An Architecture for Sentiment Analysis in Twitter

Michele Di Capua, Emanuel Di Nardo, Alfredo Petrosino

Abstract: *Social network has gained great attention in the last decade. Using social network sites such as Twitter through the internet and the web 2.0 technologies has become more affordable. The heavy reliance on social networks causes them to generate massive data characterised by some computational issues as: size, noise and reliability. These issues make social network data complex to analyse manually, resulting in the adoption of computational tools. In this paper we discuss a recent software architecture, named lambda-architecture, modified with the introduction of machine learning components, in order to perform sentiment analysis on big data streams, as the one provided by the Twitter social network.*

Key words: *E-learning, Sentiment analysis, Opinion mining, Big data, Social network, Lambda architecture.*

INTRODUCTION

The incorporation of adaptation methods and techniques allows the development of adaptive e-learning systems, where each student receives personalized guidance during the learning process (Brusilovsky, 2001). In order to provide personalization, it is necessary to store information about each student in what is called the student model.

The specific information to be collected and stored depends on the goals of the adaptive e-learning system (e.g., preferences, learning styles, personality, emotional state, context, previous actions, and so on). In particular, affective and emotional factors, among other aspects, seem to affect the student motivation and, in general, the outcome of the learning process (Shen, Wang, & Shen, 2012).

Therefore, in learning contexts, being able to detect and manage information about the students’ emotions at a certain time can contribute to know their potential needs at that time. On one hand, adaptive e-learning environments can make use of this information to fulfill those needs at runtime: they can provide the user with recommendations about activities to tackle or contents to interact with, adapted to his/her emotional state at that time. On the other hand, information about the student emotions towards a course can act as feedback for the teacher. This is especially useful for online courses, in which there is little (or none) face-to-face contact between students and teachers and, therefore, there are fewer opportunities for teachers to get feedback from the students.

In general, in order for a system to be able to take decisions based on information about the users, it is necessary for it to get and store information about them. One of the most traditional procedures to obtain information about users consists of asking them to fill in questionnaires. However, the users can find this task too time-consuming. Recently, non intrusive techniques are preferred, yet without compromising the reliability of the model built.

There are different ways to provide sentiment-based support to students – see as instance CoMoLE (Martín, Carro, Rodríguez, 2007). One of them deals with the possibility of proposing motivational tasks to students whose emotional state is detected as negative, when guiding each of them. The motivational activities may be included; they are activities whose goal can be no other than motivating the students (e.g., games, simulations, etc.). In such a way, when a student connects to the system to receive personalized advice about which educational activity to be carried out next (according not only to his/her progress but also to his/her needs and context at that time), if the system detects that he/she has a negative sentiment, then it can propose him/her a motivational activity to try to engage him/her first. The only requirements for adaptive systems to incorporate emotion-based adaptation are to include the user emotion as a new attribute in the user model, to feed this model with the results of the sentiment analysis, and to specify the adaptation/recommendation criteria based on this attribute, according to the desired support to be given to the user emotions.

Another application of sentiment analysis in adaptive e-learning systems is related to collaborative learning. The activities to be proposed to each group of students can vary depending on the group features, actions or context. Therefore, knowing the group sentiment makes it possible, for example, that when a workgroup show rather negative emotions towards a course, the system carries out specific motivational actions intended to encourage the group.

This paper reports the activities of the group in sentiment analysis to be engaged in an e-learning system. Specifically, we report the designed architecture to process data and the learning procedures adopted to design the adaptive e-learning framework. Based on the experience of the group and the actual trends in learning, we describe why and how deep learning architectures have been adopted to this aim.

SOCIAL NETWORKS

Nowadays social networks provide an important source of information. In addition to the common use, they are also used by companies and researchers to extract information that is usually not visible to the naked eye. Within the most important social platforms we can found Facebook, Twitter and YouTube. Inside these platforms a daily analysis is typically conducted over the users and over the way the platforms are used, for research activities and marketing purposes. Among all the social platforms available, Twitter can be considered a really big source of information, publicly readable. Twitter is a free microblogging service, created in the 2006 by the Obvious Corporation in San Francisco, which give the users the possibility to update a personal web page, with text messages (tweets) made by at most 140 chars. Twitter provides several techniques to recover messages published by users, and these messages (more than half a billion by day), are commonly used worldwide by different research institutions to perform sentiment analysis with excellent results.

Since Twitter allows users to write a maximum of 140 characters for each tweet people are forced to share only essential information about the topic or subject they want to discuss.

This limitation (very short documents) represents a big challenge in the sentiment analysis area due to the fact that the most common adopted techniques for text analysis perform better with long documents. Twitter users commonly adopt an informal language in posting messages, with many slang words and acronym, also due to the limitation imposed by the platform. Another trend in social networks communication is related to a massive use of emoticons (graphic symbols or recently also images), to immediately express feeling over a particular topic or news. To mitigate these problems we introduce some techniques, derived from Natural Processing, in order to pre-process tweets data, and obtain a more clean input sentence. For example, we use some publicly available slang words list or acronym list to replace ambiguous terms. We also use a weighted list of emoticons, with a pre assigned sentiment polarity, to measure the overall sentiment value of the sentence.

Twitter has also some other functions that users can apply to characterize their message:

- hashtags, that can be used to relate a message to a certain topic
- usertag, that can be used to send a message to a specific user in a public way

These features could be used, in the processing phase, to filter more efficiently post, i.e. in grouping together tweets related to the same topic or course.

The personal Twitter page of each user can be updated in different ways. Actually, apart from the classical web page, a user can post messages to its blog from mobile phone with dedicated mobile client application, from SMS (short message) and programmatically also from a rich API provided by Twitter itself.

ARCHITECTURE

The definition and modelling of an architecture dedicated to the activities of analysis of big data, as the ones produced by social networks as Twitter, is currently still at an early stage of its development and consolidation. Unlike traditional data warehouse or business intelligence systems, whose architecture is designed for structured data, systems dedicated to big data work instead with semi-structured data, or so called "raw data", i.e. without a particular structure. It should also be pointed out that such systems should be able to allow processing and analysis of data not only in batch mode, but also in a real real-time fashion.

Nowadays a huge amount of data, daily produced by social networks, can be processed and analyzed for different purposes. These data are provided with several features, among which:

- dimension;
- peculiarities;
- source;
- reliability;

By the time the need to obtain the information and the way this information must be processed has changed. Until recently it was thought that the data should be first processed and subsequently made available, regardless of the time aspect. This type of processing is commonly called batch processing. Nowadays the amount of data is increased exponentially and now real-time processing is needed to get the most advantages from this data, in different fields.

Actually batch models do not allow to work with the data in a real time fashion, due to the long time required by processing operations. Against the implementation of real-time processing architecture could lead to lower accuracy One possible solution is to merge the two concepts into a single architecture, capable of handling big data, but also with scalable and fast processing features.

A possible solution to this problem is the so called Lambda Architecture (Marz and Warren, 2015), a software architecture made by 3 different levels:

- Batch layer;
- Speed layer;
- Serving layer;

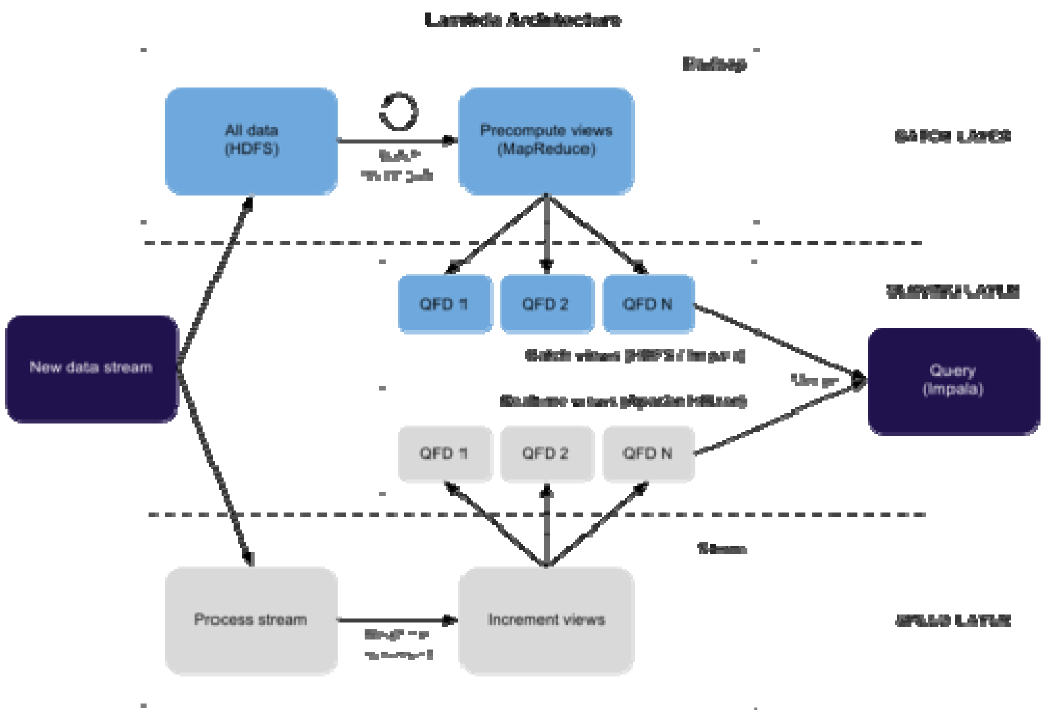

Figure: Layer organization of a Lambda Architecture

Batch Layer

This level is responsible to store the input data in a master dataset structure. These data are periodically processed, generally with a Map Reduce approach (Berlinska, 2011). The batch layer precomputes results using a distributed processing system that can handle very large quantities of data. The batch layer aims at perfect accuracy by being able to process all available data when generating views. This means it can fix any errors by recomputing based on the complete data set, then updating existing views. Output is typically stored in a read-only database, with updates completely replacing existing precomputed views. Several frameworks can be used at this level, but the two most adopted are:

a) ***Apache Hadoop*** can be used for the Map Reduce operations. It uses a proprietary file system called Hadoop Distributed File System (HDFS), which reads and writes using customizable I/O drivers. It's a distributed file system with also failure-tolerant capabilities. It's available in different language implementations as Java and Python.

b) ***Apache Spark*** can be used for the Map Reduce operations but also for general-purpose task. It's able to perform on memory operation and it supports also HDFS file system HDFS and the Hadoop drivers. It's distributed and fault tolerant. It came with different libraries for the view data generation, micro-batching, machine learning and also graph based analysis operations.

Speed Layer

This layer processes data streams in real time. This layer sacrifices throughput as it aims to minimize latency by providing real-time views into the most recent data. Essentially, the speed layer is responsible for filling the "gap" caused by the batch layer's lag in providing views based on the most recent data. This layer's views may not be as accurate or complete as the ones eventually produced by the batch layer, but they are available almost immediately after data is received, and can be replaced when the batch layer's views for the same data become available. As for the batch layer several frameworks, based on stream processing technologies, can be used at this level. The most adopted are:

a) ***Apache Storm*** is based on the idea of streaming computation, i.e. processing new data individually. It provides support for micro-batching operations, collecting new data in small data set and then executing operations on the whole dataset with the Trident API. It's distributed and it uses Apache Zookeeper for this task. The core abstraction in Storm is the "stream". A stream is an unbounded sequence of tuples. Storm provides the primitives for transforming a stream into a new stream in a distributed and reliable way. For example, you may transform a stream of tweets into a stream of trending topics. The basic primitives Storm provides for doing stream transformations are "spouts" and "bolts". Spouts and bolts have interfaces that you implement to run your application-specific logic. A spout is a source of streams. For example, a spout may connect to the Twitter API and emit a stream of tweets. A bolt consumes any number of input streams, does some processing, and possibly emits new streams. Complex stream transformations, like computing a stream of trending topics from a stream of tweets, require multiple steps and thus multiple bolts. Bolts can do anything from run functions, filter tuples, do streaming aggregations, do streaming joins, talk to databases, and more.
   Networks of spouts and bolts are packaged into a "topology" which is the top-level abstraction that you submit to Storm clusters for execution. A topology is a graph of stream transformations where each node is a spout or bolt. Edges in the graph indicate which bolts are subscribing to which streams. When a spout or bolt emits a tuple to a stream, it sends the tuple to every bolt that subscribed to that stream.

b) ***Apache Spark Streaming*** is an extension of the core Spark API that enables scalable, high-throughput, fault-tolerant stream processing of live data streams. Data can be ingested from many sources like Twitter, and can be processed using complex algorithms expressed with high-level functions like map, reduce, join and window. Finally, processed data can be pushed out to filesystems, databases, and live dashboards. It's also possible to apply Spark's machine learning and graph processing algorithms on data streams..

Serving Layer

This layer takes care of the output task, joining the results of the Batch and Speed layer, in order to obtain a single view of the data. A critical task at this level is related to data synchronization activities. It's worth here to integrate a storage engine capable of executing random reads, bulk writes, with a low latency and also in a distributed manner.

Typically these kind of operations are well performed in storage organized in key/value, such as: ElephantDB, Redis, Hbase, Druid, and Elastic search.

DEEP LEARNING FOR SENTIMENT ANALYSYS

Sentiment analysis, also called opinion mining, is the field of study that analyses people's opinions, sentiments, evaluations, appraisals, attitudes, and emotions towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes (Bing Liu, 2012).

There are two main applied techniques for sentiment analysis activity: machine learning based and lexicon based. Few research studies have also combined this two methods to gain relatively better performance and results. However, both the approaches used in literature so far have shown two fundamental problems:

1) In the context of machine learning techniques along with the support of techniques derived from NLP (Natural Language Processing), the algorithms developed work usually on data that are rendered in formats tailored to the particular algorithm

developed. For example in the area of text classification, one commonly adopted solution is to use unordered lists of words (called bag of words), ignoring thereby the relations between the words of a sentence and the grammatical structure of the sentence itself. Especially in the field of sentiment analysis, the correlation between the words close together can dramatically change the meaning of a sentence in context. In summary a major problem of the solutions adopted NLP concerns about simplifying assumptions of phrases and language analyzed.

2) Another fundamental problem relates to the representation of the features. The secret of many analytical systems depends on the type of representation of the features used, such as the choice of named entities (i.e. persons or organizations), or the use of POS (part of speech) tagging. The use of ad hoc features involves consuming computational resources and makes the system or algorithm developed less flexible for purposes other than those for which it was realized.

Therefore it's desirable to adopt a general representation of sentences, but without losing information on the grammatical structure in which these sentences are presented (Irsoy & Cardie, 2014).

In this work we have explored an architecture capable of processing large amounts of data from Twitter. For the processing phase, in order to perform sentiment analysis on data, we propose to integrate in the same software architecture, some software components that implements techniques of machine learning together with NLP algorithms.

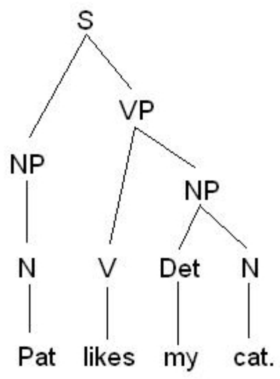

Figure: Syntactic (and morphological) structure depicted with a tree diagram

More in details we propose to adopt a lambda architecture and to use, at the batch level of the architecture, a recursive neural network model, based on a pre-trained Kohonen SOM (Self Organized Map).

A SOM is a neural network that is able to map data input received into different corresponding regions, with different regions (clusters) having different response characteristics for corresponding input mode. Generally, SOM has proven to be an efficient and suitable documents clustering method. It can map documents onto two-dimensional diagram to show the relationship between the different documents. SOM can depict text in more figurative and better visual way. Text Clustering is a high-dimensional application and closely related to the semantic features. The above characteristics of SOM make it very suitable for text clustering and subsequent sentiment analysis. By inputting a document, usually using a vector representation of its words, the neurons representing the pattern class-specific (sentiment related) in the output layer will have the greatest response.

Recursion is obtained by applying the same neural network at each node of the grammatical structure, represented by the semantic tree of the sentence. Grammatical structures, where used, can solve the problem of so-called "propositional attachment". The details on the machine learning algorithms and techniques applied here is not covered in this paper, but we simply focus our attention on the architectural components of the solution.

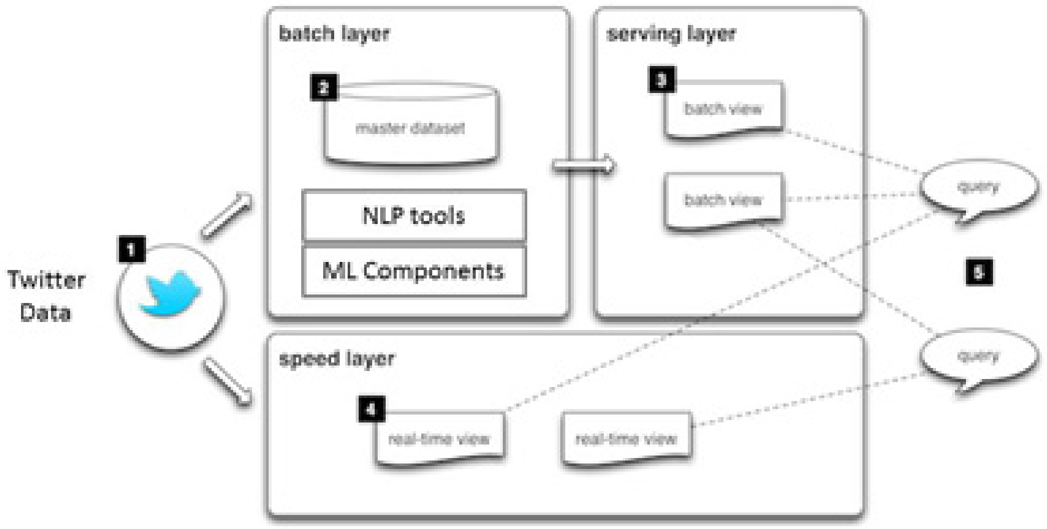

Figure: Modified Lambda Architecture for Sentiment Analysis

The models of recursive networks, not only are able to predict this type of structure of sentences, but can also learn the meaning of a composition of words within a tweet. Thus some problem can be solved problems such as learning vector of features for variable input size, without ignoring the structure or sequence of words, as it appears in the sentence. Most of these deep models are able to learn also compositional semantics, simply starting from training data without having any manual description of features.

Preliminary results testing this architecture on tweets streaming data seems to be promising. The tweets are obtained and submitted to the nodes of the architecture using the Twitter Streaming API, with some applied filtering criteria only on the region and on the language (ENG) of the authors. The architecture is able to process at a 600k tweets/sec per each node, using Apache Spark as integrated component at the batch layer.

CONCLUSIONS AND FUTURE WORK

Applying sentiment analysis techniques to mine the huge amount of unstructured data in social networks has become an important research problem. Business organizations and research institutes are putting their efforts to improve techniques for sentiment analysis. Although, some algorithms have been used in sentiment analysis with good results, there are still no technique able to resolve all the current challenges.

Focusing on the architectural aspect, we discuss here a software system that receives events as big data (tweets), archives them, performs both offline and real-time computations for sentiment assessment, and merges the results of those computations into coherent information. All of this needs to happen at the scale of millions events per second.

Further work is needed on further improving both the accuracy of the sentiment classification and also improving the rate at which this huge amount of information must be processed.

REFERENCES

[1] Brusilovsky, P. (2001) Adaptive hypermedia. User Modeling and User Adapted Interaction, Ten Year Anniversary Issue (Alfred Kobsa, ed.) 11 (1/2), 87-110.

[2] Martín, E., Carro, R.M., Rodríguez, P. (2007) CoMoLE: A Context-based Adaptive Hypermedia System for M-Learning. Proceedings of VIII Simposio Nacional de Tecnologías de la Información y las Comunicaciones en la Educación at CEDI 2007, pp. 79-86. Thomson.

[3] Marz N, Warren J. "Big Data. Principles and best practices of scalable realtime data systems". Manning Book, 2015.

[4] Berlinska J., Drozdowski M., Mickiewicz A. (2011) "Scheduling divisible MapReduce computations", Journal of Parallel and Distributed Computing 71 450-459.

[5] Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. (2011). "Sentiment analysis of twitter data". In Proceeding of ACL HLT conference, pp. 30–38.

[6] O. Irsoy, C. Cardie (2014), "Deep recursive neural networks for compositionality in language". Advances in Neural Information Processing Systems, pp. 2096-2104.

ABOUT THE AUTHORS

Michele Di Capua, Department of Computer Science, University of Milan, email: michele.dicapua@unimi.it

Emanuel Di Nardo, Department of Science and Tecnology, University of Naples Parthenope, Centro Direzionale Is. C4, I-80143, Naples, ITALY, email: emanuel.dinardo@gmail.com

Alfredo Petrosino, Department of Science and Tecnology, University of Naples Parthenope, Centro Direzionale Is. C4, I-80143, Naples, ITALY, email: alfredo.petrosino@uniparthenope.it

The paper has been reviewed.