Int. J. Appl. Math. Comput. Sci., 2017, Vol. 27, No. 4, 669–679
DOI: 10.1515/amcs-2017-0046

amcs

EXPLORING COMPLEX AND BIG DATA

JERZY STEFANOWSKI [a,*], KRZYSZTOF KRAWIEC [a], ROBERT WREMBEL [a]

[a] Institute of Computing Science
Poznań University of Technology, ul. Piotrowo 2, 60-965 Poznań, Poland
e-mail: {jerzy.stefanowski,krawiec,robert.wrembel}@cs.put.poznan.pl

This paper shows how big data analysis opens a range of research and technological problems and calls for new approaches. We start with defining the essential properties of big data and discussing the main types of data involved. We then survey the dedicated solutions for storing and processing big data, including a data lake, virtual integration, and a polystore architecture. Difficulties in managing data quality and provenance are also highlighted. The characteristics of big data imply also specific requirements and challenges for data mining algorithms, which we address as well. The links with related areas, including data streams and deep learning, are discussed. The common theme that naturally emerges from this characterization is complexity. All in all, we consider it to be the truly defining feature of big data (posing particular research and technological challenges), which ultimately seems to be of greater importance than the sheer data volume.

Keywords: big data, complex data, data integration, data provenance, data streams, deep learning.

1. Introduction

Big data can be defined in several ways. According to the prevailing opinion, it involves storage and analysis of massive and complex data sets that exceed the capacity of conventional computer systems and algorithms. However, quantity alone does not seem to be sufficient to justify such a distinction: it is the specific data *properties* rather than large volumes that make the difference and call for new approaches and technologies. This stance is aptly reflected in the 'many Vs' (*mVs*) definition, the original and much cited variant which includes (Beyer and Laney, 2012), *Volume* (referring to the huge amount of data), *Variety* (indicating multiple, heterogeneous, and complex data representations), and *Velocity* (referring to a speed at which the data are generated and analysed, as well as their dynamics and evolution in time, e.g., data streams). Subsequent extensions brought about other *mVs*, i.e., *Veracity* (referring to the worse quality and uncertainty of data) and *Value* (referring to a potential business value that big data analysis could give). Also, these and other definitions (see a comprehensive review by Mauro *et al.* (2015)) point out that big data involve data repositories that integrate multiple and spatially distributed sources.

The transition from previous approaches developed for knowledge discovery from databases to big data (characterized with the *mVs* properties) raises new challenges. The methods for data pre-processing and analysis, though arguably advanced in the last decades, are *still primarily designed for static and well structured data representations* (usually tabular). There is evident shortage of approaches, algorithms, and technologies that explicitly address other *mVs*, and the demand for them is growing continuously. It is partially driven by practice—recent years have brought a deluge of usage scenarios and applications that result in massive volumes of complex, poorly-structured, and dynamic data in many domains such as social networks, mobile services, network monitoring, smart cities, and intelligent transportation.

In this paper, we focus on one of the main aspects of big data, namely the growing *complexity* of data, which stems from the need for getting richer and more precise descriptions of the real world on the one hand, and the advancement of new technologies of data acquisition on the other. Many contemporary systems collect complex data; for instance, modern manufacturing systems may involve a broad range of data sources such as specialized sensor networks for monitoring the equipment of production lines (e.g., robots), systems monitoring the quality of produced goods, traditional databases of

*Corresponding author

orders, stocks, planning and logistics, web marketing, and even social media channels. Other examples are medical information systems, which store medical records about patients, diagnoses, treatments, and screening. Such data are inherently heterogeneous in representation, ranging from tabular (patient records, results of laboratory tests), to unstructured texts, to time series (e.g., ECG examinations), to various types of images.

Although some data analysts and past research studies have already considered data of more complex types than the simple attribute-value representation (cf. relational data mining), the demand for techniques of efficient and accurate analysis of complex and heterogeneous data is growing. Furthermore, end users expect the discovered knowledge to present a full picture of the problem, rather than an isolated result based only on a particular data representation. In this respect, not only *predictive* but also *prescriptive analytics* is gaining popularity (Soltanpoor and Sellis, 2016).

Complexity not only affects the structure of data, but also the techniques for data collecting and analyzing. On the input side, new solutions are needed for data acquisition, quality checks of measurements, preliminary filtering, compression, integration of various data sources, and efficient loading data into repositories (Hashem and Ranc, 2016). Once loaded, complex data require specialized algorithms for efficient processing and good scaling. New hardware and software architectures are typically required for large-scale efficient acquisition, storage, and analysis of data (in particular, in distributed and cloud architectures). Last but not least, an analysis of such data is also more demanding: complex data typically imply considering several different predictive models (all of which interact with or impact one another) in order to provide the best possible results. On top of that, the ever-increasing customer expectations concerning privacy and security issues must be satisfied.

For the aforementioned reasons, collecting, storing, processing, and exploring big data is fundamentally different from and more challenging than the traditional data analytics. Thus, research and technological sectors are working on new approaches. This paper is intended to discuss new and still open challenges in processing and analyzing complex, heterogeneous, and large data sets. To this end, in the following we focus on three aspects of big data that are closely related to the *mVs* mentioned above:

(i) architectures for storing, managing, and pre-processing big data,

(ii) handling complex data representations by deep learning algorithms,

(iii) mining concept-drifting data streams.

2. Types of complex and big data

Data generation and acquisition are the first steps of any knowledge discovery process. Traditionally, the relatively homogeneous data sources were verified and integrated into well-structured logical forms. However, applications of big data often require integrating data from both traditional and non-traditional sources (Che *et al*., 2013). With inexpensive sensors, mobile devices, the Internet, and social collaboration technologies, data are now generated in numerous forms such as text, web data, tweets, images, audio, video, log files, and many more. According to many reports (e.g., Gens, 2011), only about 10% of big data are well structured; the rest needs specialized tools for decoding and pre-processing (e.g., with natural language processing), and are in general not easy to mold into rigid, predefined categories and logical formats. As a consequence, data integration, pre-processing, and analysis (see the next sections) are challenging. Furthermore, analysts are forced to deal with *structured*, *semi-structured*, and *unstructured* data simultaneously, which is another difficult task, as most standard analytical tools are dedicated to one of those categories exclusively.

Historically, machine learning researchers have already studied objects that are complex in the above sense, e.g., entities with sophisticated internal hierarchies, multi-dimensional nested structures with different attribute types, featuring a variable number of attributes at particular levels, and/or containing varying-length sequences of symbols or graphs. The popular textbook by Han and Kamber (2011) distinguishes the following major types of complex objects: spatial, temporal, sequence, graph or tree structured data, multidimensional time series, text, and multimedia data. Heterogeneity of these data types forms a challenge even on a small scale; in big data analysis, it leads to further growth of complexity and difficulty in their analysis. No wonder that the interest within this area has moved to mining really very large and heterogeneous graphs or to other kinds of spatio-temporal data integrating more sources and time dimensions. Moreover, the sole nature of many new problems considered in big data analysis, such as social networks, data streams, sequence mining, leads to new kinds of data complexity which have not be considered before (Japkowicz and Stefanowski, 2016a).

Also, even the more traditional learning tasks, such as training classifiers, are now reformulated in more demanding ways, for instance, by taking into account additional constraints or data properties, like unusual distributions of examples and/or imbalance of target classes (Fernández *et al*., 2017; Napierala and Stefanowski, 2016). Such enriched input to the induction process requires more advanced and complex algorithms.

Last but not least, there are more works on learning

more advanced outputs from input data. They mainly include methods for predicting structured objects, rather than scalar discrete or real values (see, e.g., multi-labeled classification, sequence tagging or some variants of probabilistic graphical models).

3. Architectures for processing big data

Due to the multiple *Vs* discussed in Introduction, storing and processing big data is challenging and requires dedicated hardware and software architectures. It is now widely agreed that the currently available architectures that are feasible for big data are mainly *data lakes*, *virtual architectures*, and *polystores*. Each of them can be run in a cluster of workstations or on a main memory appliance, (cf. Francisco, 2012; Bayer and Edjlali, 2014).

3.1. Data lake. The most common and accepted architecture for physically integrating big data is a data lake (Russom, 2017; Terrizzano *et al*., 2015). It is a repository that stores a vast amount of heterogeneous data in their original formats. Therefore, an application that accesses a data lake has to discover and understand the data formats on the fly. In most data lake deployments, a data storage layer is based on a distributed file system (HDFS or GFS), and data are processed in parallel, typically using the MapReduce model (Chen and Zhang, 2014).

Given the volume and variety of data stored in a data lake, querying its content is one of the key challenges in big data management. A query engine needs to support additional functionalities, including means for (i) identifying relevant data sets for answering a given query, i.e., to understand the content of the data lake, (ii) discovering formats of data sets, (iii) converting data on-the-fly to the format preferred by a user (Duggan *et al*., 2015; Liu and Wang, 2016), and (iv) appropriately visualizing query results (Chen and Zhang, 2014). To this end, appropriate and rich metadata are needed.

Once a data lake is built and supplied with data, another issue is to keep it up-to-date, as the underlying data sources frequently produce new data and update their content. Depending on the application domain, a data lake has to be refreshed either periodically or in (near) real-time. Refreshing a data lake can be difficult for several reasons:

- its content typically depends on multiple heterogeneous and distributed data sources;
- the capabilities of accessing these data sources and detecting their content and structural changes may be limited;
- it is often nontrivial to devise and incorporate incremental data refreshing algorithms for data formats other than simple tables;
- heterogeneous and complex data have to be homogenized, cleaned, augmented (Ahmadov *et al*., 2015; Miao *et al*., 2017), and de-duplicated (Benjelloun *et al*., 2009) before being analyzed. These tasks are analogous to those in traditional data warehouses, where the so-called ETL software (extract-transform-load) or its ELT or ELTL variants are used for this purpose. Nonetheless, due to the heterogeneity, complexity, and volume of big data, these tasks are much more challenging and require more complex and efficient algorithms. Typically, some ETL tasks are implemented as user-defined functions (UDFs), e.g., MapReduce tasks, which are treated by the system as black boxes that produce a given output for a given input. Optimizing such UDFs has not been explored yet.

3.2. Virtual integration. On the opposite side of data integration architectures lies the paradigm of virtual integration. In a *virtual integration architecture* data are stored in their original systems but are accessed via a global schema. The schema provides a *view* on the integrated data sources. Queries are expressed on the global schema and further resolved to be sent to and executed in the underlying data sources. This architecture is similar to the well-known federated (Elmagarmid *et al*., 1999) or mediated architectures (Wiederhold, 1992). Such architectures re-gain their popularity in the context of heterogeneous and complex data, including graphs (Liu and Wang, 2016) and Web data (Langegger *et al*., 2008).

The paradigm of virtual integration has two advantages over the data lake approach. First, data being accessed are always up-to-date. Second, neither additional storage nor bespoken refreshing mechanisms are needed. The price to pay in exchange for these conveniences is poor query performance. To address this weakness, virtual architectures typically apply the caching and reusing of query results (Gessert *et al*., 2017; Zakhary *et al*., 2017), which, though undoubtedly useful, pose certain research and technological challenges, namely, (i) deciding what to cache, (ii) deciding how long to store the cached data, (iii) developing effective means to figure out whether a cache content is still valid, (iv) developing algorithms for efficient refreshing of an invalid cache, (v) developing mechanisms for proactive (in-advance) caching, which in turn requires predicting which queries are likely to be executed, (vi) deciding whether to move the cached data into a permanent storage and, if so, which cached data to move.

Other still open problems related to a virtual integration architecture for big data pertain to being able to (i) (semi-)automatically discover the relevant data sources to be integrated, (ii) figure out the structures,

content, and quality of data sources (developing methods for discovering and profiling (Naumann, 2014) data sources on the Internet), (iii) (semi-)automatically include a data source into an integration architecture and construct an integrated schema, (iv) optimize a query on an integrated schema, assuming that the underlying data sources are distributed and may offer various functionalities: from fully-functional relational databases, NoSQL engines, distributed file systems, through to Web tables, pages, and portals.

3.3. Polystore. In the work of Duggan *et al*. (2015), yet another alternative big data integration architecture was proposed, called a *polystore*. In this architecture, data sets are organized into the so-called *islands of information*. This phenotenon is defined as a collection of storage engines accessed with the same query language. For example, a collection of graph databases may form a graph island. Similarly, a collection of text databases may form a text island. An island provides a single data model, which is suitable for all storage engines in it, and a common query language, pertinent to the island's data model.

The data model and language are mapped by dedicated software modules called *shims*, into the languages and models of data management systems running specific databases/engines within the island. Users operate on an island by means of the specific island's query language. A user query is decomposed into partial queries—one query for one storage engine in the island. Next, partial queries are sent to appropriate shims. Each shim translates its partial query into a query in a native language of a storage engine. The partial queries are executed in their proper data sources, and their results are integrated by a shim and transformed into an output data model indicated in the original user's query. Multi-island queries are also allowed by means of shims from different islands.

The polystore draws upon the techniques of physical and virtual data integration. A shim virtually integrates the underlying storage engines, while data can be physically moved between islands, if requested by a user.

3.4. Data provenance. According to Gupta (2009), *data provenance* "refers to a record trail that accounts for the origin of a piece of data (in a database, document or repository) together with an explanation of how and why it got to the present place". It helps in assessing the quality of data, identifying sources from which a given piece of data was ingested, and analyzing step-by-step transformations executed on a given piece of data.

To collect, represent, store, and share provenance data, the open provenance model (OPM) (Moreau *et al*., 2011) was proposed. It recommends to use five components, namely, (i) an abstract model, (ii) a domain-specific model, (iii) mechanisms for model extension, (iv) data acquisition and mapping to RDF, and (5) APIs for querying. The specifications of these components remain under development, so the problems related to developing and implementing the OPM architecture are still open. For example, even though provenance data may be available in a system, their querying and visualizing is still a challenge. Moreover, augmenting a user query with the most pertinent provenance of data could help in explaining the obtained query results, but such a universal query mechanism is still an open issue.

In the integration architectures outlined in Sections 3.1, 3.2, and 3.3, data undergo multiple transformations in order to make their format and values suitable for analysis. Such transformations are typically modeled as transformation workflows, e.g., ETL ones. For traditional applications, provenance datasets on a transformation workflow are large—often larger than the transformed dataset itself. In big data, data transformation workflows are much more complex and generate much larger provenance data sets. For this reason, efficient querying of such provenance data is of crucial importance. Moreover, 'intelligent' methods for deciding which provenance data are essential and should be thus recorded, still to be developed (Glavic, 2014).

Big data transformation workflows are often based on MapReduce tasks, which are implemented as user-defined functions and executed in a distributed environment. Gathering provenance data for such workflows is difficult. Some approaches and software for gathering provenance data from MapReduce have already been developed, e.g., RAMP, HadoopProv, Pig Lipstick (Wang *et al*., 2015). Nonetheless, they are usually dedicated to a particular architecture (i.e., MapReduce) and a particular type of processing (i.e., batch). In a data lake architecture, data are processed not only in large batches by disk-resident MapReduce but also in the main memory, either as small batches (e.g, Spark) or data streams (e.g., Kafka, Storm, Flink). There are other specialized approaches for storing the provenance of data given in a particular format, e.g., RDF only (Wylot *et al*., 2017). However, big data solutions involve data of various formats, from simple rows to complex graphs, yet the provenance for all these data should be preferably recorded and stored in a unified manner to allow efficient and unified querying.

3.5. Open research and technological challenges. In the area of collecting and storing big data we have identified the following, fundamental and still open, research and technological challenges. They include the development of

- optimization mechanisms for user-defined functions in data transformation workflows,
- techniques for (semi-)automatic discovery of relevant data sources to be integrated in a big data architecture,
- query optimization and data caching strategies for virtual big data integration architectures,
- tracking mechanisms for data changes and structural changes in data sources for the purpose of propagating them to a big data architecture (e.g., a data lake or a polystore),
- a well-defined metadata standard capable of providing reach metadata to support the aforementioned features of a big data integration architecture,
- comprehensive mechanisms for big data provenance, including representation, maintenance, storage, and querying,
- mechanisms for gathering essential provenance data (for data of heterogeneous and complex origins) and automatically augmenting user queries with the most relevant provenance data.

4. New requirements for data mining algorithms

Most machine learning algorithms were originally developed for static and relatively small data sets, often assumed to fit into memory. Attempts to apply such methods to real-world datasets quickly met with memory limitations. Soon, a range of techniques were proposed to address this issue, including sampling, incremental processing, or training multiple models on partitioned data. Furthermore, the development of knowledge discovery from databases in the 1990s led to new solutions for more efficient communication with disk-residing data and new data structures that sped up data mining (DM) algorithms.

Nevertheless, the dawn of big data exaggerated some of the scalability problems: the efficiency challenges, which seemed to be at least partly resolved, resurfaced again when the volumes of data skyrocketed by several orders of magnitude. These problems, and those stemming from the difficult nature of data (cf. the *mVs* discussion), need to be addressed to better realize the potential of big data analysis. As follows from multiple ongoing discussions in the community (e.g., Japkowicz and Stefanowski, 2016b), the new requirements refer both to properties of data and analytical issues. We briefly review them individually here.

Scaling. The size of data remains one of the main problems. However, besides the large number of instances, it is even more challenging to efficiently handle the growing numbers of attributes and classes, e.g., the multi-labeled learning or the extreme classification.

Parallelization. Developing optimally distributed and parallel implementations of existing algorithms (e.g., in the Hadoop or Spark frameworks) is important. Nevertheless, the demand for *genuinely new algorithms* is growing, too (Bekkerman *et al*., 2011).

New data types. Most algorithms still work with attribute-valued tabular data only, but many applications require complex data types (cf. Section 2). Furthermore, some tasks involve heterogeneous data sources and need to integrate the results of their analysis, which calls for new information fusion approaches.

Mining data streams also constitutes a new challenge, and given the growing importance of this 'data modality', we elaborate more on this in a Section 6.

Data imperfectness and uncertainty. Data inconsistency and incompleteness, imprecision, ambiguity and vagueness of available descriptions, latency of getting access to important data elements—all these imperfections have been partly addressed in the machine learning and data mining (ML/DM) community. Nevertheless, together with uncertainty, they are inherent and more important in many aspects of big data, particularly when learning from the social media, text, multimedia, language translations, or summarizing human opinions (Che *et al*., 2013).

Timeliness. There are many situations in which results are required immediately or at least fast enough, e.g., in fraud detection, stock market predictions, technical diagnostics of equipment, and monitoring systems. This calls for the development of new data processing architectures (e.g., parallel and distributed, main memory, NVRAM servers) and index structures that meet specific time-related criteria (cf. *Scaling* above). Whenever such criteria cannot be met with exact algorithms, approximate solutions with sufficient theoretical error bounds need to be developed.

Trust and provenance. Early on, DM algorithms were typically applied to thoroughly pre-processed data, which were assumed to come from well-defined sources. In big data scenarios, data sources have less clear origins, are unverified, and are in general of lower quality than in traditional systems (Che *et al*., 2013). Therefore, it becomes important to be aware of the *provenance*, quality, and value of such data (cf. Section 3.4).

Privacy. Privacy preserving DM deals with drawing conclusions about entire populations while protecting the privacy of individuals. This imposes constraints on the DM practice: ways have to be found to mask the actual

data while preserving their aggregate characteristics, in particular, maintaining the accuracy of estimates (Matwin, 2013). Privacy issues, though studied earlier, have become extremely important with the emergence of big data, where achieving a desired effect often requires more personal and sensitive information, like localization-based and personalized recommendations of services, or targeted and individualized marketing. However, many contemporary privacy techniques can be hacked, allowing the user's identity to be inferred when several pieces of information from various repositories are available (Japkowicz and Stefanowski, 2016a).

Data ownership. Big data are in essence collected everywhere, which may interfere with people's rights to their electronic records. Users need to understand, and have the right to be appropriately informed, that their data may be shared and used for analytical aims other than those they envisioned. New legal and social mechanisms need to be developed to this aim. The danger is that too strong laws protecting the data and limiting data sharing may have negative impact on the value of the knowledge to be discovered (Japkowicz and Stefanowski, 2016a).

Transparency and ethics. The results of mining personal data may be applied to *predict the actions of other people*. Although more and more researchers become aware that DM techniques should be used in a fair, non-discriminating and *transparent* manner, research on these postulates is still in the premature stage (Custers *et al*., 2013).

Comprehensibility and human interaction. Although the majority of research in machine learning aims at maximizing the predictive accuracy, models' comprehensibility (interpretability and understandability) for users is very important in many applications. This aspect was in focus in the early machine learning research, but largely neglected later on, with the prevalence of *black box* models (e.g., SVMs, neural networks, ensembles) that can be hardly interpreted by users. Many big data applications demonstrated that model comprehensibility is also important for users trust and willingness to implement the outcomes in practice (Rudin, 2014). In some application domains, users need to understand systems recommendations enough to explain the reason for their decisions to other people. Last but not least, in most usage scenarios, big data analysis cannot be fully automated, but needs to include a domain expert in the loop. The expert's guidance can help narrow down the analysis to the most promising regions in data, choosing potential models, tuning their parameters, verifying data and operation correctness, and interpreting results. The above discussion shows that these expectations are hard to meet in big data practice due to its complexity and multi-faceted character. More research is needed to improve the comprehensibility of models and allow humans to seamlessly interact with the analytical processes.

For a comprehensive review of the above requirements, the reader is referred to Che *et al*. (2013) as well as Japkowicz and Stefanowski (2016b) for a detailed discussion on the differences between the earlier ML algorithms and newer big data approaches.

Furthermore, the reader should be aware of more critical discussions about the risks or even negative effects of big data analysis along with its pitfalls (cf. Boyd and Crawford, 2012). Yet another interesting issue concerns the influence of big data on methodological changes in scientific research; see, e.g., discussions in the first two chapters of the book by Japkowicz and Stefanowski (2016b).

5. Complex data representation and deep learning

As pointed out in Section 1, one of the major challenges for big data is the complexity and variety of data representations. The recent advancement in deep learning brought an unexpected ally to address this issue—artificial neural networks. Today, we witness them being surprisingly versatile when it comes to learning from data represented in a variety of forms. It has been known for long that convolutional neural networks are useful tools in image analysis. However, it is only within the last decade that conceptual progress, combined with easy access to the massive power of GPU computing and large volumes of training data, has led to qualitative breakthroughs, including superhuman performance on well-defined object/image recognition tasks. More recent approaches, particularly within recurrent models, have proven deep learning to be capable of effective learning from data representations that are arguably 'even less tabular': variable-length sequences (time series, sound, speech, text), graphs and networks (including social networks), natural language, and even the source code of computer programs (see the works of LeCun *et al*. (2015) and Schmidhuber (2015) for extensive reviews).

Another aspect of deep neural networks that seems confluent with big data is their capability to perform sophisticated *feature engineering*. The complexity of big data problems mentioned earlier often implies that well-performing predictors cannot be easily induced: training data need to be re-represented to, e.g., allow the decision classes in question to be successfully delineated. In the past, the search for useful representations had to be conducted by experts, via manual feature engineering, or explicit feature selection and construction techniques (see, e.g., Krawiec, 2016). Deep neural networks help in automating this task: feature construction becomes an inherent part of the training process, tightly bound to the

search in the hypothesis space. Tools that come in handy here include, among others, convolutions (for spatial or temporal data) and multi-dimensional embeddings. Some neural models allow synthesis of features in the form of latent variables with certain desirable properties, e.g., distributions (cf., e.g., *variational autoencoders* (Kingma and Welling, 2013)).

Modular design of deep learning architectures facilitates construction of models that handle multiple data representations simultaneously, like generating textual annotations from images, speech synthesis from text, or end-to-end translation. This enables handling tasks beyond the traditional classification and regression, and comes in particularly handy when data come from a range of sources, which is common in big data settings. Also, the representations learned from various data repositories, materialized in the form of networks' layers or feature maps, can be easily reused in other contexts, and further tuned/trained if necessary. As a matter of fact, *transfer learning*, so natural and easy in deep learning, it is now in routine use: for instance, it is now common to re-use the initial layers of AlexNet (Krizhevsky *et al*., 2012), the network that once famously advanced deep learning performance on ImageNet benchmarks.

In a sense, the big data framework is a natural environment for deep learning, where models typically require substantial volumes of data to learn effectively—large numbers of models' parameters, often in the order of millions, make overfitting almost inevitable when learning from small data sets. It seems rightful to claim that the dawn of big data actually *started* the deep learning era. Though this convergence seems fortunate, one must admit that massive training data combined with complex deep learning architectures lead to exorbitant demands for computing power. The continuous progress in hardware, including the ever more powerful GPUs, capacious and fast NVRAMs, and other specialized hardware, seems to partially satisfy this demand. Nevertheless, more work is needed to seek the opportunity for reducing the computational requirements. Also, deep learning is clearly not a panaceum to all big data challenges; for instance, deep learning models hardly ever provide for transparency and human-readable explanation.

6. Challenges for mining data streams

We discuss data streams as they perfectly represent many *Vs* properties of big data. In particular, compared with Section 5, we focus more on challenges of dealing with huge volumes of data, their rapid speed or arrival into a system, limited memory usage, and changes of data distribution in time.

6.1. Data characteristics. A data stream is a potentially unbounded sequence of data items available over time. It is typically assumed that the items are continuously produced at a rapid rate so that the data arrival speed is relatively high compared with the computational capabilities of a processing system. Furthermore, the items often cannot be stored in memory and must be processed immediately upon their arrival and discarded after being processed, in order to make space for new ones.

The processing of a data stream is different from the conventional batch, static processing and implies new requirements for algorithms, such as constraints on memory usage, restricted processing time, and scanning the incoming items only once. Another challenge is a dynamic, non-stationary environment, where data and a target concept may change over time due to the so-called *concept drift*.

Many standard algorithms cannot meet the data stream requirements, so several new proposals have been introduced (see the work of Gama (2010) for a review). The simplest algorithms for *processing* or *querying streams* extend the earlier incremental algorithms for counting event occurrences in the stream or the number of distinct values, basic statistics, frequency moments, etc. It is not an easy task, as a stream is unbounded in length and the domain of possible values may change in time. These and the aforementioned stream constraints may mean that an algorithm produces an approximate answer based on a summary or a 'sketch' of the data stream in memory (Gama, 2010). To this end, new techniques for sampling (e.g., reservoir sampling) have been developed.

Data stream mining is the process of discovering knowledge structures from streams. It includes such tasks as clustering, discovering frequent patterns and associations, classification, regression, change detection, novelty identification, and time series mining.

6.2. Classification of streams. Classification is the most widely studied task in data stream mining. Apart from the aforementioned general difficulties of streams, the concept drift is particularly important challenge. It deteriorates the predictive accuracy of classifiers, as the current data differ from the instances they were trained on. Thus, the algorithms should monitor the 'evolution' of incoming data and properly react to changes.

The new proposed streaming classifiers can be categorized with respect to different criteria. Most former researchers distinguish between *active* (trigger-based) and *passive* (adaptive) approaches. The *Active* approaches include drift detectors that analyze the incoming examples and indicate the need for rebuilding a classifier. Drift detectors are usually implemented using statistical tests based on a sequential analysis, process control charts, or monitoring the differences between distributions (Gama

et al., 2014). The latter do not contain drift detectors and continuously update a classifier each time a new data item arrives. Some of the adaptive approaches exploit *forgetting* mechanisms, such as *sliding windows* or fading functions assigned to incoming examples.

Another categorization makes a distinction between single classifiers and ensembles. The prominent representative of the former is the very fast decision tree (VFDT), originating from the Hoeffding tree algorithm (Domingos and Hulten, 2000), which induces a decision tree from a data stream incrementally, without the need for storing examples after they have been used to update the tree. It is based on exploiting the Hoeffding bound to select an attribute good enough for a split test in the tree nodes, which is done without viewing all the examples but guarantees the split to be correct at a user-specified probability. Enhancements to the basic VFDT algorithm include limiting memory usage, dealing with numerical attributes, pruning mechanisms, and sliding windows or drift detectors, to adapt the algorithm to non-stationary settings (Gama, 2010).

An *ensemble* of classifiers, also called a multiple classifier, is a set of individual component classifiers whose predictions are combined to assign a class label to a new instance. The ensembles are inherently capable of adapting to changing streams by introducing new component classifiers created using batches of incoming examples, updating existing components, or changing the weights in the aggregation phase.

Depending on the way the incoming examples are processed and the component classifiers updated, data stream ensembles are categorized into *block*-based (process small chunks of examples) and *online* (process single instances). The former re-evaluate the component classifiers with the recent block of incoming items and usually replace the worst component with a new candidate classifier trained on the most recent items. The latter integrate several incremental single classifiers, and are often based on extending standard ensembles (like many variants of online bagging) or the weighted majority algorithm (Brzezinski and Stefanowski, 2014). Comprehensive reviews of various ensembles can be found in some recent surveys (Ditzler *et al*., 2015; Krawczyk *et al*., 2017).

6.3. Open research problems. Though mining data streams have experienced increased interest in recent years, several problems still need to be addressed. For instance, most of the current research on stream classifiers assumes full and immediate access to class labels of incoming examples. This, however, may be unrealistic in situations when getting true class labels for all examples in the stream may be too costly or even impossible. Moreover, in some other situations, information about labels may be delayed. Thus, the following scenarios of processing data streams should be more deeply investigated: (i) *delayed labeling* (the classifier has to adapt to changes without knowing labels but then it may exploit them to update its model; (ii) *semi-supervised learning*, where labels are not available for all incoming examples (but they may be provided in limited portions from time to time, or active learning strategies are applied to query the most useful examples); (iii) *unsupervised framework*, where the initial classifier is learned from a limited number of examples and then it has to process the stream of unlabeled examples (this scenario is the most challenging; see the discussion by Ditzler *et al*. (2015)).

Modern applications often involve more *complex* data representations than flat attribute vectors. Streams of documents, text or tweets are more and more commonly examined. Researchers have also started studying streams of graphs and more sophisticated relational structures (e.g., relational learning in streams) as well as multi-labeled data streams or sequence predictions. In recent years, class imbalances (also with variable class cardinalities or swapping roles of minority vs. majority classes) and detection of novel, appearing classes have been receiving increased interest (Sun *et al*., 2016). However, research on these topics is still in the early phase (Krawczyk *et al*., 2017).

Some other open problems include (i) handling incomplete information (mainly on missing attribute values), (ii) studying parallel streams (where pieces of information are linked between streams), (iii) integrating both offline and online approaches, (iv) dealing with censored event streams (e.g., multi-dimensional adaptation of a survival analysis (Shaker and Hüllermeier, 2014)), (v) developing new measures for stream evaluation (e.g., for imbalanced streams or for a multiple criteria aggregation of prediction and fast reaction to drifts), and (vi) estimation procedures (also including better adaptive scenarios to tune parameters). These and other open challenges are discussed by Krempl *et al*. (2014).

7. Final remarks

Big data, arguably one of the terms that have recently won immense popularity in computer science and beyond, also happens to be one of the most abused notions, with all too many and too hazy definitions, and hard-to-define extent. In this paper, while building upon the past attempts, we aimed at concise and possibly precise characterization of the big data issue as it is being contemporarily practiced. We covered, among others, the essential features of big data domains, discussed the types of data involved, system architectures for their efficient processing, and touched upon the particularly relevant trends of deep learning and data streams. Furthermore, we highlighted open

challenges in these sub-fields.

Big data are clearly not a closed chapter in the history and present of data analytics. Quite the opposite, at least for the time being: it is hard to see why the volumes of data to be handled should ever shrink and their complexity decrease. Growing bandwidth, storage, and processing power will only facilitate this trend. In this light, persistence in meeting the requirements discussed in Sections 4 and 3, as well as addressing the challenges outlined in other sections, is the only way to go.

Acknowledgment

The work of Robert Wrembel is supported with the National Science Center grant no. 2015/19/B/ST6/02637. Krzysztof Krawiec acknowledges the support from the National Science Center under the grant no. 2014/15/B/ST6/05205. Jerzy Stefanowski acknowledges the support from the Poznań University of Technology through statutory funds.

Literatura

Ahmadov, A., Thiele, M., Eberius, J., Lehner, W. and Wrembel, R. (2015). Towards a hybrid imputation approach using web tables, *IEEE/ACM International Symposium on Big Data Computing (BDC), Limassol, Cyprus*, pp. 21–30.

Bekkerman, R., Bilenko, M. and Langford, J. (2011). *Scaling Up Machine Learning: Parallel and Distributed Approaches*, Cambridge University Press, New York, NY.

Benjelloun, O., Garcia-Molina, H., Menestrina, D., Su, Q., Whang, S.E. and Widom, J. (2009). Swoosh: A generic approach to entity resolution, *The VLDB Journal* **18**(1): 255–276.

Bayer, M.A. and Edjlali, R. (2014). *Magic Quadrant for Data Warehouse Database Management Systems*, Gartner Publications, Stamford, CT, `https://www.gartner.com/doc/2678018/magic-quadrant-data-warehouse-database`.

Beyer, M. and Laney, D. (2012). *The Importance of "Big Data": A Definition*, Gartner Publications, Stamford, CT.

Boyd, D. and Crawford, K. (2012). Critical questions for big data, *Information, Communication and Society* **15**(5): 662–679.

Brzezinski, D. and Stefanowski, J. (2014). Combining block-based and online methods in learning ensembles from concept drifting data streams, *Information Sciences* **265**: 50–67.

Che, D., Safran, M. and Peng, Z. (2013). From big data to big data mining: Challenges, issues, and opportunities, *in* B. Hong *et al.* (Eds.), *International Conference on Database Systems for Advanced Applications*, Lecture Notes in Computer Science, Vol. 7827, Springer, Berlin/Heidelberg, pp. 1–15.

Chen, C.L.P. and Zhang, C. (2014). Data-intensive applications, challenges, techniques and technologies: A survey on big data, *Information Sciences* **275**(10): 314–347.

Custers, B., Calders, T., Schermer, B. and Zarsky, T.Z. (Eds.) (2013). *Discrimination and Privacy in the Information Society—Data Mining and Profiling in Large Databases*, Studies in Applied Philosophy, Epistemology and Rational Ethics, Vol. 3, Springer, Berlin/Heidelberg.

Ditzler, G., Roveri, M., Alippi, C. and Polikar, R. (2015). Learning in nonstationary environments: A survey, *IEEE Computational Intelligence Magazine* **10**(4): 12–25.

Domingos, P. and Hulten, G. (2000). Mining high-speed data streams, *ACM SIGKDD International Conference on Knowledge Discovery Data Mining, Boston, MA, USA*, pp. 71–80.

Duggan, J., Elmore, A.J., Stonebraker, M., Balazinska, M., Howe, B., Kepner, J., Madden, S., Maier, D., Mattson, T. and Zdonik, S. (2015). The BigDAWG polystore system, *SIGMOD Record* **44**(2): 11–16.

Elmagarmid, A., Rusinkiewicz, M. and Sheth, A. (Eds.) (1999). *Management of Heterogeneous and Autonomous Database Systems*, Morgan Kaufmann, San Francisco, CA.

Fernández, A., del Río, S., Chawla, N.V. and Herrera, F. (2017). An insight into imbalanced big data classification: Outcomes and challenges, *Complex & Intelligent Systems* **3**(2): 105–120.

Francisco, P. (2012). Oracle Exadata and IBM Netezza data warehouse appliance compared, *IBM White Paper*, `www.ibmbigdatahub.com/pdf/Oracle_Exadata_IBMNetezza_Compared_WP_EN.pdf`.

Gama, J. (2010). *Knowledge Discovery from Data Streams*, Chapman and Hall, Boca Raton, FL.

Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M. and Bouchachia, A. (2014). A survey on concept drift adaptation, *ACM Computing Surveys* **46**(4): 44:1–44:37.

Gens, F. (2011). IDC predictions 2012: Competing for 2020. IDC analyze the future, `https://www.virtustream.com/sites/default/files/IDCTOP10Predictions2012.pdf`.

Gessert, F., Schaarschmidt, M., Wingerath, W., Witt, E., Yoneki, E. and Ritter, N. (2017). Quaestor: Query web caching for database-as-a-service providers, *PVLDB* **10**(12): 1670–1681.

Glavic, B. (2014). Big data provenance: Challenges and implications for benchmarking, *in* T. Rabl *et al.* (Eds.), *Specifying Big Data Benchmarks*, Springer, New York, NY, pp. 72–80.

Gupta, A. (2009). Data provenance, *in* L. Liu and M.T. Özsu (Eds.), *Encyclopedia of Database Systems*, Springer, Berlin, pp. 608–608.

Han, J. and Kamber, M. (Eds.) (2011). *Data Mining. Concepts and Techniques*, Morgan Kaufmann, San Francisco, CA.

Hashem, H. and Ranc, D. (2016). Pre-processing and modeling tools for bigdata, *Foundations of Computing and Decision Sciences* **41**(3): 151–162.

Japkowicz, N. and Stefanowski, J. (2016a). A machine learning perspective on big data analysis, *in* N. Japkowicz and J. Stefanowski (Eds.), *Big Data Analysis: New Algorithms for a New Society*, Springer, Cham, pp. 1–31.

Japkowicz, N. and Stefanowski, J. (Eds.) (2016b). *Big Data Analysis: New Algorithms for a New Society*, Studies in Big Data, Vol. 16, Springer, Cham.

Kingma, D.P. and Welling, M. (2013). Auto-encoding variational Bayes, *ArXiv e-prints*, 1312.6114a.

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

Krempl, G., Zliobaite, I., Brzezinski, D., Hüllermeier, E., Last, M., Lemaire, V., Noack, T., Shaker, A., Sievi, S., Spiliopoulou, M. and Stefanowski, J. (2014). Open challenges for data stream mining research, *SIGKDD Explorations* **16**(1): 1–10.

Krizhevsky, A., Sutskever, I. and Hinton, G.E. (2012). Imagenet classification with deep convolutional neural networks, *in* F. Pereira *et al.* (Eds.), *Advances in Neural Information Processing Systems 25*, Curran Associates, Inc., Red Hook, NY, pp. 1097–1105.

Krawiec, K. (2016). Evolutionary feature selection and construction, *in* S. Claude and G. Webb (Eds.), *Encyclopedia of Machine Learning and Data Mining*, Springer, Boston, MA.

Langegger, A., Wöß, W. and Blöchl, M. (2008). A semantic web middleware for virtual data integration on the web, *European Semantic Web Conference on the Semantic Web: Research and Applications (ESWC), Tenerife, Canary Islands, Spain*, pp. 493–507.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning, *Nature* **521**(7553): 436–444.

Liu, M. and Wang, Q. (2016). Rogas: A declarative framework for network analytics, *International Conference on Very Large Data Bases (VLDB), New Delhi, India*, pp. 1561–1564.

Matwin, S. (2013). Privacy-preserving data mining techniques: Survey and challenges, *in* B. Custers *et al.* (Eds.), *Discrimination and Privacy in the Information Society*, Vol 3. Springer, Berlin/Heidelberg, pp. 209–221.

Mauro, A.D., Greco, M. and Grimaldi, M. (2015). What is big data? A consensual definition and a review of key research topics, *International Conference on Integrated Information, Madrid, Spain*, pp. 97–104.

Miao, X., Gao, Y., Guo, S. and Liu, W. (2017). Incomplete data management: A survey, *Frontiers of Computer Science*, DOI: 10.1007/s11704-016-6195-x.

Moreau, L., Clifford, B., Freire, J., Futrelle, J., Gil, Y., Groth, P., Kwasnikowska, N., Miles, S., Missier, P., Myers, J., Plale, B., Simmhan, Y., Stephan, E. and den Bussche, J.V. (2011). The open provenance model core specification (v1.1), *Future Generation Computer Systems* **27**(6): 743–756.

Napierala, K. and Stefanowski, J. (2016). Types of minority class examples and their influence on learning classifiers from imbalanced data, *Journal of Intelligent Information Systems* **46**(3): 563–597.

Naumann, F. (2014). Data profiling revisited, *SIGMOD Record* **42**(4): 40–49.

Rudin, C. (2014). Algorithms for interpretable machine learning, *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA*, pp. 1519–1519.

Russom, P. (2017). Data lakes: Purposes, practices, patterns, and platforms. *TDWI White Paper*, `https://info.talend.com/rs/talend/images/WP_EN_BD_TDWI_DataLakes.pdf`.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview, *Neural Networks* **61**(C): 85–117.

Shaker, A. and Hüllermeier, E. (2014). Survival analysis on data streams: Analyzing temporal events in dynamically changing environments, *International Journal of Applied Mathematics and Computer Science* **24**(1): 199–212, DOI: 10.2478/amcs-2014-0015.

Soltanpoor, R. and Sellis, T. (2016). Prescriptive analytics for big data, *Australasian Database Conference on Databases Theory and Applications (ADC), Sydney, Australia*, pp. 245–256.

Sun, Y., Tang, K., Minku, L.L., Wang, S. and Yao, X. (2016). Online ensemble learning of data streams with gradually evolved classes, *IEEE Transactions on Knowledge and Data Engineering* **28**(6): 1532–1545.

Terrizzano, I., Schwarz, P., Roth, M. and Colino, J.E. (2015). Data wrangling: The challenging journey from the wild to the lake, *Conference on Innovative Data Systems Research (CIDR), Asiloma, CA, USA*.

Wang, J., Crawl, D., Purawat, S., Nguyen, M.H. and Altintas, I. (2015). Big data provenance: Challenges, state of the art and opportunities, *IEEE International Conference on Big Data, Santa Clara, CA, USA*, pp. 2509–2516.

Wiederhold, G. (1992). Mediators in the architecture of future information systems, *IEEE Computer* **25**(3): 38–49.

Wylot, M., Cudré-Mauroux, P., Hauswirth, M. and Groth, P.T. (2017). Storing, tracking, and querying provenance in linked data, *IEEE Transactions on Knowledge and Data Engineering* **29**(8): 1751–1764.

Zakhary, V., Agrawa, D. and El Abbadi, A. (2017). Caching at the web scale, *International Conference on World Wide Web Companion, Perth, Australia*, pp. 909–912.

Jerzy Stefanowski is an associate professor in the Institute of Computing Science, Poznań University of Technology. His research interests include machine learning, data mining and intelligent decision support—in particular, rule induction, multiple classifiers, class imbalance, concept drift, classification of data streams and big data. For more information, refer to `http://www.cs.put.poznan.pl/jstefanowski`.

Krzysztof Krawiec is an associate professor in the Institute of Computing Science, Poznań University of Technology. His recent work includes program synthesis (in particular, by means of genetic programming) evolutionary computation for machine learning, learning game strategies and pattern recognition; deep learning for image analysis and game playing; coevolutionary algorithms and test-based problems. He is an associate editor of the *Genetic Programming and Evolvable Machines* journal, and has participated in research projects at the University of California and the Massachusetts Institute of Technology. For more information, refer to `http://www.cs.put.poznan.pl/kkrawiec`.

Robert Wrembel is an associate professor in the Faculty of Computing, Poznań University of Technology, Poland. His main research area is data warehouse systems. He has been involved in 7 research projects in the area of databases and data warehouses as well as in 7 industrial projects in the field of information technologies. He has prepared and delivered numerous courses in the area of programming languages, database administration, and information systems designing, for multiple companies and institutions in Poland, including Oracle, Microsoft, IBM, BAE Systems. He has been a visiting professor at Loyola University (New Orleans, USA), an invited lecturer at Universidad de Costa Rica (San Jose, Costa Rica), a graduate in the Stanford University post-graduate programme on entrepreneurship and innovation, an intern at the BI company Targit (Tampa, USA), a prizewinner of the IBM Faculty Award for highly competitive research. For more information, refer to `http://www.cs.put.poznan.pl/rwrembel`.

Received: 7 September 2017
Accepted: 9 September 2017