PyramidTags: Context-, Time- and Word Order-Aware Tag Maps to Explore Large Document Collections

Johannes Knittel, Steffen Koch, and Thomas Ertl

Abstract—It is difficult to explore large text collections if no or little information is available on the contained documents. Hence, starting analytic tasks on such corpora is challenging for many stakeholders from various domains. As a remedy, recent visualization research suggests to use visual spatializations of representative text documents or tags to explore text collections. With PyramidTags, we introduce a novel approach for summarizing large text collections visually. In contrast to previous work, PyramidTags in particular aims at creating an improved representation that incorporates both temporal evolution and semantic relationship of visualized tags within the summarized document collection. As a result, it equips analysts with a visual starting point for interactive exploration to not only get an overview of the main terms and phrases of the corpus, but also to grasp important ideas and stories. Analysts can hover and select multiple tags to explore relationships and retrieve the most relevant documents. In this work, we apply PyramidTags to hundreds of thousands of web-crawled news reports. Our benchmarks suggest that PyramidTags creates time- and context-aware layouts, while preserving the inherent word order of important pairs.

Index Terms—Visual analytics, information retrieval, text analysis, layout

1 Introduction

Analysts from different fields have to deal with large document collections with the goal to get a general overview of the data, but also to find interesting aspects and stories, often with little a-priori knowledge about the content. Business analysts, for instance, have to constantly monitor news reports and trending topics on social media to react to specific developments and make informed decisions. Journalists investigating an unauthorized document leak usually need to process large amounts of textual data in a short amount of time, which is labour-intensive [1].

Interactive visualizations help to provide compact summaries of large data sets and can support analysts to study promising aspects in detail. *Tag clouds* (or *word clouds*) are popular choices to visualize the most prominent terms in large text collections. However, there is some discussion whether tag clouds are appropriate for analytical tasks [2], [3], [4]. Hu et al. [5] argue that traditional tag clouds often only convey basic concepts. They stress that analysts require longer, connected phrases to grasp more complete ideas. Sinclair and Cardew-Hall [6] found out that tag clouds are considered useful for browsing and information discovery, but less for seeking specific information.

Several methods have been introduced in recent years to improve the analytical capabilities of tag-based approaches, such as adding interaction [7], clustering tags semantically [8], or animating the temporal evolution [9]. It has been shown that context-aware tag cloud layouts can improve the understanding of the underlying documents [10], [11]. To quickly reveal clusters of terms in a matrix visualization, Chuang et al. [12] introduced a seriation technique [13] that also preserves the natural reading order. Recent work [14], [15] investigated how spatial text visualization in combination with semantic interaction facilitates exploring large data sets with thousands of tags. In all these cases, though, the approaches mainly focus on bringing forward a specific advancement such as context-aware layouts *or* visualizing syntactical structures *or* conveying temporal information. Additionally, even though many approaches have dealt with context-awareness in the past, Hearst et al. [10] claim that there is still a lack of automated tools that reliably produce semantically grouped word clouds.

We propose a novel technique, called *PyramidTags*, which combines several advantages of different approaches into one visualization without animations. Single- and multi-word tags from a time-stamped document collection, e.g. news reports, are extracted and placed onto a 2D spatialization (a *tag map*). Related tags are placed nearby, the position on the map indicates corresponding date ranges, and the word order is preserved to stimulate longer phrases. Interaction possibilities enable analysts to further explore concepts, reveal relationships within the collection and retrieve relevant documents. Figure 1 depicts an example which we generated using around 200,000 news articles from mid-January 2019. In this case, the analyst has already selected four tags of interest as indicated by the colored rectangles, and the remaining tags are shaded from black to nearly-transparent according to their semantic relation to the selection.

The main contribution of PyramidTags is three-fold: First, the initial visualization promotes the understanding of large collections through a context-, time- and word order-aware layout. Second, interaction mechanisms with visual cues and suggestions guide analysts to dive deeper into topics of interest and retrieve relevant documents. Third, our approach is highly scalable and applicable to several hundreds of thousands of news reports using different time spans.

• *J. Knittel, S. Koch, and T. Ertl are with the Institute of Visualization and Interactive Systems, University of Stuttgart, Germany. E-mail: firstname.lastname@vis.uni-stuttgart.de*

Manuscript received xxxx xx, xxxx; revised xxxx xx, xxxx.

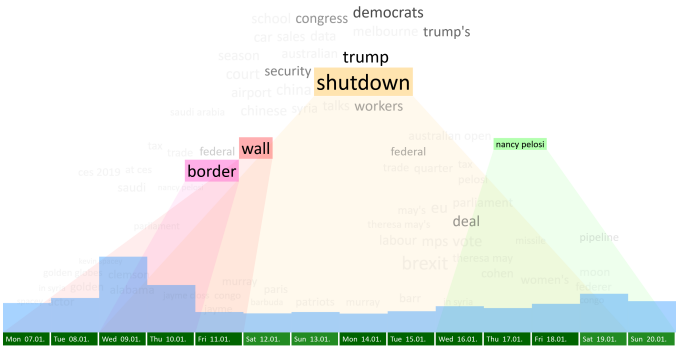


Fig. 1. PyramidTags visualization generated from more than 200k news articles mid-January 2019. The analyst has selected four tags of interest highlighted in color. The trapezoid beneath each tag indicates the prevalent date range of this tag in the data set, while the remaining tags are shaded according to their semantic connection with the tag selection. The bar chart at the bottom above the timeline visualizes the number of documents containing the selected tags per day.

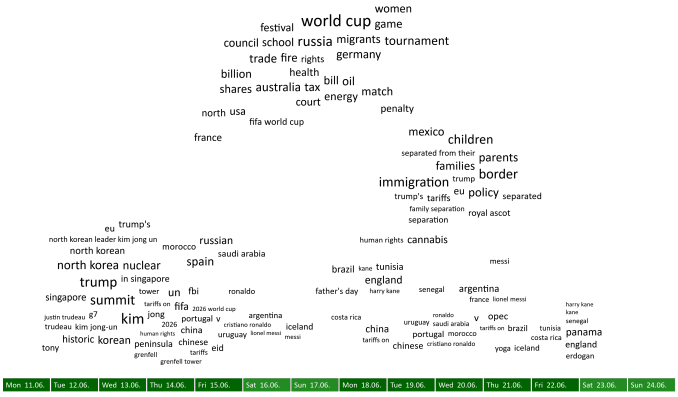


Fig. 2. PyramidTags visualization with 100 multi-word tags that was generated on 230k news articles from two weeks mid-June 2018.

2 Related Work

2.1 Tag Clouds for Analytical Tasks

The concept of tag clouds dates back to 1976, but beginning with the late 1990s it started to become popular on the web as a way to present frequent search terms or user-generated tags [4]. Initially, tag clouds were often regarded as a tool for designers and there was a strong focus on aesthetics and visual appearance [16], [17]. Bateman et al. [18] found out that font size, font weight and color have a strong influence on which tags users typically select.

Later on, visualization research investigated the use of tag clouds for analytical tasks. Schrammel et al. [19] conducted a task-driven study which showed that while semantically structured tag cloud layouts worked better than random layouts, a simple alphabetical list still performed best for finding specific tags. There was no difference regarding the search time for finding tags related to a topic or the ability to recall tags. However, they use a static WordNet-based method [20] to infer similarity, while our approach adapts to the data set and captures dynamically changing word relationships. Lohmann et al. [21] concluded that the best way to arrange tags depends on the task at hand, which was recently corroborated [22]. Research from Sinclair and Cardew-Hall [6] suggests that tag clouds have strengths for non-specific information discovery, and Wang et al. [11] found out that semantically clustered word clouds can improve the understanding of large document collections. In our approach we shade related words accordingly if analysts select one or multiple tags. Liu et al. [23] proposed a new technique for word cloud navigation which changes the word sizes dynamically while considering the mental map of users. Dörk et al. [24] introduced VisGets to explore news items with linked visualizations. Dynamic search queries can be defined using an alphabetically sorted word cloud, a geographic map, and a date/time slider. Hovering over a tag highlights related tags and visual elements. Heimerl et al. [7] presented a visual text analytics tool that heavily relied on word clouds with interactive features for filtering, co-occurrence highlighting and statistical insights. Word clouds typically only use single words (unigrams), however, descriptive tags to summarize documents for analytical tasks should include multi-word phrases [25], [26].

2.2 Context-Aware Layouts

Several efforts were made to generate more context-aware layouts. Ada et al. [27] applied multidimensional scaling to create distance aware tag clouds, although on a relatively small data set. Hassan-Montero et al. [28] clustered user-generated tags semantically based on their target resource. Cui et al. [29] presented context-preserving word clouds, again using multidimensional scaling. Terms occurring in the same sentence are considered to be related. Wu et al. [8] improved upon this to generate layouts more reliably. ReCloud [11] applies a force-directed graph layout to produce semantically clustered word clouds on restaurant reviews. Xu et al. [30] first constructed a similarity graph using word embeddings, transformed it then with multidimensional scaling and finally used force-directed methods to obtain dense layouts. TagSpheres [31] aims to visualize hierarchical relations by arranging tags in circular bands, while at the same time placing related tags nearby. Endert et al. [14] generated a 2D spatialization of the entire English Wikipedia with about four million documents. Several thousand important terms, phrases and snippets are extracted and placed in such a way that semantically similar tags are placed nearby. Analysts can zoom in until they finally retrieve individual Wikipedia articles. PyramidTags visualizes the temporal evolution in addition to semantic similarity without having the analyst switch through different time steps.

2.3 Time-Aware Layouts

To visually compare document collections based on the time or location, Collins et al. [32] proposed Parallel Tag Clouds in which each category (date, location, etc.) is represented by a column containing a list of extracted relevant terms. Analysts can hover over tags to highlight and follow its trajectory. The authors proved its scalability by applying it to 600,000 court documents. PyramidTags, though, places the tags in a way that semantically related tags are positioned nearby, while aiming to preserve temporal appearance. SparkClouds [33] enriches tag clouds with line charts under each tag to show its popularity over time. PyramidTags summarizes the temporal distribution of terms in a chart as

well, but also facilitates the recognition of tag clusters with similar occurrence patterns. Chi et al [9] applied rigid body dynamics to smoothly morph word clouds over time. Cui et al. [29] combined their context-preserving word clouds with a significance line chart and generated multiple clouds for a selection of time steps that were deemed most significant. Binucci et al. [34] proposed animated word clouds to show the temporal evolution of real-time streaming data while trying to preserve the mental map of the user. ConcentriCloud [35] merges word clouds from different documents and is optimized for the comparison of a few long documents such as books.

In contrast to previous work, the placement strategy of PyramidTags is inspired by the Triangular Model to express time ranges without animations or interactive sliders. The Triangular Model was originally introduced by Van de Weghe et al. [36] to visualize interval-based data, based on work from Kulpa et al. [37].

2.4 Linguistic Patterns and Document Embeddings

Apart from tag cloud approaches, tree-based techniques were proposed to let analysts inspect structural patterns in large document collections. The work of Burch et al. [38] combines similar tags that share a prefix to generate word cloud layouts that save space and quickly reveal different variants of terms. Wattenberg et al. [39] presented an interactive word tree which renders the structure of sentences as an hierarchical tree, enabling analysts to explore how sentences continue in different variants. It usually requires a keyword search first to find an interesting starting point. Phrase Net [40] generates node graphs from text to reveal syntactic or lexical relations based on a pattern query. SentenTree [5] extracts frequent sequential patterns from tweets and generates a node-link diagram trying to preserve the word order. These concepts do not take into account the semantic relationship between graphs.

Wise et al. [41] introduced the concept of transforming text to a spatial representation, enabling a more natural way of perceiving thematic patterns and relationships among documents. The Galaxies visualization projects high-dimensional representations of documents to points in a 2D scatter plot, and this inspired a variety of work [15], [42], [43]. As opposed to Wise et al. we do not visually represent text documents and their similarity, but common content.

2.5 Topic Modeling

Termite [12] helps analysts to assess topic models in a tabular layout. They propose a seriation method that favors the natural reading order to quickly reveal relevant patterns. In general, many text visualization approaches apply some sort of topic or event extraction [12], [15], [44], [45], [46], [47], [48], [49], [50], [51], [52]. In this case, it is more straightforward to visually express temporal evolution and relationships. However, this makes the strong assumption that relevant topics matching the expectation of analysts can be algorithmically found with sufficient precision. Many of these approaches cannot be scaled to large data sets because of high (pre)processing costs. With our approach we aim to support creative and unbiased interactive exploration of large document collections by avoiding to introduce predefined boundaries.

3 PyramidTags

Our approach is a novel technique that presents analysts a time-, context- and word order-aware overview of large time-stamped document collections to support interactive exploration, topic selection and document retrieval. In this paper we mainly focus on applying our approach to several hundreds of thousands of news articles that we collected over time.

PyramidTags extracts important single- and multi-word tags from documents within a specified date range, for example a week or a month, and lays out these tags considering several objectives:

Context-Awareness (O1): Related tags as indicated by the underlying data should be placed nearby

Word Order-Awareness (O2): If the underlying data implies a certain word order, then the placement of the affected tags should adhere to that order

Time-Awareness (O3): The placement should reveal at which date range the respective tags mainly appear in the underlying data

At the bottom of the visualization a timeline is shown (Figures 1 – 8). The vertical position of a tag indicates its temporal extent (duration), and the horizontal position the mid-point of the time range in which this tag mainly appears in the data. Tags placed at the bottom, right above the timeline, mainly occur on the specific day that is shown underneath. The higher a tag is placed, the longer its corresponding time range, i.e. tags at the top are consistently mentioned throughout the entire processed data set. In contrast to simple time-to-space mappings, this placement strategy also visualizes data associated with intervals of time (time spans). It should be noted that the layout does not necessarily imply a topic hierarchy.

Each objective aims to improve the layout such that analysts gain a more thorough overview of the data set they want to explore. Placing related tags nearby (**O1**) helps analysts to better understand the content [10], [11]. If tags regularly appear close to each other then we consider them as related (Section 5.6). Furthermore, we want to preserve the word order of the most important pairs in our map (**O2**) to visualize more descriptive and complex concepts of the data with longer phrases [5], [12], [25], particularly if analysts hover over tags. Finally, the triangular layout (**O3**) offers two major benefits for analysts. On the one hand, it instantly provides more details of the document collection by revealing in which time range tags are mainly mentioned. This typically corresponds to when events happened (in case of news reports, social media, diaries, or written protocols, for instance). On the other hand, we argue that the triangular layout stimulates clusters of topics, because surrounding tags exhibit a similar time range in which they are mainly mentioned in the document collection. For instance, two tags which both only appear at one specific day are less likely to be related if the dates are far apart. Our quantitative evaluation in Section 7 corroborates this: even if we completely ignore the context- (**O1**) and word order-awareness (**O2**) objectives the triangular layout still results in a more context-aware placement of tags compared to a conventional, random layout.

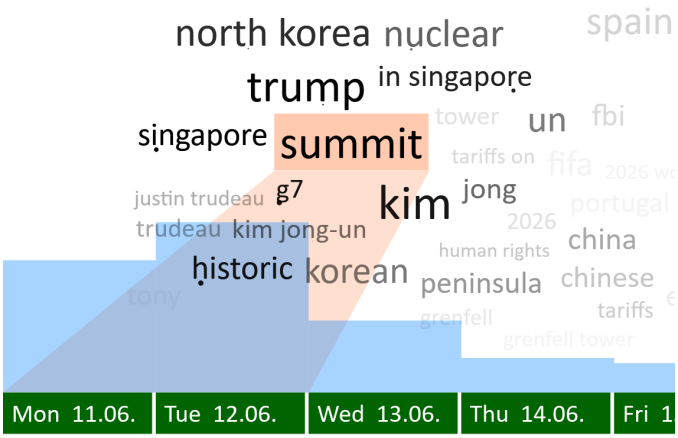


Fig. 3. User is hovering over the tag `summit`. The trapezoid in light red indicates the peak time range of the term, the blue bar chart at the bottom depicts the number of documents that contain this tag per day. The other tags are shaded according to their relatedness. Little dots appearing under some tags reveal the word order that was determined in regard to the hovered tag.

Hence, PyramidTags enables not only an instant visual overview of prominent words and phrases in a big corpus similar to multi-word tag clouds, it also conveys semantic relationships between tags (**O1, O2**), linguistic structures (**O2**), and temporal patterns (**O3**), revealing internal structures of the data set in a novel, comprehensive way. It is a closely coupled process of data analytics, visualization and interaction, extracting statistical and temporal relationships in the data first, which are then visually presented in an interactive environment. Optimizing for several objectives that often conflict with each other is a challenging task. The optimal position of tags according to their temporal pattern (**O3**) may lead to violations of **O1** and **O2**, for instance. Thus, we need to establish a viable compromise between all three objectives, which we achieve by optimizing an energy function.

Our system serves as a starting point to interactively explore large document collections. It is provided with several interaction mechanisms to reveal relationships in detail, select topics of interest, and retrieve relevant documents.

3.1 Overview

Figure 2 shows the generated layout after opening the application. In this case, 230,000 news articles spanning two weeks from mid-June 2018 were processed and 100 multi-word tags were extracted. Due to the positioning, analysts can immediately see that the Soccer World Cup 2018 was prominently featured in the news the entire time, whereas the Trump-Kim summit in Singapore seems to be mainly mentioned in the first days. The second week was dominated by topics around Trump's border and immigration policy. Analysts can toggle whether spaces are replaced with underscores to better discern multi-word tags from accidental alignments (Figure 8).

3.2 Hovering Tags

When analysts hover over a tag, the remaining tags are shaded depending on how related they are to that tag, from black (very related) to nearly-white (data does not suggest relation). Figure 3 shows an example of this behavior. While related tags should be placed nearby, not every tag i that is placed next to tag j is actually related to it, and in some cases there could even be a strong connection to a different tag h that is placed much further away. Hovering over terms enables analysts to debunk false friends and sense how strong the connection really is. For instance, `tower` appears right next to `summit` in Figure 3, but the barely visible shading reveals that these terms do not frequently occur close to each other in the data set.

In addition, a little dot under each term appears if the data implies an ordering regarding the currently hovered tag. We use circles to encode both the direction and the manifestation of the word order in the data set. The horizontal position of the dot depends on the percentage of occurrences in the collection with the indicated order. If the dot is placed to the left, that tag mainly appeared before the tag which is currently hovered. Analogously, if it is on the right, then the tag mainly appeared after the tag of interest in the document collection. The size of the dot indicates how sure we are that there is a suggested word order, i.e., how often the respective pair occurs in that order. These hints tell users whether the order of the tags as it is shown on screen is relevant and consistent with the statistics from the underlying data. For instance, the term `immigration` probably appears often before `policy` and not the other way round if both are closely mentioned (Figure 2).

Each tag is associated to a specific time range during which it was strongly mentioned in the corpus. When hovered, this range is illustrated with a trapezoid that spans from the tag to the start and end date on the bottom at the timeline. Furthermore, a more detailed view regarding the temporal evolution of the hovered tag is presented with a bar chart popping up right above the timeline. Each bar sitting on its date is mapped to the corresponding number of documents the tag of interest appeared in on that date.

Figure 3 shows the updated visualization when hovering over the term `summit`. The shading of the surrounding terms indicate several strongly related tags such as `trump`, `kim` and `singapore`. The associated time range reveals that `summit` was most frequently mentioned on June 11-12, the bar chart peaks at June 12. The dots under some tags reveal the assumed word order. Therefore, one could read *'historic ... summit ... in singapore'*, or *'g7 ... summit'*, for instance. The analyst could reason that there was a summit concerning Kim and Trump on that Tuesday, which is indeed true. On June 12, 2018 the highly anticipated meeting between Trump and Kim Jong Un took place in Singapore shortly after the G7 summit.

3.3 Multiple Tag Selection and Document Retrieval

Analysts can select and deselect several tags by clicking on them. Then, the selected tags with their corresponding triangles remain highlighted. The opacity of the other tags is updated to reflect the lowest relatedness to any of the selected tags, which is an upper bound of the actual relatedness to the selection. Analogously, the bar chart on the bottom is updated with the number of documents containing all selected terms for each day. Hence, analysts are supported

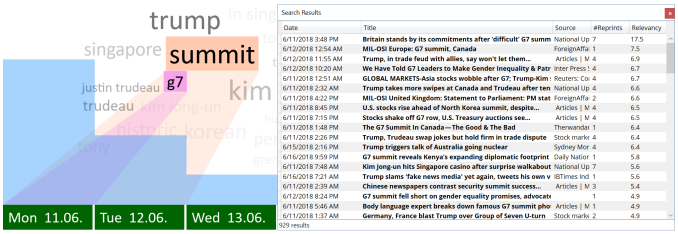


Fig. 4. After selection of tags `summit` and `g7`, a list of search results is presented with the most relevant documents for the selection.

in picking a topic they would like to explore more deeply. Suitable selections are suggested by highlighting relevant tags according to the document collection. In many cases, analysts have to select only few tags to greatly narrow down the search results even in big data sets comprised of millions of documents.

If analysts select one or several tags, a separate window opens presenting a sorted list of the most relevant news articles as specified by the selection. The list contains the date, title and source of the documents as well as the number of reprints (i.e. how many different news outlets published the same story) and the relevancy as determined by our system. Only documents that contain all the selected tags are shown in the list. The number of occurrences but also the distances between the tags in the document influence the relevance score. Documents are considered to be more relevant if the selected tags occur closer to each other rather than being scattered throughout the document, because this indicates that they are semantically connected in the document. The total number of documents matching the selection is shown at the bottom of the window.

An example is shown in Figure 4. In the depicted case, the analyst has decided to learn more about the G7 summit in the data collection by clicking on `G7+summit`. As expected, news related to the G7 summit show up, even though the Singapore summit happening roughly at the same time was much more prominently featured in the press.

Full articles can be retrieved by double clicking on the respective list item. The document explorer shows the requested article in a new tab, revealing the full text and the link to the source page among other meta data, as depicted in Figure 5.

4 Preprocessing and Data Analysis

PyramidTags is built on a pipeline with three main steps. First, the data at hand is algorithmically analyzed and relationships are extracted. Then, based on the collected statistics the actual visualization is created as explained in Section 5. Finally, our visual analytics approach displays the generated visualization and provides several interaction possibilities as described in the previous section.

Technically, our method works with any collection of timestamped documents. However, we aim to visualize content relating to events that span different time ranges, which is typical for social media posts or news articles. We collect around 100,000 distinct English news articles each week from various online news sources. We apply PyramidTags to

Fig. 5. Document browser showing documents in tabs with textual content and meta-data, for instance the URL of the article.

several data sets extracted from this vast collection covering a specific date range. This data is particularly challenging due to the size but also due to the fact that it contains noisy, real-world articles crawled from the web.

4.1 Cleaning and Reprint Detection

Web-crawled articles often contain additional content that is not part of the actual article, e.g., related news articles or advertisements. We strip paragraphs from the document if several other articles from this news outlet also contain the same paragraph, assuming that only text that is specific to this document is considered as useful content. If a cleaned article largely contains the same content as a previously processed article, but it is from a different source, then it is considered to be a reprint. If it is from the same source we discard it as duplicate. Reprints are still considered as being part of the corpus for subsequent data analytics and the visualization, because the decision of news sites which agency reports they distribute is an indication for the importance of the content. The reprint detection is mainly used to save computing resources and to improve the usability of the news retrieval scenario.

4.2 Multi-Word Tag Extraction

We first extract the K most important words from the subset of documents within the desired time frame. We apply the widely used tf-idf weighting scheme [53]. For the term frequency we look at the number of occurrences within the selected subset of the data, while the inverse document frequency is based on a bigger data set with several million articles spanning roughly half a year. A list of stop words is used to filter out very common words. Before the text is split into tokens it is first converted to lowercase.

The extraction yields single word tags $t_1, t_2, ..., t_K$. The next step is to explore whether these words often appear as part of a multi-word phrase. If `cup` is an important tag and the documents often contain `world cup`, then we would like to add `world cup` to the set of tags, for instance.

Let $\mathrm{count}(x)$ be the number of occurences of a word or phrase x in the underlying data set. If a phrase p_i contains a tag t_j as a word and

$$\mathrm{count}(p_i) \geq \frac{\mathrm{count}(t_j)}{\mu}$$

then we add p_i to the set of extension candidates C^j of the tag j. This means we regard every phrase as possible multi-word tag, if it contains a tag j and occurs at least as often as a some specified fraction of tag j. While the threshold can be set flexibly, testing several values showed that $\mu = 3$ is a viable trade-off between the number of different variants and their overall importance.

The resulting candidate set contains many redundant items, because for each phrase every possible sub-phrase is also part of the set by design. To reduce the number of redundant tags, each extension candidate c_m^j associated with tag j is added to the final set of tags iff there is no other candidate c_n^j in the same set C^j which *properly contains* c_m^j. *Properly contains* means that there is a subset of matching words, e.g., `using` does not properly contain `us`, but `like us` does. This means we only add the longest variants of these candidates.

The original single-word tag t_j is removed from the final set of tags iff

$$\text{count}(t_j) - \sum_{m=0}^{M} \text{count}(c_m^j) < \frac{\text{count}(t_j)}{\mu}$$

where each c_m^j was added to the final set of tags.

To implement this we need to count the occurrences of all phrase variants that contain one of the original tags. A naive approach would lead to an explosion of combinations. Therefore, we have to use word and word pair count statistics of the data set at hand in order to make this computationally feasible. For each match of an original single-word tag t_j in the data set, we look to the left and right and count every phrase combination, but only as long as the number of overall occurrences of the current word or current bigram is at least as high as the threshold $\frac{\text{count}(t_j)}{\mu}$, up to four words in each direction. Word and bigram counts are upper bounds for the final phrase count, so we only keep track of promising phrases that have a high chance of meeting the final threshold. This makes the approach computationally tractable.

Extracting multi-word tags is beneficial in several ways. First, longer phrases (that may include stop-words) offer richer context information that help to understand the content. Second, recognizing obvious connections between words lowers the pressure on the subsequent optimization routine. As we already explained earlier, our objectives may contradict each other, and this preprocessing step helps to speed up the process. Third, it allows us to compare our approach with traditional multi-word tag cloud layouts. We designed our extraction algorithm such that it can easily process hundreds of thousands of documents, but PyramidTags is agnostic as to which keyword extraction method is used. RAKE [54] is one possible alternative which is commonly used. However, it assumes that keywords are 'delimited' (e.g., by stop-words), it may extract many variants of related keyphrases, and it is patented. The more complex approach by Frantzi et al. [55] creates lists of bigrams, trigrams, and so on during the process, and this may take a while on collections with hundreds of thousands of documents.

4.3 Tag Relationship Analysis

Our goal is to visualize large text corpora while preserving important relationships and structures within documents. Tags that often appear together in documents should also be closer to each other in the visualization, tags that often appear in a certain order should also be ordered that way in the resulting visualization. Furthermore, the temporal evolution should be visible, for example, if several tags mainly occur in a specific time range.

We need to process the data set that we want to visualize again to analyze the relationship between tags. For each document, we first search for the tags that were extracted in the previous step and note their position in the text. For each search result, we increment the count for the specific tag and date which we need later on. Then, we look at each pair (t_i^p, t_j^q) in that document where p and q denote the respective positions in the text of tag t_i and t_j, and t_i has a lower lexicographic rank than t_j to avoid counting the same pair twice.

We disregard this match if there is another match (t_i^p, t_j^s) with $|s - p| < |q - p|$ or (t_i^s, t_j^q) with $|q - s| < |q - p|$. This means, if we replace one tag of this pair with the same tag but at a different location, this must not lead to a closer pairing. For example, the pair *'[John] Doe was seen outside. [Doe] wore a black jacket'* is ignored, since there is a closer pairing with one of the tags involved: *'[John] [Doe] was seen outside...'*.

If all these conditions are fulfilled, we calculate the distance weight d^w. We define the distance weight as the inverse distance between the two matches plus one:

$$d^w(t_i^p, t_j^q) = \begin{cases} \frac{1}{1+q-p-\text{numWords}(t_i)} & \text{if } q > p \\ \frac{1}{1+p-q-\text{numWords}(t_j)} & \text{otherwise} \end{cases}$$

The distance weight d^w is added to the order-aware tag distance weight w_{ij} if $q > p$ and to w_{ji} if $p > q$. We also add the distance weight to the distance weight count for the specific pairing and date. The order-aware tag distance weight w_{xy} is then the sum of the inverse distances of valid tag pair matches (t_x, t_y) where t_x appeared before t_y.

After processing all documents, $w_{ij} + w_{ji}$ indicates how often the tags t_i and t_j appeared nearby in the data. If one summand is high compared to the other one, we can assume that the underlying documents suggest a specific word order, e.g. `john doe` is more likely than `doe john`.

5 Visualization Generation

To generate a PyramidTags visualization, we define several training objectives that should be fulfilled. The location on the map should correspond with the associated time range (*location*), tags should not overlap (*collision, repelling*), related tags should be placed close to each other (*proximity*), and the natural reading order should be respected (*wordOrder*). Unfortunately, these objectives often contradict each other. For instance, if all tags are placed at the same position, we would achieve the best result concerning the context-awareness, i.e., related tags are indeed close together. However, this drastically violates the collision objective that tags should not overlap each other. Hence, we need a viable trade-off between these objectives.

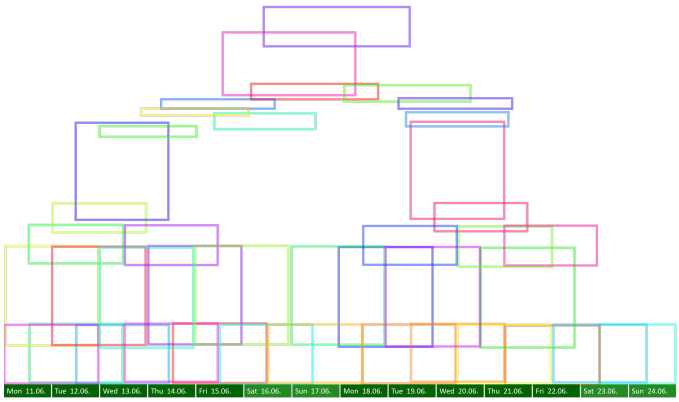


Fig. 6. Non-empty location boxes of the PyramidTags visualization from Figure 2. The energy function pushes each tag to its assigned location box indicating the prevalent date range. These boxes have dynamic widths and heights to optimize space usage while trying to preserve the triangle shape. Here, colors have been assigned randomly to location boxes to make them discernible.

We define the following energy function f_θ that we want to minimize representing all training objectives with parameters θ as input:

$$f_\theta = \lambda_1 \text{location} + \lambda_2 \text{collision} + \lambda_3 \text{proximity} + \lambda_4 \text{repelling} + \lambda_5 \text{wordOrder} \quad (1)$$

The individual components are weighted with the meta-parameters λ_x, allowing to balance the importance between the objectives. For instance, if we set $\lambda_3 = \lambda_4 = \lambda_5 = 0$ and only have one location box spanning the entire image, then the optimization generates traditionally dense, multi-word tag clouds.

In the following parts we first describe the general layout of our approach. Then, we present in detail the components our energy function is comprised of.

5.1 Layout and Map Locations

We set the font size of each tag proportional to the square root of the respective tf-idf weight. In the ideal case that all tags have equal aspect ratios, this means that the area of the tag x is proportional to this count which we denote with weightedCount(t_x).

We want to track the evolution of certain tags and their relationship over time. Each tag is mapped to a specific *location box* that indicates when and for how long this tag mainly appeared in the underlying data set. On the bottom of the visualization, a timeline shows the dates within the processed time range of n_d days. There are n_d location boxes on the first row right above the timeline, representing each day. If a tag is mapped to one of these boxes on the first row, it mainly occurs on that day. The second row (counted from the bottom) is comprised of $n_d - 1$ location boxes that represent durations of two days. This can be continued until the top row is reached with one location box in which the tags associated with it span the whole n_d days. The resulting structure resembles a pyramid-like shape in 2D, hence the name PyramidTags. An example of this structure is shown in Figure 6, in which each non-empty location box (i.e., at least one tag is mapped to the box) is drawn as a rectangle.

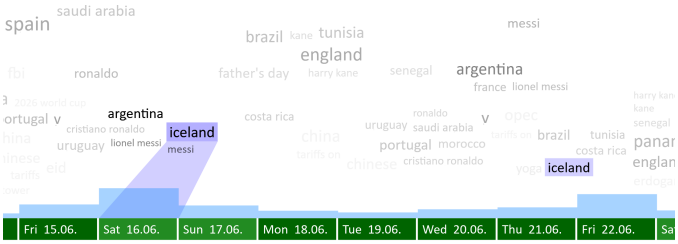


Fig. 7. Analysts hovers over the tag `iceland` to the left. Another `iceland` tag is highlighted on the right. The bar chart above the timeline reveals that there are two distinct peaks in the data set, therefore, the tag was split such that each peak can be assigned to a tag.

The rows do not necessarily have equal heights, this is determined based on the space requirements of the most occupied box of that row. The width of the boxes increases with each row from bottom to top, because there are less boxes the tags are distributed to. However, the width does not grow linearly with the number of days the boxes represent. As a result, the box in the top row does not occupy the entire horizontal space that is available to preserve the pyramid metaphor. The analyst should still be able to guess the probable time range of each tag, even with the static visualization, by spanning a right-angled triangle from the mid-point of the tag to the timeline at the bottom. The dynamic sizing is clearly visible in Figure 6. Each rectangle in the same row also has the same height, however, the height differs among rows.

Figure 6 shows that rows and columns can overlap each other, particularly with increasing number of total days. This is necessary to fit regular tags within the bottom row for large n_d. The flexible, dynamic approach for determining the height and width of location boxes optimizes the use of white space and preserves the pyramid-like structure at the same time.

5.2 Tag Splitting and Mapping

We have to determine when and for how long tags mainly appear in the document collection to assign tags to location boxes. However, there can be multiple distinct time spans, e.g. some tag can be mainly mentioned in the beginning of the month as well as in the end of the month, but rarely in-between. Furthermore, two tags can be strongly connected on a particular day, but one of it is mentioned similarly often throughout the week as well. These are cases we want to represent with tags, but in which the same tags should be placed at distinct locations to reflect the semantics adequately. For this purpose we introduce the concept of tag splitting. Figure 7 shows an example of tag splitting in the resulting visualization. The term `iceland` mainly appears on two days which are a week apart. Splitting the tag allows to assign each tag to one of the respective peaks. This avoids a centered, misleading placement between those two peaks.

To determine in which time ranges each tag is strongly mentioned, we first count the number of occurrences per day for each tag in the relationship analysis phase. Then, we extract possible time spans by finding contiguous regions in the corresponding occurrences histogram where each count is over a threshold of 60% of the maximum value. The resulting one or more time spans are added to the candidate

set of spans for this tag. Each time span is defined by a *start day* and *duration* which can be mapped to the corresponding location box.

We also want to collect time spans in which one tag often occurs nearby another tag. For instance, `world cup` may appear consistently throughout the week, but `england` and `world cup` often appear closely together at one particular day. Then we add this day to the candidate set of time spans for the tag `world cup`. To realize this, we compute for each tag i the sum of the order-aware distance weights per day where this tag is involved, i.e.

$$\sum_j w_{ij}^d + w_{ji}^d$$

where d stands for the respective day. Please note, w_{ij} is the weight in cases where tag i appeared *before* tag j, and w_{ji} where tag i appeared *after* tag j. Again, we extract contiguous regions in the resulting histogram that indicate time spans in which this tag strongly correlates with at least one other tag, and add them to the candidate set of spans for this tag.

This can result in many similar time ranges, e.g., two week-long spans that are just shifted by one day. For each span candidate, the respective accumulated count of the underlying histogram (i.e., how many occurrences in the data this time span covers) indicates the significance of the span for its tag. We only select those candidates that are similarly significant as the strongest candidate and sufficiently distinct to the remaining time ranges. If there is only one time span left, the tag is assigned to the corresponding location box. Otherwise, we split the tag so that each time span can be assigned to one split tag. The document counts of the resulting split tags (for the tag size) are distributed according to the relative significance of the corresponding time range (i.e., how many occurrences it covers). The related order-aware distance weights w_{xy} stay the same.

5.3 Particle Swarm Optimization

We employ Particle Swarm Optimization (PSO) [56], [57] to find a reasonable local minimum of our energy function (1). PSO does not use the gradient, hence, we are less restricted concerning the design of our energy function including the use of non-differentiable features.

Let A be the total area of all tags combined. To set the working image plane dimensions (p_w, p_h), we define that the area of the image plane should be $4A$ and enforce an aspect ratio of 16:9. The exact size is not important, but the smaller the area, the more dense the resulting layout, with less room for the context- and word order-objectives.

The parameters of the energy function are the positions (p_i^x, p_i^y) on the image plane for each tag i. Thus, the parameter dimension is $2N$ if we want to place N tags on our plane. The goal is to apply PSO to find parameters that yield a reasonable minimum of the energy function. Technically, the parameters are the positions encoded as fractions relative to the plane width and height, respectively. Before calculating the energy function, however, we define rectangles that represent the tags and compute their position on the plane.

5.4 Location Force

Let (c_i^x, c_i^y) be the center of the rectangle representing tag i, (l_i^x, l_i^y) the center of the associated location box and (α_x, α_y) the horizontal and vertical distance of the tag's center to the box boundary if the tag is currently placed outside the box (otherwise 0). The following term pushes tags to be placed inside their associated location box:

$$d_i^l = \sqrt{\left(|l_i^x - c_i^x| + \alpha_x(1+\alpha_x)\right)^2 + \left(|l_i^y - c_i^y| + \alpha_y(1+\alpha_y)\right)^2}$$

$$\text{location} = \frac{1}{\sqrt{A}} \sum_i^N d_i^l$$

If the tag is placed inside the boundary of its location box, the regular distance between the center of the tag and the center of the location box is penalized, pushing tags slightly towards the center of the box. If it is outside, however, we strongly increase the weight of the penalty with an additional squared term based on the distance to the boundary of the box.

The square root of the total area of all tags combined ($\sqrt{A}$) is used for normalization. If tag splitting is disabled and every tag is assigned to the same 'location box' spanning the entire image plane, then this method generates classic tag cloud layouts in which tags are densely placed in the center of the image.

5.5 Collision avoidance

Let I, J be the area of tag i and j, respectively. We compare the rectangles representing tags with each other and compute the intersecting area $|I \cap J|$. The total intersecting area makes up the collision avoidance component, again normalized by $\sqrt{A}$:

$$\text{collision} = \frac{1}{\sqrt{A}} \sum_i^N \sum_{j=i+1}^N |I \cap J|$$

5.6 Proximity

Tags that are related to each other should also be close to each other, i.e. we want to minimize the distance between every two tags i and j weighted by their relatedness r_{ij}. In section 4.3 we explain how we calculate the order-aware distance weights w_{ij} using the inverse distance between pairs of tags in the source data. We apply these weights to calculate the relatedness:

$$r_{ij} = \frac{1}{Z}(w_{ij} + w_{ji}) \max(a_i, a_j) \Phi_{ij}$$

$$a_x = \log \text{IDF}(t_x) \quad Z = \frac{1}{N} \sum_i^N \max_j (w_{ij} + w_{ji})$$

We normalize the order-aware distance weights with Z to retrieve relative values between 0 and 1. Phrases that generally appear often in documents are more likely to occur nearby just by chance. Hence, we further apply a correction factor a_x which is the logarithm of the inverse document frequency of the less frequent term. This restricts the influence of high-frequency tags, but also prevents a

strong emphasis of low-frequency outliers, which is similar to the idea of tf-idf, where the influence of terms occurring in many documents is diminished. As a result, the relatedness between two tags is high if they appear unusually often close to each other in the documents.

The previously described tag splitting may result in multiple occurrences of the same two tags (content-wise) at different time ranges. The connection between each pair as expressed by the relatedness should also depend on the overlap of the time spans they have been associated to. For instance, if *australian open* and *andy murray* are assigned to a location box on the left and there are two additional tags *australian open* and *djokovic* which are assigned to the right, then the first *australian open* should mainly be related to *andy murray* and not to *djokovic* on the right. Therefore, we weight the order-aware distance-weights of a pairing (calculated on the whole time range) with Φ_{ij} that represents the share of the *global* relatedness for this particular pair based on the overlap of their associated time spans.

Let l_i^c be the column and l_i^r the row of location box assigned to tag i. The proximity component of our energy function penalizes large distances between tags that have a high relatedness:

$$\hat{d}_{ij}^o = \frac{d_{ij}^o}{\sqrt{A}} \qquad \phi_{ij} = \frac{1}{1 + (|l_i^c - l_j^c| + |l_i^r - l_j^r|)\frac{5}{n_d}}$$

$$\text{proximity} = \sum_i^N \sum_{j=i+1}^N \hat{d}_{ij}^o (1 + \hat{d}_{ij}^o) r_{ij} \phi_{ij}^2$$

Rather than calculating the distance between the center coordinates of the rectangles, we instead use the outer distance d^o that is defined as the closest distance between two points on the border of each rectangle. This ensures that we do not penalize horizontal stacking, i.e. tags placed next to each other horizontally have the same outer distance of zero like tags that are stacked on top of each other.

If two tags are assigned to different location boxes that are far away, it is nearly impossible to place them nearby. The proximity force is reduced in these cases by multiplying with the square of ϕ_{ij} which is the inverse difference between row and column indexes of the respective location boxes. The term $5/n_d$ is just for normalization to retrieve values independent from the total number of days.

5.7 Repelling Force

On the one hand, we would like to have a compact visualization. On the other hand, we would also like to slightly separate tags if the underlying data suggests that these tags are not related to each other. Whereas the previous proximity component rewards close distances of related tags, the following repelling force encourages tags to be placed slightly apart. In combination with the proximity term, this should lead to better visual clusters of related concepts. Let d_{ij} be the distance between the center points of the tags t_i and t_j. We then define:

$$\text{repelling} = \sum_i^N \sum_{j=i+1}^N \frac{1}{1 + (d_{ij})^{1.5}\sqrt{A}}$$

The intuition behind the exponent $1.5 > 1$ is to let this force quickly diminish with increasing distance.

5.8 Word Order

In some cases, the data implies a certain ordering of tags, e.g., if *tree* and *christmas* appear next to each other, *tree* comes after *christmas*. We want to preserve these word order characteristics in the visualization as best as possible to improve the readability and support sensemaking tasks. As previously explained, if the order-aware distance weight w_{xy} is much greater than w_{yx}, then tag x often appears before tag y when they occur nearby in the documents.

We define the word order o_{ij} which ranges from zero (tag i is right to tag j) to one (tag i is left to tag j), and the strength of the ordering γ_{ij} to express how certain we are that the data implies an ordering between tag i and j:

$$o_{ij} = \frac{w_{ij}}{w_{ij} + w_{ji}}$$

$$\gamma_{ij} = (0.5 - \min(o_{ij}, 1 - o_{ij})) \left(\frac{10}{z} \cdot |w_{ij} - w_{ji}|\right)^{0.3}$$

The first part of γ is zero if there is no implied word order, i.e., tag i appears equally often before and after tag j. It reaches its maximum at 0.5 if all occurrences are in the same order. The second part expresses how strong the evidence is and uses the absolute difference between the order-aware distance weights. The intuition behind having an exponent < 1 is that these differences do not follow a uniform distribution. A linear relationship would underestimate the evidence except for the pair with the highest difference. The normalization constant z is the maximum number of tag pair occurrences and, therefore, an upper bound for w. In combination, γ is high when tag i and j often appear in close proximity with one specific order for the most part. The normalization constant 10 and the exponent 0.3 were empirically determined such that the resulting forces are reasonable across different data sets, but they can be changed to shift the emphasis. For instance, a higher exponent increases the required evidence from the data, resulting in less tag pairs that are considered to have a significant word order.

We penalize the horizontal distance if the tag is on the wrong side according to the word order, and weight this by the strength of the ordering:

$$x_{ij} = \begin{cases} \max\left(-\text{width}(t_i), p_i^x - p_j^x\right) & \text{if } o_{ij} >= 0.5 \\ \max\left(-\text{width}(t_j), p_j^x - p_i^x\right) & \text{otherwise} \end{cases}$$

$$\text{wordOrder} = \frac{1}{\sqrt{A}} \sum_i^N \sum_{j=i+1}^N x_{ij} \gamma_{ij} r_{ij} \phi_{ij}$$

The lowest and best value for x_{ij} is reached if the tag that should appear first is completely placed before the other tag. The worst value is only limited by the image plane boundaries. We cap values at the optimal value $-\text{width}_{i/j}$, because we do not want to encourage tags to be pushed too far away again. Similar to the proximity component, we take the relatedness and the location box distances of the tags (if applicable) into account as well.

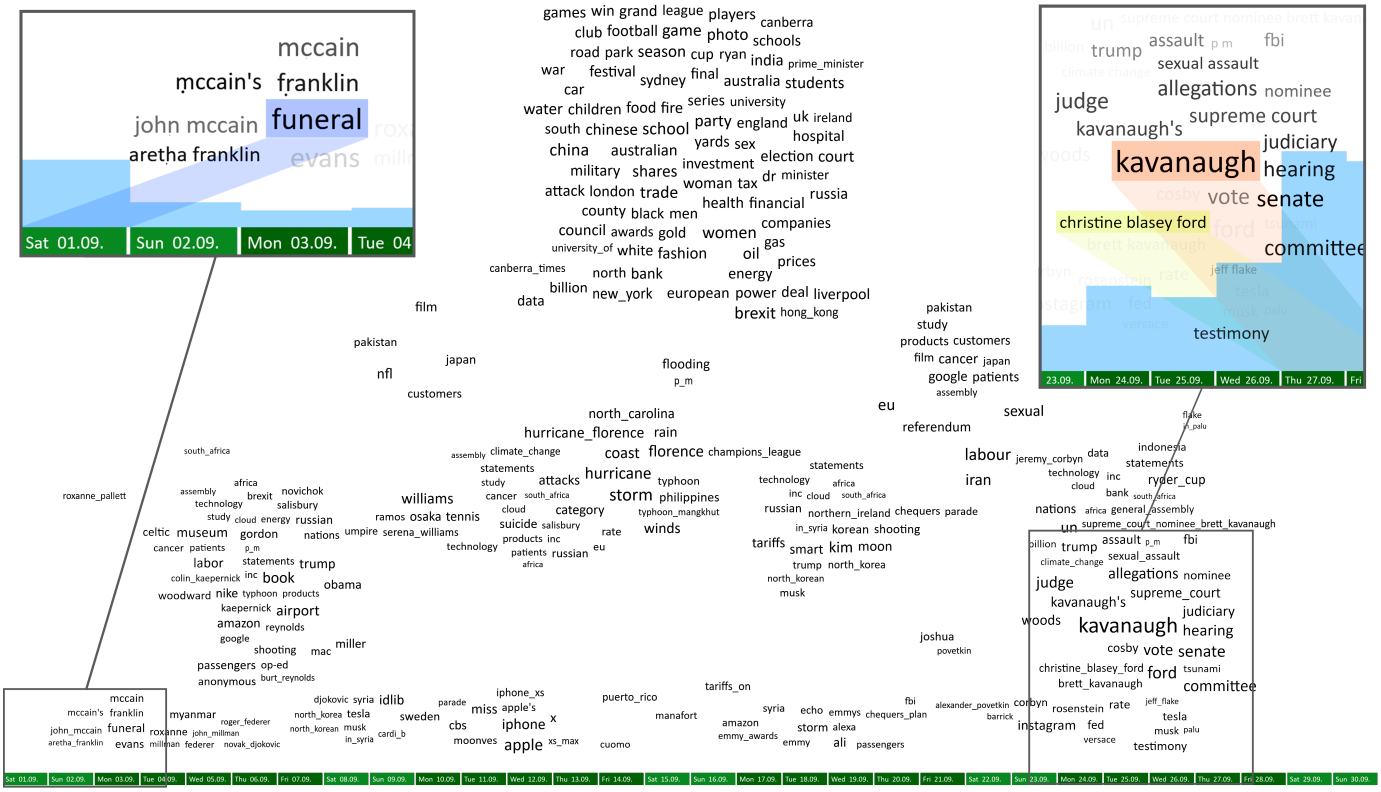


Fig. 8. PyramidTags with 250 tags, generated on roughly half a million documents from September 2018. Mid-month, news about Hurricane Florence dominated, while the hearing of Brett Kavanaugh was mainly covered at the end of September. Here, spaces in multi-word tags have been replaced with underscores. Top left: Cutout from the lower-left part with tag `funeral` highlighted. Top right: Cutout from the lower-right part after analyst selected `christine blasey ford` and `kavanaugh`.

6 Use Case Scenario

Figure 8 presents the resulting visualization of about half a million English news articles from around the world during the whole month of September 2018. From this static visualization (i.e., before any interaction is performed) the analyst can already form several hypotheses. She reckons that football-related news dominated the whole month, because tags like *football*, *games* and *cup* appear at the top. Further down, a cluster of tags appear that seem to be related to a hurricane named *florence*. From the position of the tags on the map she hypothesizes that North Carolina was hit by a storm in the middle of the month. She wants to investigate this further and hovers over *hurricane florence*. The shading of co-occurring tags such as *storm* and *north carolina* confirms her hypothesis. She then clicks on the tags *hurricane florence* and *north carolina* to scan through all related news stories that contain these tags. From the bar chart at the bottom she learns that the stories peaked around September 14. She right-clicks to clear her selection. At the bottom to the left she recognizes *aretha franklin* and *john mccain* next to *funeral* and remembers that both died around the same time. She wonders whether the tag *roxanne* is also related to the funerals, but upon hovering over it she realizes that this is a separate story that just happened around the same time. Further to the right, around September 11, *cbs* and *moonves* catch her interest. She selects both tags and learns that CBS head Les Moonves has apparently stepped down because of misconduct allegations.

7 Evaluation

The difficulty to evaluate tag cloud layouts and similar approaches dealing with text spatializations partly stems from the lack of annotated real-world data sets and the labour-intensive work to generate one. In addition, task-driven studies often fail to cover non-specific information seeking needs, because it is hard to artificially force and stage such needs.

To tackle these challenges, we developed a method to quantitatively evaluate our approach by utilizing meta-data that we could collect for a subset of our data set. We used about 60,000 articles, annotated with a list of associated *topics*, from June 2018 to January 2019 from the British newspaper `The Guardian`. Typically, this list contains both broad categories such as 'US News' and also more specific ones, e.g. 'Immigration Policy'. We use these annotations to test the context-awareness of our approach.

We evaluate three configurations: a time span of two weeks with 100 tags, two weeks with 200 tags, and one month with 200 tags. For each configuration we ran the pre-processing pipeline on four different date ranges, resulting in a total of 12 different date ranges. Next, for each configuration and date range, we trained several different visualization types by successively activating components of our energy function (1) (setting respective $\lambda_x \neq 0$). The first column in Table 1 lists the types. **Simple** is the regular multi-word tag cloud (only collision component activated, one location box). **PROX** or **WO** means that the `proximity` or `wordOrder` component is enabled, respectively. **Pyramid**

	Density	Context	Context4	Context7	WO5	WO10	WO50	DateCos	DaysOff
Simple	7.6	0.17	0.10	0.03	0.50	0.46	0.52	0.55	0.0
WO + REP	7.4	0.19	0.14	0.05	1.00	0.99	0.94	0.55	0.0
PROX + REP	7.8	0.29	0.28	0.15	0.45	0.43	0.51	0.57	0.0
PROX + WO + REP	7.7	0.27	0.25	0.13	1.00	0.94	0.74	0.57	0.0
Pyramid + REP	3.3	0.23	0.20	0.10	0.63	0.61	0.56	0.67	0.0
Pyramid + WO + REP	3.9	0.23	0.20	0.10	0.95	0.93	0.73	0.66	0.0
Pyramid + PROX + REP	4.5	0.31	0.30	0.18	0.73	0.65	0.56	0.69	0.0
Pyramid + PROX + WO + REP	4.5	0.31	0.28	0.17	1.00	0.93	0.71	0.69	0.1

TABLE 1
Benchmark results for different layouts with 200 tags, two weeks (higher is better). *WO*, *PROX* and *Pyramid* denote whether the respective `wordOrder`, `proximity` and `location` components of our energy function are activated. The last row represents the full PyramidTags layout.

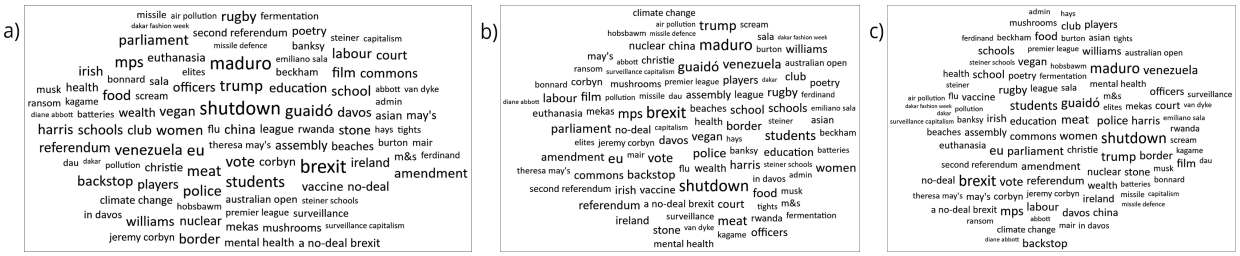


Fig. 9. Different layouts without location boxes, based on activated components of our energy function: a) Simple, b) PROX, c) PROX + WO

indicates types with the triangular layout, different location boxes and tag splitting. Hence, the last row represents the full PyramidTags variant with all $\lambda_x \neq 0$.

Figure 9 shows three visualization types without location boxes in action. All depicted layouts were trained on the same data set. Similar to neural network training, the output may differ due to the random initializations. Thus, we ran the generation routine twice to mitigate the effect of outliers, resulting in six different visualization instances on three different date ranges per visualization type.

7.1 Benchmarks

We calculated the following benchmarks on the generated output:

Density: Average number of tags in the neighborhood of any tag. The neighborhood of a tag consists of all tags that are located within an outer distance (shortest distance between the bounding boxes) of the height of the respective tag. Choosing a slightly higher threshold increases the number of tags in the neighborhood, but the relations of the benchmark values between different variants stay largely the same.

Context-Overlap (Context): For each tag, we fetch those articles that contain the tag. Each article has a list of categories provided by the newspaper. We use this to build a *topic vector* with the occurrence count of each topic, representing the thematic landscape of the respective tag. *Context* is the average cosine similarity between the topic vectors of the center and a neighboring tag. A higher value indicates that nearby tags are more likely to be related thematically. To make these values more interpretable, we further calculate the probability that a neighboring topic vector has a cosine similarity ≥ 0.4 (*Context4*) or ≥ 0.7 (*Context7*), i.e. how likely it is that a neighboring tag is thematically related according to different thresholds.

Word-Order (WO*x*): For each tag set, we create a ranking of pairs according to the evidence that they appear in a certain order as explained in Section 5.8. To evaluate how well different layout strategies perform in preserving the word order, we calculate the fractions of the top 5 (*WO5*), 10 (*WO10*) and 50 (*W50*) pairs appearing in the correct order.

Time-Awareness (DateCos): For each tag, we build a date vector comprised of the number of articles per day. *DateCos* denotes the average cosine similarity between date vectors of a tag and one of its neighbors. A higher value indicates that neighboring tags have a higher similarity regarding the temporal evolution of related articles.

Horizontal Displacement (DaysOff): The average displacement in days per tag to its associated time range.

7.2 Results

Table 1 shows the results of the benchmarks for the configuration of 200 tags and a time span of two weeks. Results for the other configurations can be found in the supplementary material. All visualization types with activated proximity component show a notable improvement in the Context-Overlap score. The triangular layout results in a distinct increase of the cosine similarity between date vectors of neighboring tags compared to the dense word cloud layout, as expected. Interestingly, the benchmarks show that it is also more context-aware even without enabling the proximity component. Most of the tags are placed within their associated horizontal range, but the displacement increases with the number of tags and the overall time span.

The word order of the most important pairs is preserved in all variants that use the `wordOrder` component. Conversely, the word order probabilities of the remaining vari-

ants are on par with the expected random odds. However, it can be seen that there is a trade-off between context-awareness and word-order preservation.

As expected, the *Pyramid* layouts have a lower density compared to the other variants. Nevertheless, our PyramidTags approach stands out for its unique ability to express temporal relationships while at the same time preserving the word order at least as good as the *PROX+WO* variant.

These benchmarks reveal that our approach is clearly more context-, word order- and time-aware than traditional tag clouds. Furthermore, they show that our proposed placement strategy to visualize the temporal evolution of tags structures the data in a meaningful way. In addition, the visual comparison in Figure 9 reveals that our *PROX+WO* approach offers semantically clustered tag clouds while also better preserving the word order compared to a random layout, but with the same high density and no increase in screen space, making it well-suited for data sets without timestamps or in which the temporal evolution only plays a marginal role in analyzing the data.

7.3 Qualitative Feedback

In addition, we individually presented PyramidTags to two visualization experts working in a company that deals with social media monitoring. We used a configuration of 250 tags that were extracted from our news data set of January 2019. First, they had to inspect the visualization without any introduction and tell us what they thought the data set was about, how they think the layout is supposed to work, and which insights they gained. After explaining the system and which kind of interactions it supports, we then asked them to further explore the data set and try to find out more about specific themes they are interested in.

Both experts stated that the triangular layout is intuitive, however, one expert initially thought that the pyramid represented a hierarchical representation of topics before we explained the system. They found it helpful that many tags were placed in their reading direction and that the dots indicate the assumed word order. Both quickly noticed several clusters of tags and hypothesized about related stories and their temporal evolution. They rated the interface as responsive, with no lag while interacting with it.

One expert was wondering why there were duplicate tags, but found it useful after we revealed the reason behind it. They noted that, in some cases, nearby tags wrongly appeared to belong together, and suggested some sort of coloring or a visual border to separate unrelated tags, if possible at all.

One expert would have found it useful to highlight tags based on a selected date range at the bottom. Furthermore, they stated that also offering a negative selection, i.e. selecting tags that are not relevant, would improve the utility. Both proposed to implement a filtering mechanism that would allow analysts to explicitly search for keywords, resulting in an updated visualization solely based on the filtered documents.

They thought that the presented search results were relevant to the selection they performed and liked the fact that documents open in a new tab instead a new window. One suggestion was to highlight occurrences of selected tags in the retrieved document. Both regarded the overall system as helpful to explore large time-stamped collections. One expert was particularly impressed that the approach made it easy to find specific topics with just two or three clicks, even though, at first glance, the visualization had not looked like it would really represent half a million documents.

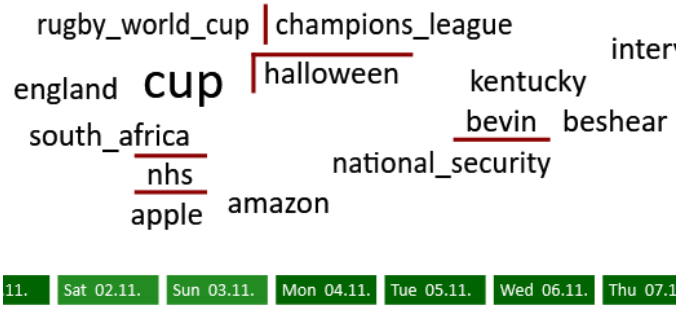


Fig. 10. Early support of visual borders to separate unrelated tags which have been placed nearby.

8 Discussion

We developed PyramidTags to let analysts visually explore large, time-stamped document collections without requiring prior knowledge about the data set at hand. We chose a tag-based method to directly show important snippets from the corpus and to avoid predefined topics. We proposed a novel spatialization technique that incorporates several design goals. Semantically connected tags should be placed in close proximity on the map and the position should give hints about the temporal evolution. This enables analysts to not only track individual tags over time, but also to quickly see at which date range a group of possibly related tags is most present in the collection.

Our multi-word tag extraction algorithm in combination with the intention to preserve the word order leads to the appearance of longer phrases that help users form a mental map of the content at hand, while still considering computational costs. The relatedness between tags is induced by their distance patterns in the underlying data, resulting in a more fine-grained concept of relationship compared to a range of previous approaches that define semantic similarity based on whether tags appear in the same sentence.

Our interaction techniques support analysts to select appropriate tags of interest and retrieve related documents. If many tags are selected at the same time, the (optional) trapezoids visualizing the respective time ranges become less useful due to overlap. However, it rarely happens that this many tags have to be selected, because every new item drastically reduces the search space.

A disadvantage is the increase of whitespace compared to dense tag cloud layouts. This increase is still modest, but it can limit the usability and utility in certain cases, for instance on mobile devices. However, whitespace does not necessarily mean wasted space, because the lack of content can also express information, e.g., less activity in certain date ranges, or less semantic coherence. Furthermore, the space on the upper left and right can be used to place the search results window and document viewer.

In contrast to many related approaches, we apply our method to large, real-world data sets in order to prove its scalability. On a modern 6-core CPU with a parallel implementation, it takes about 15 seconds to process 10,000 documents extracting 100 tags, analyzing their relationship and generating an index for document retrieval. Processing 500,000 documents extracting 250 tags lasts 10-20 minutes. Generating the final visualization additionally takes several minutes. Once the visualization is generated, users can interact with it and retrieve documents instantly.

Tags that are placed close to each other do not necessarily relate to each other. Additionally, in some cases viewers might 'read' a certain phrase which makes sense grammatically, but is actually not backed by the document collection. These are common disadvantages if complex non-linear relationships are projected onto flat visualizations. In our case, users can interactively debunk such false friends, for example by hovering over tags to reveal relationships in detail. In addition, we currently investigate how we can implement the idea to draw borders between unrelated tags, which was raised by one of the experts during the interview. A preliminary result is shown in Figure 10.

Aiming for several objectives at the same time is challenging. Nevertheless, our quantitative evaluation showed that our approach reliably produces semantically clustered layouts which also convey temporal patterns *and* preserve the word order for the most important pairs.

9 Conclusion

We presented PyramidTags that generates context-, word order- and time-aware tag maps. The visualization combined with rich interactions enables analysts to explore and gain insights into large document collections.

In the future, we want to investigate how we can further speed up the process, possibly by designing a differentiable energy function to apply stochastic gradient descent methods. This could enable exciting interaction mechanisms, in which more specific tag maps could be interactively generated based on a selection of tags. In addition, we would like to reduce the number of similarly worded tags without loosing the ability to encourage longer, connected phrases.

Acknowledgments

This work was funded by the German Science Foundation DFG as part of the 392087235 project 'Visual Analytics of Online Streaming Text' (VAOST).

References

[1] S. M. Yimam, H. Ulrich, T. von Landesberger, M. Rosenbach, M. Regneri, A. Panchenko, F. Lehmann, U. Fahrer, C. Biemann, and K. Ballweg, "new/s/leak – Information Extraction and Visualization for Investigative Data Journalists," in *Proceedings of ACL -2016 System Demonstrations*, 2016, pp. 163–168.

[2] M. J. Halvey and M. T. Keane, "An assessment of tag presentation techniques," in *Proceedings of the 16th International Conference on World Wide Web*, 2007, pp. 1313–1314.

[3] A. W. Rivadeneira, D. M. Gruen, M. J. Muller, and D. R. Millen, "Getting our head in the clouds," *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '07*, p. 995, 2007.

[4] F. B. Viégas and M. Wattenberg, "TIMELINES: Tag clouds and the case for vernacular visualization," *Interactions*, vol. 15, no. 4, p. 49, 2008.

[5] M. Hu, K. Wongsuphasawat, and J. Stasko, "Visualizing Social Media Content with SentenTree," *IEEE Transactions on Visualization and Computer Graphics*, vol. 23, no. 1, pp. 621–630, 2017.

[6] J. Sinclair and M. Cardew-Hall, "The folksonomy tag cloud: When is it useful?" *Journal of Information Science*, vol. 34, no. 1, pp. 15–29, 2008.

[7] F. Heimerl, S. Lohmann, S. Lange, and T. Ertl, "Word cloud explorer: Text analytics based on word clouds," in *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2014, pp. 1833–1842.

[8] Y. Wu, T. Provan, F. Wei, S. Liu, and K. L. Ma, "Semantic-preservingword clouds by seam carving," *Computer Graphics Forum*, vol. 30, no. 3, pp. 741–750, 2011.

[9] M. T. Chi, S. S. Lin, S. Y. Chen, C. H. Lin, and T. Y. Lee, "Morphable Word Clouds for Time-Varying Text Data Visualization," *IEEE Transactions on Visualization and Computer Graphics*, vol. 21, no. 12, pp. 1415–1426, 2015.

[10] M. Hearst, E. Pedersen, L. P. Patil, E. Lee, P. Laskowski, and S. Franconeri, *An Evaluation of Semantically Grouped Word Cloud Designs*, mar 2019.

[11] J. Wang, J. Zhao, S. Guo, C. North, and N. Ramakrishnan, "ReCloud: semantics-based word cloud visualization of user reviews," *Proceedings of Graphics Interface 2014*, pp. 151–158, 2014.

[12] J. Chuang, C. D. Manning, and J. Heer, "Termite: Visualization techniques for assessing textual topic models," in *Proceedings of the Workshop on Advanced Visual Interfaces AVI*, 2012.

[13] I. Liiv, "Review seriation and matrix reordering methods: An historical overview," 2010.

[14] A. Endert, R. Burtner, N. Cramer, R. Perko, S. Hampton, and K. Cook, "Typograph: Multiscale spatial exploration of text documents," in *Proceedings - 2013 IEEE International Conference on Big Data, Big Data 2013*, 2013, pp. 17–24.

[15] C. L. Paul, J. Chang, A. Endert, N. Cramer, D. Gillen, S. Hampton, R. Burtner, R. Perko, and K. A. Cook, "TexTonic: Interactive visualization for exploration and discovery of very large text collections," 2018.

[16] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer, "On the beauty and usability of tag clouds," in *Proceedings of the International Conference on Information Visualisation*, 2008, pp. 17–25.

[17] F. B. Viégas, M. Wattenberg, and J. Feinberg, "Participatory visualization with wordle," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, 2009, pp. 1137–1144.

[18] S. Bateman, C. Gutwin, and M. Nacenta, "Seeing things in the clouds: the effect of visual features on tag cloud selections," in *Proceedings of the nineteenth ACM conference on Hypertext and hypermedia*, 2008, pp. 193–202.

[19] J. Schrammel, M. Leitner, and M. Tscheligi, "Semantically structured tag clouds," *Engineering*, p. 2037, 2009.

[20] S. Banerjee and T. Pedersen, "Extended gloss overlaps as a measure of semantic relatedness," in *IJCAI International Joint Conference on Artificial Intelligence*, 2003.

[21] S. Lohmann, J. Ziegler, and L. Tetzlaff, "Comparison of tag cloud layouts: Task-related performance and visual exploration," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 5726 LNCS, no. PART 1, 2009, pp. 392–404.

[22] C. Felix, S. Franconeri, and E. Bertini, "Taking Word Clouds Apart: An Empirical Investigation of the Design Space for Keyword Summaries," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 1, pp. 657–666, 2018.

[23] X. Liu, H. W. Shen, and Y. Hu, "Supporting multifaceted viewing of word clouds with focus+context display," *Information Visualization*, vol. 14, no. 2, pp. 168–180, 2015.

[24] M. Dörk, S. Carpendale, C. Collins, and C. Williamson, "VisGets: Coordinated visualizations for web-based information exploration and discovery," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008, pp. 1205–1212.

[25] J. Chuang, C. D. Manning, and J. Heer, ""Without the clutter of unimportant words"," *ACM Transactions on Computer-Human Interaction*, vol. 19, no. 3, pp. 1–29, 2012.

[26] P. D. Turney, "Learning algorithms for keyphrase extraction," *Information Retrieval*, 2000.

[27] I. Adä, K. Thiel, and M. R. Berthold, "Distance aware tag clouds," in *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, 2010, pp. 2316–2322.

[28] A. A. Talin, F. Léonard, A. M. Katzenmeyer, B. S. Swartzentruber, S. T. Picraux, M. E. Toimil-Molares, J. G. Cederberg, X. Wang, S. D.

Hersee, and A. Rishinaramangalum, "Transport characterization in nanowires using an electrical nanoprobe," *Semiconductor Science and Technology*, vol. 25, no. 2, 2010.

[29] Weiwei Cui, Yingcai Wu, Shixia Liu, Furu Wei, M. X. Zhou, and Huamin Qu, "Context-Preserving, Dynamic Word Cloud Visualization," *IEEE Computer Graphics and Applications*, vol. 30, no. 6, pp. 42–53, 2010.

[30] J. Xu, Y. Tao, and H. Lin, "Semantic word cloud generation based on word embeddings," in *IEEE Pacific Visualization Symposium*, vol. 2016-May, 2016, pp. 239–243.

[31] S. Jänicke and G. Scheuermann, "On the visualization of hierarchical relations and tree structures with tagspheres," in *Communications in Computer and Information Science*, 2017.

[32] C. Collins, F. B. Viégas, and M. Wattenberg, "Parallel tag clouds to explore and analyze faceted text corpora," in *VAST 09 - IEEE Symposium on Visual Analytics Science and Technology, Proceedings*, 2009, pp. 91–98.

[33] B. Lee, N. H. Riche, A. K. Karlson, and S. Carpendale, "SparkClouds: Visualizing trends in tag clouds," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1182–1189, 2010.

[34] C. Binucci, W. Didimo, and E. Spataro, "Fully dynamic semantic word clouds," in *IISA 2016 - 7th International Conference on Information, Intelligence, Systems and Applications*, 2016.

[35] S. Lohmann, F. Heimerl, F. Bopp, M. Burch, and T. Ertl, "Concentri cloud: Word cloud visualization for multiple text documents," in *Proceedings of the International Conference on Information Visualisation*, vol. 2015-Septe, 2015, pp. 114–120.

[36] N. Van de Weghe, R. Docter, P. De Maeyer, B. Bechtold, and K. Ryckbosch, "The triangular model as an instrument for visualising and analysing residuality," *Journal of Archaeological Science*, vol. 34, no. 4, pp. 649–655, 2007.

[37] Z. Kulpa, "Diagrammatic Representation of Interval Space in Proving Theorems about Interval Relations," *Reliable Computing*, vol. 3, no. 3, pp. 209–217, 1997.

[38] M. Burch, S. Lohmann, D. Pompe, and D. Weiskopf, "Prefix tag clouds," in *Proceedings of the International Conference on Information Visualisation*, 2013, pp. 45–50.

[39] M. Wattenberg and F. B. Viégas, "The word tree, an interactive visual concordance," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, 2008, pp. 1221–1228.

[40] F. Van Ham, M. Wattenberg, and F. B. Viégas, "Mapping text with phrase nets," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 15, no. 6, 2009, pp. 1169–1176.

[41] J. Wise, J. Thomas, K. Pennock, D. Lantrip, M. Pottier, A. Schur, and V. Crow, "Visualizing the non-visual: spatial analysis and interaction with information from text documents," *Proceedings of Visualization 1995 Conference*, pp. 51–58,, 2002.

[42] J. Alsakran, Y. Chen, Y. Zhao, J. Yang, and D. Luo, "STREAMIT: Dynamic visualization and interactive exploration of text streams," in *IEEE Pacific Visualization Symposium 2011, PacificVis 2011 - Proceedings*, 2011, pp. 131–138.

[43] F. Heimerl, M. John, Q. Han, S. Koch, and T. Ertl, "DocuCompass: Effective exploration of document landscapes," in *2016 IEEE Conference on Visual Analytics Science and Technology, VAST 2016 - Proceedings*, 2017, pp. 11–20.

[44] Weiwei Cui, Shixia Liu, Li Tan, Conglei Shi, Yangqiu Song, Zekai Gao, Huamin Qu, and Xin Tong, "TextFlow: Towards Better Understanding of Evolving Topics in Text," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2412–2421, 2011.

[45] W. Cui, S. Liu, Z. Wu, and H. Wei, "How hierarchical topics evolve in large text corpora," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 2281–2290, 2014.

[46] W. Dou, X. Wang, D. Skau, W. Ribarsky, and M. X. Zhou, "LeadLine: Interactive visual analysis of text data through event identification and exploration," in *Proceedings of the 2012 IEEE Conference on Visual Analytics Science and Technology (VAST)*, 2012, pp. 93–102.

[47] S. Havre, E. Hetzler, P. Whitney, and L. Nowell, "ThemeRiver: Visualizing thematic changes in large document collections," *IEEE Transactions on Visualization and Computer Graphics*, vol. 8, no. 1, pp. 9–20, 2002.

[48] M. Krstajić, E. Bertini, and D. A. Keim, "Cloudlines: Compact display of event episodes in multiple time-series," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2432–2439, 2011.

[49] M. Krstajić, M. Najm-Araghi, F. Mansmann, and D. A. Keim, "Story tracker: Incremental visual text analytics of news story development," *Information Visualization*, vol. 12, no. 3-4, pp. 308–323, 2013.

[50] G. Sun, Y. Wu, S. Liu, T. Q. Peng, J. J. Zhu, and R. Liang, "EvoRiver: Visual analysis of topic coopetition on social media," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1753–1762, 2014.

[51] F. Wei, S. Liu, Y. Song, S. Pan, M. X. Zhou, W. Qian, L. Shi, L. Tan, and Q. Zhang, "TIARA: A Visual Exploratory Text Analytic System," *ACM SIGKDD International Conference on Knowledge Discovery and Data Minining*, pp. 153–162, 2010.

[52] Y. Wu, S. Liu, K. Yan, M. Liu, and F. Wu, "OpinionFlow: Visual analysis of opinion diffusion on social media," pp. 1763–1772, 2014.

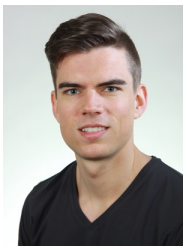
[53] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, no. 5, pp. 513–523, 1988.

[54] S. Rose, D. Engel, N. Cramer, and W. Cowley, "Automatic Keyword Extraction from Individual Documents," in *Text Mining: Applications and Theory*, 2010.

[55] K. Frantzi, S. Ananiadou, and H. Mima, "Automatic recognition of multi-word terms: The C-value/NC-value method," *International Journal on Digital Libraries*, 2000.

[56] B. Chopard and M. Tomassini, "Particle swarm optimization," in *Natural Computing Series*, vol. 4, 2018, pp. 97–102.

[57] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360)*, 2002, pp. 69–73.



Johannnes Knittel received the M.Sc. degree in computer science from the University of Stuttgart. He is working toward the doctoral degree in the Institute for Visualization and Interactive Systems at the University of Stuttgart. His research interests include visual analytics, visual text analytics, data mining and machine learning.

Steffen Koch earned his doctorate degree in computer science from the University of Stuttgart in 2012. He currently has a permanent position as a research associate at the Institute for Visualization and Interactive Systems at University of Stuttgart. His research interests comprise visualization in general, with foci on visual analytics for text/documents, visualization in the digital humanities, and interactive visualization support for data mining/machine learning.

Thomas Ertl is a full professor of Computer Science at the University of Stuttgart in the Visualization and Interactive Systems Institute (VIS) and director of the Visualization Research Center of the University of Stuttgart (VISUS). He is coauthor of more than 500 scientific publications. He served as editor-in-chief of IEEE TVCG and as chairman of the Eurographics Association. He received the Outstanding Technical Contribution Award of the Eurographics Association and the Technical Achievement Award of the IEEE Visualization and Graphics Technical Committee in 2006.