The Thirty-Third International
FLAIRS Conference (FLAIRS-33)

Towards Real-Time and Unsupervised Campaign Detection in Social Media

Dennis Assenmacher, Lena Adam, Heike Trautmann, Christian Grimme
University of Münster
Information Systems and Statistics
Münster, Germany

Abstract

The detection of orchestrated and potentially manipulative campaigns in social media is far more meaningful than analyzing single account behaviour but also more challenging in terms of pattern recognition, data processing, and computational complexity. While supervised learning methods need an enormous amount of reliable ground truth data to find rather inflexible patterns, classical unsupervised learning techniques need a lot of computational power to handle large amount of data. This makes them infeasible for real-time analysis. In this work, we demonstrate the applicability of text stream clustering for the real-time detection of coordinated campaigns.

Introduction

Social media platforms have become central institutions for social and private content sharing as well as network building. This may foster democratic principles such as open sharing of opinions and freedom of speech. At the same time, the tight coupling of global networks, information propagation, and underlying technical infrastructures allow for strategic manipulation of the public opinion. Automation and coordinated human campaigns can amplify topics worldwide without being recognized as such by users or even classical multipliers such as journalists or deciders like politicians, who lack a global view on the networks. To make it even more complicated, the amount of data and information running through the networks is too large for human analytical capabilities.

A lot of research has been put into computer-aided detection of automated content spreaders in the context of opinion manipulation. While research started with the attempt to detect automation of single accounts (Ferrara et al. 2016; Varol et al. 2017a), recent approaches increasingly focus on the detection of collaboration of multiple (not necessarily automated orchestrated) actors (Cresci et al. 2019; Grimme, Assenmacher, and Adam 2018) in campaigns. As one of the first, Lee et al. discriminate these campaigns into *organic campaigns* that arise from classic human interaction in social media and *non-organic* campaigns that are promoted by artificial mechanisms or purchased from (and then supported by) the social platform (Lee et al. 2014).

While the research in campaign detection started with a-posteriori analysis of network topologies, the clustering of content, and the temporal investigation of topic development, this mainly aimed for the identification of features for machine learning approaches. More recent detection approaches afterwards focused on the application of machine learning in campaign detection to identify characteristic patterns of organic and non-organic campaigns (Varol et al. 2017b). However, there are some major disadvantage of these approaches: first, they have to be trained using labelled data. This data is usually not sufficiently available leaving the approaches imprecise. Second, the learned patterns can only capture the characteristics found in available input and learning data and may become inflexible regarding new kinds of orchestrated campaigns.

Only very recent work (Cresci et al. 2019; Chen and Subramanian 2018; Yang et al. 2019) addresses the application of unsupervised detection methods like clustering and network analysis as solutions to some of the issues. These approaches do not need initial training and can detect unknown characteristics. However, as correctly pointed out in (Yang et al. 2019), these methods are computationally too complex to handle observed social media content in real-time.

In this paper, we argue and empirically show that unsupervised stream clustering can solve the issue of analyzing real-time data. Different to classical unsupervised techniques, stream clustering approaches continuously update current clusters without complete and expensive re-clustering of all data. We apply and extend a new text stream clustering approach (Carnein, Assenmacher, and Trautmann 2017b) to discover and observe topics and their importance in real time social media discussions. The temporal patterns of the topics can then be considered as indicators to support human analysts in detecting organic and non-organic campaigns. Interestingly, the results match offline detected patterns from earlier work (Lee et al. 2014).

Text Stream Clustering

The main goal of stream clustering in general is to apply clustering in in an online fashion. The general assumption that the data stream is potentially unbounded requires the algorithm to iterate over the data only once (Silva et al. 2013; Carnein, Assenmacher, and Trautmann 2017a).

Usually, stream clustering algorithms follow a two-step approach. During the online phase, observations are fetched from a data stream and are directly processed into micro-clusters. Micro-clusters are aggregations of multiple observations which are located in dense areas. Hence, it is not necessary to store each observation but only a set of micro-clusters is maintained which strives to represent the original data. In a second step, the offline-phase, micro-clusters can be again clustered on-demand by traditional clustering techniques. This re-clustering phase can be triggered at any point in time and is independent of the online phase. Therefore the restriction that data should be only processed once does not apply in this step. Originally micro-clusters were designed to aggregate numeric data. However, the idea can be extended to handle textual data as well (Aggarwal 2014).

A crucial concept that differentiates stream clustering approaches from incremental algorithms is the explicit notion of time. In an online stream, the underlying data distribution may change over time (known as concept drift). Therefore, micro-clusters have some mechanisms to account for these changes. Usually, an associated cluster weight is slowly decayed, if a micro-cluster is not updated anymore. Ultimately, a micro-cluster will be removed from the clustering, if the weight falls below a certain threshold.

textClust: While a variety of different stream-clustering algorithms were proposed for metric data, little research is put into the development of text-based stream algorithms. For our experiments, we employ the textClust algorithm, presented in (Carnein, Assenmacher, and Trautmann 2017b)[1]. However, it should be stressed, that the metric which is introduced in this paper to detect campaigns is generalizable and can be applied on any stream clustering algorithm that utilizes fading techniques to deal with concept drift. The textClust algorithm produces micro-clusters mc which are represented as 3-tuples:

$$mc = (w, t, TF)$$

The weight w of a micro-cluster reflects its relative importance. Each time a new observation is added to an existing micro-cluster, the weight is increased by 1. To account for concept-drift, the weight is exponentially decayed at each time step using

$$f(w) = w * 2^{-\lambda(t_{now}-t)},$$

where λ denotes the fading factor, t_{now} the current time and t the time the respective micro-cluster was last updated. Every t_{gap} timesteps a cleanup procedure is called and all micro-clusters below a predefined threshold are removed from the clustering result. The same applies for all tokens within a respective micro-cluster.

The TF vector contains the term frequency of representative words as n-grams (for our experiments we employed bi-grams). To calculate the distance between two micro-clusters the cosine similarity between the micro-clusters $tfidf$ vectors is computed. Tfidf extends the traditional term frequency by weighting down words that do appear in many documents, since they are less important. To obtain the inverse document frequency, the terms over all micro-clusters are used. When a new observation arrives, a new micro-cluster is created and the distance to all other micro-clusters is calculated. The closest micro-cluster to the newly created one is selected for a merge, if it falls below a predefined radius threshold r. Otherwise the new micro-cluster is added to the set of all clusters. The parameters r, λ and t_{gap} are parameters which have to be manually set by the user.

Adjusted Cosine Similarity: While the original textClust algorithm utilizes the cosine-similarity for calculating the distance between two micro-clusters, we choose a variant which is frequently utilized in the area of Collaborative Filtering: the adjusted cosine-similarity. Often, micro-clusters that represent some kind of trend, i.e. which are updated frequently, exhibit high token weights. A new observation that arrives from the stream naturally consists of low weights, because of its novelty. Although a trend micro-cluster and a new micro-cluster might be similar in terms of tokens, the overall weight distribution completely differs. To overcome this issue, we take the average weight of each micro-cluster into account, by computing each single token weight relative to the average weight. Let A and B represent two $tfidf$ vectors from two different micro-clusters, with their respective means being μ_A and μ_B. The adjusted cosine similarity between both vectors is then defined as follows:

$$cos(\alpha) = \frac{\sum_i (A_i - \mu_A)(B_i - \mu_B)}{\sqrt{\sum_i (A_i - \mu_A)^2} \cdot \sqrt{\sum_i (B_i - \mu_B)^2}}$$

Micro-cluster monitoring to detect campaigns

Monitoring micro-clusters over time and by this means identifying suspicious cluster evolutions that differ from normal trending content and therefore indicating potential campaigns, is one of the key goals of this work. Since a large number of micro-clusters are created during clustering, manual inspection of each individual one is often infeasible. Therefore, it is necessary to automatically reduce the amount of clusters of interest to a size that can be manually inspected by human experts assisted by sophisticated indicators.

Per definition a micro-cluster represents a topic that has been recently discussed in the text stream. The corresponding token weights furthermore indicate how many text messages were merged into that respective micro-cluster. A naive approach for monitoring the evolution of topics would be to only observe those micro-clusters with the highest cluster weights. While for plain trend detection, the approach might be valid, we are more interested in unusual patterns that might indicate non-organic campaigns. Varol et al., already differentiated between organic and promoted trends and showed that both exhibit some distinct temporal characteristics. In our work we assume that unusual, and therefore interesting micro-clusters show some detectable abnormalities during their lifespan. Although a deeper investigation of a suspicious micro-cluster has do be done manually afterwards, we define a selection criterion to detect candidate micro-clusters automatically.

[1] An implementation of the algorithm in Python can be found here: `https://wiwi-gitlab.uni-muenster.de/d_asse011/textclustpy`

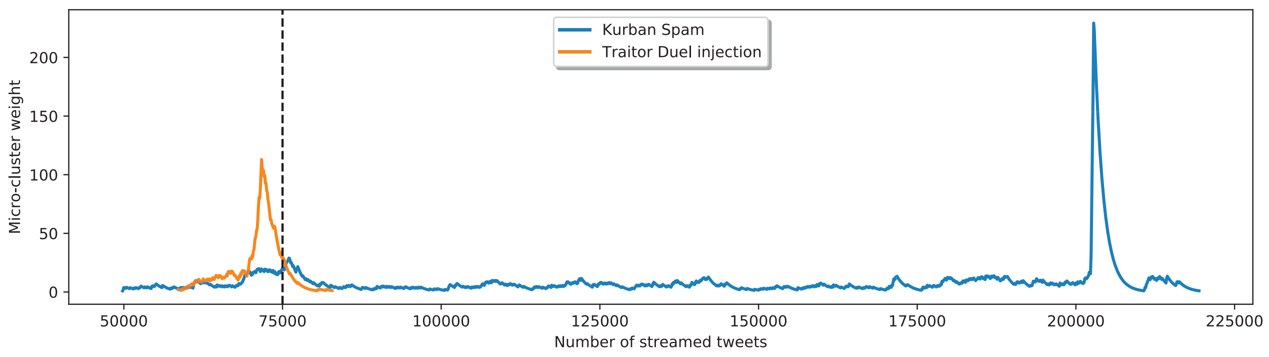

Figure 1: Three of all 55 filtered micro-cluster developments. The dashed vertical line indicates the beginning of the broadcasted tv-debate.

The change of the micro-cluster weight $\Delta_w = w - w_{last}$ indicates how the weight of a micro-cluster changes within t_{gap} cluster updates (w_{last} indicates the cluster weight from t_{gap} timesteps before). The average weight change over all k micro clusters $\mu = \frac{\sum_i \Delta_{w_i}}{k}$ and the corresponding standard deviation $\sigma = \sqrt{\frac{1}{k-1}\sum_{i=1}^{k}(\Delta_{w_i} - \mu)^2}$ at one specific point in time can be used to identify clusters with unusual temporal patterns. Since we cannot assume normally distributed changes, we utilize Chebyshev's inequality to memorize clusters that do exhbit cluster changes with small probabilities (Mood, Graybill, and Boes 1974). Let X be a random variable with expected value μ, standard deviation σ and k any positive number. The inequality states that:

$$P(|X - \mu| \geq k * \sigma) \leq \frac{1}{k^2}$$

Concretely we choose a 6σ ($k = 6$) threshold to ensure that only 2.7778% of observations are selected as special and thus interesting. To conclude, the set I of clusters that should be furthermore investigated is defined as follows:

$$I = \{mc||\Delta_w - \mu| \geq 6 \cdot \sigma\}$$

To implement the proposed changes, we extend a micro cluster to a 4-tuple with an additional w_{last} component:

$$mc = (w, w_{last}, t, TF)$$

w_{last} cannot be directly calculated from w, since it is constantly updated and we are interested in the change within t_{gap} timesteps and not in the change since the last update.

Case Study Evaluation

As already discussed before, evaluating our approach is problematic because of missing labelled data. In this preliminary work, we show within two example cases that our approach of detecting campaigns works. First, we look at Twitter conversations during the German Federal election in 2017. During a television debate between Angela Merkel and her competitor Martin Schulz, a coordinated hashtag injection attack was observed and confirmed by media afterwards. Using this well known attack, we evaluate whether our approach is capable of identifying this coordinated campaign from the stream of all 500,000 Tweets that were gathered during the debate. Second, we look at a stream of about 400,000 tweets containing the keyword `trump` between the 4th and the 8th of November, 2019. For both scenarios we use a standard parameter setting for textClust with $\lambda = 0.001, r = 0.6, t_{gap} = 100$, which perform well for both cases. The data was captured from the Twitter Streaming API by filtering for the most popular topic-related hashtags (`#tv-duel` and `#trump`). Preprocessing steps such as lemmatization and stopword removal were applied to improve the clustering results.

Known Campaign during German Election: By utilizing the micro-cluster change criterion, we reduced the total amount of interesting clusters from 260,000 to 55, which is a manageable size for manual inspection. Most importantly, the micro-cluster that reflects the hashtag injection attack is among those filtered clusters. Within Figure 1 the change of weights for three selected micro-clusters is displayed. The orange timeline reflects the hashtag injection attack. We see that the micro-cluster first slowly gained some weight because the topic started trending directly before the debate. Then, the cluster's importance drastically increased and completely faded out afterwards within a few hundred iterations. As reported in media, a group of about 380 trolls tried to establish a new hashtag `traitor-duel` by injecting it into tweets that contained the popular `#tv-duel` hashtag. However, the attack was not successful because the hashtag was not adapted by other users as the rapid fading of the micro-cluster reflects. Another attack was observed later at night (blue timeline). In this case, a single account spammed messages.

Campaign Discovery in Live Stream: Analysing the live stream about `trump`, we found mechanisms of content amplification by single and multiple accounts. The orange and blue time series both depict spreading of content by single accounts. In line with (Lee et al. 2014), we observe sharp spikes which are frequently repeated until they are slowly removed from the clustering. In both cases, single accounts simply promoted statements about Trump (positive and negative). The green timeline on the other hand shows a dif-

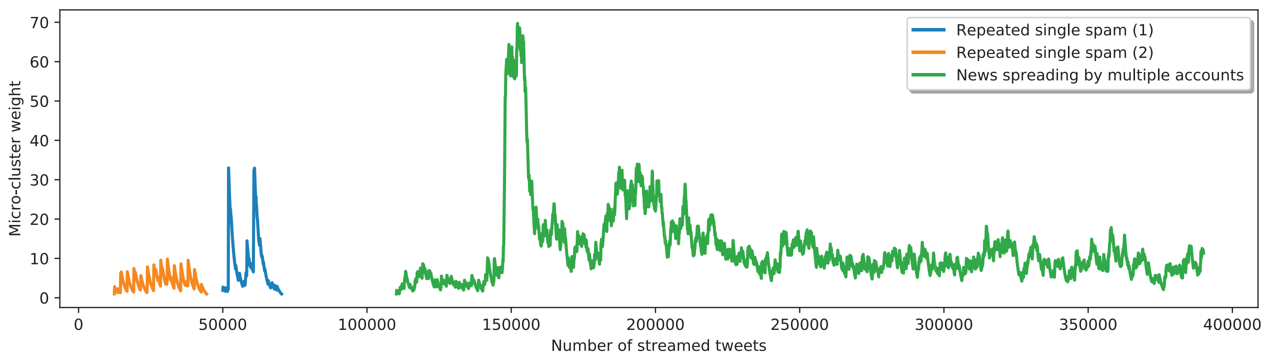

Figure 2: Non organic micro-cluster evolution with spiking cluster weights within the `trump` stream

ferent amplification approach. Herein, a large news outlet (Reuters) posted a tweet that Trump's "conscience" rule for healthcare workers was struck down by a U.S. judge. Afterwards, the tweet text was copied and posted by multiple other accounts. All of these accounts posted a large number of tweets in the past ($\mu = 15.000$). We assume that these accounts use the Twitter API to automatically redistribute content that supports specific political views (here anti-Trump).

Discussion and Future Work

Our experiments exemplarily show that the approach of utilizing text stream clustering and focusing on unusual patterns of micro-cluster changes can be used to reduce the amount of data that has to be manually inspected for real time insights into suspicious campaigns in a text data stream. In contrast to approaches that only focus on account level, we are able to identify coordinated strategies of multiple accounts. This essentially enables us to identify the actors in a top-down approach: after the suspicious campaign topic has been identified, it is easy to subsequently identify the actors - may it be single (amplifying) robots or a group of distributed collaborating human actors.

Due to the inherent lack of benchmark campaign data for text streams, we are currently not able to conduct in-depth quantitative evaluation of the approach. Future research also comprises automatic parameter configuration for the algorithm and our metric. Ideally parameters would be automatically adjusted while the data-stream is processed. Moreover, the experiments show the capabilities of real time campaign detection but leaves an evaluation of campaign to the user. Here, automatic tools could be added to simplify the discrimination of harmless and malicious campaigns.

References

Aggarwal, C. C. 2014. Mining text and social streams: A review. *SIGKDD Explor. Newsl.* 15(2):9–19.

Carnein, M.; Assenmacher, D.; and Trautmann, H. 2017a. An empirical comparison of stream clustering algorithms. In *Proceedings of the ACM International Conference on Computing Frontiers (CF '17)*, 361 – 365. ACM.

Carnein, M.; Assenmacher, D.; and Trautmann, H. 2017b. Stream clustering of chat messages with applications to twitch streams. In *Proceedings of the 36th International Conference on Conceptual Modeling*, 79–88. Springer.

Chen, Z., and Subramanian, D. 2018. An unsupervised approach to detect spam campaigns that use botnets on twitter. *CoRR* abs/1804.05232.

Cresci, S.; Petrocchi, M.; Spognardi, A.; and Tognazzi, S. 2019. On the capability of evolved spambots to evade detection via genetic engineering. *Online Social Networks and Media* 9:1 – 16.

Ferrara, E.; Varol, O.; Davis, C.; Menczer, F.; and Flammini, A. 2016. The rise of social bots. *Commun. ACM* 59(7):96–104.

Grimme, C.; Assenmacher, D.; and Adam, L. 2018. Changing perspectives: Is it sufficient to detect social bots? In Meiselwitz, G., ed., *Social Computing and Social Media. User Experience and Behavior*, 445–461. Springer.

Lee, K.; Caverlee, J.; Cheng, Z.; and Sui, D. Z. 2014. Campaign extraction from social media. *ACM Trans. Intell. Syst. Technol.* 5(1):9:1–9:28.

Mood, A. M.; Graybill, F. A.; and Boes, Duane C., . 1974. *Introduction to the theory of statistics*. McGraw-Hill New York, 3rd ed. edition.

Silva, J. A.; Faria, E. R.; Barros, R. C.; Hruschka, E. R.; Carvalho, A. C. P. L. F. d.; and Gama, J. a. 2013. Data stream clustering: A survey. *ACM Comput. Surv.* 46(1):13:1–13:31.

Varol, O.; Ferrara, E.; Davis, C. A.; Menczer, F.; and Flammini, A. 2017a. Online human-bot interactions: Detection, estimation, and characterization. In *International AAAI Conference on Web and Social Media*, 280–289. AAAI.

Varol, O.; Ferrara, E.; Menczer, F.; and Flammini, A. 2017b. Early detection of promoted campaigns on social media. *EPJ Data Science* 6(1):13.

Yang, K.-C.; Varol, O.; Davis, C. A.; Ferrara, E.; Flammini, A.; and Menczer, F. 2019. Arming the public with artificial intelligence to counter social bots. *Human Behavior and Emerging Technologies* 1(1):48–61.