. The study of real-world problems highlighted shortcomings of existing methodologies and showcased previously unaddressed research issues.

Consequently, we call the data stream mining community to consider the following action points for data stream research:

- developing methods for ensuring privacy with incomplete information as data arrives, while taking into account the evolving nature of data;
- considering the availability of information by developing models that handle incomplete, delayed and/or costly feedback;
- taking advantage of relations between streaming entities;
- developing event detection methods and predictive models for censored data;
- developing a systematic methodology for streamed preprocessing;
- creating simpler models through multi-objective optimization criteria, which consider not only accuracy, but also computational resources, diagnostics, reactivity, interpretability;
- establishing a multi-criteria view towards evaluation, dealing with absence of the ground truth about how data changes;
- developing online monitoring systems, ensuring reliability of any updates, and balancing the distribution of resources.

As our study shows, there are challenges in every step of the CRISP data mining process. To date, modeling over data streams has been viewed and approached as an extension of traditional methods. However, our discussion and application examples show that in many cases it would be beneficial to step aside from building upon existing offline approaches, and start blank considering what is required in the stream setting.

Acknowledgments

We would like to thank the participants of the RealStream2013 workshop at ECMLPKDD2013 in Prague, and in particular Bernhard Pfahringer and George Forman, for suggestions and discussions on the challenges in stream mining. Part of this work was funded by the German Research Foundation, projects SP 572/11-1 (IMPRINT) and HU 1284/5-1, the Academy of Finland grant 118653 (ALGODAN), and the Polish National Science Center grants DEC-2011/03/N/ST6/00360 and DEC-2013/11/B/ST6/00963.

9. REFERENCES

[1] C. Aggarwal, editor. *Data Streams: Models and Algorithms*. Springer, 2007.

[2] C. Aggarwal and D. Turaga. Mining data streams: Systems and algorithms. In *Machine Learning and Knowledge Discovery for Engineering Systems Health Management*, pages 4–32. Chapman and Hall, 2012.

[3] R. Agrawal and R. Srikant. Privacy-preserving data mining. *SIGMOD Rec.*, 29(2):439–450, 2000.

[4] C. Anagnostopoulos, N. Adams, and D. Hand. Deciding what to observe next: Adaptive variable selection for regression in multivariate data streams. In *Proc. of the 2008 ACM Symp. on Applied Computing*, SAC, pages 961–965, 2008.

[5] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *Proc. of the 21st ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, PODS, pages 1–16, 2002.

[6] C. Brodley, U. Rebbapragada, K. Small, and B. Wallace. Challenges and opportunities in applied machine learning. *AI Magazine*, 33(1):11–24, 2012.

[7] D. Brzezinski and J. Stefanowski. Reacting to different types of concept drift: The accuracy updated ensemble algorithm. *IEEE Trans. on Neural Networks and Learning Systems.*, 25:81–94, 2014.

[8] D. Chakrabarti, R. Kumar, F. Radlinski, and E. Upfal. Mortal multi-armed bandits. In *Proc. of the 22nd Conf. on Neural Information Processing Systems*, NIPS, pages 273–280, 2008.

[9] D. Cox and D. Oakes. *Analysis of Survival Data*. Chapman & Hall, London, 1984.

[10] T. Dietterich. Machine-learning research. *AI Magazine*, 18(4):97–136, 1997.

[11] G. Ditzler and R. Polikar. Semi-supervised learning in non-stationary environments. In *Proc. of the 2011 Int. Joint Conf. on Neural Networks*, IJCNN, pages 2741 – 2748, 2011.

[12] W. Fan and A. Bifet. Mining big data: current status, and forecast to the future. *SIGKDD Explorations*, 14(2):1–5, 2012.

[13] M. Gaber, J. Gama, S. Krishnaswamy, J. Gomes, and F. Stahl. Data stream mining in ubiquitous environments: state-of-the-art and current directions. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(2):116 – 138, 2014.

[14] M. Gaber, A. Zaslavsky, and S. Krishnaswamy. Mining data streams: A review. *SIGMOD Rec.*, 34(2):18–26, 2005.

[15] J. Gama. *Knowledge Discovery from Data Streams*. Chapman & Hall/CRC, 2010.

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

[17] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy, and A. Bouchachia. A survey on concept-drift adaptation. *ACM Computing Surveys*, 46(4), 2014.

[18] J. Gantz and D. Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east, December 2012.

[19] A. Goldberg, M. Li, and X. Zhu. Online manifold regularization: A new learning setting and empirical study. In *Proc. of the European Conf. on Machine Learning and Principles of Knowledge Discovery in Databases*, ECMLPKDD, pages 393–407, 2008.

[20] I. Guyon, A. Saffari, G. Dror, and G. Cawley. Model selection: Beyond the bayesian/frequentist divide. *Journal of Machine Learning Research*, 11:61–87, 2010.

[21] M. Hassani and T. Seidl. Towards a mobile health context prediction: Sequential pattern mining in multiple streams. In *Proc. of , IEEE Int. Conf. on Mobile Data Management*, MDM, pages 55–57, 2011.

[22] H. He and Y. Ma, editors. *Imbalanced Learning: Foundations, Algorithms, and Applications*. IEEE, 2013.

[23] T. Hoens, R. Polikar, and N. Chawla. Learning from streaming data with concept drift and imbalance: an overview. *Progress in Artificial Intelligence*, 1(1):89–101, 2012.

[24] IBM. An architectural blueprint for autonomic computing. Technical report, IBM, 2003.

[25] E. Ikonomovska, K. Driessens, S. Dzeroski, and J. Gama. Adaptive windowing for online learning from multiple interrelated data streams. In *Proc. of the 11th IEEE Int. Conf. on Data Mining Workshops*, ICDMW, pages 697–704, 2011.

[26] A. Kotov, C. Zhai, and R. Sproat. Mining named entities with temporally correlated bursts from multilingual web news streams. In *Proc. of the 4th ACM Int. Conf. on Web Search and Data Mining*, WSDM, pages 237–246, 2011.

[27] G. Krempl. The algorithm APT to classify in concurrence of latency and drift. In *Proc. of the 10th Int. Conf. on Advances in Intelligent Data Analysis*, IDA, pages 222–233, 2011.

[28] M. Last and H. Halpert. Survival analysis meets data stream mining. In *Proc. of the 1st Worksh. on Real-World Challenges for Data Stream Mining*, RealStream, pages 26–29, 2013.

[29] F. Nelwamondo and T. Marwala. Key issues on computational intelligence techniques for missing data imputation - a review. In *Proc. of World Multi Conf. on Systemics, Cybernetics and Informatics*, volume 4, pages 35–40, 2008.

[30] E. Noack, W. Belau, R. Wohlgemuth, R. Müller, S. Palumberi, P. Parodi, and F. Burzagli. Efficiency of the columbus failure management system. In *Proc. of the AIAA 40th Int. Conf. on Environmental Systems*, 2010.

[31] E. Noack, A. Luedtke, I. Schmitt, T. Noack, E. Schaumlöffel, E. Hauke, J. Stamminger, and E. Frisk. The columbus module as a technology demonstrator for innovative failure management. In *German Air and Space Travel Congress*, 2012.

[32] M. Oliveira and J. Gama. A framework to monitor clusters evolution applied to economy and finance problems. *Intelligent Data Analysis*, 16(1):93–111, 2012.

[33] D. Pyle. *Data Preparation for Data Mining*. Morgan Kaufmann Publishers Inc., 1999.

[34] T. Raeder and N. Chawla. Model monitor (m^2): Evaluating, comparing, and monitoring models. *Journal of Machine Learning Research*, 10:1387–1390, 2009.

[35] P. Rodrigues and J. Gama. Distributed clustering of ubiquitous data streams. *WIREs Data Mining and Knowledge Discovery*, pages 38–54, 2013.

[36] T. Sakaki, M. Okazaki, and Y. Matsuo. Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Trans. on Knowledge and Data Engineering*, 25(4):919–931, 2013.

[37] C. Salperwyck and V. Lemaire. Learning with few examples: An empirical study on leading classifiers. In *Proc. of the 2011 Int. Joint Conf. on Neural Networks*, IJCNN, pages 1010–1019, 2011.

[38] B. Settles. *Active Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan and Claypool Publishers, 2012.

[39] A. Shaker and E. Hüllermeier. Survival analysis on data streams: Analyzing temporal events in dynamically changing environments. *Int. Journal of Applied Mathematics and Computer Science*, 24(1):199–212, 2014.

[40] C. Shearer. The CRISP-DM model: the new blueprint for data mining. *J Data Warehousing*, 2000.

[41] Z. Siddiqui, M. Oliveira, J. Gama, and M. Spiliopoulou. Where are we going? predicting the evolution of individuals. In *Proc. of the 11th Int. Conf. on Advances in Intelligent Data Analysis*, IDA, pages 357–368, 2012.

[42] Z. Siddiqui and M. Spiliopoulou. Classification rule mining for a stream of perennial objects. In *Proc. of the 5th Int. Conf. on Rule-based Reasoning, Programming, and Applications*, RuleML, pages 281–296, 2011.

[43] M. Spiliopoulou and G. Krempl. Tutorial "mining multiple threads of streaming data". In *Proc. of the Pacific-Asia Conf. on Knowledge Discovery and Data Mining*, PAKDD, 2013.

[44] D. Waterman. *A Guide to Expert Systems*. Addison-Wesley, 1986.

[45] W. Young, G. Weckman, and W. Holland. A survey of methodologies for the treatment of missing values within datasets: limitations and benefits. *Theoretical Issues in Ergonomics Science*, 12, January 2011.

[46] B. Zhou, Y. Han, J. Pei, B. Jiang, Y. Tao, and Y. Jia. Continuous privacy preserving publishing of data streams. In *Proc. of the 12th Int. Conf. on Extending Database Technology*, EDBT, pages 648–659, 2009.

[47] I. Zliobaite. Controlled permutations for testing adaptive learning models. *Knowledge and Information Systems*, In Press, 2014.

[48] I. Zliobaite, A. Bifet, M. Gaber, B. Gabrys, J. Gama, L. Minku, and K. Musial. Next challenges for adaptive learning systems. *SIGKDD Explorations*, 14(1):48–55, 2012.

[49] I. Zliobaite and B. Gabrys. Adaptive preprocessing for streaming data. *IEEE Trans. on Knowledge and Data Engineering*, 26(2):309–321, 2014.