SEAMLESS AUTOMATION AND INTEGRATION OF MACHINE LEARNING CAPABILITIES FOR BIG DATA ANALYTICS

Amril Nazir

Department of Computer Science, Taif University, Saudi Arabia

ABSTRACT

The paper aims at proposing a solution for designing and developing a seamless automation and integration of machine learning capabilities for Big Data with the following requirements: 1) the ability to seamlessly handle and scale very large amount of unstructured and structured data from diversified and heterogeneous sources; 2) the ability to systematically determine the steps and procedures needed for analyzing Big Data datasets based on data characteristics, domain expert inputs, and data pre-processing component; 3) the ability to automatically select the most appropriate libraries and tools to compute and accelerate the machine learning computations; and 4) the ability to perform Big Data analytics with high learning performance, but with minimal human intervention and supervision. The whole focus is to provide a seamless automated and integrated solution which can be effectively used to analyze Big Data with high-frequency and high-dimensional features from different types of data characteristics and different application problem domains, with high accuracy, robustness, and scalability. This paper highlights the research methodologies and research activities that we propose to be conducted by the Big Data researchers and practitioners in order to develop and support seamless automation and integration of machine learning capabilities for Big Data analytics.

KEYWORDS

Big Data, Machine Learning for Big Data, High-frequency Machine Learning, High-Dimension Machine Learning

1. INTRODUCTION

We are entering the era of Big Data — a term that refers to the explosion of available information. Such a Big Data movement is driven by the fact that massive amounts of very high-dimensional or unstructured data are continuously produced and stored with much cheaper cost than they used to be. For example, in analysis of surveillance videos, it is effectively possible to capture images from fifty to over one hundreds video cameras and store them for conducting analysis. This is also true in other areas such as high-frequency financial market data analysis, social media analysis, biomedical imaging, and retail sales. Existing trend that data can be produced and stored more massively and cheaply is likely to maintain or even accelerate in the future. This trend will have deep impact on science, engineering and business.

While the potential benefits of Big Data are real and significant, there remain many technical challenges that must be addressed to fully realize this potential. First, the massive samples in big data are typically aggregated from multiple sources at different time points using different technologies. Nowadays, data is not collected in batch and then processed offline. Instead, data arrives continuously and must be processed online and in real-time to gain useful insight. For example, financial market data need to be continuously streamed and synchronized during the

DOI:10.5121/ijdps.2017.8301

opening market session until the market closes. As new data arrives, the data must synchronize and combine with past historical market data before it can be processed. In some financial market use-case scenarios, multiple analysis must be done simultaneously in different investment period (e.g., minutes, hours, daily, monthly, etc.) and the results must be synchronized to compute the final aggregated output. Hence, there is a need for efficient data-storage methods in storing and managing a dynamic and massive amount of datasets to support such complexity of online data-processing.

Second, handling and managing the sheer size and the characteristics of the data is a major computational challenge. Big Data are often characterized by large sample size and/or high dimensionality. For example, financial market data generates at a high frequency – the data arrival velocity for S&P stock market may generate more than 5,760,000 data instances for only one stock symbol. In another example, a typical imaging dataset is highly dimensional since each image frame of 1024x1024 dimension can be structured into 1,048,576 data attributes and each image is typically sampled at 6 frames per second. In terms of computational efficiency, Big Data motivates the development of new computational infrastructure that can provide high-performance distributed and parallel processing, and fast learning algorithms that are scalable to massive data with high dimensionality.

Third, the challenge relates to the algorithmic instability of current machine learning algorithms in handling large-scale datasets and/or high data dimensionality. Many traditional methods that perform well for moderate sample size do not scale to massive data. Similarly, many algorithms that perform well for low-dimensional data incur low accuracy performance under large volumes and/or high-dimensional data. In particular, when large-scale data is involved, experimental variations and statistical biases emerge and this leads to the need of developing more scalable and robust methods. An efficient mechanism is needed to conduct all the required data pre-processing tasks such as data sampling, data formatting, data cleaning, data transformation, noise elimination, and feature selection. Such tasks are crucial to obtain high accuracy and robustness of the machine learning performance when analyzing large volumes and/or high-dimensional data.

Finally, building a machine learning model for a given dataset involves a series of phases such as data sampling, treating missing value, data transformation, feature selection, machine learning algorithm selection and others. The algorithms and methods that are available in every phase are well described individually by various literatures. However, not every algorithm is best suited for every type of dataset and it is a general consensus that there is no single algorithm in each phase can consistently yield best result when applying to different dataset. It is a crucial task to select the most suitable algorithm to be used in every phase based on the characteristics of the dataset so that the machine learning model being build has the optimum accuracy and robustness for the specific problem. Unfortunately, there is currently no standard framework which is able to act as guidance for choosing the best algorithm in each phase for a given dataset. Current practice for most practitioners to tackle this problem is to test the performance of several algorithms that are well known to them in each phase and select the one which happen to give the best results among the few that are tested. This approach usually leads to suboptimal performance and is very inefficient since only selective algorithms were tested for a given dataset. It is advocated that the machine learning selection to be systematically built by analyzing the problem domain and the given dataset. This phase should be automated and the most appropriate and optimal solution should be determined for the given problem and dataset with minimal intervention from the end user.

Given the sheer complications of developing machine learning capabilities for Big Data, this paper aims to provide research methodologies with a number of proposed research activities that could potentially realize the development of a seamless automation and integration of machine learning capabilities for Big Data mining and analytics.

2. LITERATURE REVIEW

Due to high number of big data producers and high frequency of data generation, the gap between data available to organization and data an organization can process is getting wider all the time as illustrated in Figure 1. It can be observed that the amount of data from an organization will continue to increase across time but there is huge disparity between the amounts of data that the organization can process versus the data generated. As the data volume continues to increase, there is a sudden need for a technology update to keep up with the momentum of data processing and analysis.

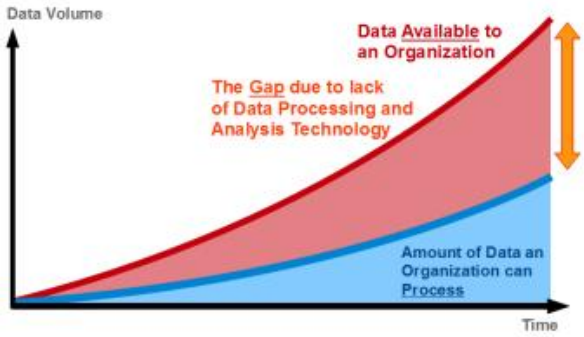

Figure 1: Gap increases due to lack of data processing and analysis technology.

The challenge in automating large-scale machine learning and data analysis for big data lies in the assimilation of data from different sources and processing the data in real-time to derive valuable information. The lack of a common and standardized platform to meet this challenge severely restricts the ability for the domain experts including sensor scientists, security officers, and quantitative financial analysts.

There are many steps involved in building a machine learning model. The first step in building a machine learning model is to define the problem, assumptions and attributes that build logical inferences from the assumptions. This step will determine whether a particular problem is suitable to be addressed using machine learning approach. If the answer is affirmative, a list of assumptions about the problem is specified. They could be rules or domain specific information that needs to be incorporated into the data structure for machine learning model construction. Based on the assumptions, a set of attributes that form logical inferences and class label will be defined. This information will be used in the later stage of the machine learning model construction phases. Typically, domain experts/researchers/practitioners will need to map their problem onto one of the existing machine algorithms, often influenced by their familiarity with specific algorithms and by the availability of corresponding software implementations. This strategy is inefficient and leads to suboptimal performance. There is a strong need to automate the mapping tasks by integrating the various machine learning capabilities.

To our knowledge, there are very few solutions that have been proposed to solve this issue. [1] provide the methodology for analyzing the problem domain and the corresponding data sets for software testing of machine learning applications. They try to determine equivalence classes based on the properties of real-world data sets looking for traits that may not have been considered by the algorithm designers, such as data set size, the potential ranges of attribute and label values, and the sort of precision is expected when dealing with floating point numbers. Alternatively, [2] propose probabilistic graphical models and efficient inference algorithms that offer uniform programming APIs that the domain experts/researchers can use to create highly tailored machine learning models for specific scenarios, as well as rapid prototyping and comparison of a range of alternative learning models. In such way, the practitioners do not have to learn about the huge range of traditional methods, but instead can focus their attention on understanding a single programming. Nonetheless, both solutions do not resolve the issues of determining the correct problem category for a particular problem domain. In particular, it does not determine whether a problem domain belongs to clustering, classification, and/or regression problem. Further, the practitioners are not given the flexibility to provide useful inputs to assist in determining such category.

The machine learning research community has arrived to a consensus that there is no single algorithm that is better than all the others on all the application problem domains [3]. The accuracy of the algorithms is largely influenced by the type of data characteristics. However, it is impossible to support all possible machine learning algorithms due to the vast amount of machine learning algorithms in the literature. For example, it was reported that there are more than 179 implementation of supervised algorithms alone in 2006 [4]. Hence, there is a need for unified and coherent machine learning framework that can analyze Big Data of different data characteristics and different application problem domains. Moreover, machine learning algorithms are heavily influenced by data quality, number of features, weight of features and the amount of pre-processing involved. Since this is the case, given a supervised problem like classification problem and a dataset, fundamental machine learning algorithms like decision trees, semi-supervised learning and ensemble learning algorithms need to be experimentally trained and evaluated in order to find the algorithm which give highest accuracy for the given problem. During the testing phase of the learning algorithm, the parameters of the learning algorithm also need to be tuned so as to get the best results out of the learning algorithms. Several studies have been conducted to make empirical comparison of many of the machine learning algorithms. [5] present a study that compares several learning algorithms (including SVMs) on a handwriting recognition problem using three performance criteria: accuracy, rejection rate, and computational cost. [4] present results from a study that evaluates nearly a dozen learning methods on a real medical data set using both accuracy and an ROC-like metric. However, such tasks require extensive manual intervention and human supervision. Hence, current machine-learning libraries will need to focus on automation of these tasks.

Currently, traditional machine learning software such R [6] and Weka [7] do not provide automated large machine learning capabilities. They only have direct capability to handle small volume and high-dimensional data. Several extensions have been developed by external researchers to provide parallel implementations. Such tools can be categorized into 1) Parallel API bindings method (e.g., rJava [8], pbdR [9], Rmpi [10], and pbdMPI [11]), 2) Parallel Interpreters (i.e., Renjin [12], and Oracle R [13]), and 3) distributed framework (e.g., Storm [14], Spark [15], Mahout [16], etc.). The Parallel API binding's method offers an interface that allows R/Weka code to distribute parallel processes into distributed programming paradigm such as MPI and RPC. On the other hand, parallel interpreters offer a compiler that compile R/Weka code into specific application-specific parallel code in Java, SQL query or others. Further, distributed framework enables R/Weka code to be spawned into multiple processes onto distributed servers. Since the choices are varied, it is not obvious to researchers and practitioners on how to make

good use of such tools. None of the current machine learning tools provide automated and integrated machine learning capabilities.

Therefore, there is a strong need for researchers and practitioners to explore mechanisms of providing a seamless automated and integrated machine learning framework that allows the integration of many of the existing application-specific customized APIs, binding APIs, application interpreters, parallel and distributed libraries. Various mechanisms will need to be explored to examine how such automation and integration can be realized.

3. REQUIREMENT ANALYSIS FOR BIG DATA FRAMEWORK

In order to realize the development of seamless integration of Big Data framework, there are a number of activities and research actions that we propose researchers/practitioners should conduct. Figure 2 demonstrates an overview of our proposed components required for supporting big data analytics. The research challenges involve in automating and integrating the processes between many of the components: Big Data Sources (Repositories), Preprocessing, Problem Domain and Data Understanding, and Machine Learning Algorithm Selection. The components will be described in detail in the following subsections.

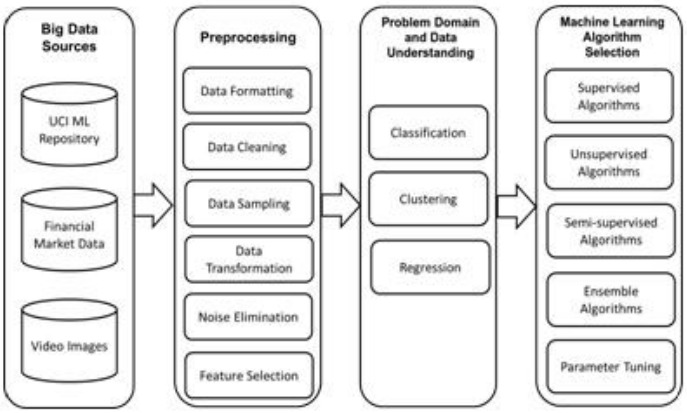

Figure 2: An overview of Proposed Components for Automated Machine Learning Capabilities.

3.1. Big Data Sources: Data Collection and Preparation

Researchers and practitioners will need to employ many different types of datasets when evaluating the Big Data machine learning framework. In particular, they will need to use real datasets to evaluate the effectiveness of their framework implementation. For example, the UCI Machine Learning Repository [17] provides a collection of databases and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. It is currently the world's primary source of machine learning data sets. The repository currently maintains up to 307 data sets with different data characteristics for the machine learning community. Most of the datasets provided by the repository are not considered as Big Datasets since each data set collection comprises not more than 1 million data instances. There are also many sources for image and video datasets. For example, Kaggle provides large files of text, images and videos datasets for research use.

3.2. Big Data Sources: Big Data Extraction, Integration, and Retrieval

First, in the extraction process, big data are sampled and recorded from various data sources. For example, in the case of sensor networks, big data maybe be streamed in real-time from devices, sensors and actuators used in different smart city applications such as transport planning, energy conservation, water and waste management, environmental monitoring, public safety, and health care. In financial domain use-case, big data may be streamed from data providers such as banks, stock exchange, and other financial data providers. In video analysis domain, the images maybe captured via security video camera at the highway, airports, government buildings, military facilities etc. Due to the diversity of the data collected, it is therefore necessary to automatically extract and organize the data for archival, regardless of whether it is real-time data from sensors and other inputs, or data from external systems. Organizing data is also referred to as data integration. Due to the high volume of big data, it is also essential to manage the data effectively to ensure that it can be manipulated in the most flexible way and it should be able to facilitate manipulation of raw data into a large variety of data formats (e.g., unstructured and structured etc.) for the analysis phase. However, there are several research issues and challenges in achieving these.

First, the images from streamed video must be captured and process immediately for the pattern detection and recognition. The actual video cameras are often located at different locations and data volumes may continuously grow. Since typical data mining algorithms require all data to be loaded into the main memory, this is becoming a clear technical barrier because moving data across different locations is expensive (e.g., subject to intensive network communication and other IO costs). Therefore, such communication overhead, data synchronization, and data management should be taken into account. Second, big data acquired directly from the sources are often not readily available for machine learning algorithm to analyze. The big datasets are often sparse, heterogeneous, uncertain, incomplete, missing and derive in multi-source data. Data pre-processing task is required to process the data in an accepted form for machine learning algorithms. However, data pre-processing tasks on big data can take enormous time. Finally, the main challenges on big data mining is on the machine algorithm execution. The execution time can be very time consuming due to the Big Data volumes, large data transfer overheads, and by high-dimensional, high frequency, complex and dynamic data characteristics.

It is important for researchers and practitioners should explore the most effective methods for the collection, integration, and the distribution of real-time high-frequency and high-dimensional data that come from various disparate data sources. Data sources should be varied to include different types of forms such as text, images, and videos (e.g., financial market data and video streaming images).

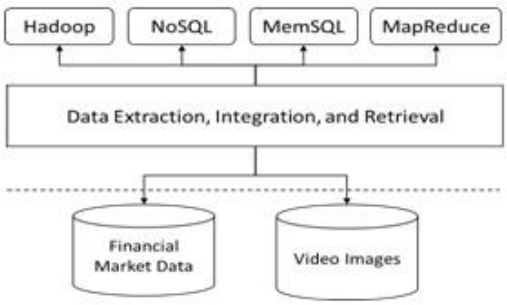

Figure 3: An overview of big data extraction, integration, and retrieval architectural.

Figure 3 demonstrates an overview of the proposed architecture for big data extraction, integration, and retrieval. The most widely-used technology to handle big data is Hadoop [18], an open source project based on Google's MapReduce and Google File System. Hadoop is a distributed batch processing infrastructure which consists of the Hadoop kernel, Hadoop Distributed File System (HDFS), and MapReduce programming framework. There are also alternative technologies such as Spark [15], Vertica [19], and MemSQL [20] databases to acquire, store, and retrieve big data. Notably, it is very important for researchers and practitioners to conduct evaluation of the different alternative middleware available for data extraction, integration, and retrieval. For example, the use of specific software libraries should be evaluated whether they are optimized for exceedingly fast data collection and query execution.

3.3. Data Preprocessing

Based on the input provided by the domain expert, a substantial amount of the computational effort is spent mostly on preprocessing. Incorrect data preprocessing steps will significantly affect the machine algorithm performance.

Figure 4 shows an example of how R and Weka data mining libraries are linked together with the Data Preprocessing component of the framework. The data preprocessing component is highlighted in gray whereas existing data mining software libraries that provide big data support for R and Weka are highlighted in blue. Researchers and practitioners should explore many of the existing libraries and tools that can be used to extend both R and Weka to handle big data. For example, R currently offers a number of methods including Parallel API bindings (i.e., rJava [8], pbdR [9], Rmpi [10], and pbdMPI [11]), Parallel Interpreters (i.e., Renjin [12] and Oracle R [13]), and MapReduce framework (e.g., HadoopStreaming [18] and Rhipe [21]). On the other hand, Weka provides a number methods including Weka direct memory management APIs (i.e., CLI, Knowledge Flow), Java bindings (i.e., Jython [22] and Groovy [23]), MapReduce (i.e., distributedWekaBase [24] and distributedWekaHadoop [25]), and data sampling tool (i.e., Reservoir [26]). Researchers and practitioners should conduct complete evaluation on many of these components to determine the most effective methods to handle big data in both R and Weka. Based on their research findings, various tools, components, and software libraries should be selected to perform the required data pre-processing such as data sampling, data formatting, data cleaning, data transformation, and feature selection.

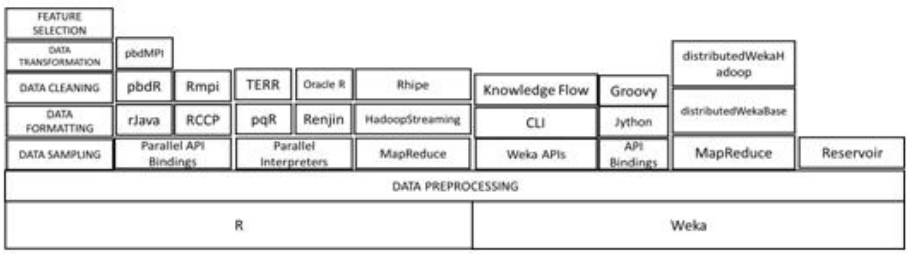

Figure 4: Software stacks of the data preprocessing component in relation to R and Weka libraries.

3.3.1 Data Sampling

Data sampling may be required for three reasons. First, data sampling is able to reduce the computational time for running the algorithm by selecting a subset of data. The selected subset of data must be balance and of suitable size for fast prototyping but is still representative to the original population. More data can result in much longer running times for algorithms since they

require larger computational and memory requirements. Therefore, a smaller representative sample of data can be selected to run the algorithms with fast response. Second, it has been discovered that data sampling can improve the performance accuracy for some algorithms [27, 28, 29]. For example, the data category type of either balance and imbalance dataset can have huge influence on what sampling method should be selected. However, very little known on what sampling method is the most suitable for specific data characteristics. Currently, the machine learning expert is required to conduct full evaluation of the many different type of sampling methods e.g., random sampling, selective sampling, stratified sampling etc. and apply the resulting sampled output to many different type of machine learning algorithms. Such manual process is very time consuming task and error-prone. Therefore, researchers and practitioners should aim to develop a generic framework that should intelligently and automatically determines the type of sampling method required for each different machine learning algorithm. They should explore mechanisms in selecting the appropriate sampling method for different type of machine learning algorithms and different data characteristics. Finally, data sampling can be useful to validate the algorithm using smaller dataset before considering running the whole big dataset on the algorithm, which will take much longer computation time. Hence, researchers and practitioners should also explore the behaviors of a set of machine algorithms under different data scale, ranging from small and large. They should validate through experiments with real datasets on how the machine algorithms behave under different size of datasets.

3.3.2 Data Formatting

The first step of data preprocessing involves data formatting/data conversion. Data formatting is the process of formatting the datasets obtained from various sources to a format which can be used directly by the data mining software e.g., R, Weka, Matlab, C etc. The selected data should be in a format in accordance to: (1) data structure (i.e., data specification) format that the data mining software use, and (2) the data format that the specific algorithm can accept.

Researchers and practitioners should aim to automate the above two requirements. Such automation is a daunting task due to a number reasons. First, data are streamed from multiple sources of many different formats. Since the data can be derived in different formats, how can one ensure that raw data format from many data format can be converted in a standard and uniform way for further processing? Furthermore, every data mining application demands different data structure as its input. For example, Weka format is based on *arff* data structure, R format is based on object-type structure, and Matlab format is based on matrix structure etc.

Second, each data mining software tool comprises different algorithms and not every data mining tool provides every algorithm. For example, Weka does not have fuzzy c-means implementation whereas R provides a generic implementation of the algorithm. Further, even if the same algorithm is provided, each data mining software offers a different variation of algorithm implementations and employ different parameters. Therefore, in the current way, the human expert will need to comprehend the different implementation of each software in order to input the correct parameters. This requires knowledge of the many of the different algorithms available. Instead, this process should be automated based on choices of data mining software. The proposed framework should be able to convert raw data to any of the data specification required by any of the software. Finally, the framework should be sufficiently intelligent to discover what data format each specific algorithm requires, and perform the data conversion appropriately without any manual intervention from the human experts.

3.3.3 Data Cleaning

Data cleaning is the removal or fixing of missing data. Real world data are generally suffer from missing values, outliers and discrepancies in class name. There may be data instances that are incomplete and may affect the resulting accuracy of the algorithm. For example, some classification algorithms are very sensitive to missing values in the data [30, 31]. This could impede the algorithm accuracy. In order to ensure that the algorithm gives the maximum accuracy, the appropriate incomplete instances will need to be removed or fixed. However, automation of data cleaning is not a trivial task. First, different methods of treating missing values can cause the treated dataset to work best on one learning algorithm but not the others. It is not clear what cleaning methods are suitable for specific machine learning algorithm. Hence, this research will explore the literature review on finding the best strategies and algorithms to treat the missing values, identify outliers, smooth out noisy data, detecting and making correction on inconsistent data so that the dataset will be suited for the selected learning algorithms. Researchers and practitioners should also further conduct experiments with real data to validate their findings. The goal of this step is to produce a clean dataset which will enable the machine learning model to attain the highest accuracy and robustness.

3.3.4 Data Transformation

The data transformation process involves several course of action like normalization, creating derived attributes, data discretization and others. Researchers and practitioners should focus on summarizing the available ways of performing data transformation, the pros and cons of the available algorithms, and formulating a framework to detect the needs of conducting transformation to the dataset and the best algorithms to perform the task. The goal is to produce a new dataset which consists of original and/or derived attributes with the attributes value being scaled to fall within a specified range. The new dataset will enjoy the advantages of being able to converge faster during the training phase and allow the model to achieve higher accuracy and robustness.

3.3.5 Noise Elimination

Real world data is not perfect. It is often subject to errors and irregularities that are usually referred to as noise. Noise may be introduced to a dataset due to reasons like data-entry errors, equipment's malfunction during the data collection, transmission and storage phase, inadequacy of information used to label each instance and others. The presence of noise in a dataset will generally reduce the accuracy of the classification algorithms. However different machine learning algorithms have different sensitivity towards the presence of noise in the dataset. Several strategies have been suggested to handle noise data. One strategy is to remove instances that contain class noise [32]. Second strategy is to apply inductive learning algorithms which handle noise implicitly during the machine learning model construction stage [33, 34]. Third approach is to use filtering techniques to detect and eliminate noisy instances before the construction of machine learning model [35]. These filtering methods handle noise explicitly and have the advantage that noisy instances will not negatively affect the structure of the induced model. Researchers and practitioners will need to experimentally test and validate many of these approaches to determine which approaches will actually improve the learning accuracy.

3.3.6 Feature Selection

The feature selection process involves selecting a subset of relevant features from the dataset for use in building the machine learning model. The central conjecture of performing this step is that the original datasets contains many redundant or irrelevant features. This assumption is

reasonable as the dataset being given might come from different sources which originally being collected for some purposes other than the one which is trying to answer by the machine learning model. By performing feature selection when building the machine learning model, the accuracy and robustness of the model can be improved through reducing over fitting. Besides that, the training time can be shortened since there are fewer features in the new dataset. However, for given dataset, different learning algorithms will attain the highest accuracy for different set of features. Researchers and practitioners should aim at formulating a framework that are able to point out the best feature selection algorithms to be used by taking into consideration the characteristics of the dataset and the learning algorithms that will be selected for evaluation in the later stage of machine learning model construction.

The complexity in providing automation feature for the data preprocessing is due to many of intermediary output involved from data formatting, data cleaning, data sampling, data transformation, and feature selection. Several research questions include: what is the appropriate order to conduct the whole data preprocessing process? Should data sampling be conducted after data transformation and feature selection? Should feature selection be conducted without data sampling for increase accuracy and robustness? What is the impact of final machine learning algorithm accuracy when using a specific feature selection algorithm? Despite decade of research in machine learning algorithms, very little known in the literature on the impact of conducting one sub-process of data preprocessing after another. Such evaluation should be experimentally evaluated with real use-case application domains to fully understand its effectiveness. Researchers and practitioners should aim to fill this gap.

3.4. Problem Domain and Data Understanding

The purpose of domain understanding is to gain useful knowledge from the domain expert in order to get a better understanding of the learning task. This will help to reduce the risk and increases the possibility to mine knowledge that is useful, interesting and relevant.

The proposed framework should be automated as much, but domain experts should have the flexibility to intervene the processes if necessary. Domain experts should provide the following input to improve the machine learning capabilities:

- Should one use clustering, classification, and/or regression problem to solve the problem? What is the expected output by the practitioners?
- Which data source(s) and what data parameter(s) to be used? In particular, what data is needed to improve performance? What data can be excluded, which is not important? Removing redundant data can improve accuracy for some algorithms.
- What data that is consider crucial but is not available from the data source. If such data is crucial, are one able to simulate the missing data instead? Is there any alternative parameter that can be used?
- What is the class label? Has it been provided? Even if the class label is provided, is there a need to use class label for machine learning?

There are many factors and issues that need to be considered as can be seen above. First, the domain expert will need to specify the type of problem domain that the algorithms are trying to solve. The problem domain could either belong to clustering, classification, and/or regression problem. The clustering problem domain generally indicates that there is no label data involves, and that the user wishes the algorithms to discover unseen patterns. On the other hand, the classification problem domain generally indicates that historical data are provided as labels, and the user wishes the algorithms to classify future data based on historical labels. Alternatively, the

regression problem domain requires the algorithms to make forecasting of future data value based on historical data. A problem domain may belong to one, some, or all of these categories.

Second, the domain expert should specify data source(s) and the data parameter(s) which are considered to be appropriate to apply the algorithms. Options and flexibility should be provided to select or remove parameters that are not considered suitable from the datasets. Third, the domain expert should have the ability to add new parameter(s) with simulated and/or external data if necessary. Finally, the expert domain should be able to specify the class label for any of the parameters from the dataset.

The biggest challenge is to provide the platform that connects, inter-links and integrates the problem domain category with existing data mining software. For example, Weka has a good range of classification algorithms but is lacking in clustering algorithms. The proposed framework must systematically determine the appropriate data mining software with the most machine learning algorithms to solve a particular problem domain. This requires explorations and experimentation of the many available machine learning algorithms from existing data mining software. Therefore, researchers and practitioners should aim to explore the pros and cons of the widely known data mining and data analysis software (such as R, Weka, Matlab, and possibly C, etc.) with regards to solving different problem domains. The goal is to offer a systematic automation of these tasks as a single platform in a seamless integrated Big Data framework.

Based on the parameters including the label selected, the data can be divided into different categories. For example, it has been reported in the literature that the accuracy of some machine learning algorithms such as Support Vector Machine (SVM) is very sensitive to imbalance dataset [36, 37, 38]. Hence, the dataset can be categorized into the category of either balance or imbalance dataset. Researchers and practitioners should therefore develop a framework with the capability to analyze the characteristic of each algorithm to determine whether it is suitable to run for a particular dataset based on its category. In particular, it should focus on proposing suitable methods that can be used to validate and categorized the datasets. By categorizing the dataset into different category, appropriate algorithms and methods can be selected with ease in the construction of machine learning model and conclusion can also be achieved in a shorter time frame.

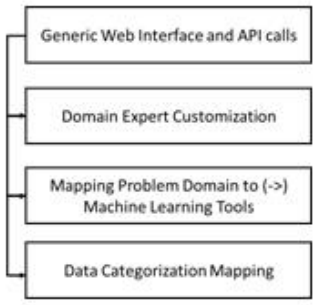

Figure 5: The framework employs a systematic approach to capture useful information on the problem domain and the dataset characteristics.

Figure 5 demonstrates how the framework employs a systematic approach in capture useful information on the problem domain and the dataset characteristics from the domain expert. Researchers and practitioners should aim to offer a generic framework implementation with the following features: (1) the ability for domain experts/users to input additional knowledge the system either by specifying problem category, data source, data parameters, class label etc. (2) to determine which data mining software is best suited for specific problem domain based on their

research exploration of many of the data mining and data analysis software conducted (3) to provide automation of data categorization based on the characteristics of the datasets (4) to expose the framework as a web interface calls and/or generic Application Programming Interfaces(APIs) that can be invoked directly by external applications.

The end goal is for researchers and practitioners to develop a functioning prototype of a customizable and extensible API enabling the above described features in a systematic manner. To our knowledge, software libraries with such features (either commercial and/or open-source) are not currently available.

3.5. Machine Learning Tools and Algorithm Selection

After the data has been prepared and preprocessed, it is now ready to run on the machine learning algorithms for solving problem. The research community has come to the consensus that there is no single learning algorithm that is suitable for all types of dataset. The accuracy of the algorithms is largely influenced by the type of data characteristics. However, it is impossible to support all possible machine due to the vast amount of machine learning algorithms in the literature. For example, [4] reported that there are more than 179 different variations of supervised algorithms implementations. This number continues to grow. Therefore, to begin with, researchers and practitioners should conduct literature reviews on the type of problem domains and the dataset characteristics associated with image data and financial data. There have been several research studies on the evaluation of the most suitable machine learning algorithms for specific application domains and/or data characteristics. Researchers and practitioners should aim to conduct a survey of all such studies. Based on their research findings, they should then explore implementations of such algorithms on well-known data mining software tools primarily, Weka and R. If such algorithms are not supported in Weka and R, researchers and practitioners will also need to consider implementations from other data analysis software such as Matlab, Scikit-learn, and possibly pure C code implementations.

Next, researchers and practitioners should explore various methods for supporting such software to handle big data. Not all of the software can work on large data sets (tera-petabytes of data) due to scalability limitations since they are limited by the non-distributed nature of the implementations. Researchers and practitioners should investigate the various mechanisms for enabling Big Data support for Weka, R, and others. Moreover, researchers and practitioners should explore other recent parallel and distributed machine learning tools such as Mahout [16], RapidMiner [39] and Pentaho [40]. These tools provide parallel machine learning execution support on multi-core physical machines. More recent Big Data libraries and tools also include Spark [15], Twister [41], HaLoop [18], Hama [42], GraphLab [43], Jubatus [44], and many others. They all provide the capability to perform distributed machine learning execution on multiple heterogeneous machines. There is a wide variety of choices but these tools offer overlapping features, which make the conventional framework overly complex. Figure 6 demonstrates existing machine learning software libraries and tools that can be used to support big data machine learning analysis in the framework. The end goal of the Big Data framework is to simplify the usage of such tools, and to provide a coherent and uniform interface that allows the integration of many of the existing application-specific customized APIs, binding APIs, application interpreters, parallel and distributed libraries for supporting fast execution of the machine learning algorithms. There is a strong need to develop a framework with clearly defined separation of concerns, with flexible bindings between each component.

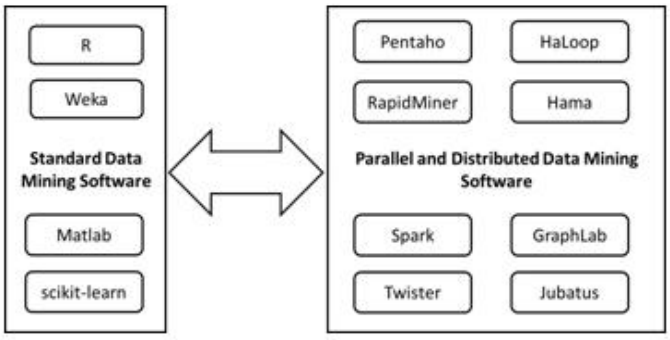

Figure 6: Standard and Parallel & Distributed Data Mining Software.

4.DESIGN & IMPLEMENTATION: SEAMLESS AUTOMATED AND INTEGRATED FRAMEWORK

Having all the components in place, it is therefore crucial to integrate all these components in seamless manner. For the design & implementation state, efforts should be made to design and implement the process execution flow for each component. The design should aim to support the following features: (1) systematically selects the most suitable machine learning algorithm(s) to perform analytics for a given problem domain (based on the information from data characteristics, domain expert input, and data pre-processing component), (2) systematically selects the most suitable machine learning software tools to execute such algorithm(s), and (3) accelerates machine learning computations that scale with large datasets.

Figure 7 demonstrates an example of how Big Data machine learning framework can be integrated seamlessly, which comprises various existing machine learning software libraries and tools that interconnect together. For example, the Data Sampling component may select the algorithm to use based on the data mining algorithm (e.g., Decision Tree/SVM, EM/K-Mean). Based on the Data Sampling rule, either a Random or Spatial sampling method might be selected if the algorithm chosen for the problem domain is either EM or K-Mean. Further, the data mining task (i.e., classification, regression, clustering) will be selected based on the chosen data sampling algorithm. Next, the proposed framework should determine the most appropriate method to use for handling missing value treatment. In an example given in Figure 7, an EM imputation algorithm may be used or other available algorithms such as Mean/Mode imputation may also be applied. These choices must be conducted automatically by the framework based on the problem domain.

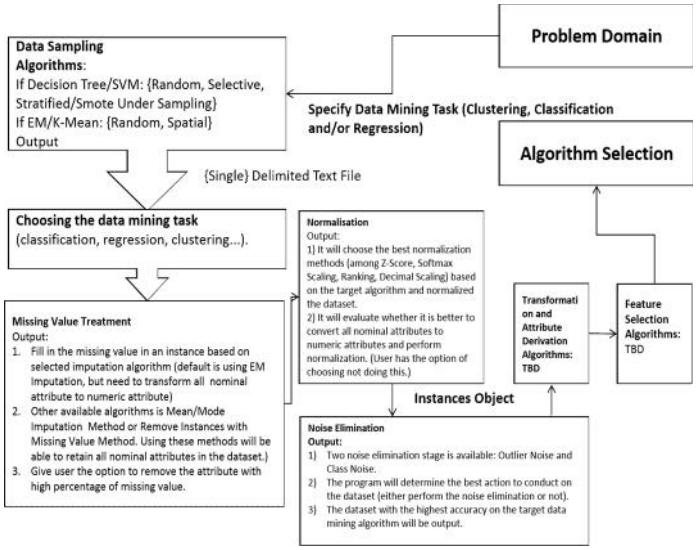

Figure 7: Process Execution Flows between Framework Components.

Moreover, the choices of software tools also play crucial role in ensuring a seamless integration of the design and implementation. Figure 8 further demonstrates an example of how various existing machine learning software libraries and tools can be integrated to support seamless integration of machine learning capabilities for Big Data in our framework. It can be observed that there are many different categories of software tools. In this example, Solr [45] and SpagoBI [46] are used as the visualization platforms. For the analysis part, the Storm, Mahout and Spark can be integrated together to support different types of analysis problem domains such as clustering, classification, and regressions. Furthermore, Management components can be used to control the information flows between other components using Oozie [47], Sentry [48], Zookeeper [49] and Cloudera Manager [50]. To support Big Data extraction and retrieval, the framework may employ both RDD and conventional RDBMS as storage solutions, which can be integrated with Sqoop [51] and YARN/HDFS [52]. The connections of various RDD and RDBMS storage solutions can be handled by Hbase [53], Pig [54], Hive [55], Impala [50] and MongoDB [56]. The framework could also support both CPU and GPU (Graphics Processing Unit) to provide accelerated computations for high-frequency and high-dimensional datasets. The CPU and GPU can be utilized by High Performance Computing libraries such as TensorFlow [57] (for deep learning), HPCC clusters, and parallel libraries of Spark.

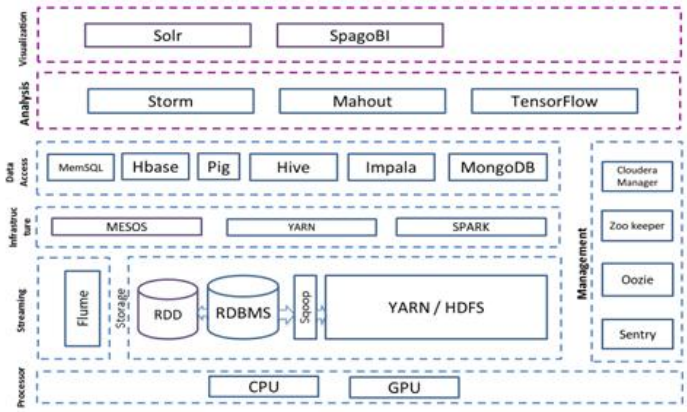

Figure 8: Our Proposed Seamless Integration of Machine Learning Capabilities for Big Data.

5. EVALUATION

In final phase, researchers and practitioners should aim to test and validate many of the different ways to integrate the data extraction, and data pre-processing, and machine learning algorithm selection tasks. The goal of the final integration validation is to construct the most effective information flow between each component and task.

Researchers and practitioners should conduct a set of experiments to evaluate that the framework based on three important metrics: (1) accuracy, (2) robustness, and (3) scalability. The accuracy refers to how well the final output is able to perform based on the problem domain. For example, the accuracy performance of supervised algorithms can be analyzed based on the correct predictions on the test data.

However, the accuracy of unsupervised algorithms is not straight forward to measure. It requires inspection on how well the output of data instances were partitioned into a number of clusters, and evaluation on the output is required to determine the effectiveness of the algorithm. Therefore, to provide automation on the performance measurements for the framework, this will require some further research explorations.

Robustness test is also crucial to validate that accuracy of the algorithm can continue to sustain with other similar data characteristics and different data size. However, validating the robustness of the algorithm would require extensive experimental run with many different datasets that will need to be automated. The scalability refers to the ability for the framework to maintain high accuracy for a given problem domain regardless of the dataset size. It is possible that high accuracy can be achieved for a smaller dataset but fails completely on Big Datasets. Hence, the selected machine learning algorithm should be tested and validated on real datasets to validate its effectiveness and scalability.

6. CONCLUSION

In this paper, we have proposed research methodologies with a number of suggested research activities that researchers and practitioners should focus on in developing a novel framework that offer automated machine learning capabilities to systematically determine the steps and procedures needed to arrive at the best learning and/or prediction performance for a given problem domain and dataset. The specific objectives are as follows: (1) design and implement a framework for supporting big data extraction, big data integration, big data retrieval, and big data machine learning analytics tasks, (2) propose solutions for the essential phases of data pre-processing including data sampling, data formatting, data cleaning, data transformation, noise elimination and feature selection, and evaluate the proposed solutions in terms of accuracy, robustness, scalability and computational performance, and (3) evaluate the performance of the proposed automated big data mining and machine-learning framework and assess the benefit and applicability of the proposed framework for generic problem domains and specific application/problem domains.

Some merits of the above are as follows: (1) domain experts, researchers and practitioners do not have to learn about the huge range of traditional machine learning methods and algorithms, but instead can focus their attention on understanding a problem domain. This avoids the need for them to explore and experiment with many machine learning algorithms for every new application/problem domain, which is a time-consuming process and very ineffective, (2) such framework can be employed to perform automated analysis on a wide variety of application and problem domains, and (3) such framework provides the flexibility to handle different types of big

data characteristics from variety of application domains thus allowing domain experts, researchers, and practitioners to focus on other important tasks, and finally (4) such framework provides the ability to mine data automatically, discover problem domains, determines the most optimal machine learning algorithm systematically, and tunes the best parameters to give the best accuracy and robustness, which can save considerable expense from the trial-and-error machine learning experimentations.

REFERENCES

[1] Christian Murphy, Gail E Kaiser, Marta Arias: "An approach to software testing of machine learning applications", , 2007.

[2] Christopher M Bishop: "Model-based machine learning", Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, pp. 20120222, 2013.

[3] David H Wolpert: "The lack of a priori distinctions between learning algorithms", Neural computation, pp. 1341—1390, 1996.

[4] Gregory F Cooper, Constantin F Aliferis, Richard Ambrosino, John Aronis, Bruce G Buchanan, Richard Caruana, Michael J Fine, Clark Glymour, Geoffrey Gordon, Barbara H Hanusa, others: "An evaluation of machine-learning methods for predicting pneumonia mortality", Artificial Intelligence in Medicine, pp. 107—138, 1997.

[5] Yann LeCun, LD Jackel, L Bottou, A Brunot, C Cortes, JS Denker, H Drucker, I Guyon, UA Muller, E Sackinger, others: "Comparison of learning algorithms for handwritten digit recognition", International conference on artificial neural networks, pp. 53—60, 1995.

[6] Ross Ihaka, Robert Gentleman: "R: a language for data analysis and graphics", Journal of computational and graphical statistics, pp. 299—314, 1996.

[7] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H Witten: "The WEKA data mining software: an update", ACM SIGKDD explorations newsletter, pp. 10—18, 2009.

[8] Simon Urbanek: rJava: Low-level R to Java interface. R package version 0.9-6, URL http://CRAN. R-project. org/package= rJava, 2013.

[9] Andrew M Raim: Introduction to Distributed Computing with pbdR at the UMBC High Performance Computing Facility, 2013.

[10] James Psota, Anant Agarwal: rMPI: message passing on multicore processors with on-chip interconnect in High Performance Embedded Architectures and Compilers. Springer, 2008.

[11] WC Chen, G Ostrouchov, D Schmidt, P Patel, H Yu: "pbdMPI: Programming with Big Data—Interface to MPI", R Package, URL http://cran. r-project. org/package= pbdMPI, 2012.

[12] Tomas Kalibera, Petr Maj, Floreal Morandat, Jan Vitek: "A fast abstract syntax tree interpreter for R", Proceedings of the 10th ACM SIGPLAN/SIGOPS international conference on Virtual execution environments, pp. 89—102, 2014.

[13] OracleR.http://www.oracle.com/technetwork/database/database-technologies/r/r-enterprise/overview/index.html, 2015.

[14] Wenjie Yang, Xingang Liu, Lan Zhang, Laurence T Yang: "Big Data Real-Time Processing Based on Storm", Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on, pp. 1784—1787, 2013.

[15] Matei Zaharia, Mosharaf Chowdhury, Michael J Franklin, Scott Shenker, Ion Stoica: "Spark: cluster computing with working sets", Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, pp. 10—10, 2010.

[16] Sean Owen, Robin Anil, Ted Dunning, Ellen Friedman: Mahout in action. Manning, 2011.

[17] Arthur Asuncion, David Newman: UCI machine learning repository. 2007.

[18] Yingyi Bu, Bill Howe, Magdalena Balazinska, Michael D Ernst: "HaLoop: efficient iterative data processing on large clusters", Proceedings of the VLDB Endowment, pp. 285—296, 2010.

[19] Andrew Lamb, Matt Fuller, Ramakrishna Varadarajan, Nga Tran, Ben Vandiver, Lyric Doshi, Chuck Bear: "The vertica analytic database: C-store 7 years later", Proceedings of the VLDB Endowment, pp. 1790—1801, 2012.

[20] Nikita Shamgunov: "The MemSQL In-Memory Database System", Proceedings of the 2nd International Workshop on In Memory Data Management and Analytics, IMDM 2014, Hangzhou, China, September 1, 2014., 2014.

[21] S. Guha, R. Hafen, J. Rounds, J. Xia, J. Li, B. Xi, W. S. Cleveland: “Large Complex Data: Divide and Recombine (D&R) with RHIPE”, Stat, 2012, to appear.
[22] Sergei V Chekanov: Scientific data analysis using Jython Scripting and Java. Springer Science & Business Media, 2010.
[23] Dierk Koenig, Andrew Glover, Paul King, Guillaume Laforge, Jon Skeet: Groovy in action. Manning, 2007.
[24] J Uma Mahesh, R Madan Mohan: “Real World Application of Big Data in Data Mining Tools”.
[25] Aditya B Patel, Manashvi Birla, Ushma Nair: “Addressing big data problem using Hadoop and Map Reduce”, Engineering (NUiCONE), 2012 Nirma University International Conference on, pp. 1—5, 2012.
[26] Brian Babcock, Mayur Datar, Rajeev Motwani: “Sampling from a moving window over streaming data”, Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms, pp. 633—634, 2002.
[27] Nele Verbiest, Enislay Ramentol, Chris Cornelis, Francisco Herrera: “Preprocessing noisy imbalanced datasets using SMOTE enhanced with fuzzy rough prototype selection”, Applied Soft Computing, pp. 511—517, 2014.
[28] Guohua Liang, Chengqi Zhang: “An efficient and simple under-sampling technique for imbalanced time series classification”, Proceedings of the 21st ACM international conference on Information and knowledge management, pp. 2339—2342, 2012.
[29] Wei Pan, Kun She, Pengyuan Wei, Kai Zeng: Nearest Neighbor Condensation Based on Fuzzy Rough Set for Classification in Rough Sets and Knowledge Technology. Springer, 2014.
[30] Yufeng Ding, Jeffrey S Simonoff: “An investigation of missing data methods for classification trees applied to binary response data”, Journal of Machine Learning Research, pp. 131—170, 2010.
[31] Qiang Yang, Charles Ling, Xiaoyong Chai, Rong Pan: “Test-cost sensitive classification on data with missing values”, IEEE Transactions on Knowledge and Data Engineering, pp. 626—638, 2006.
[32] Senzhang Wang, Zhoujun Li, Wenhan Chao, Qinghua Cao: “Applying adaptive over-sampling technique based on data density and cost-sensitive SVM to imbalanced learning”, Neural Networks (IJCNN), The 2012 International Joint Conference on, pp. 1—8, 2012.
[33] Dragan Gamberger, Nada Lavrač, Sašo Džeroski: “Noise elimination in inductive concept learning: A case study in medical diagnosis”, International Workshop on Algorithmic Learning Theory, pp. 199—212, 1996.
[34] Dragan Gamberger, Nada Lavrac, Ciril Groselj: “Experiments with noise filtering in a medical domain”, ICML, pp. 143—151, 1999.
[35] Sofie Verbaeten, Anneleen Van Assche: “Ensemble methods for noise elimination in classification problems”, International Workshop on Multiple Classifier Systems, pp. 317—325, 2003.
[36] Rehan Akbani, Stephen Kwek, Nathalie Japkowicz: “Applying support vector machines to imbalanced datasets”, European conference on machine learning, pp. 39—50, 2004.
[37] Yuchun Tang, Yan-Qing Zhang, Nitesh V Chawla, Sven Krasser: “SVMs modeling for highly imbalanced classification”, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), pp. 281—288, 2009.
[38] Konstantinos Veropoulos, Colin Campbell, Nello Cristianini, others: “Controlling the sensitivity of support vector machines”, Proceedings of the international joint conference on AI, pp. 55—60, 1999.
[39] Markus Hofmann, Ralf Klinkenberg: RapidMiner: Data mining use cases and business analytics applications. CRC Press, 2013.
[40] Roland Bouman, Jos Van Dongen: Pentaho solutions: business intelligence and data warehousing with Pentaho and MySQL. Wiley Publishing, 2009.
[41] Jaliya Ekanayake, Hui Li, Bingjing Zhang, Thilina Gunarathne, Seung-Hee Bae, Judy Qiu, Geoffrey Fox: “Twister: a runtime for iterative mapreduce”, Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, pp. 810—818, 2010.
[42] Sangwon Seo, Edward J Yoon, Jaehong Kim, Seongwook Jin, Jin-Soo Kim, Seungryoul Maeng: “Hama: An efficient matrix computation with the mapreduce framework”, Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on, pp. 721—726, 2010.
[43] Yucheng Low, Joseph E Gonzalez, Aapo Kyrola, Danny Bickson, Carlos E Guestrin, Joseph Hellerstein: “Graphlab: A new framework for parallel machine learning”, arXiv preprint arXiv:1408.2041, 2014.

[44] Shohei Hido, Seiya Tokui, Satoshi Oda: “Jubatus: An Open Source Platform for Distributed Online Machine Learning”, NIPS 2013 Workshop on Big Learning, Lake Tahoe, 2013.
[45] Trey Grainger, Timothy Potter, Yonik Seeley: Solr in action. Manning Cherry Hill, 2014.
[46] Andrea Gioia, Grazia Cazzin, Ernesto Damiani: “SpagoBI: A distinctive approach in open source business intelligence”, Digital Ecosystems and Technologies, 2008. DEST 2008. 2nd IEEE International Conference on, pp. 592—595, 2008.
[47] Mohammad Islam, Angelo K Huang, Mohamed Battisha, Michelle Chiang, Santhosh Srinivasan, Craig Peters, Andreas Neumann, Alejandro Abdelnur: “Oozie: towards a scalable workflow management system for hadoop”, Proceedings of the 1st ACM SIGMOD Workshop on Scalable Workflow Execution Engines and Technologies, pp. 4, 2012.
[48] Masoumeh Rezaei Jam, Leili Mohammad Khanli, Morteza Sargolzaei Javan, Mohammad Kazem Akbari: “A survey on security of Hadoop”, Computer and Knowledge Engineering (ICCKE), 2014 4th International eConference on, pp. 716—721, 2014.
[49] Patrick Hunt, Mahadev Konar, Flavio Paiva Junqueira, Benjamin Reed: “ZooKeeper: Wait-free Coordination for Internet-scale Systems.”, USENIX annual technical conference, pp. 9, 2010.
[50] Marcel Kornacker, Justin Erickson: “Cloudera Impala: Real Time Queries in Apache Hadoop, For Real”, ht tp://blog. cloudera. com/blog/2012/10/cloudera-impala-real-time-queries-in-apache-hadoop-for-real, 2012.
[51] Kathleen Ting, Jarek Jarcec Cecho: Apache Sqoop Cookbook. " O'Reilly Media, Inc.", 2013.
[52] Vinod Kumar Vavilapalli, Arun C Murthy, Chris Douglas, Sharad Agarwal, Mahadev Konar, Robert Evans, Thomas Graves, Jason Lowe, Hitesh Shah, Siddharth Seth, others: “Apache hadoop yarn: Yet another resource negotiator”, Proceedings of the 4th annual Symposium on Cloud Computing, pp. 5, 2013.
[53] Mehul Nalin Vora: “Hadoop-HBase for large-scale data”, Computer science and network technology (ICCSNT), 2011 international conference on, pp. 601—605, 2011.
[54] Christopher Olston, Benjamin Reed, Utkarsh Srivastava, Ravi Kumar, Andrew Tomkins: “Pig latin: a not-so-foreign language for data processing”, Proceedings of the 2008 ACM SIGMOD international conference on Management of data, pp. 1099—1110, 2008.
[55] Ashish Thusoo, Joydeep Sen Sarma, Namit Jain, Zheng Shao, Prasad Chakka, Suresh Anthony, Hao Liu, Pete Wyckoff, Raghotham Murthy: “Hive: a warehousing solution over a map-reduce framework”, Proceedings of the VLDB Endowment, pp. 1626—1629, 2009.
[56] Alexandru Boicea, Florin Radulescu, Laura Ioana Agapin: “MongoDB vs Oracle-Database Comparison.”, EIDWT, pp. 330—335, 2012.
[57] Martın Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, others: “Tensorflow: Large-scale machine learning on heterogeneous distributed systems”, arXiv preprint arXiv:1603.04467, 2016.