Gossip Insights

A Graph-based Keyword Extraction & Visualisation for Contextual Analyses of Social Media Trends

Felix Heck
Stuttgart, Germany
hi@whotheheck.de

November 2018

***Abstract*—When analysing social media trends with software like Brandwatch Analytics different visualisations such as word clouds are available, which represents the most prominent key words and sentences in the selected time interval. Although the word clouds provide a good overview of discussed topics and diverse insights into trends, further information, including contextual relationships between word groups, is lost in this visualisation. These can be useful for the rapid exploration of several discussed topics and their significance in the selected time interval. The aim of this thesis was to design and implement a prototype to the problem, so that important topics can be extracted, visualised and explored in a contextual manner.**

For this purpose, the problems were analysed and requirements were specified as well as prioritised. In the scope of the conceptual design and implementation, a multi-stage process was developed which extracts significant keywords from a collection of tweets and ranks them by various heuristics and a graph; the constructed graph was also used for the visualisation of the keywords. Thereupon, the implementation was qualitatively evaluated.

The result of this paper is a prototype which compensates for conceptual disadvantages of word clouds, and also optimises the underlying keyword extraction by considering various statistical heuristics.

***Index Terms*—Information Retrieval; Web Mining; Natural Language Processing; Graph Theory; Data Visualisation; Twitter**

I. Introduction

A. Motivation

Brandwatch Analytics is a software that enables real-time analyses of social media data by collecting mentions[1]. For this purpose the application continually gathers data using queries[2] defined by customers or internal research analysts. On top, provided dashboard[3] templates and components[4] allow to get quick insights into trends and further to customise the visualised data. In case of significant peaks in line charts visualising the volume of mentions per time interval, it might be of great use to quickly identify the most discussed topics to be able to react responsively to trends [1].

The use of the word cloud enables the user to identify the most common terms in relation to the peak data[5]. More precisely, word clouds are usually an accumulation of n-grams, which occur most frequently in the corresponding corpus. Particularly frequently occurring n-grams are usually displayed in larger font sizes and are more likely to be placed in the centre (figure 1).

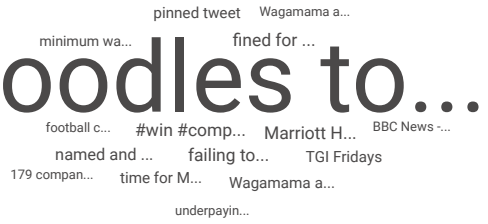

Figure 1. Exemplary Word Cloud Component

Although the word cloud provides a good overview of discussed topics and a wide range of insights into trends, such as the visualisation of growth or sentiment, some disadvantages have been worked out in cooperation with data scientists and research analysts. In the following, these points of criticism are described in detail.

Since the axes of the word clouds often have no particular meaning, terms are arranged randomly and placed in such a way that only a minimum of free space remains. Due to this comparatively simple visualisation, a lot of mostly context-related information is lost. This leads to the first problem to be addressed in this paper. The word cloud often contains several terms that belong to the same topic; however, this fact is not immediately apparent. To be able to recognise these relations, it is usually necessary to

[1] A content, e.g. a comment on social media, matching queryies.
[2] Search string including operators used to match mentions.
[3] Multiple components enabling to analyse found mentions.
[4] Modular visualisation tools providing different data.
[5] Data relating to the period of the actual peak or event.

analyse the content and context of the respective terms by examining the mentions more closely.

Apart from the visualisation itself, the algorithm that extracts the terms from the corpus is also responsible for the loss of contextual information. The algorithm filters and scores the keyword candidates mainly based on their frequency. With this background, it is now possible to describe the remaining problems.

Interactions between users are ubiquitous in social media so that single posts are both referred to or commented on directly. This results in threads, whose individual posts might differ substantially in their choice of words. The reason for this is that, as in normal conversations, reference is usually made to the initial statement and the use of specific context-giving terms is not essential. As a result, terms are not seen as relevant because they do not occur frequently enough, although a single topic is discussed in the narrower sense.

Furthermore, especially on Twitter, it is quite usual to share other posts. These shared tweets are also known as retweets and may also become viral in some cases. However, other pages enable the sharing of articles with prepared texts as well. This leads to the fact that there might be multiple mentions with the same choice of words. In the case of particularly popular retweets, their terms often emerge in the word cloud, whereby other terms are entirely suppressed or partly disappear, even though other topics were also discussed more frequently.

The problems outlined above result in analysts being deprived of contextual information that can be useful for analysing trends and identifying topics of interest to the company. As a result, those analysts must spend additional time extracting such information or to verifying that the visualisation is complete. Addressing these problems can lead to financial and time savings.

B. Objectives

This paper aims to reduce the additional time-consuming tasks by conceptualising and realising a prototypical application. All identified problems were addressed as far as possible to provide analysts with intuitive and straightforward contextual insights into discussed topics and discussions in general in the future.

Thus, both the extraction of terms and their corresponding visualisation are part of the implementation. The detection of events is explicitly not part of the work, as it would exceed the scope of this paper and is already being researched internally. Which particular techniques and approaches are used remained open; these decisions were based on the analysis of related work and were made as part of the design process.

C. Requirements

Firstly, the number of mentions which are required for the algorithm to extract the keywords should be kept as low as possible to enable fast processing without losing accuracy and completeness. Secondly, the stored data is not annotated with keywords, so the approach has to be unsupervised. Additionally, the mentions should be preprocessed and normalised adequately to ensure the highest possible quality of the extracted keywords; this is especially important for social snippets like tweets due to the high noise [2].

For the prototypical implementation, the focus will be on English Twitter data, since Brandwatch provides full coverage and Twitter is one of the most important microblogging services that enables users to express their opinions or discuss topics interactively. This results in large amounts of information, which can be caused by personal as well as local or global events such as disasters and social movements [1][3][4][5][6]. For instance, 1% of the public Twitter stream already covers about 95% of all events deposited with news agencies [7][8]. Moreover, English is both the most widely used language on Twitter and Brandwatch customers' favourite one [9].

Furthermore, the keyword extraction should extract n-grams ($n \geq 1$) with the help of linguistic characteristics, as additional surrounding words provide a more detailed context. The extraction itself as well as the weighting of the keywords must take into account the problems mentioned above, so that those do not have any negative influence on the future prototype. Finally, the extracted keywords must be clustered in such a way that contextual relationships are taken into account. In addition to the algorithmic process, the resulting clusters must also be visualised and, if possible, assigned a single n-gram per cluster. An interactive visualisation is intended to enable the user to examine specific parts in more detail and to highlight these in presentations.

The extraction is the core of the prototype and is therefore the most important; the focus is on contextual features in particular. Clustering for visualising contextual information and interactions for a more straightforward examination are mandatory and thus corresponds to the extraction requirements.

II. Related Work

A. Keyword Extraction

The process of keyword extraction is a simplistic topic model and attempts to automatically identify the terms in an unstructured text that best describe the topic of the document [10][11][12]. Keywords serve to characterise topics discussed in the document. More precisely, keywords can be used to index, classify and summarise a document or collection of documents [13].

Keyword extraction can be structured according to Ping-I, Shi-Jen and Zhang into statistical, linguistic, machine learning-based and other approaches [14][15]. Machine-learning based approaches are mostly disregarded in the subsequent consideration. The reason for this is that supervised approaches are particularly common in this area, but cannot be used due to the absence of annotations. Therefore, the focus is on unsupervised approaches that do not contain any learning components [11][16]. Below, these approaches are briefly characterised and relevant research in the respective field is outlined.

1) Statistical Approaches: Statistical approaches use simple methods that require no training and are also independent of language and domain. Statistics of words or n-grams are used to identify keywords in documents [14][15]. The term frequency (TF) is one of the most important factors and is the basis for further statistics such as term frequency-inverse document freuqncy (TF-IDF) [3][17]. Other examples include word co-occurrences which expresses n-grams co-occurring within a defined window, a sentence, paragraph or document [11]. Statistical features that seem to be interesting are those that mainly relate to TF such as varying weightings of the TF-IDF score. These features can be extended by further features. For example, both the subsumption count and the length of the term are of interest [18][19][20].

2) Linguistic Approaches: Linguistic approaches that use linguistic properties of text parts include lexical, syntactic and discourse analysis [15]. For this purpose, the results of part of speech (POS) tagging and named-entity recognition (NER) are used as well. The linguistic features are especially interesting in terms of named entities and POS tags. However, since these entities are essentially nouns, these can be captured with POS tag patterns targeting noun phrases (NP). Moreover, the surrounding words provide more context [21][22].

3) Graph-based Approaches: A graph is a mathematical model, which enables to explore relations and structures efficiently [23]. Graphs have in common that a text source is modelled by representing terms by nodes and connections by edges. The edges can represent various metrics and relations like co-occurrence, syntax and semantics [18][24][25][26][27]. The basic idea is to evaluate the graph by ranking the importance of individual nodes [26][28]. Therefore, the graph-based approaches tend to combine several of the approaches already mentioned [11]. The approach of Keygraph, which uses co-occurrences and basic statistics to build a graph, forms the basis of many graph-based approaches [29]. Various extensions are considered for the prototype, such as the use of POS filters, clustering methods and PageRank to identify the most important keywords. The two-stage ranking, consisting of TF-IDF and PageRank, is also considered as part of the prototyping [10][20][27][30].

B. Community Detection

In order to identify topics and related keywords in a graph-based approach, the keywords have to be clustered. The resulting clusters are also called communities. Communities are groups of nodes within a network which have a higher intra-connectivity and a relatively weak inter-connectivity [31][32][33]. The intra-group connections are therefore much denser. Various community detection approaches – which can essentially be broken down into modularity-based, spectral and random walks-based algorithms as well as label propagation and information-theoretical measures [31] – were considered.

The evaluations of the different community detection algorithms differ in some parts. Mothe *et al.* identified Louvain and the Leading Eigenvector algorithm as the best performing algorithms for communities with high modularity [33]. Günce *et al.* argues that Infomap outperforms all other algorithms, even if algorithms like Walktrap or Louvain yield excellent results. Infomap, Infomod and Louvain seem to work best on larger networks [34]. Emmons *et al.*, in contrast, conclude that Louvain also surpasses Infomap's performance and thus contradicts Günce *et al.* [35]. However, all evaluations have in common that Louvain delivers excellent to the best results. Based on these evaluations, Louvain is used in the prototype.

III. Conceptual Design

The following chapter covers the conceptual design prototype and its individual steps. It is a summary of the initial conceptual design of the pilot experiment as well as the final prototype, which was improved using the results of a UX research to further fulfil the demands of the users. During the development of the pilot experiment, an iterative evaluation took place in order to compare the advantages of different approaches. Due to the prototype character, this is explicitly not about selecting the best model for tokenisation and POS tagging, but about the general approach of data processing, keywork extraction and ranking as well as the contextual visualisation.

The basic concept of Gossip Insights is inspired by the concepts of SGRank which extracts keywords in several stages: extraction of n-grams and removal of those keywords that are unlikely to be keywords; multiple rankings of the remaining n-grams with a modified TF-IDF heuristics and additional ones; and final ranking using a graph [20].

The Gossip Insights algorithm first extracts all possible n-grams using POS tag patterns and removes all candidates that are unlikely to be keywords. In addition, the terms are lemmatised to make it easier to group them. Subsequently, the terms are ranked with the help of a score, which is mainly based on frequency but also on further statistics. A graph is then generated showing the co-occurrences of the remaining keywords. With the help of this graph, keywords can not only be clustered into conversations and discussion

topics, but also the most important keywords per cluster can be determined. The resulting visualisation shows the relations of the keywords, taking into account not only the weights of the nodes but also those of the edges.

A. Data Preparation

As a first step, all texts are pre-processed and normalised to compensate for specific characteristics of social snippets and to generalise the algorithm better. To simplify the text and reduce the number of possible characters, all corrupted Unicode symbols are replaced, and all characters are transliterated. In this case, incorrect Unicode symbols are the result of unintended character encoding, which often replaces characters that are unrelated to the original one; or HTML entities that are not displayed correctly. Transliterating characters means that all characters that do not conform to ASCII are converted to characters that most closely match the original character. In the case of accents, these are replaced by the corresponding character without an accent or ellipses by three dots.

Moreover, all URLs are removed from the texts. Those URLs are already extracted and provided as an additional property by Twitter. To continue, mail addresses with `EMAIL`, phone numbers with `PHONE` and URLs with `URL`; even if email addresses and phone numbers are rarely published on Twitter and URLs should have been already removed. Besides, the detached string `&` is replaced with the equivalent `and` to normalise texts even more. As a final step, English contractions are replaced by the corresponding initial words. This also serves to standardise the texts better as well as simplify the identification of keywords.

To avoid having to look for extraction of the keywords, which of the Twitter handles originate from retweets and which are actual @-mentions, those flags are already removed during preprocessing using regular expressions. The reason why authors are in the graph besides @-mentions is that Twitter flags retweets with `RT <author_handle>:`. Thus the authors, whose tweets have been retweeted very often, would get into the visualisation. Unlike when modifying the generated tokens, a simplified pattern can be used here. The reason for this is that by combining `RT` and `<author_handle>` at the beginning of the mention, the probability of removing other text fragments is very low. Optionally, a trailing string consisting of colon and whitespace is removed.

As the last step in data preparation, the data is restructured to simplify the further process. The restructuring takes place in two steps: grouping the mentions by days and reducing the hierarchy of these groupings. These mentions are not the mentions of the peak itself, but those of background data[6] which are used to draw more peak-specific conclusions by comparing their statistics with those of the peak. For this purpose, the mentions grouped by days can be merged into so-called pseudo documents.

[6]Data relating to the period before the actual peak or event.

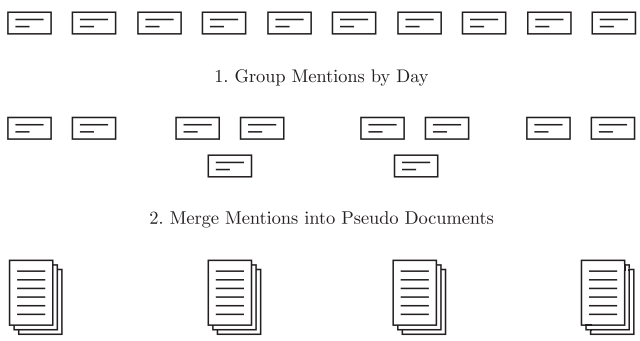

Figure 2. Schematic Grouping and Merging into Pseudo-Documents

B. Data Processing

In addition to tokenisation, data processing mainly deals with Twitter-specific steps. In addition to the tokenisation and its correction with regard to Twitter-specific features, a Twitter thread tree must also be created.

1) Tokenisation and POS Tagging: For further processing, such as extracting the keywords, ranking and visualisation, it is necessary to transform the pre-processed texts into tokens and to assign POS tags to these tokens. For this purpose, the library spaCy is used, which has several built-in and pre-trained models for different use cases. spaCy is primarily based on pipelines, which consist of the default of a Tokeniser, POS Tagger, Dependency Parser and Named Entity Recogniser. Only the first two steps are required for the prototype. The choice of the model for POS tagging is the English model of medium size. This is a neural model trained with blogs, news and comments. It has an accuracy of 97,11% regarding POS tagging. Since there is currently no POS model for spaCy that has been trained using Twitter data, and there is no necessity for such a model within the scope of the prototype, the available one is used instead.

2) Twitter-specific Tokenisation: The tokeniser usually cannot handle Twitter usernames, the commonly known @-mentions, and hashtags, because the respective prefix is separated from the rest of the token. However, since they each form a unit, the tokens must be adjusted manually in these cases. To adjust the tokens, the procedure is as follows. Two regular expressions are used to search the mentions for hashtags and @-mentions. The respective matches are less interesting, but rather the position of the matches. With the help of the start and end position of the match, all tokens within this window can be merged into a single one. Both regular expressions are based on the `TweetTokenizer` of the Natural Language Tool Kit.

3) Twitter Thread Tree: To consider threads on Twitter, this information must be provided for each tweet. For this purpose, the algorithm uses the Twitter API, which allows querying the necessary information per tweet. This approach requires a large number of requests, but

caching attempts to minimise the disadvantages for the development of the prototype. To create the thread tree, the superordinate tweets are requested from the API. Superordinate tweets are commented ones as well as with a comment retweeted ones. This step is performed iteratively until no superordinate tweet is available. All queried tweets are now part of a thread while the last tweet of a thread represents its origin and thus its identifier. With the help of the grouping of tweets in threads, pseudo-documents can now be created again. Both the pseudo-documents and the thread volumes, the number of tweets per thread, allow to define co-occurrences in threads later and to calculate corresponding weights.

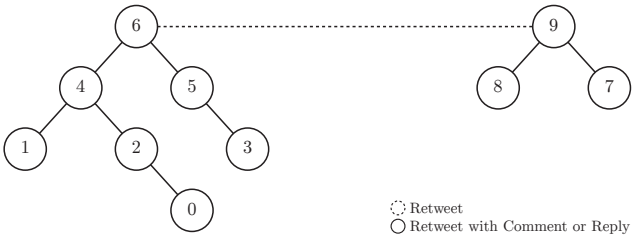

Figure 3. Exemplary Twitter Thread Tree

C. Extraction of Keyword Candidates

1) Definition of POS Tag Patterns: As already mentioned above, NP chunks based on POS tag patterns are suitable for extracting keywords. This is proven by Hulth, Li Z. *et al.* as well as Alrehamy and Walker. In addition to the Twitter-specific tokens such as hashtags and handles, the following patterns are defined – the notation is based on the Universal POS Tagset, which generalises the widely used Penn Treebank notation [36]. This universal tagset is also used by spaCy.

- `(ADJ)?(NOUN|PROPN)*(STOP|X)?(NOUN|PROPN)+`
- `(SYM)?(NUM)+(SYM)?(NOUN)*`

These patterns are derived from the patterns used by SemCluster [37]. Nouns and proper names are used synonymously in many patterns since the model tags proper names mostly on a case-sensitive basis – since this is often not taken into account in social media, it cannot be relied upon. The first pattern combines all SemCluster patterns: individual nouns and proper names as well as the concatenation of these. These can also occur in combination with a leading adjective so that those are described more specifically. The last section is for entities, a sequence of nouns or proper names that contains an optional stopword in the middle; in addition to the stop words, unknown tokens that are not Twitter-specific are also taken into account. The second pattern handles numeric tokens and consists of two composite patterns. One for currencies or units that includes optional symbols before or after the numeric sequence. Another that describes subsequent nouns in more detail by defining the quantity.

The extracted candidates are subsequently cleaned up to compensate for incorrect tagging. This removes leading or trailing stop words as well as candidates which are part of a blacklist or stop word list. Keywords consisting of only one character are also removed. In order to be able to summarise terms better downstream and thus minimise duplicates such as pluralisation, another representation of the term is created which consists of the lemma of the keyword and which no longer contains whitespaces.

2) Define Frequency Measures: For the terms of the filtered list, the frequency within the mentions is determined subsequently while taking word boundaries into account. Additionally a subsumption count is determined, which defines how many terms are the superset of a specific term:

$$\mathrm{ssc}(t,d) = 2.25 \cdot \sum f_{t',d}\,[t \subset t']$$

This subsumption count is used later to reduce the weighting of terms that offer less context, thus minimising overlaps and duplicates. This concept is based on SGRank [20]. In contrast, however, the leading factor of 2.25 is used to prioritise supersets more strongly and to not only reduce the weighting of shorter terms but also to exclude them completely in some cases.

3) Group Candidates: As already mentioned, there are several representations stored per extracted keyword to minimise duplicates. The lemma of the keyword without whitespaces is utilised in this step to unite keyword candidates that overlap in terms of representations. Previously determined heuristics, such as the frequency and the subsumption count, have to be merged.

D. Ranking and Selection of Keywords

1) Calculate Modified TF-IDF Score: The TF-IDF score is essentially based on the functions $tf(t,d)$, the term frequency of the term t in document d and $idf(t,D)$, the inverse document frequency of the same term in document corpus D; several variants exist for both functions. The term frequency $tf(t,d)$ is basically the absolute occurrence frequency ($f_{t,d}$) of a term t in document d. This metric can be normalised using the maximum occurrence frequency, so that $0 \leq tf(t,d) \leq 1$ applies. To reduce the weight of word sequences that represent subsets of other word sequences, the frequency is also reduced by the subsumption count $ssc(t,d)$. Thus, $tf(t,d) \leq 1$ applies. The inverse document frequency is the inverse ratio of d documents in the D corpus, which contain the term t, to the total number of documents in the corpus.

$$tf(t,d) = \frac{f_{t,d} - ssc(t,d)}{\max(f_{t',d} : t' \in d)}$$

$$idf(t,D) = \log_{10}\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right)$$

The final TF-IDF score usually is the product of the functions $tf(t,d)$ and $idf(t,D)$, but is varied here. To prefer n-grams with a larger n, the TF-IDF score is extended by the square root of n. This non-linear factor has only an insignificant influence on the result but emphasises the effect of the subsumption count. This concept is based on the approach proposed by Alrehamy and Walker [37].

$$tfidf(t,d,D) = tf(t,d) \cdot idf(t,D) \cdot \sqrt{|t|}$$

In addition, the normal weighting for determining the inverse document frequency is preferred, as this causes $tfidf(t,d,D) = 0$ for words that appear in all documents. This allows those words to be filtered independently of the term frequency. The filtering includes removing those keywords with $tfidf(t,d,D) < 0.005$, and selecting up to 150 remaining ones with the highest scores.

2) Calculate Edge Weights: As in many other approaches, the edges in the graph are based on co-occurrences. However, not on co-occurrences in individual documents, but rather in pseudo-documents based on threads. This ensures that keywords that are part of a wide-ranging conversation are also connected in the graph. The weights of the edges are based on the previously determined volumes of the individual threads, more precisely, it is the sum of the volumes of all threads in which those terms co-occur. At this point, the edge weights are also normalised so that they are within the range $[0, 1]$.

E. Graph Creation & Clustering

1) Graph Creation: The creation of the graph and the corresponding community detection take several steps. In addition to the actual creation and clustering, this includes filtering, customisations and optional steps. The edges and their weights created based on the co-occurrences in threads are used to create an undirected graph, where the nodes are automatically generated by the start and end points of the edges. The resulting graph can be processed directly, and the communities detected using the Louvain approach. This results in a mapping that assigns each node a community. The nodes are extended with metadata based on this mapping definition and the previously calculated heuristics. In addition to identifiers, some attributes describe the community and the weighting. The weighting consists of the product of the previously calculated TF-IDF and the degree of the respective node; the degree is determined by the number of outgoing/incoming edges [38]. This is based on the observations of Palshikar that central nodes in the network are often keyword candidates [13]. The degree centrality is, therefore, an easy to calculate but efficient way. Afterwards, the generated nodes and communities are cleaned up by removing all communities that are exclusively based on hashtags or Twitter handles. Such communities tend to be spam and are therefore negligible.

2) Graph Filtering: In order to make the visualisation simpler and more transparent, the number of edges and nodes must be reasonably reduced. However, since removing the edges causes missing context, another attempt is made based on the TF-IDF scores. After creating the graph and thus the node weights, the nodes are filtered again by selecting only the up to thirty nodes with the highest weighting. This ensures that all necessary contextual information is retained between the nodes, while at the same time the graph becomes simpler to read.

3) Assign Single Keyword: To meet the requirement to assign a single keyword to communities, the community structure is simplified first. The aim is to temporarily group all communities that are connected with edges for determining the keyword. In addition, all communities with only two nodes are removed to filter insignificant ones. To extract finally the most important node per community the algorithm is taking advantage of the degree centrality, as Palshikar recommends [13].

4) Embed Mentions: Similar to existing topic components, it should be possible to display the mentions for each keyword in which the respective keyword occurs. In previous components the twenty most current mentions were displayed and it is possible to apply various filters; a pagination is usually implemented to get further mentions. This should also be implemented in the graph, apart from the fact that neither pagination nor duplicates should be displayed; retweets are reduced to a single tweet to preserve the simplicity. Furthermore, filters are not necessary for the prototype. To find the identifiers of the corresponding mentions, the pipeline is extended. In this step, the program searches in reverse chronological order for matching mentions for each node; the identifiers are mapped to the nodes. Up to 250 unique mentions are assigned to the nodes, even if only a fraction of them is displayed. Since the mentions should not only be embedded in nodes but also in edges – to show mentions in which both keywords occur – it is ensured that both nodes have been assigned sufficient mentions to obtain meaningful intersections. However, it is not the list of unique mentions assigned to the nodes, but the complete one. This can be used to determine at a later stage how much is made up of retweets.

5) Detect Retweet-based Clusters: Even if most participants of the UX research sessions were able to independently identify the different nature of the clusters and thus in particular to find retweet-based clusters, these should be explicitly marked as such by embedding this information in the nodes. Since the colour is already used as an attribute for the segmentation of the keyword types, other shapes are used in this case; all retweet-based clusters or subclusters are therefore displayed as squares. To identify these clusters, this step uses the embedded list of associated mentions. For this purpose, all mentions are unified per subcluster, not the aggregated ones, and the ratio of uniques to all mentions is determined. The threshold value is set at 90%

so that those subclusters whose unique tweets represent only 10% are marked as retweet-based.

$$retweetRatio = 1 - \frac{|\{x_1, \ldots, x_n\}|}{|(x_1, \ldots, x_n)|}$$

Also, retweet-based clusters should be hideable to be able to focus on more wide-ranging clusters. Initially, however, all clusters are displayed so that no information is withheld.

F. Visualisation

After the most important keywords have been extracted, ranked and exported, these are visualised in the next and final step. Since the exported graph is already a contextual visualisation that maps the information thoroughly, it is used as a baseline and conceptualised below.

1) Force-directed Graph Layout: The layout of the network or graph should be based on a force-directed layout [39][40]. Force-directed graphs are primarily built on an attracting force between connected nodes and a repulsive force between nodes in general. This creates different clusters based on the edges, which provide insight into connections between nodes [41]. This considers the statements of Borgatti *et al.* that poorly laid out networks not only convey too little information but can also be misunderstood [23]. Instead of the usual force layouts based on spring forces and Coulomb's law, charge-based forces are used. While negative charges, i.e. low weightings or unconnected nodes repulse, positive charges, i.e. nodes that are connected with strongly weighted edges, attract [40][42] (figure 3).

Figure 4. Random versus Force-directed Layout in Equilibrium State

The distance is also influenced by a weak geometric constraint, where a function determines the optimal distance between the respective nodes [42]. This results in natural subclusters according to the data, which, among other things, visualise conversations on the topic of a cluster.

2) Color-based Segmentation: During the feedback sessions, reference was made to various segmentations already existing in Brandwatch Analytics. These include @-mentions, hashtags, sentiments (positive, neutral and negative), named entities (organisations, people and locations) and remaining usual keywords. The colours are used according to the style guide to embedding such segmentation into nodes. In the first step, particular focus is placed on hashtags, @-mentions and usual keywords are highlighted accordingly. Sentiment and named entities are therefore initially neglected and considered for future steps and

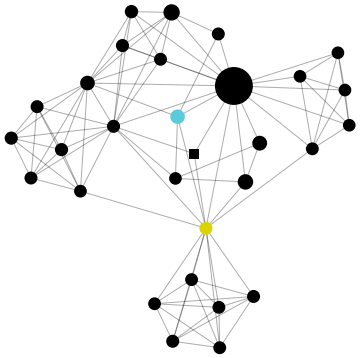

Figure 5. Subclusters within a Common Topic

extensions. This is mainly due to the fact that this would exceed the time and scope of the prototype. Hashtags and @-mentions can be easily identified by their prefix.

3) Interactions: The basic navigation concept should be based on the depth of the interaction and be reflected in the provided information. No inconsistencies should be created, i.e. the interactions start with the hovering, can be extended by a single click and end with a double click. When hovering a node, the respective label is displayed directly above the node; when clicking once, all nodes and edges that are not directly connected to the node are hidden; when double-clicking, the corresponding mentions are displayed. Equivalent to the interactions with the nodes, these are also introduced for edges, so that the edge and the two associated nodes are shown with one click, while the others are hidden. A double click, displays the mentions belonging to the edge. This edge-specific behaviour is almost equivalent to the previous solution, so that it can be easily extended. If the whitespace is the target of a click and not a node, one step back is taken. So it is possible to get back from the view of the mentions and the subgraph.

4) Legend for Navigation & Segmentation: To simplify the introduction and the working with the visualisation, Gossip Insights is extended by a legend in the form of a further sidebar. In addition to explaining the main interactions, this should also show the segmentation and provide the opportunity to hide and show the retweet-based clusters. To illustrate the depth of the interactions, this is reflected in the legend by starting with the hovering and listing clicks and double-clicks afterwards. However, since the sidebar can be distracting and irritating, especially at a higher zoom level, its state is also linked to the zoom level. As soon as the zoom threshold is exceeded, the sidebar disappears, allowing the user a better overview.

IV. Evaluation

To evaluate the prototypical implementation finally, the defined requirements, the approaches derived from related work and the UX research findings are examined and

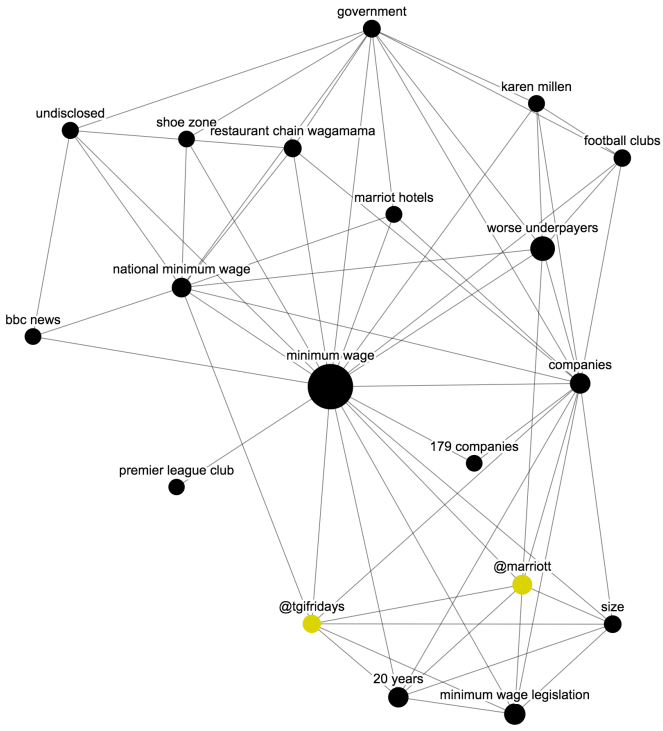

Figure 6. Detail View of a Single Cluster

evaluated. At this point it is explicitly emphasised that this is a prototypical implementation and especially the functionality and the concept are in focus; the quality of the implementation or the selection of the best models is of minor importance.

The final version of Gossip Insights is an unsupervised approach that extracts n-grams of different lengths out of a sampled and normalised Twitter dataset using POS tag patterns and multi-stage rankings. By using graphs, contextual relationships are included and displayed, resulting in clustering as well as nodes and edges; both the edges and the clusters take threads into account. The visualisation in the form of a graph also enables new metrics for weighting the nodes, so that retweets are less significant. Gossip Insights is completed by a variety of interactions with the visualisation and the mapping of the most important node per cluster.

Reviewing the approaches that were classified as reasonable and worth considering in the analysis of related work, it results that nearly all approaches and features were meaningfully implemented in Gossip Insights without unnecessarily increasing the complexity. Through co-occurrences, sufficient contextual relationships can be created to visualise the keywords properly.

Regarding the findings of the UX research, almost all weaknesses have been eliminated within the context of the prototype, so that it can be assumed that this adds further value; various brief practical tests with analysts confirm this impression. Due to the limited time and the narrow scope, two features are not implemented in the prototype. In addition to the statistics for nodes and clusters, this also includes the segmentation of named entities.

In summary, Gossip Insights meets the results of the requirements analysis and user feedback. Only two of the subsequently requested features are not implemented due to the limited time frame. In addition to the formal requirements, the prototype also proves itself in practice with various datasets.

V. Conclusion

In the process of researching approaches, technologies and implementations that deal with the given problem, only individual aspects and not an entirely suitable solution were identified. On closer examination of these, approaches were extracted which seemed reasonable and promising in combination. The challenges became evident only in the conceptual design and implementation of the pilot experiment.

First, the extraction of keywords based on POS tag patterns turned out to be difficult because models with noisy social media data partly encounter difficulties. Second, the balanced and meaningful selection of keywords through ranking and selection of keyword candidates, nodes and edges. Moreover, last but not least, contextual visualisation, which embeds various metrics while maintaining the balance between detail and overview. However, the use of the pilot experiment for several datasets and its evaluation in the form of a UX research session revealed, apart from proof of the concept, that the implementation has to be extended by further features, which increase in particular the UX, but also the confidence of the users in the visualisation. For this purpose, solutions could be realised immediately after that. Their implementation, the fulfilment of all requirements, objectives and almost all user requests as well as the convincing way the prototypical implementation works with different datasets ensure that analysts can already use it on a trial basis.

VI. Future Work

For the further development of the prototype, various modifications and extensions are possible; a summary of some of them is given below, which is intended to provide a perspective for the future.

First of all, the features that were requested during the UX research sessions, such as embedding statistics on clusters, nodes and edges – like the volume, share or sentiment – as well as extending the segmentation by named entities or emojis, which would need a NER step. The mentions and the associated sidebar can also be adjusted. It is feasible to replace the Twitter widget with a custom implementation to highlight the corresponding keyword in the individual tweet. Also, interactions and features of

existing components could be adapted to make mentions filterable and sortable as well as to introduce pagination.

As Borgatti *et al.* demonstrate, graphs can also be extended to scatter plots by using axes to visualise attributes. Thus, the volume, the cluster size, impact, sentiment, trend related factors or others can be mapped [23]. Even if the prototype already allows to highlight 1-degree ego networks, it would be potential to extend this functionality. For example, with n-degree-highlights or the temporary removal of ego nodes to identify subclusters more easily. This also includes highlighting nodes which are connected to all other nodes within a cluster to simplify the identification of nodes worth removing. The last functional extension is the visualisation of time series to be able to observe the evolution of a graph within an interval.

References

[1] F. Atefeh and W. Khreich, "A survey of techniques for event detection in twitter," *Computational Intelligence*, vol. 31, no. 1, pp. 132–164, 2015.

[2] A. O. Steinskog, J. F. Therkelsen, and B. Gambäck, "Twitter topic modeling by tweet aggregation," in *Proceedings of the 21st nordic conference of computational linguistics*, 2017, pp. 77–86.

[3] Shuangyong Song, Y. Meng, and J. Sun, "Detecting keyphrases in micro-blogging with graph modeling of information diffusion," in *13th pacific rim international conference on artificial intelligence: Trends in artificial intelligence*, 2014, pp. 26–38.

[4] J. Hurlock and M. L. Wilson, "Searching twitter: Separating the tweet from the chaff," in *Proceedings of the 5th international aaai conference on weblogs and social media*, 2011.

[5] H. Kwak, C. Lee, H. Park, and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th international conference on world wide web*, 2010, pp. 591–600.

[6] A. Ritter, Mausam, O. Etzioni, and S. Clark, "Open domain event extraction from twitter," in *Proceedings of the 18th acm sigkdd international conference on knowledge discovery and data mining*, 2012, pp. 1104–1112.

[7] D. Zhou, L. Chen, X. Zhang, and Y. He, "Unsupervised event exploration from social text streams," *Intelligent Data Analysis*, vol. 21, no. 4, pp. 849–866, 2017.

[8] S. Petrović, M. Osborne, R. McCreadie, C. Macdonald, I. Ounis, and L. Shrimpton, "Can twitter replace newswire for breaking news?" in *Proceedings of the 7th international aaai conference on weblogs and social media*, 2013.

[9] M. Orcutt, "The many tongues of twitter." MIT Technology Report, 2013.

[10] R. Mihalcea and P. Tarau, "TextRank: Bringing order into texts," in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004.

[11] S. Beliga, A. Meštrović, and S. Martinčić-Ipšić, "An overview of graph-based keyword extraction methods and approaches," *Journal of Information and Organizational Sciences*, vol. 39, no. 1, pp. 1–20, 2015.

[12] D. Sarkar, *Text analytics with python: A practical real-world approach to gaining actionable insights from your data*, 1st ed. New York, NY, USA: Apress Media LLC, 2016.

[13] G. K. Palshikar, "Keyword extraction from a single document using centrality measures," in *Pattern recognition and machine intelligence*, 2007, pp. 503–510.

[14] P.-I. Chen and S.-J. Lin, "Automatic keyword prediction using google similarity distance," *Expert Systems with Applications: An International Journal*, vol. 37, no. 3, pp. 1928–1938, 2010.

[15] C. Zhang, H. Wang, Y. Liu, D. Wu, Y. Liao, and B. Wang, "Automatic keyword extraction from documents using conditional random fields," in *Journal of computational information systems*, 2008, vol. 4, pp. 1169–1180.

[16] K. Bennani-Smires, C. Musat, M. Jaggi, A. Hossmann, and M. Baeriswyl, "EmbedRank: Unsupervised keyphrase extraction using sentence embeddings," *Computing Research Repository*, vol. abs/1801.04470, 2018.

[17] Y. Liu, B. J. Ciliax, K. Borges, V. Dasigi, A. Ram, S. B. Navathe, and R. Dingledine, "Comparison of two schemes for automatic keyword extraction from medline for functional gene clustering," in *Proceedings of the ieee computational systems bioinformatics conference*, 2004, pp. 394–404.

[18] H. Sayyadi, M. Hurst, and A. Maykov, "Event detection and tracking in social streams," in *In proceedings of the international conference on weblogs and social media*, 2009.

[19] J. Li, Q. Fan, and K. Zhang, "Keyword extraction based on tf/idf for chinese news document," *Wuhan University Journal of Natural Sciences*, vol. 12, no. 5, pp. 917–921, 2007.

[20] S. Danesh, T. Sumner, and J. H. Martin, "SGRank: Combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction," in *Proceedings of the fourth joint conference on lexical and computational semantics*, 2015, pp. 117–126.

[21] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the 2003 conference on empirical methods in natural language processing*, 2003, pp. 216–223.

[22] Z. Li, D. Zhou, Y.-F. Juan, and J. Han, "Keyword extraction for social snippets," in *Proceedings of the 19th international conference on world wide web*, 2010, pp. 1143–1144.

[23] S. P. Borgatti, M. G. Everett, and J. C. Johnson, *Analyzing social networks*, 2nd ed. Thousand Oaks, CA, USA: SAGE Publications, Inc., 2018.

[24] S. S. Sonawane and P. A. Kulkarni, "Graph based representation and analysis of text document: A survey of techniques," *International Journal of Computer Applications*, vol. 96, no. 19, pp. 1–8, 2014.

[25] R. Mihalcea and D. Radev, *Graph-based natural language processing and information retrieval*, 1st ed. New York, NY, USA: Cambridge University Press, 2011.

[26] K. S. Hasan and V. Ng, "Automatic keyphrase extraction: A survey of the state of the art," in *Proceedings of the 52nd annual meeting of the association for computational linguistics*, 2014, vol. 1x, pp. 1262–1273.

[27] M. Grineva, M. Grinev, and D. Lizorkin, "Extracting key terms from noisy and multitheme documents," in *Proceedings of the 18th international conference on world wide web*, 2009, pp. 661–670.

[28] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *Proceedings of the 7th international conference on world wide web*, 1998, pp. 110–117.

[29] Y. Ohsawa, N. E. Benson, and M. Yachida, "KeyGraph: Automatic indexing by co-occurrence graph based on building construction metaphor," in *Proceedings of the advances in digital libraries conference*, 1998, pp. 12–18.

[30] A. Bougouin, F. Boudin, and B. Daille, "TopicRank: Graph-based topic ranking for keyphrase extraction," in *Sixth international joint conference on natural language processing*, 2013, pp. 543–551.

[31] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.

[32] B. S. Khan and M. A. Niazi, "Network community detection: A review and visual survey," *Computing Research Repository*, vol. abs/1708.00977, 2017.

[33] J. Mothe, K. Mkhitaryan, and M. Haroutunian, "Community detection: Comparison of state of the art algorithms," in *Computer science and information technologies*, 2017, pp. 125–129.

[34] G. K. Orman, V. Labatut, and H. Cherifi, "On accuracy of community structure discovery algorithms," *Computing Research Repository*, vol. abs/1112.4134, 2011.

[35] S. A. G. Emmons Scott AND Kobourov, "Analysis of network clustering algorithms and cluster quality metrics at scale," *PLOS ONE*, vol. 11, no. 7, pp. 1–18, 2016.

[36] S. Petrov, D. Das, and R. T. McDonald, "A universal part-of-speech tagset," *Computing Research Repository*, vol. abs/1104.2086, 2011.

[37] H. H. Alrehamy and C. Walker, "SemCluster: Unsupervised automatic keyphraseextraction using affinity propagation," in *Advances in computational intelligence systems: Contributions presented at the 17th uk workshop on computational intelligence*, 1st ed., F. Chao, S. Schockaert, and Q. Zhang, Eds. Springer International Publishing AG, 2017.

[38] T. Opsahl, F. Agneessens, and J. Skvoretz, "Node centrality in weighted networks: Generalizing degree and shortest paths," *Social Networks*, vol. 32, no. 3, pp. 245–251, 2010.

[39] M. J. Bannister, D. Eppstein, M. T. Goodrich, and L. Trott, "Force-directed graph drawing using social gravity and scaling," *Computing Research Repository*, vol. abs/1209.0748, 2012.

[40] S. G. Kobourov, "Spring embedders and force directed graph drawing algorithms," *Computing Research Repository*, vol. abs/1201.3011, 2012.

[41] S. Wu, "Understanding the force." Medium, 2015.

[42] C. Wu, M. Marchese, J. Jiang, A. Ivanyukovich, and Y. Liang, "Machine learning-based keywords extraction for scientific literature," *Journal of Universal Computer Science*, vol. 13, no. 10, pp. 1471–1483, 2007.