A distributed framework for early trending topics detection on big social networks data threads

Athena Vakali, Kitmeridis Nikolaos, Panourgia Maria

Informatics Department, Aristotle University, Thessaloniki, Greece

{avakali,nkitmeri,panomari}@csd.auth.gr

Abstract Social networks have become big data production engines and their analytics can reveal insightful trending topics, such that hidden knowledge can be utilized in various applications and settings. This paper addresses the problem of popular topics' and trends' early prediction out of social networks data streams which demand distributed software architectures. Under an online time series classification model, which is implemented in a flexible and adaptive distributed framework, trending topics are detected. Emphasis is placed on the early detection process and on the performance of the proposed framework. The implemented framework builds on the lambda architecture design and the experimentation carried out highlights the usefulness of the proposed approach in early trends detection with high rates in performance and with a validation aligned with a popular microblogging service.

1. Introduction

Big data threads are rapidly produced in social networks, with emerging textual data streams, evolving unpredictably, in a non pre-determined manner. Such data threads offer fertile ground for analytics which can reveal trends, phenomena, and knowledge. Already in social networks, such as in Twitter microblogging service[1], data threads unfolding over time are characterized as trending topics when they exhibit viral cascading data patterns.

This work is motivated by the fact that predicting whether a social network's topic will become a trend, prior it is declared as a trend by the social network itself (i.e. Twitter) can be addressed as a classification problem, tackling with evolving big data requirements. Predicting topics which will actually end up as trends in social networks has as major obstacles the real time, often bursty, data production, along with the missing exemplar time series to ignite data processing.

[1] https://twitter.com/whatstrending

While previous research has analyzed trending topics mostly on a long term, recent efforts have focused on detection of tweets' trend, in an almost real time fashion due to ongoing and continuous social networks events [1, 2]. Setting the appropriate distributed frameworks to support trend detection in social networks, embeds big data demands and real time requirements coverage [3, 4]. Such frameworks must be tunable to offer the appropriate testbeds for important applications highly related with trend detection, such as fraud and emergencies [5, 6].

This paper addresses the problem of popular topics and trends prediction out of social networks data streams, under an online time series classification model, which is implemented over a flexible and adaptive distributed framework. Trend prediction is employed in a real time fashion, utilizing big data principles and technologies, under a framework designed in Lambda architecture outline. Lambda architecture has been chosen due to its flexibility in processing both evolving and static data threads, based on a tunable latent source model. The proposed model sets the so called latent source "signals" corresponding to an exemplar event of a certain type, and a clustering process is combining with the classification tasks of labeling the data threads over specific categories (either detected as trends or not). Different similarity measures are used to support classification via the latent source model. The proposed work is validated under an experimentation setting with actual Twitter large scale data threads, having the Twitter declared trending topics, as the ground truth for detected viral topics evaluation. Under the experimentation carried out on the proposed distributed architecture, trending topics prediction has reached an accuracy of 78.4%, while in more than one third of studied cases the prediction is successful in advance of the respective Twitter trends declaration.

The structure of the paper is as follows. Next section outlines the principles of both early trend detection in social networks and of the suitability of the architecture to carry out such a process. Section 3 has the details for the proposed micro-blogging trending topics prediction implementation, while experimentation and results are discussed in Section 4. Section 5 has conclusions, indicating future work.

2. A framework for early trend detection in social networks

Trend detection problem as a time series classification problem has been studied in several earlier efforts [7, 8], with a focus on classifying a topic as a trend based either on the Euclidean distance of topic's time series and the time series of trend (or not trend) training sets, respectively or streaming time series into a vector which is then used for clustering over the evolving time series. Social networks (Twitter in particular) data stream emergence in real time, has set the floor for research approaches capable of predicting phenomena, such as disease outbreaks, emergencies, and opinions shifts [9, 10, 11]. Popularity alone is not adequate for a topic to break

into the trends list since Twitter favours novelty over popularity. It is the velocity of interactions around a specific topic which should be monitored in relation to a baseline interactions level [12], [1]. To resolve such issues earlier efforts have focused on the so called latent source model for time series, which naturally leads to a "weighted majority voting" classification rule, approximated by a nearest-neighbor classifier [7]. The proposed work is inspired by this latent classification model which is advanced here with novel distributed data processing methodologies, with the inclusion of social features related to a process favoring the early detection of a trending topic. To support this approach two distance metrics of cosine and squared Euclidean are utilized advancing earlier efforts which have favored the use of only Euclidean space metrics.

Complex architectures have been designed and tested on evolving big data management use cases extracting useful insights from large scale data collections [13, 14]. Among such distributed frameworks, in this work Lambda Architecture[2] is proposed since it gains ground due to its well defined set of architecture principles allowing both batch and real-time stream data processing [15, 16]. The three Lambda architecture's layers, i.e. the Batch, the Serving, and the Speed layer support the proposed work's framework since they enable processing of high volumes of data in either batch or on a real time mode. Data streams initially enter the system and are dispatched to batch the speed layer.

In our case, since data are collected from social networks, data arrival originates from the Twitter streaming API, and at the batch layer side, a master dataset, proceeds with an immutable, append-only set of raw data, computing arbitrary views. Hadoop[3] and its implementation of MapReduce model has been chosen since it is ideal for this batch layer, leveraging on HDFS. Serving layer gets the output from the batch layer as a set of flat files containing pre-computed views and exposes views for querying. Bridging among these layers and the support of a fast database is handled with specific tools (like Apache Drill[4]) in combination with an in-memory key-value datastore. Speed layer computes data views as well, but it does so incrementally by balancing high latencies of the batch layer with real time views computations. Real time views are of transient nature along with the data propagation rates (from the batch and serving layers), under the so called "complexity isolation" process which pushes into the layer only temporary results [17]. In response to users queries, merging results from batch and real-time views are delivered to the user under an integrated common interface. This architecture is next exploited for

[2] http://lambda-architecture.net/

[3] https://hadoop.apache.org/

[4] https://drill.apache.org/

the specific social networks trend detection problem, due to its capabilities of handling both static and evolving big data threads.

3. A micro-blogging trending topics prediction implementation

The proposed framework builds on Lambda Architecture, with a batch, a speed and a serving layer. However, the key difference between them arises from the framework's execution work flow which is depicted in Figure 1. Both batch and speed layers are fed with the same incoming data stream, but batch layer executes periodic processing on the whole of the dataset, until the specified time, while speed layer performs the same processing, in real time, in the part of data acquired while the former layer is busy. Finally, the results of the batch layer processing stored in the serving layer are aggregated with the results of the serving layer processing and the outcome are global views of the dataset in response to specific queries. Data processing procedure on batch and speed layers of the implemented framework differ in that batch layer is used for both the creation of the training set and the periodic update of the training set, while speed layer covers the topics classification into trends through a series of execution steps. Big volume of tweets is used to define the training set, in an asynchronous execution at the speed layer, accessing results stored at the service layer.

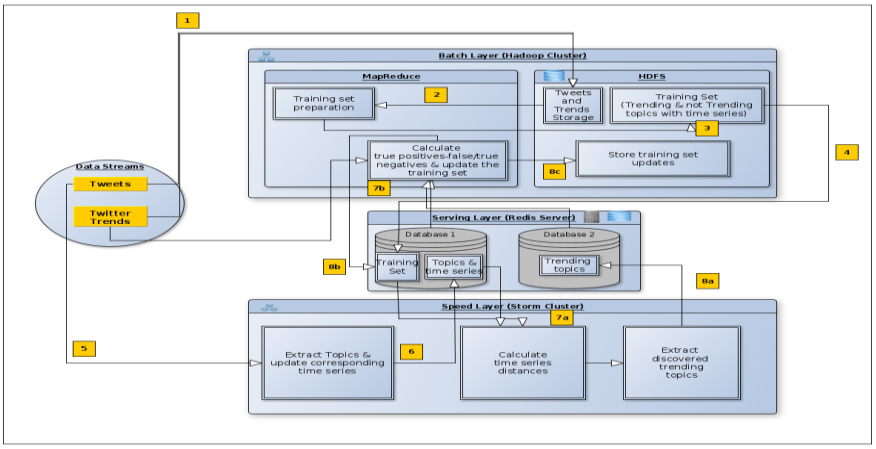


Fig. 1: Twitter trend prediction framework

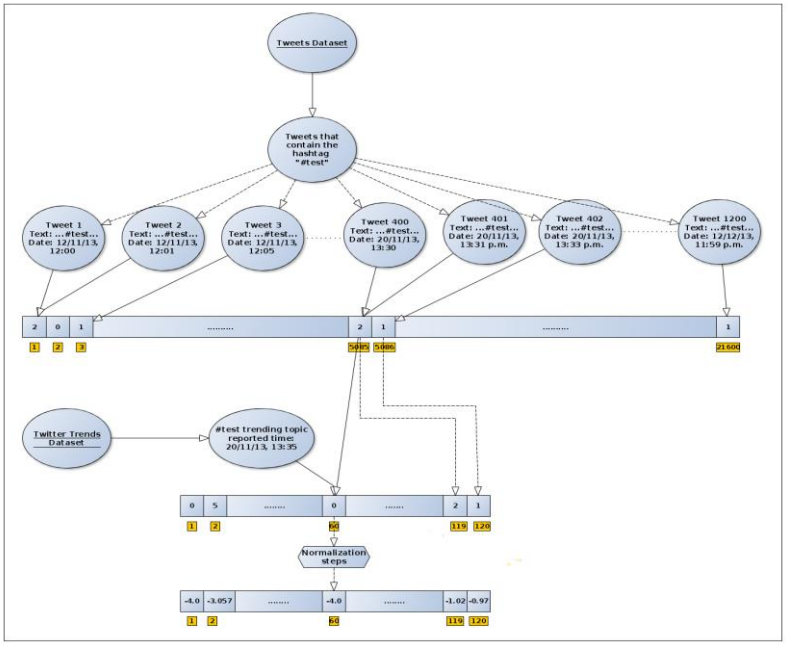


Fig. 2: Trending topic's time series generation example

In this work, early prediction of Twitter's trending topics involves a series comparison process, requiring a sample of the overall tweets dataset over a specific period. This sample is built on the basis of the Twitter declared trending topics and by performing retrospective analysis on the dataset, to generate the time series for each topic. For each topic, the ratio of summarized distances is calculated and once it is above a threshold the topic is classified as a potential trending topic. Proposed execution approach novelty is due to the next key points:

- training set's time series are constructed by a small rate of the overall tweets dataset;
- the time series of topics to be tested are not static, but are generated in real time in a form of a sliding window, maintaining low percentage of tweets for constructing the time series.

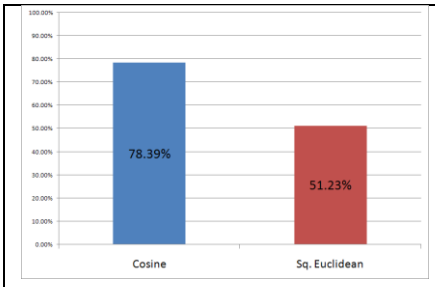
Creation of the final form of time series is given in Figure 2, which visualizes the flow of a time series generation example, in line with the next proposed steps:

- timestamps and exact date of the topic being declared as a trending topic;
- definition of a time range from the moment just before the topic became trended, i.e. the time of the last kept time slot;

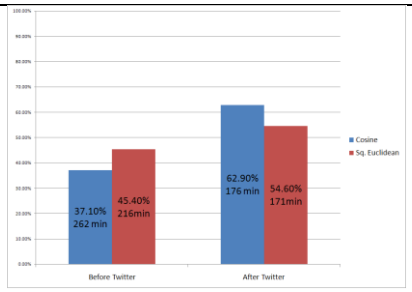
- removal of all time slots preceding the one that signifies the start of the aforementioned time range;
- production of the final form of the time series by applying a set of normalization filters.

Time series of non trending topics is generated under a similar procedure. Density of time series to be chosen may range from too sparse to too dense, and the normalization filters proposed here also range from baseline to spike ones, accordingly [7].

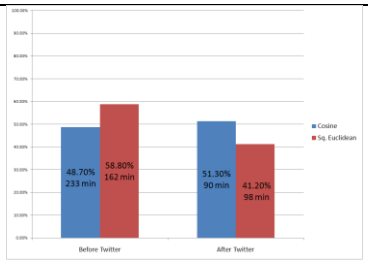
4. Experimentation and Results



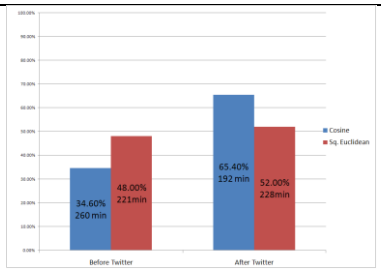
(a) true positives percentages at the end of the 48 hour execution, for Cosine and Squared Euclidean



(c) 36 hours of execution



(b) 24 hours of execution



(d) 48 hours of execution

Fig. 3: Rates of trending detection prior and upon Twitter declaration

The overall framework (depicted in Figure 2) has been installed at the GRNET's Cloud service, called Okeanos[5], and it consisted of a Hadoop cluster (batch layer),

[5] https://okeanos.grnet.gr

a Storm cluster (speed layer) and a single node (serving layer) . The available resources from Okeanos were 48 CPU cores, 46 GB of RAM and 600 GB of disk storage and in total, 14 virtual machines have been created, using these resources, for the implementation of above clusters and servers.

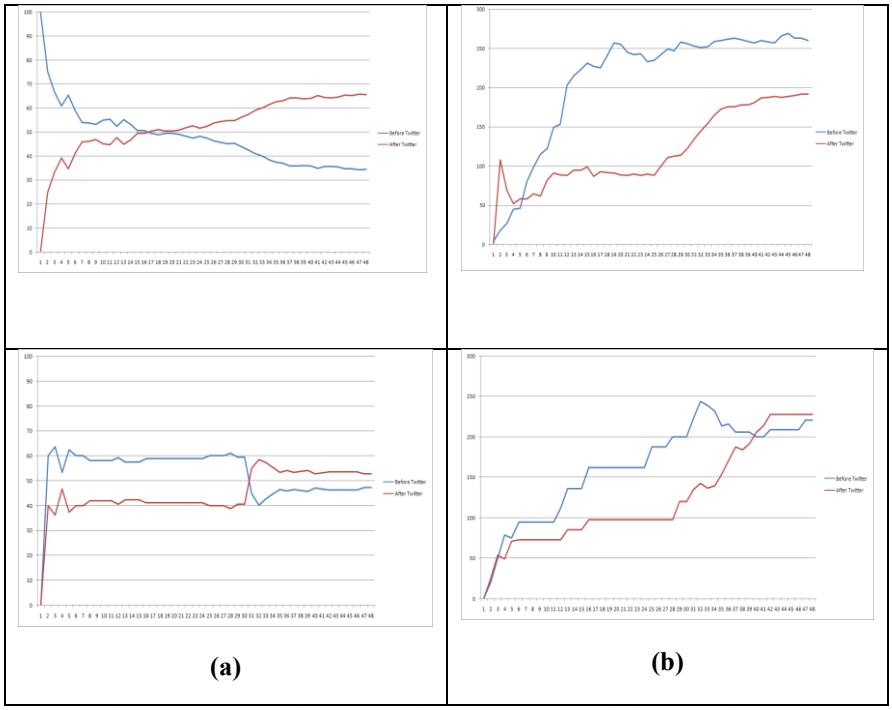


Fig. 4: Hourly Percentages (a) and Mean times (b) fluctuation for trends detection prior and after Twitter declaration

The proposed framework has been stress tested under various big data tweets threads collected via Twitter streaming API, over several time windows. Due to lack of space, here a sample of 1 month period (Nov-Dec 2013) is discussed, of its total size exceeding 300 GB, along with the Twitter trending topics announced for the same time window. The parallel collection of tweets and the trending topics followed the proposed time series generation, by dividing the time window of the one month in fine grained time slots of 2 minutes. The evaluation of the proposed methodology implemented on the lambda inspired architecture has reached improved performance since almost 80% of the actual trending topics were classified as potential trending topics by the method (Figure 3a). Specifically, 79% of the correctly classified trending topics, were claimed as trending topics by the method under cosine similarities, and this was actually performed earlier than the Twitter API, with the mean time to be 1.43 hours earlier and only 4% of the potential trending topics

were false positive. Figure 3(b-d), also depicts the rates of trending detection ignition for true positives prior and upon Twitter trending topics declaration, with the similarities estimation based on either Euclidean or cosine metrics, in the regular intervals of 24^{th}, 36^{th}, and 48^{th} execution hours.

The percentage of the reports of trending topics by the implemented framework been done before the respective reporting from Twitter follows a downward path throughout the execution (as partially depicted in Figure 4) which illustrates the whole of the execution flow.

On the contrary, mean times of the true positive's early reporting in comparison to the timing of Twitter's reports, are quite satisfactory and such improved performance is enhanced by the fact the mean times of the later reports, in comparison to Twitter's, are significantly smaller.

5. Conclusion and Future Work

In summary, the overall percentage of the trending topics of the proposed framework validates its positive performance which flows similarly to Twitter's respective percentages of trending topics detection, its strong contribution lies in its ability to early detect trending topics, and it can further be extended and improved.

Use of different distance metrics for the time series comparison can be beneficial since time series oriented distance metrics like, edit distance with Real Penalty, move-split-merge and sparse dynamic time wrapping are currently under experimentation in the proposed framework. Normalization and filtering of the time series can also be extended and refined while the implemented algorithms execution workflow, can also become more "loyal" to the principles of the Lambda Architecture. In such a case a more powerful infrastructure and the inclusion of a distributed NoSQL database to serve the scope of the serving layer are under consideration. In terms of its applicability, observing and analyzing social networks textual threads as evolving time series, has many future relevant applications, such as recommenders and other personalized services.

References

1. Z. Arkaitz, "Real-time classification of Twitter trends.", *Journal of the Association for Information Science and Technology 66,* pp. 462-473, 2015..

2. G. Salvatore, G. Lo Re and M. Morana, "A framework for real-time Twitter data analysis.," *Computer Communications 73,* pp. 236-242, 2016.

3. J. Li, "Bursty event detection from microblog: a distributed and incremental approach.," *Concurrency and Computation: Practice and Experience,* p. 3115–3130, 2015.

4. D. Manirupa, "Towards methods for systematic research on big data.," in *Big Data (Big Data), IEEE International Conference on. IEEE*, 2015.

5. M. Giatsoglou, D. Chatzakou, N. Shah, C. Faloutsos and A. Vakali, "Retweeting Activity on Twitter: Signs of Deception," *Advances in Knowledge Discovery and Data Mining, Springer,* 2015.

6. T. Sakaki, M. Okazaki and Y. Matsuo, "Earthquake shakes twitter users: Real-time event detection by social sensors," in *In Proceedings of the Nineteenth International WWW Conference, ACM*, 2010.

7. N. Stanislav , G. Chen and D. Shah, "A latent source model for nonparametric time series classification.," in *Advances in Neural Information Processing Systems.*, 2013.

8. M. Kontaki, A. Papadopoulos and Y. Manolopoulos, "Continuous trend-based classification of streaming time series," *LNCS, Vol. 3631 Springer,* p. 294–308, 2005.

9. M. Szomszor, P. Kostkova and E. Quincey, "#swineflu: Twitter predicts swine flu outbreak in 2009," *eHealth,* 2010.

10. S. Lei, "Predicting US primary elections with Twitter.," [Online]. Available: http://snap. stanford. edu/social2012/papers/shi.pdf. [Accessed 2012].

11. Y. Wang, "To Follow or Not to Follow: Analyzing the Growth Patterns of the Trumpists on Twitter.," *arXiv:1603.08174,* 2016.

12. M. Mathioudakis and N. Koudas, "Twittermonitor: trend detection over the twitter stream.," in *SIGMOD ACM*, 2010.

13. I. Gorton and K. Klein, "Distribution, data, deployment: Software architecture convergence in big data systems.," *Software, IEEE 32.3,* pp. 78-85, 2015.

14. B. Tang, "A Hierarchical Distributed Fog Computing Architecture for Big Data Analysis in Smart Cities.," in *Proceedings of the ASE BigData & SocialInformatics, ACM*, 2015.

15. K. Mariam, "Lambda architecture for cost-effective batch and speed big data processing.," in *IEEE Big Data Int. Conf.*, 2015.

16. M. Martínez-Prieto, "The solid architecture for real-time management of big semantic data." *Future Generation Computer Systems 47,* 2015.

17. N. Marz, Big data : principles and best practices of scalable realtime data systems., O'Reilly Media, 2013.