Analysis of Social Media Streams

B.Sc. Florian Weidner
Fakultät Informatik
Technische Universität Dresden
Nöthnitzer Straße 46
01187 Dresden, Deutschland
Florian.Weidner@tu-dresden.de

ABSTRACT

Social media streams like Twitter[1] feeds are good sources of information. The information retrieval community is very interested in the provided data, especially in the following four ways: Detecting topics, track topics, cluster social media streams and summarize social media streams. Though these tasks are related to each other, every single one requires different measurements to achieve good results. In this article, the general problems of social media stream clustering and summarizing as well as topic detection and tracking are outlined and several state-of-the-art approaches are presented. Furthermore, some drawbacks and major contributions are mentioned. We conclude with the statement, that, although there's much motion in the research community, an efficient solution which covers all problems in an computational efficient manner is still not available.

Categories and Subject Descriptors

H.3 [**General**]: [Miscellaneous]; H.3.1 [**Content Analysis and Indexing**]: [Abstracting methods]; H3.3 [**Information Search and Retrieval**]: [Clustering, Relevance feedback, retrieval models, information filtering]

1. INTRODUCTION

The rapid growth of the Internet and the emerging rise of social media technology has changed people's live all over the world. Many platforms like Facebook[2], Twitter or Waibo[3] have made it easy for the user to exchange, share and publish information easily. Besides images, videos, links and geotags, textual information has a large share on the published information. Moreover, the tweets and status updates arrive with a high frequency providing information at a high rate. This makes text stream analysis an extraordinarily good tool for clustering, summarizing and analysing this data. In addition to the textual content of the text, many meta data can be analysed. Information about the user, the user's contacts like followers or friends, and the timestamp of the status update are used for information retrieval. This analysis can have many objectives. Here, they are divided into two main categories according to the main data element: *Social Media Streams* (section 2) and *Topics* (section 3).

A social media stream represents continuously arriving status updates of users from a service like Twitter. Applications and algorithms which work on social media streams need to be very efficient because they have to deal with a huge amount of incoming data. The work with social media streams is divided by us into two main categories: *Clustering* and *Summarization*. Clustering describes the procedure of dividing the incoming data and sorting it according to certain criteria. This can be a first step in information retrieval and helps to filter out irrelevant data. The resulting categories can be important for further tasks. Furthermore, several algorithms try to automatically produce a summary of the the stream or the cluster. This can be a single sentence, just some words or several sentences.

The term *topic* is an umbrella term and describes an object of the analysis or rather a theme. It might be something like *C++* or *New York*. However, a topic can also represent a real-life event like the *Super Bowl* or an *earthquake*. Existing algorithms distinguish between existing, hot, emerging and new topics. Existing topics have already been detected by the framework. Hot topics receive a lot of attention by the users of the social media platform whereas emerging topics might become hot topics. New topics describe topics for which no data is available yet. The detection of those topics is an important step during information retrieval. Another task when working with social streams is the tracking of these detected topics. Tracking implies not only the tracking of bare content but also spatial and temporal tracking of the tweets and status updates of the social media stream.

This article presents a state-of-the-art overview about techniques which try to solve these task. The structure of the article is adapted to the presented aspects. Section 2 describes the format of social media streams as well as algorithms which cluster and summarize such a stream. Section 3 characterizes the main features of topics. Furthermore, measurements for detecting and tracking topics are presented. Section 4 presents several unsolved problems in the field of information retrieval and specifically concerning social media streams and topics.

[1] http://www.twitter.com
[2] http://www.facebook.com
[3] http::/www.waibo.com

2. SOCIAL MEDIA STREAMS

Social media streams consist of continuously arriving and short pieces of information. In the domain of social media, these pieces of information are posts, tweets or status updates from a social network. Especially microblogging systems like *Twitter*, *Tumblr*[4] or *Weibo* but also *Facebook* are sources of such social media streams.

As mentioned before, these streams consist of several kinds of information, however, text information "...is still the most fundamental and main form of information in the Internet"[11], so this article and the presented techniques will mostly focus on textual information. Text clustering has been extensively studied by Gibson [10], and Silverstein [22] and others. However, they tackle the problem only in the off-line or non-stream context. And according to Gong et al. [11] "...text data stream clustering is still in its early stage". However, on-line or stream clustering is very important for news group segmentation, text crawling in social networks, and target marketing for electronic commerce and others. [2]

During the processing of text social media streams, all measurements have to avoid the pitfalls of human language: spelling and grammatical errors, internet slang, unknown abbreviations and the existence of different languages. Furthermore, the status updates are often very short (i.e. a Tweet has just 140 characters) and some contain links to other pages. These facts make it hard for well-established natural language processing (NLP) techniques to process this data [18].

Common tasks performed on this data are clustering and summarizing the status updates in real-time The following sections provide an introduction to clustering and presents two approaches how to cluster text streams. After a short introduction to clustering (section 2.1), two algorithms are presented in section 2.1.1 and (section 2.1.2). In section 2.2, some main problems concerning summarization are described. Then, three algorithms with different approaches are explained in section 2.2.1, 2.2.2 and 2.2.3.

2.1 Clustering

Clustering is a learning mechanism in the field of data mining. The goal is to find groups of similar instances. The difference to classification is that clustering does not need to or have predefined classes. The following three main clustering approaches exist: (1) Partitioning methods, (2) hierarchical methods and (3) density-based methods. Partitioning based methods use distance-based metrics like k-means and k-medoids to cluster data. They produce non-overlapping partitions on one level. Hierarchical methods partition the data following a bottom-up or top-down approach in levels. Density-based algorithms, partition the data points according to the their position in regions. A region with dense data points will form a cluster. This results in arbitrary shapes of the cluster. [3]

A problem in clustering is the *curse of dimensionality* [13]. A dimension of a social media stream is described by a feature. For example, the number of a word in a status update can be a feature. If a data set A has the dimension of n and another one has the dimension of m, this will lead to a complexity of $n \cdot m$. This and the continuously arriving data leads to a high computational effort. Looking at text streams, each word can represent a dimension or feature. With this high-dimensional data it is computational expensive to calculate clusters but many dimensions may also degrade the learning performance. This is because a lot of irrelevant data is taken into account. In terms of the text clustering example, words like conjunctions are often not as important as nouns. Two main methods have been developed in order to reduce the dimensions: *Feature Extraction* and *Feature Selection*. Feature extraction techniques project features into a new space with lower dimensionality (cf. Principle Component Analysis, Linear Discriminant Analysis, Singular Value Decomposition). Feature selection techniques select a subset of the original feature set trying to minimize redundancy and maximize relevance. [3]

Another problem are outliers. Outliers are data points that do not match any existing cluster and not enough other data points are similar to them to form a new cluster. Clustering algorithms have to deal with such data in order to avoid wrong or misleading clusters. A simple technique to deal with them is to ignore or rather delete those values. Two further methods which deal with outliers are presented in section 2.1.1 and section 2.1.2.

The following two sections describe some recent approaches for clustering continuously arriving textual data.

2.1.1 Cluster Droplets, Fading and Similarity Functions

Aggarwal et al. [2] define the text clustering problem as follows: "for a given set of data points, we wish to partition them into one or more groups of similar objects'". The similarity between data points is defined with the use of an objective function or distance measurement. Possible functions mentioned by Aggarwal et al. are *Cosine Similarity*, *Dice Coefficient* and *Jaccard Coefficient*. The *Cosine Similarity* describes the value between two vectors[5]. If the result is one, the both vectors will be equal. *Dice Coefficient* and *Jaccard Coefficient* both measure the set agreement. In other words, the result is a value between zero and one. Zero indicating a no overlap and one a total overlap of a set of two values. Other examples for similarity functions are the *Hamming distance*, the *Euclidian distance* or *Soergel distance*.

The approach presented by Aggarwal et al. has three main steps: (1) Receive text stream. (2) Process data point and (3) build or update a *Cluster Droplet*. Here, a novel data structure called *Cluster Droplet* (CD) is proposed by Aggarwal et al. Each cluster has a CD and this CD stores statistical information for its cluster and is updated every time a data point is inserted into this cluster. With them the authors try to solve two problems: (1) time dependency of clusters and (2) enabling analysis in different time horizons. The first problem concerns the fact that the incoming data is time dependent. Old data is not as important as new data. The latter describes the fact that a user should be able to analyse not only the most recent state of the data, but also

[4] http://www.tumblr.com

[5] Here, vector means a document or status update in feature space

segments of older data. To solve the first problem, each CD contains a weight for each data point. The sum of this weight is also stored in the *CD* and represents "...the importance of the historical information in the data stream"[2]. The single weights are updated with a so-called *fading function.* This function decays uniformly with the time t. When the global weight is below a threshold, the according cluster will be removed. For the second problem, *CDs* are stored periodically in order to enable the user to perform an off-line analysis of the social media stream according to a time window. Storing the *CD* can either be performed within uniform intervals or by using the pyramidal time frame concept. Here, the snapshots are stored at differing levels of granularity depending upon recency[1]. In the proposed algorithm, *CDs* are stored in-memory for a fast write and read access.

Aggarwal et al. solve the problem of outliers with a simple scheme of cluster management. If a data-point arrives and does not fit into an existing cluster, a new one will be created. This new cluster is a so-called *trend-setter.* If other data points arrive and are put in this cluster, the *trend-setter* will become *mature.* If this does not happen, it will become inactive and will probably decay after a period of time. If a cluster decays this will be referred to as *Cluster Death.* With the fading of a cluster, an implicit outlier detection is introduced.

With this approach, the authors provide a stable algorithm that allows clustering evolving text streams and also an offline analysis of the data using the *CDs.* The authors found the algorithm to be scalable and adaptive. Also, it outperforms other stream clustering methods like OSKM[6] but the overall cluster precision for text streams is only 51.07%. Besides that, the analysis performed by the authors only works with recorded data and not with real social media streams. Therefore, the results do not represent a real world example. Furthermore, the authors say that the algorithm allows analysis of arbitrary time windows but the analysis of old data depends on the snapshotting interval.

2.1.2 Variable Features Sets

Gong et al. [11] propose another method facing the same challenges (much data arriving continuously). They state that algorithms, based on fading functions, droplets and traditional approaches do not perform sufficiently in order to process the streaming and evolving data set. To solve these problems, they propose an algorithm which varies its features for clustering over time and on each topic subrogation or rather change. Features (or attributes) are the working units in this approach. A plain example for features is the following: Each word with its number of occurrence in the text is a feature.

For evolving streams, it is possible that the features chosen at the beginning are, after an amount of time, not appropriate any more. The algorithm from Gong et al. solves this problem with an adaptive feature set. The general approach works as follows: At first, they select an initial feature set and cluster N texts. Each cluster contains relative data, but also cluster centres and a cluster *Validity Index* (VI) according to equation (1).

$$VI = \frac{Avg_i(Radius(C_i)) \times 2))}{Avg_{ij}(Distance(C_i, C_j))} \tag{1}$$

In the equation, C represents a cluster. The radius of a cluster is the maximum distance between all data points within the cluster. The distance represents the distance or similarity between the clusters and can be a Hamming distance, Jaccard distance or another distance measurement. After calculating the *VI*, a new text is evaluated. The nearest cluster for this text is determined with a distance function and the *VI* will be evaluated: If the *VI* is smaller than a predefined *Clustering Threshold* (CT), everything will be fine. Otherwise, the result of the clustering is getting worse but is still tolerable. In both cases, the text is inserted into a cluster. However, in the latter case, the *VI* is compared with an, also predefined, *Reselection Threshold* (RT). If the *VI* is greater than the *RT*, new features will be selected The original cluster result is saved and a new cluster set is generated. The *CT* and *RT* are manually determined during an initial training phase with an iterative adjusting scheme. For feature selection, the authors propose the following five possibilities: Information gain, X^2 statistics, document frequency, term strength or word variance-based selection. In their evaluation, which shows the effectiveness and feasibility of the proposed algorithm, the word variance-based selection approach was chosen.

With this algorithm, Gong et al. proposed a method for long-term social media stream clustering. Previous methods tend to deliver an unsatisfying result because of the feature drift. Despite the good test results, several possible improvements are mentioned by the authors. Here, another strategy for choosing the *Validity Index* and thresholds as well as the introduction of incremental computing might be the most important ones.

2.2 Summarization

Clustering text streams like Twitter timelines is important to order and categorize the enormous incoming data. However, the resulting cluster might still contain a huge amount of data, or in other words, more data than one person can scan and analyse. To solve this problem, summarization algorithms try to automatically generate a summary out of these clusters or out of the whole stream.

Possible application domains for such algorithms can be news extraction platforms, the detection of highlights in sport events, or displaying news from social networks according to the interests of the user. The general idea of every algorithm which tries to summarize streams of incoming data can be described in the following way: (1) Continuously receive data, then (2) sort, order, cluster or categorize these pieces of data and finally (3) generate a summary.

An important issue when summarizing streams is the question of which data should be incorporated. To summarize always all incoming text is way too expensive regarding the computational effort. If the algorithm only processes data in time intervals, the diversity between time intervals must be considered because differences between intervals might also be important. [6]

[6] Online spherical k-means algorithm

Furthermore, the incoming data from social networks is often very noisy. Spam, many languages, spelling errors and unimportant content like insults will influence the result and are therefore often deleted during a preprocessing stage. Besides that, many algorithms are only able to work with streams that contain status updates regarding only one topic or in other words, pre-filtered streams. For example, a Twitter stream filtered according to a specific *Hashtag*[7]. [6]

The output of summarization algorithms can be categorized as follows: (1) A summary consisting of pre-existing text, (2) the summary is a simplification of pre-existing text or (3) the summary is a complete new text. Some algorithms are able to produce visual summaries (cf. [16], [8] and [15]) but we focus on textual output.

It is noteworthy, that most of the approaches which work with social network streams use Twitter. The very open and broad access to the data is, compared to for example the Facebook API very developer friendly. Furthermore, most Twitter data is publicly available whereas Facebook posts are mostly private [18].

The following sections describe three methods for summarizing by means of examples.

2.2.1 Word-Variance

Sharifi et al [20] propose an algorithm based on a *Phrase Reinforcement Algorithm* (PR). They try to find the most commonly used phrase of a filtered Twitter stream. In this context, filtered stream is defined as follows: A query with a starting phrase (topic or Hashtag) is sent to Twitter. The answer is a list of posts. This list of posts gets preprocessed, meaning that duplicates, spam and other non-relevant content is removed. The resulting list of posts is now scanned for the longest sentence in each post. The list of longest sentences is the input of the PR algorithm which builds a tree. The root node of the tree is the starting phrase which was sent to Twitter. The child nodes are words that appear in the selected sentences before (left children) or after (right children) the starting phrase. The edges in this tree represent adjacency between words. For example, *Information is beautiful!* with the starting phrase *is* would result in a tree with *is* as root node, *Information* as left child and *beautiful* as right child. Besides the actual word, the word count is added or updated, representing the number of appearance of this word in all posts processed so far. Note, that the algorithm does not assign edge weights. After the tree is complete (all sentences are processed), weights are assigned to the nodes in order to prevent the domination of longer sentences regarding the output (cf. equation 2).

$$W(N) = Count(n) - [rootDist(N) * log_b Count(N)] \quad (2)$$

N in the equation means *Node*. The higher the word count of a word and the nearer the word is to the starting phrase, the higher the assigned weight is. After calculating all weights, a final tree is available. One path from one child to another via the root node has maximum weight. This path represents the most occurring phrases before or after the starting phrase. Then, a new tree with this phrase as root node is built. The result is a complete summary, containing the starting phrase and the most occurring words or phrases before and after the starting phrase.

Though the authors had the intention to generate summaries for static data only, the approach easily applies to streaming data when the tree is built incremental with a hierarchical topic detection approach [18]. Nevertheless, the presented approach does not consider semantic properties of the textual data. The final summary might be a meaningless concatenation of words. Besides that, some other features like the latest news or current events or the location of the status updates might improve the generated summary. Furthermore, the authors do not mention any measurements for the case when two or more final paths have the same weight.

Nichols et al. [17] extend this solution and produce a multi-sentence output. For this, they choose the k phrases with the highest weights that do not share words with the same stem. With this approach, they hope to provide a more complete summary. Filippova [9] uses a similar word-graph based approach but another weighting scheme.

2.2.2 Hidden Markov Models

In contrast to the PR based approaches, *SummHMM* from Chakrabarti et al. [6] is designed to work over a long period of time. It is based on *Hidden Markov Models* (HMM) which learns the structure and vocabulary of topics and subtopics.

In general, a HMM has a number of states, observables and state transition probabilities. The authors try to model reoccurring topics with such an HMM. The usage scenario used by the authors are the football games of one season from the Green Bay Packers and Chicago Bears. The algorithm should be able to automatically produce summaries for each game of this teams from a social activity stream and provide the user with the most important information regarding the game and team. The states of the HMM are football games and in-game events like touchdowns, interceptions or fouls. The observables are the word distribution of the incoming tweets and the burstiness of words in them. Furthermore, it combines information from multiple topics, meaning different games that take place in parallel.

The algorithm works as follows: (0) It starts with a preprocessing of the incoming data. Here, the tweets are filtered according to Hashtags. In detail, the authors just use Tweets with the Hashtags "#GreenBayPackers" and "#ChicagoBears". Furthermore, the incoming and filtered stream needs to be cleaned. Unimportant and noisy data is deleted in order to reduce the computational effort. It is noteworthy that Chakrabarti et al. did not only remove duplicates and obvious spam, but also tweets from users who posted more than 100 or less than 2 Tweets per game. Furthermore, they delete Porter-stemmed (cf. [23]) words which occur less than five times. This broad preprocessing and cleaning decreases the final set of tweets and words and therefore candidates for the summary dramatically. (1)Next, the algorithm needs to be trained and the model parameters Θ are calculated. Here, an Expectation-Maximization algorithm is used (cf. [5]). They consist of three general sets:

[7] Hashtag: a keyword used by Twitter users which indicates the topic of a tweet

- $\Theta^{(s)}$: Parameters valid for all topics
- $\Theta^{(sg)}$: Parameters valid for some topics
- $\Theta^{(bg)}$: Noisy and irrelevant parameters

With this three sets of parameters, they take into account that some keywords only occur for some subtopics (i.e. names of players or trainers) while other keywords are typical for all topics (i.e. "touchdown" or "interception") The model parameters contain word distributions and transition parameters. (2) With Θ, a perfect segmentation is computed with the Viterbi algorithm (cf: [19]). Each segment represents a subtopic and can be summarized. In detail, the summary is the very Tweet, which describes the subtopic best. (3) These segments are than summarized. The final summary is the union of all summaries of the subtopics.

With *SummHMM* Chakrabarti et al. provide a general approach for summarizing tweet streams. An evaluation shows, that their algorithm outperforms strong baselines. However, they did not perform any experiments on real world data. The results are based on a filtered and recorded twitter stream. Furthermore, they note that the HMM easily adapts to other topics. However, the discrete nature of football games (and other sport events) is a clear benefit for this algorithm. *SummHMM* might not work with events without clear and reoccurring keywords as well as non-repeating events. They state, that they want to evaluate this in the future.

2.2.3 Distance Metrics

The Tweet summarization algorithm *Sumblr* proposed by Shou et al. [21] is related to the clustering approach explained in section 2.1.1. Shou et al. use a so-called *Tweet-Cluster-Vector* (TCV) which stores meta data of the cluster like timestamps, number of tweets in a cluster and a set of tweets that are nearest to the centroid of the cluster. Sumblr "...aims to extract k tweets from T [a set of Tweets], so that they can cover as many tweet content as possible"[21]. They show that this is an NP-hard problem. The algorithm enables the user to perform an online as well as an offline summarization for arbitrary time durations. To achieve this, snapshots of the clusters are taken regularly.

The summarization algorithm works as follows: During clustering, for each cluster a set called *ft_set* is maintained. (1) At the beginning of the summary, these sets are retrieved. Each *ft_set* contains m tweets whose distance to the cluster centroid according to cosine similarity is smallest. (2) With LexRank [12], centrality scores are calculated for each tweet. (3) Finally, from each TCV the Tweet with the highest score t is chosen and inserted into the final set of Tweets: the summary.

$$t = argmax[\lambda \frac{n_{t_i}}{n_{max}} LR(t_i) - (1 - \lambda) \cdot avgSim(t_i, t_j)] \quad (3)$$

In Equation 3, the first part increases the weight of tweets from big clusters and with high scores, the second part devalues Tweets with content that is already in the final set. With this approach, coverage and novelty of Tweets are considered during generation of the summary. This is related to the *Maximal Marginal Relevance* approach by Carbonell et al. [4] which considers relevance and novelty. (5) If the resulting summary has not the required length, it starts again with a global set of Tweets. This set consists of all tweets in the clusters, except those which are already in the summary.

Sumblr is, according to the evaluation of Shou et al., an effective and efficient framework. It provides a good balance between summary quality and efficiency. It consists of a clustering module and TCV-Rank summarization algorithm. They compared Sumblr against several baseline methods. However, the authors mention that these methods are not adequate for such a comparison because they rely on different principles. Furthermore, the experiments were only performed on filtered streams, meaning that the input only consists of Tweets with a specific keyword.

3. TOPICS

The analysis of topics is important for banks, universities, NGOs, companies and other organizations. For example crime and disaster management. A topic can have two main manifestations. First, it can be an abstract theme like a discussion and second it can represent a real life event. The organizations mentioned in the beginning have at least three main intentions: The detection of novel topics, the detection of novel hot topics and the tracking of existing topics. Here tracking describes two aspects: track topics and receive information about it and track one or several topics in order to predict future topics. The latter utilises patterns in the topic distribution to detect reoccurring topics.

Algorithms which offer solutions to these problems have several characteristics. They are either designed for small-scaled or large-scaled topics. Small-scaled solutions work with topics with short durations like small emergencies or local celebrations. Whereas the latter work with big and long-lasting topics like the Super Bowl or a major election. The former avoids the problem of working with large data but due to this, less information is available. The latter receives a lot of data and needs to process it to find the important information.

Possible data sources which might provide essential information for tracking, detecting and predicting geographical annotations like longitude and latitude, timestamps, semantic data but also data from other data sources like news pages or blogs. Besides that, several features are calculable. For example, the influence of a user or a status update according to its receivers. Predefined data sources like key users respective accounts or a set of important words helps to identify new topics and track them. If a topic has already been detected, it will help to define or retrieve words which often co-occur with terms describing the topic. For example, if a topic is described by the word *Election* it might be useful to consider the names of the candidates as well. If a status update contains links to images, web-pages or videos, it will be useful to extract (meta) data from the connected media. If a link points to a blog post, the content, timestamps and owner of this blog will also be a useful resource of information.

After we have explained the domain of working with topics according to application domains, data sources and algorithm characteristics, we will now describe *Topic Detection* (section 3.1) and *Topic Tracking* (section 3.2) in detail and with examples.

3.1 Detection

A social media stream offers a huge amount of status updates to the user and to algorithms. Processing these posts and extracting topics is a computational expensive operation. Walther et al. mention in [24] two main approaches for detecting topics. First, *topic augmentation*. Here, a topic from an external source like a news ticker is the input or starting point of the algorithm and information form the social media stream is added with the help of keywords or meta data of this initial headline. Second, *topic detection*. Here, topics are extracted from the stream without an external source and, at best, without training or fixed reference data.

Algorithms which try to detect topics are often based on clustering techniques. Here, the input stream is clustered and the fastest growing cluster is the trending topic. For example, a clustering approach could use a word-frequency based approach and offer the keywords which arrive most often as trending topics.

The rest of this section outlines exemplary three algorithms which detect trending topics.

3.1.1 Word-Variance

Olariu et al. present in [18] a word-variance based approach. They start with a rather easy preprocessing of a Twitter stream where URLs and usernames of the incoming data are replaced by placeholders. Furthermore, the #-symbol for Hashtags is removed. Therefore, Hashtags are treated like normal words. Finally, they reduce the set of words with a stemming function and remove stopwords.

The topic detection approach processes words within a time window. In this time window, the word frequencies are calculated and compared with the frequencies of the previous window. If there is a significant increase for one or several words, these words are possible emerging topics. Note that they only compare words which frequency exceeds a certain threshold. For all words which have an increased usage in the current time window, a correlation between them and the words from the previous window is calculated. These correlation scores are then used to cluster the words. The resulting clusters represent topics. In their work, Olariu et al. use these clusters to summarize the emerging topics hierarchical. The output of their algorithm is a tree of sentences which outlines the cluster. For this, they use a cosine similarity measurement.

Olariu et al. propose an algorithm which detects emerging topics by means of word frequency in a certain time window. However, dependent on the length of this window, it is not possible to detect slowly emerging topics. Furthermore, they have only evaluated their algorithm with a rather small data set and therefore no statement about applicability in the domain of social media streams has been given. The very stringent preprocessing eliminates Hashtags. Here, the semantic information of these keywords might have a positive influence in detecting topics. Besides that, they don't use the geotags of some posts to relate the status updates to their location and use this to detect emerging topics.

3.1.2 Location

In contrast to Olariu et al. Walther et al propose in [24] an algorithm which makes use of the location. In detail, the location is used to tackle the problem of the enormous incoming data from social media streams. As stated before, the large data permits it to compute an complex textual similarity score based on vectors. The algorithm initially collects all Tweets available via the Twitter API[8] and stores them in a Database. It then checks longitude and latitude of these Tweets. If x Tweets within the radius of y have arrive in the last z minutes, a new cluster would have been created. A cluster represents a possible topic. If clusters overlap, they will be merged.

Via Machine Learning, each cluster gets assigned a score based on several features. These features can be classified into two main categories: *Textual Features* and *Other Features*. Most important textual features are the theme of the cluster, possible duplicates within the cluster, sentiment in a cluster based on emoticons and predefined keywords, the tense of the contained Tweets as well as the semantic categories which is defined with pre-selected dictionaries. Other features are special locations like subway stations or bus stops and unique coordinates. Besides that, Poster and Tweet count in a cluster is considered for detecting topics. The scores of Tweets and Clusters are than ranked and, if the final rank of a cluster exceeds a threshold, marked as a real life event.

The approach of Walther et al. makes an extensive use of the location information provided by the Tweet. It is noteworthy, that also the tense of the contained Tweets is considered during evaluation which might give a hint for emerging topics. However, they only use the longitude and latitude tag and no other keywords like names of cities or areas of cities. This might improve the quality of the algorithm.

3.1.3 Hot Emerging Topics

Detecting topics like the approaches of Walter et al. and Olariu et al. are doing is an essential step for the algorithm of Chen et al. [7]. They try to detect not only new topics about an organization in a Twitter stream, but want to detect hot emerging topics, meaning topics which will receive a lot of attention in the future.

For this, they define a set called *orgKeyUsers* which contains officials and representative accounts which post important information about an organization. Another set, *orgKeyWords*, contains important words which are always related to the organization and the Tweet containing this word most likely contains information about the organization. Their algorithm clusters the incoming Tweets frequently with a centroid/distance based similarity function whereas each cluster represents a possible topic. They do not mention any specific interval for doing this. Then, the *Topical User Authority Score* and *Topical Tweet Influence* are calculated. Both scores indicate if the cluster contains Tweets from important users or Tweets with relevant content. With those measurements, a weight for each word is calculated. The weighted words are than ranked, the highest ranked word representing an emerging topic (not hot emerging topics). To get emerging topics, two possible learners are proposed: a co-training learner and a semi-supervised learner which can be

[8] https://dev.twitter.com/docs

used alternatively. Based on six features (increase of influence, overlap between fixed an selected keywords, increase of user number, increase of Tweets count, increase of Retweets count, overlap between fixed and influential users), these learners extract the hot emerging topics from a given set of Tweets.

Chen et al. present a technique which performs at least as well as other state-of-the-art methods, but seldom better than those approaches. Hence, the perofmrance of the proposed approach needs to be improved. Besides that, the limited usage scenario (only organizations) is a drawback. Furthermore, the output of the current algorithm is not well-suited for organization user according to readability.

3.2 Tracking

After detecting a topic, it is often interesting to track this topic. This can be used to perform two tasks: (1) track the topic during a period of time in order to receive more information concerning the theme. (2) Track the spatial development of the topic.This is interesting and useful in order to analyse social phenomena and signal unusual topics. This section exemplary present solutions for topic prediction.

Displaying only the content which is related to a certain topic is the goal of the approach of Hong et al. [14]. With tracking the interesting topic, they select interesting information. Tweets enter the system and the algorithm decides if the Tweet should be shown to the user or not. Essentially it is a topic-based real-time filtering system.

As in the system from Wang et al. described in section **??**, the set of tweets is divided into two main parts: the background and foreground corpus. The starting point of topic is the split word or phrase, dividing the tweet. The background corpus is dynamically updated with each arriving Tweet and represents all Tweets which arrived before the starting point. It is used for building an index with the Lemur toolkit[9] This index is the basis for topic tracking and contains information like term frequency, document (or Tweet) frequency and the total number of tweets.

With each arriving Tweet which is added to the background corpus, preprocessing (stemming, and so on) is triggered. Then, the semantic and quality features of the tweets are calculated. Semantic features will be retrieved if a status update contains a URL. Then, the system retrieves effective information from the web pages these links point to. Examples are the URL keyword (*cnn* from *www.cnn.com*), the information in the *title*-Tag and so on. Quality features are the number of Retweets[10], Hashtags, Mentions[11] and shortened URLs contained by a Tweet. With these features, a *Content Model* is updated. Besides this model, a *Feedback Model* is taken into account. The *Feedback Model* is based on the result of a *Pseudo Relevance Feedback* (PRF) approach. It handles topic alteration and drift with the use of a temporal sliding window which contains the latest tweets relevant to the current topic. The algorithm calculates a similarity score between these Tweets and the incoming one and outputs the information on whether the Tweet is relevant or not according to a threshold.

With the information provided by the *Content Model* and the *Feedback Model*, the algorithm decides if the Tweet is relevant or not and therefore should be shown to the user or not.

The dynamic filter system of Hong et al. tracks topic despite topic drift in real-time. It utilizes a lot of information of the Tweets content and the semantic information. However, location is not taken into account.

4. CONCLUSION

Social media streams contain a lot of information that is important to a huge variety of stakeholders. NGOs, governmental organizations, market researchers, advertising agencies but also sociologists can use the information retrieved through clustering, summarization, topic detection and topic tracking. Moreover, the application domains for the end-user are manifoldly.

This article provides an overview about these topics. Recently proposed approaches in the mentioned fields are described and shortly evaluated. Furthermore, the overall problem of the single tasks has been described.

The clustering and summarization tasks are well researched. However, the approaches only work with restrictions. Most of them only work on filtered streams. A generic approach which summarizes all status updates of the incoming stream would be a good tool. There are also drawbacks because only a few social media platforms can be observed. For example, the Twitter API allows developers to use the information for generating summaries and working with topics. However, the APIs from Facebook, Instagram or LinkedIn aren't that open and accessible. Connecting these sources might lead to a better accuracy of the results.

Furthermore, the presented algorithms are only narrow solutions. They don't use all possible and available information sources but just some of them. A algorithm which utilizes geotags, semantic information, textual information, the information provided by linked pages, information based on the social ties of users and so on would, in our opinion, achieve good results in tracking, detecting, clustering and summarizing. A semantic approach with ontologies might also improve the mentioned tasks. However, the algorithm needs to be computational efficient in order to operate in real-time.

Except for summarization (summify[12], prismatic[13]) there are currently no open source frameworks for performing the other tasks. Besides that, no framework exists, that allows an overall analysis of social media streams combining topic detection, tracking and as well as clustering and summarization.

In this field of information retrieval and analysis, more re-

[9] http://www.lemurproject.org/lemur.php

[10] How often a Tweet was shared

[11] Addressing another user directly in a Tweet is called a *Mention*

[12] http://summify.com/

[13] http://getprismatic.com/

search is necessary in order to offer well-designed and efficient algorithms that cover a variety of tasks and operate in real-time. However, single-purpose solutions like the summarization frameworks mentioned before have already been available.

5. REFERENCES

[1] C. Aggarwal. *Data Streams – Models and Algorithms.* Springer, 2007.

[2] C. C. Aggarwal and P. S. Yu. On clustering massive text and categorical data streams. *Knowledge and Information Systems*, 24(2):171–196, Aug. 2009.

[3] S. Alelyani, J. Tang, and H. Liu. Feature Selection for Clustering: A Review, 2013.

[4] J. Carbonell and J. Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336, 1998.

[5] D. Chakrabarti and K. Punear. Event summarization using tweets, 2011.

[6] D. Chakrabarti and K. Punera. Event Summarization Using Tweets. *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, pages 66–73, 2011.

[7] Y. Chen, H. Amiri, Z. Li, and T.-S. Chua. Emerging topic detection for organizations from microblogs. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, page 43, 2013.

[8] B. Connor, M. Krieger, and D. Ahn. TweetMotif: Exploratory Search and Topic Summarization for Twitter. *ICWSM*, 2010.

[9] K. Filippova. Multi-sentence compression: finding shortest paths in word graphs. *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 322–330, 2010.

[10] D. Gibson and J. Kleinberg. Clustering categorical data: An approach based on dynamical systems. *The VLDB Journal - The International Journal on Very Large Data Bases*, pages 311–322, 2000.

[11] L. Gong, J. Zeng, and S. Zhang. Text stream clustering algorithm based on adaptive feature selection. *Expert Systems with Applications*, 38(3):1393–1399, Mar. 2011.

[12] E. Günes and R. Dragomir R. exRank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.(JAIR)*, 22(1):457–479, 2004.

[13] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer, 2001.

[14] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis. Discovering geographical topics in the twitter stream. *Proceedings of the 21st international conference on World Wide Web - WWW '12*, page 769, 2012.

[15] A. Marcus, B. Michael, B. Osama, K. David, M. Samuel, and M. Robert C. TwitInfo: Aggregating and Visualizing Microblogs for Event Exploration. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 227–236, 2011.

[16] D. Marian and E. Al. Visual Backchannel for Large-Scale Events. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1129–1138, 2010.

[17] J. Nichols, J. Mahmud, and C. Drews. Summarizing sporting events using twitter. *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*, pages 189–198, 2012.

[18] A. Olariu. Hierarchical clustering in improving microblog stream summarization. *Proceedings of the 14th international conference on Computational Linguistics and Intelligent Text Processing*, 2(March):424–435, 2013.

[19] L. Rabiner. A tutorial on hmm and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[20] B. Sharifi, M.-a. Hutton, and J. Kalita. Summarizing Microblogs Automatically. *HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, (June):685–688, 2010.

[21] L. Shou, Z. Wang, K. Chen, and G. Chen. Sumblr: continuous summarization of evolving tweet streams. *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval*, pages 533–542, 2013.

[22] C. Silverstein and P. Jan. Almost-constant-time clustering of arbitrary corpus subsets. *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 60–66, 1997.

[23] C. van Rijsbergen, J. Cornelis, S. Robertson, and M. Porter. New models in probabilistic information retrieval. 1980.

[24] M. Walther and M. Kaisser. Geo-spatial Event Detection in the Twitter Stream. *Proceedings of the 35th European conference on Advances in Information Retrieval*, pages 356–367, 2013.

[25] Y. Wang, E. Agichtein, and M. Benzi. Tm-lda: efficient online modeling of latent topic transitions in social media. *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 123–131, 2012.