SocialTree: Socially Augmented Structured Summaries of News Stories

Gevorg Poghosyan
Insight Centre for Data Analytics
University College Dublin
Dublin, Ireland
gevorg.poghosyan@insight-centre.org

Georgiana Ifrim
Insight Centre for Data Analytics
University College Dublin
Dublin, Ireland
georgiana.ifrim@insight-centre.org

ABSTRACT

News story understanding entails having an effective summary of a related group of articles that may span different time ranges, involve different topics and entities, and have connections to other stories. In this work, we present an approach to efficiently extract structured summaries of news stories by augmenting news media with the structure of social discourse as reflected in social media in the form of social tags. Existing event detection, topic-modeling, clustering and summarization methods yield news story summaries based only on noun phrases and named entities. These representations are sensitive to the article wording and the keyword extraction algorithm. Moreover, keyword-based representations are rarely helpful for highlighting the inter-story connections or for reflecting the inner structure of the news story because of high word ambiguity and clutter from the large variety of keywords describing news stories. Our method combines the news and social media domains to create structured summaries of news stories in the form of hierarchies of keywords and social tags, named *SocialTree*. We show that the properties of social tags can be exploited to augment the construction of hierarchical summaries of news stories and to alleviate the weaknesses of existing keyword-based representations. In our quantitative and qualitative evaluation the proposed method strongly outperforms the state-of-the-art with regard to both coverage and informativeness of the summaries.

CCS CONCEPTS

• **Computing methodologies** → **Information extraction**; • **Information systems** → *Document topic models*; Association rules.

KEYWORDS

social indexing; news summarizarion; association rules

This is the authors' version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *Proceedings of 30th ACM Conference on Hypertext & Social Media (HT '19)*, https://doi.org/10.1145/3342220.3343668.

HT '19, September 17–20, 2019, Hof, Germany

ACM ISBN 978-1-4503-6885-8/19/09…$15.00
https://doi.org/10.1145/3342220.3343668

1 INTRODUCTION

The rapid development of news media sites has led to large volumes of news content being produced every day by a huge number of sources. News consumers on both traditional news media and social media platforms have difficulties finding and following the news stories of interest from the unstructured streams of news. Journalists, on the other hand, need help for writing explainer articles[1] and summaries for complex stories. Furthermore, for such stories it is often difficult to quickly deduce the relationship between entities, events and connections with other stories over time.

Recently, there have been many efforts in the research community to address the well-known problem of *information overload* [11, 13, 14]. The proposed solutions aim to reveal the relationships between news stories to help the news consumers better understand, interpret and summarize the news and to track their evolution in time. The existing methods include evolutionary clustering for storyline extraction [11], graph analytical approaches [13], generative models [5, 16, 18] and more complex approaches combining multiple models [1, 14] to enable the extraction of topics, events or clusters, which we collectively refer to as *stories*[2]. Some of the solutions achieve high accuracy, but do not operate in real-time [16, 18] or do not scale up for tracking stories longer than few weeks [10]. The common weaknesses of existing methods are (i) the used representations are sensitive to article wording and the keyword extraction algorithm, (ii) the resulting summaries are either bulky and thus unhelpful for human interpretation, or miss on smaller but still important aspects or events of the story.

In this paper we propose a methodology for creating structured summaries of news stories. Different from the extractive or abstractive textual summarization tasks, we do not aim to produce an abridged textual version of the documents of interest. The goal of our method is to help the user navigate the news stream by providing an overview and the relatedness structure of key entities, time periods and popular discussion aspects in a story. Given a large news document collection annotated with social tags and keywords, and a user-defined query that describes the story of interest, our algorithm generates a concise structure of the story that maximizes the coverage of relevant substories. Apart from being consumed by the end-user for news understanding or by a journalist as an aid, our summary structures can also be used for query expansion, query refining or for textual summary extraction.

[1] https://www.fipp.com/news/opinion/what-is-explainer-journalism

[2] In order to avoid confusion between the terms *event*, *topic*, *cluster*, *episode* or *saga* used for different methods in correspondingly event detection, topic modeling and clustering tasks, we use the term *story* to refer to a group of related news articles. The term *substory* is used to refer to different aspects of a news *story*.

Our method builds on the coupling of social media and the mainstream news, as complementary means to tell a story. This allows our method to produce better summaries than state-of-the-art methods by (i) reducing the sensitivity to document wording and keyword extraction algorithm, (ii) connecting to relevant discussions on social platforms for a more complete news experience.

Many social platform users annotate their posts using social tags, also known as hashtags. These are keywords, preceded by the '#' symbol, meant to provide context to a social post and connect users involved in the same discussion. Hashtags were proven to be useful for browsing social media. They are widely used on Twitter, and are making their way to traditional media, e.g., AJ+ and TheJournal.ie use hashtags to organize their content. Furthermore, news editors attach hashtags to the news headlines they publish on social media, in order to reach relevant communities. Essentially, social tags are a new form of metadata. Yet, despite the society's growing dependence on digital content and increasing availability of it, metadata is at a premium [8].

In our experiments we compare the proposed summarization method to the state-of-the-art in terms of efficiency and also the compactness, document coverage and informativeness of the produced summaries. We address the following research questions. (RQ1) – **"Does a document representation augmented with social annotations enable better (e.g., denser, more informative) summaries compared to keyword-only summaries?"** Recent recommendation tools, like *Hashtagger* [15], have advanced the automated tagging of news with social tags. These tags are sourced by merging news and social media streams in real-time, followed by a high-precision method (achieving 87% Precision@1 as reported by the authors) for recommending social tags to news articles. The key idea is that the social discourse on Twitter and news media can be captured via hashtags attached to news articles, and this can enable us to do better story summarization, since people tend to use only a few hashtags to label and to provide context to the stories. For example, simple keyword matching and co-occurrence for the story "blacks racial conflicts in the USA" is confined to the vocabulary used in news articles and may retrieve many off-topic documents. If we have relevant hashtags attached to news articles, we can exploit the social discourse expressed via hashtags (such as #blacklivesmatter) to enable better story retrieval and summarization. (RQ2) – **"Do hierarchical structures provide better (e.g., more comprehensive) summaries compared to flat structures?"** Here we test our hypothesis suggesting that hierarchical organization in structures allows users to explore substories at various levels of abstraction. We design a methodology and a user study to allow us to answer these questions in detail.

We summarize our main **contributions** as follows:

- We propose techniques for extracting a hierarchical structure built of keywords and social tags that summarizes the news story described in a given collection of news articles.
- We show that we can efficiently extract informative structured summaries of long-running (e.g., years) stories from large collections of news documents, by adapting Frequent Itemset Mining and Maximum Spanning Tree algorithms.
- We conduct a quantitative and qualitative assessment of the effect of using socially augmented document representations in a multi-document summarization task.

2 RELATED WORK

In this section we discuss existing approaches for retrospective story summarization. Among the prior work, we distinguish the methods which produce unstructured summaries of news article collections from the ones that represent the extracted news stories in a meaningful structure to help story understanding.

The related work is summarized and ordered by publication year in Table 1 and can be compared by the document representation, produced story structure, ability to capture the temporal relationship of substories, time complexity, ability to update stories with incoming data and whether the method allows accessing a document from different substories, denoted in the table as "multi-label".

Unstructured Summaries of News Stories: The state-of-the-art methods for topic detection and tracking, storyline extraction and summarization methods group news articles into stories based on the similarity of the articles. Many of these methods represent news stories on a linear timeline. A major disadvantage of these methods is that they rely on the assumption that the substories are disjoint in time. The linear timelines are thus limited to connect only adjoining substories. In some cases metadata (e.g., location, organizations, people, topics) extracted from the articles is used in the story representation. Zhou et al. [18] present a non-parametric model combined with a Chinese Restaurant Process (named *Storyline* in Table 1), which extracts metadata-rich representations and evolution patterns of storylines. Ahmed et al. [1] proposed a non-parametric evolutionary clustering method (named *RCRP* in Table 1), which models news storyline clustering by applying a topic model to the clusters, while simultaneously generating single-linkage clusters using the Recurrent Chinese Restaurant Process. This approach allows the number of stories to be determined by the data, which is an important feature in practice. Hua et al. [5] introduced a Bayesian model, called *ASG*, generating storylines from news articles using Twitter data for "supervision". They propose a tripartite graph of linked topics, events and storylines as a representation. *NewsMiner* introduced by Hou et al. [4] represents each news article in the dimensions of entities, topics and events. A knowledge base is used for linking topics and linking entities and thus creating links between the articles.

Structured Summaries of News Stories: Methods in this category reveal the inter-story and intra-story relations in addition to grouping articles into topical and/or temporal clusters. CHARCOAL presented by Tang et al. [16] is a non-parametric probabilistic model, which uses a three-level distance-dependent Chinese Restaurant Process for clustering the articles into stories, and the stories into topics. CHARCOAL produces location-oriented topics with directed acyclic graphs of articles. Due to the restricted nature of extractable topics by this method, we do not include it in the evaluation performed in our study. *Story Forest*, presented in [11], clusters the articles into events and connects the extracted events in a directed acyclic graph, where the edge directions indicate the time flow. Sayyadi and Raschid [13] have proposed a topic detection method called *KeyGraph*, which represents the given document collection as a keyword co-occurrence graph. The method creates connections between the events and also allows articles to be linked to multiple events. *KeyGraph* performs community detection on the keyword co-occurrence graph, and the keywords in each community are

Method	Representation Space	Story Structure	Temporal	Time Complexity	Online, Evolutionary	Multi-label
RCRP (Ahmed et al. 2011) [1]	noun phrases, named entities	clusters linked to topics	-	$O(\|\mathbb{D}\|)$	+	+
MetroMap (Shahaf et al. 2012) [14]	named entities	intertwining timelines	+	$O(\|\mathbb{D}\|)$	-	+
KeyGraph (Sayyadi and Raschid 2013) [13]	noun phrases, named entities	linked graph node communities	-	$\sim O(\|\mathbb{D}\|)$	-	+
NewsMiner (Hou et al. 2015) [4]	named entities	list of documents linked to entities, events, topics	-	N/A	-	-
CHARCOAL *(Tang et al. 2015) [16]*	named entities	branching timeline (DAG) of headlines	+	N/A	-	-
ASG (Hua et al. 2016) [5]	N/A	tripartite graph of storylines, events, topics	-	N/A	-	-
Storyline (Zhou et al. 2016) [18]	categorized named entities	timeline	+	$O(\|\mathbb{D}\|)$	-	-
Story Forest (Liu et al. 2017) [11]	classified keywords	branching timeline (DAG) of headlines	+	$O(N_{events})$	+	-
***SocialTree** (this paper)*	**keywords, hashtags**	spanning tree	+	$O(\|\mathbb{D}\|)$	-	+

Table 1: Comparison of news story summarization methods. $|\mathbb{D}|$ is the size of the document collection. "+" and "-" are used to indicate the existence of the feature and "N/A" indicates method characteristics not reported by the authors.

used to represent topics (substories). *MetroMap* proposed by Shahaf et al. [14] had objectives most similar to ours. The method first detects article clusters in each time window, and then connects these clusters into stories. The temporally ordered clusters (metro stops) are connected into "metro lines" which represent the substories. The "metro lines" intersect in some clusters and form a *MetroMap*, which represents the story structure.

MetroMap produces complex structures which are not easy for the users to interpret. Both CHARCOAL and *Story Forest* represent stories as directed acyclic graphs, where the branches are timelines and the edge directions indicate the time flow. The timeline branching eliminates a major disadvantage of linear timelines, by allowing topical links between substories in addition to the temporal links. Nevertheless, these approaches do not scale for long stories, because the number of extracted events usually scales linearly with the coverage period, which, in turn, makes the summaries unreadable. *Story Forest* also requires supervision and heavy parameter tuning. In our experiments we could not reproduce the results presented for *Story Forest* in [11]; in particular that method was not evaluated on English corpora and has many parameters that require extensive tuning.

SocialTree's representation of stories is a tree as well, but unlike the above mentioned methods, we use trees of keywords and hashtags (later referred to as *tags*) to unveil their hierarchical relationship. We regard these tags as substories and aspects of the user interest. In *SocialTree* the temporal information is visualized orthogonally to the tree depth, by ordering and sizing the substories on each hierarchy level according to their time periods. The links between the substories are undirected and encode substory relatedness, query relevance and duration. The hierarchical structure allows users to explore substories at various levels of abstraction [2]. Similar to *KeyGraph* and *MetroMap* we use a graph analytical approach and leverage metadata co-occurrence between documents for substory extraction. Our work goes beyond existing methods in the following:

(i) the problem is framed and solved in a new representation space combining hashtags and keywords,
(ii) the story representation is created prior to and independent from structure extraction (unlike all other methods),
(iii) introduction of new criteria for story-aspect relatedness and aspect pruning,
(iv) encoding aspect importance in the inter-aspect relatedness,
(v) the structure produced by our method is a hierarchical organization, as opposed to a hierarchical linkage of substories as in hierarchical clustering — it encodes substory relatedness as opposed to subdivision, which enables higher compression[3].

3 DESIGN GOALS FOR STRUCTURED SUMMARIES

Our goal is to extract informative summaries of long-running complex stories from a news collection, given a user's interest expressed in the form of a text query and a time period. The objective is to reduce the summary's sensitivity to the article wording and keyword extraction, as well as to improve its informativeness to a human user, thereby addressing the main weaknesses of state-of-the-art methods. For this purpose, we propose to leverage the connections of hashtags in the social discourse by augmenting the document representation with social tags.

Hashtags used by big crowds most often depict the most important aspects of the story and are often interpretable, e.g., #impeachzuma is used with posts related to the impeachment of the fourth President of South Africa Jacob Zuma. Some news stories, however, don't get annotated with any hashtags, because these stories either get too little attention on social media or the related discussions do not converge to representative hashtags. Keywords are often noisy, but, on the other hand, are able to capture details of the story that are not annotated as hashtags. For example, social

[3]As a result some of the data in child nodes may not be accessible from their parent nodes and vice-versa.

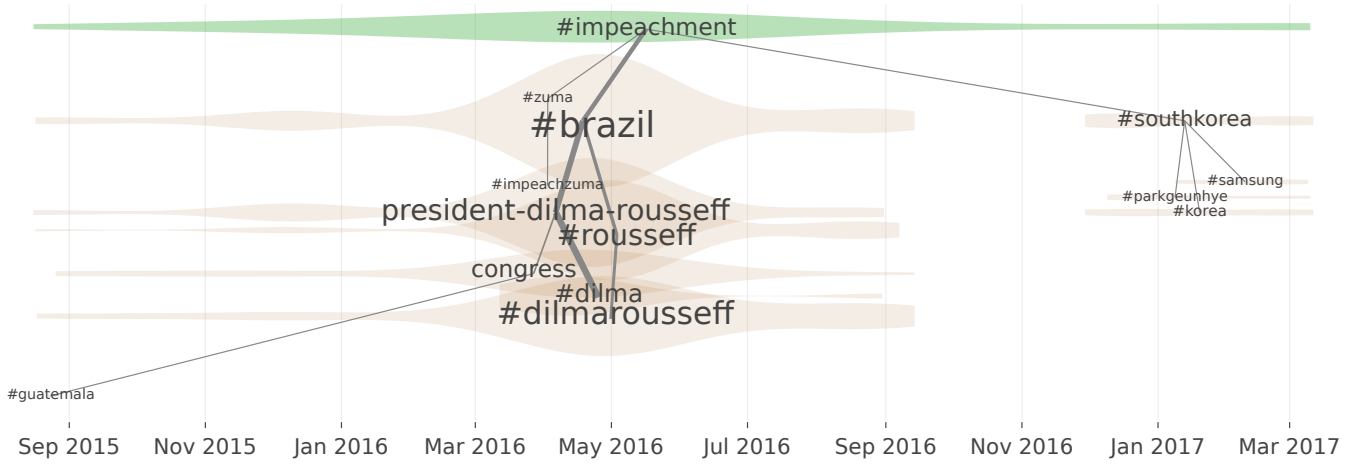

Figure 1: An example *SocialTree* illustrating the "impeachment" story in the period from August 2015 to May 2017. The green node on top (#impeachment) is the tree root. The horizontal dimension is the time and a node's horizontal stretch illustrates the time range in which the tag was assigned to the articles. The vertical dimension indicates the node levels and is a proxy for how specific/generic is the tag in the current story context. A node's level in the hierarchy has a meaning only within the scope of its branch and does not carry a story-wide information. Hierarchy levels of nodes are not comparable across different branches. The vertical size of nodes is proportional to the number of articles annotated with the tag in a given time period. The users can interact with a *SocialTree* by zooming and hovering over the nodes and edges for additional information.

discourse around a terrorist attack peaks right after the event, and people usually use hashtags like *#terrorist*, *#attack*, *#paris* etc. Nevertheless, details like the names of the terrorists don't appear as hashtags because they may surface long after the peak social activity and also because people rarely use terrorists' names as hashtags. To get the best of the two worlds, we use a mix of hashtags and keywords for document representation and refer to them as *tags*[4]. The lexically duplicate tags, e.g., *#paris* and *"Paris"* or *#dilmarousseff* and *"Dilma Rousseff"* are merged into the hashtag, resulting in *#paris* and *#dilmarousseff* respectively, in order to allow the user to connect to social discussions directly.

As selected by the social media crowd, the hashtags represent the most important aspects of the story. This nature of hashtags allows us to interpret them as substories. The high density of this representation (compared to a keyword-only representation) enables us to enforce strict connectivity requirements without breaking the storyline. Other methods, like *KeyGraph* and *MetroMap*, rely on high connectivity of the communities of keywords to extract substories. Considering the social tag characteristics, we are looking for a hierarchical story structure in the form of a spanning tree, where the tree nodes are not documents, but substories. Linear timelines allow only temporal distances in the story summary. *MetroMap*, on the other hand, allows only binary linked/not-linked relationship of substories. A hierarchical organization creates a notion of distance between any pair of tags reflected both by the path connecting them and by their corresponding distances from the tree root. The more dissimilar are the hashtags, the longer should be the path connecting them in the tree.

The generic tags which represent the core of the story and appear several times in the storyline should be the connecting rings for tangent or minor substories and short-lived small events. For example, #impeachment is the connecting link for South African #zuma, #brazil and #southkorea as shown in Figure 1. For the ease of navigation in a summary, the more specific tags should represent substories of the generic tags, and the latter should appear in higher hierarchy levels. The hierarchy level of a tag serves as an indicator of how generic or specific it is in the story context. Note, that an aspect's level in the hierarchy has a meaning only within the scope of its branch and does not carry a story-wide information. Hierarchy levels of aspects are not comparable across different branches, because those reflect the story's aspect relations only within a branch. For example, although #zuma and #rousseff are similar in nature[5], they appear on different hierarchy levels as shown in Figure 1.

The pace of news content generation and its dynamic nature make the real-time operation of summarization methods a key feature for reaching web users. Liu et al. [11] advocate for online operation both for allowing visualization of breaking news, and for enabling the delivery of consistent story development structures to the users. The real-world usability and near-real-time operation is a key design requirement for our algorithm.

4 SOCIALTREE EXTRACTION

We explain the proposed methodology for the generic case when a user issues a query q with a time period T resulting in a retrieval of ranked document collection $\mathbb{A}$.

We use superscripts for article indices and subscripts for tag indices. Each document $d \in \mathbb{A}$ is associated with a set of weighted tags $\mathbb{H}^d$ assigned either by a human (e.g., by the author of the document) or by a tool like *Hashtagger* [15]. The weights of tags in

[4]In our experiments we also demonstrate that the proposed method works with any kind of tags, e.g., keywords selected based on their TF-IDF profiles, as in related work.

[5] Both aspects represent country presidents. Jacob Zuma was the fourth President of South Africa who was a target of a failed impeachment attempt on 5 April 2016. Dilma Rousseff was impeached on 31 August 2016 while serving as the 36th President of Brazil.

$\mathbb{H}^d$ represent their contribution to the article and add up to 1, i.e. $\sum_j w_j^d = 1$, where w_j^d is the weight of the j^{th} tag assigned to the document d.

Article Headline	Tags	Relevance Score
Brazils congress set to impeach president Rousseff	{#rousseff: 0.45, #brazil: 0.41, congress: 0.24}	0.134
Zuma survives impeachment vote in South Africa	{#impeachzuma: 0.41, #impeachment: 0.59}	0.071

Table 2: Example documents retrieved with q ="impeachment" for input to the algorithm.

Each document $d \in \mathbb{A}$ has a query relevance rel^d computed using the BM25 ranking function [12] and is independent from the document's publishing time. Two example documents retrieved with the "impeachment" query are shown in Table 2. The collection $\mathbb{A}$ contains a big number of non-relevant articles that may bias the summary towards a tangent or even a completely irrelevant (sub)story. For example, the *"Kim Jong-Nam assassination"* query may retrieve a large number of articles about Kim Kardashian and Luuk de Jong. This problem can often be solved using entity disambiguation, as it is often done in related work. However, this may result in performance difference between the queries that contain named entities and the ones which do not. Entity disambiguation also creates a dependency on a knowledge base and potentially slows down the summary extraction process. Moreover, in the fast-changing domain of news, the knowledge base must be kept up to date to allow comparable performance on breaking news.

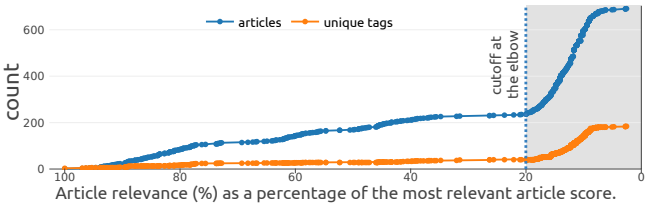

Figure 2: Filtering the irrelevant documents in the "impeachment" story selecting 236 out of 691 articles, which reduced the number of unique tags from 276 to 40.

Figure 2 shows the cumulative distribution of the retrieved document scores for the "impeachment" query, normalized to the most relevant document's score. The figure shows that two thirds of the documents are not even 20% as relevant as the document at rank one. We have observed a similar picture for all the other queries used in the study. In the work of [6, 7] it was shown that BM25 scores have a long-tail behavior and can be fitted a Gamma distribution, which confirms our expectations of a large number of very low relevance documents retrieved for each query. We therefore consider the documents beyond the relevance curve elbow, to be not relevant to the query. To select the cutoff point, we calculate the point on the curve that is the farthest from the line connecting the first and the last articles on the relevance curve. This approach offers a reasonable trade-off between the core and the tangent articles of the story. We denote the set of relevant documents as $\mathbb{D}$.

The relevance scores of articles $d \in \mathbb{D}$ are normalized, i.e., $\sum_d rel^d = 1$ and represent the contributions of the articles to the story defined by q. We denote by $\mathbb{D}_h$ the subset of $\mathbb{D}$ comprised of all the articles which had the tag h assigned to them. We call a set of two or more tags $\mathbb{H}_c$ co-occurring in the set of documents $\mathbb{D}_{\mathbb{H}_c}$ if every tag in $\mathbb{H}_c$ has been assigned to every document in $\mathbb{D}_{\mathbb{H}_c}$.

The coverage period L_h of a tag h is the period between the first and the last appearance of h in $\mathbb{D}$. The life duration $|L_h|$ is the number of days in L_h. $\mathbb{T}_h$ is the set of timestamps of articles in $\mathbb{D}_h$.

The pseudocode in Algorithm 1 presents the high-level overview of *SocialTree* extraction, which we elaborate in more detail further below in this section.

Algorithm 1 *SocialTree* extraction

Require: The query q with a time period T has retrieved a collection of ranked and tag-annotated articles $\mathbb{D}$

function EXTRACT_SOCIALTREE($\mathbb{D}$)

```
ℍ ← construct_story_profile(𝔻)                    ▷ Phase 1
ℝ ← extract_association_rules(𝔻)                  ▷ Phase 2
ℍ_pruned ← remove_duplicate_tags(ℍ, ℝ)            ▷ Phase 2
ℙ ← extract_frequent_patterns(𝔻)                  ▷ Phase 3
SocialGraph ← construct_graph(ℙ, ℍ_pruned)        ▷ Phase 3
SocialGraph ← reweight_edges(SocialGraph)         ▷ Phase 4
SocialTree ← MST(SocialGraph)                     ▷ Phase 5
root ← select_tree_root(SocialGraph)              ▷ Phase 6
```

4.1 Phase 1: Maximizing Compactness and Coverage — Constructing the Story Tag Profile

The socially augmented representation of keywords and hashtags, like any keyword representation, comes with a challenge that not all the tags are equally informative and useful. Hashtags are being created, popularized and abandoned by the crowds on social networks like Twitter. For relatively simple stories, often the crowd will converge to a few hashtags fully describing the event or the entity. For example, England vs Ireland rugby matches will be discussed with #ENGvsIRE, #coybig. For more complex stories with multiple events, locations or political stances, the set of hashtags may be very big. For instance, for the "US elections" story our dataset contains 1056 hashtags of various degrees of relevance including #imwithher, #hillary, #hillaryclinton, #neverhillary, #trump, #trumptrain, #trump2016, #maga, etc. In order to reduce the summary to a human-friendly size, we further filter the remaining tags.

In this phase we construct the story *profile*: a representative set of key entities, events and discussion topics extracted from the set of all tags in $\mathbb{D}$.

A tag is often being assigned to multiple articles. To quantify how much does h describe a story defined by query q and article collection $\mathbb{D}$, we sum the tag weights across $\mathbb{D}_h$, re-weighting the weights by the relevances of the corresponding articles.

$$rel_h = \sum_{d \in \mathbb{D}_h} \sum_{h_j^d = h} rel^d \times w_j^d ,$$

where h_j^d is the j^{th} tag of d. We normalize the tag relevances ($\sum_h rel_h = 1$) and interpret rel_h as the contribution of h to the story.

Our experiments have shown that the profile tag scores also follow a Gamma distribution. Thus we follow the same intuition for filtering the tags as we did with the articles, and remove the tags with scores beyond the L-curve's elbow. For the "impeachment" story, this step reduced the number of profile tags from 40

to 14. This filtering strategy offers a reasonable trade-off between significant reduction of the tag profile size and its high document coverage, because rel_h profile scores reflect both the frequency of tag assignment and also their association strengths with articles.

This tag relevance is used for *SocialGraph* edge re-weighting to embed the query-relevance in the story structure (Phase 4).

4.2 Phase 2: Reducing the Redundancy in the Tag Profile

Even after the lexical redundancy removal, a large number of relevant tags often create a situation where there are tags with (sometimes even fully) overlapping document coverage, e.g., #maga and #makeamericagreatagain. The spam tags (e.g., #BreakingNews) or tags weakly-related to the story (e.g., #CNN, BBC), although relevant, introduce noise from other stories. Also, the number of relevant tags is often large and is cognitively not acceptable for summaries[6].

In *KeyGraph* a small conditional co-occurrence of keywords is penalized by removing the corresponding edges. This results in omission of small, but yet relevant substories and in a high redundancy in the document representation. Instead, we remove all but one of the nodes (i.e., tags) with a conditional co-occurrence probability $\geqslant max\_rule\_conf$ considering these tag pairs redundant. We use association rules of tag co-occurrences in articles, extracted with the Eclat algorithm[7] [17] as an efficient way to compute the conditional co-occurrence of tags. From each of the redundant sets, only the tag with the highest profile score rel_h is kept. The rest are removed from the story extraction pipeline, and are added in the visualization as synonyms to the corresponding profile tags. The profile scores of the removed tags are added to the score of the remaining tag of the redundant set, in order not to alter the substory proportions in the story.

4.3 Phase 3: Frequent Pattern Mining for Graph Construction

We use frequent itemset mining with the Eclat algorithm[7] as an efficient and scalable method for counting co-occurrences of tags. Similar to *KeyGraph* and *Story Forest*, the document collection is represented as a graph of tags. The graph nodes correspond to the tags in the profile $\mathbb{H}_{pruned}$ (as noted in Algorithm 1), an edge between two tags marks the co-occurrence of those tags in at least min_cooc_freq documents, and the weight of the edge is equal to the number of documents where the pair of tags co-occurs. We call this graph a *SocialGraph*. Co-occurrence of tags captures their contextual relation. The *SocialGraph* is constructed from the frequent patterns of length two, i.e., frequent pairs of tags[8]. Our findings show that in the dense representation which includes hashtags, a minimum required co-occurrence frequency of $min\_cooc\_freq = 1$ yields useful and interpretable patterns and substory relationships.

[6] Keywords extracted from news articles have similar challenges associated with them. Nevertheless, the number of existing words is significantly larger, and, compared to hashtags, the words have less specific meanings.

[7] We use the Eclat implementation by Christian Borgelt http://www.borgelt.net/eclat.html.

[8] The frequent co-occurrences of three or more tags correspond to cliques in the *SocialGraph*, which can be used for an alternative (non-iterative and efficient) pruning of edges and merging of nodes.

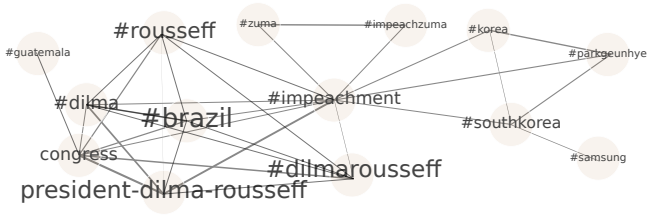

Figure 3: The *SocialGraph* for the "impeachment" story.

4.4 Phase 4: Maximizing Informativeness

The *SocialGraph* constructed from frequent pairs of tags has edge weights equal to the number of articles, where the two tags co-occur. However, these weights do not represent the tag relatedness correctly due to the following reasons. First, different substories often get unequal coverage in news media. This affects the popularity-based co-occurrence patterns of tags, thus, making them unreliable for representing the substory relationships correctly. Second, highly overlapping document collections may be retrieved with related, but different queries bound to the same time period. This can be seen as different points of view on the same data or as an expression of a higher interest towards certain aspects of the story by the user.

Each edge of the graph is re-scaled proportional to the *importances* and inversely proportional to the temporal distance of its ends, in order to compensate for the possible biases in tag co-occurrence patterns. Both the longevity and the query-relevance of a substory characterize its *importance*, which we define as follows:

$$importance_h = rel_h \times |L_h| \ ,$$

where rel_h and $|L_h|$ are correspondingly the query relevance and the life duration of tag h.

To optimize the resulting hierarchical structure for our *informativeness* needs, we incorporate the temporal information, query-relevance information in addition to the contextual co-occurrence information in the *SocialGraph* edge weights, prior to the hierarchical structure extraction. As a result, the weight of an edge connecting tags h_1 and h_2 in *SocialGraph* becomes

$$W_{h_1,h_2} = |\mathbb{D}_{\{h_1,h_2\}}| \times \frac{importance_{h_1} \times importance_{h_2}}{|importance_{h_1} - importance_{h_2}|} \times \frac{1}{|median(\mathbb{T}_{h_1}) - median(\mathbb{T}_{h_2})|} \ ,$$

where $|\mathbb{D}_{\{h_1,h_2\}}|$ is the number of articles that both h_1 and h_2 are assigned to, $median(\mathbb{T}_{h_1})$ and $median(\mathbb{T}_{h_2})$ are the timestamps of the middle articles of $\mathbb{D}_{h_1}$ and $\mathbb{D}_{h_2}$ respectively.

The intuition behind this is that the less important edges of weakly related tags would be broken easier by the hierarchy extraction algorithm and the more relevant edges will be kept instead. The denominators encourage connectivity between the tags belonging to the same substory. As a result, the tags that are more relevant to the query and more often assigned to the relevant articles will be higher in the hierarchy. At the same time, the enforced inter-substory connections will allow organization of the more specific tags closer to the hierarchy leaves. Note that this optimization will not change the size and the content of the summary, but may significantly alter its edges and hierarchical ordering of the tags.

4.5 Phase 5: Spanning Tree Extraction

To extract the most informative tree from the *SocialGraph*, we need to select the subset of edges that maximizes the total importance carried in edge weights. A maximum spanning tree (MST) is a subset of the edges of a connected, edge-weighted undirected graph that connects all the vertices together, without any cycles and with the maximum possible total edge weight. We use the widely adopted Kruskal's algorithm [9] for MST extraction. A single news story given by query q may result in multiple trees (a spanning forest) if the graph is not connected.

4.6 Phase 6: Tree Root Selection

As discussed previously in Section 3, our goal is to extract a hierarchy, where the more generic tags would appear higher in the tree. Our root selection strategy follows the intuition that the more generic an aspect of the story (represented as a tag) is, the more connections it is likely to have with more specific aspects of the story. A graph centrality measure, like betweenness centrality [3], can be used to select the most central tag as a hierarchy root. The betweenness centrality for each vertex is the number of shortest paths that pass through the vertex. However, as we have already justified in previous sections, the tag connectivity alone, may present an unreliable view of the connectivity. To compensate for the bias in media coverage, we weight the betweenness centrality of each tag by the total weight of all its edges in *SocialGraph*, and select the node with the highest value as a root.

4.7 Algorithm Complexity

The construction of the tag profile (*Phase 1* in Figure 4) has $O(|\mathbb{D}|)$ complexity. The Eclat algorithm for frequent itemset and association rule mining (*Phase 2* and *Phase 3*) has a complexity of $O(|\mathbb{D}_{\mathbb{H}}| \sum_{k=1}^{l} \binom{|\mathbb{H}|}{k})$, where $\mathbb{D}_{\mathbb{H}}$ is the set of articles containing at least one tag from story profile $\mathbb{H}$, and l is the longest itemset length. In the worst case $l = |\mathbb{H}|$, but the algorithm allows to limit the search space to tag pairs, which makes the complexity of this step $O(|\mathbb{D}_{\mathbb{H}}| \times |\mathbb{H}|^2)$. The complexity of edge re-weighting is $O(|\mathbb{H}_{pruned}|)$. Betweenness centrality calculation requires $O(|\mathbb{H}_{pruned}|^3)$.

Our experiments show that on average $|\mathbb{D}|/|\mathbb{D}_{\mathbb{H}}| \sim 10$ in our dataset. Considering that after duplicate removal $|\mathbb{H}_{pruned}| < |\mathbb{H}|$, and $|\mathbb{H}| \ll |\mathbb{D}_{\mathbb{H}}|$ because of the initial greedy profile selection, the total complexity of our algorithm is $O(|\mathbb{D}|)$.

4.8 Comparison to the State-of-the-Art

Existing summarization methods suffer from sensitivity to keyword extraction, poor noise filtering and redundancy in the representation. In summary, the key differences of our algorithm with respect to state-of-the-art methods, aiming to address the above mentioned issues, are the following:

- We use socially augmented representations of documents.
- We perform a greedy story profile selection prior to other processing steps, like graph construction. This helps the noise reduction and algorithm efficiency.
- We use high-confidence association rules of tag co-occurrence to prune the story profile from duplicate tags. *SocialTree* removes the strong connections in the co-occurrence graph, which is an opposite strategy to *KeyGraph* and *Story Forest*, where the weak connections are pruned away instead.
- We use frequent pattern mining for efficient graph construction.
- *KeyGraph* and *Story Forest* perform the structure extraction in co-occurrence space alone, whereas in *SocialTree* the co-occurrence graph is projected to temporal and relevance axis, therefore eliminating the need of multiple passes in the algorithm.

The key steps of the algorithms are illustrated in Figure 4.

5 METHOD EVALUATION

We work with a dataset of news in English (UK-based and Ireland-based news media) collected over a 21-month period between August 2015 and May 2017. The dataset includes 290,657 news articles, of which 174,347 have at least one hashtag assigned to it by *Hashtagger*. *Hashtagger* [15] was chosen for its availability[9], the reported high precision and the capability of near-real-time operation, which is a design requirement for our algorithm. Only English-language tweets were used as a source for hashtags, nevertheless hashtags composed of any unicode characters (e.g., in Chinese, Arabic, Portuguese and other alphabets) were assigned to the news articles.

The news articles were preprocessed to meet the demands of all the methods evaluated in this study. After stopword removal and named entity recognition, the identified noun phrases from article headlines and contents were stemmed with Porter Stemmer. The stemmed noun phrases and named entities extracted with NLTK's classifier-based named entity recognizer were used to construct TF-IDF profiles of the articles for keyword extraction. The TF-IDF scores of named entities were boosted by a factor of two. We selected 5 keywords with the highest TF-IDF values to represent each news article. The same set of news articles was fed to each method. All the experiments have been performed on a computer with 16GB of RAM and Intel Core i7-4790 CPU to compare method runtimes.

We choose stories of different popularity and complexity, as shown in Table 3, to evaluate our method's performance for various story types. As we are evaluating the performance in a summarization context, we use the full temporal span of the dataset for all the stories. To showcase *SocialTree* performance on a short, local story, we have also included the "Pope Francis visit" story, covering a period from Sept. 15 to Oct. 1, 2015, which includes Pope's visit to Cuba and USA. All these stories and their summaries are available at https://gevra.github.io/socialtree/.

Query	Duration	Locality	Complexity	Articles
impeachment	long	global	low	236
Pope Francis visit	short	local	low	141
plane crash	long	global	high	979
earthquake	long	global	low	573
hurricane	long	global	high	274
Brazil	long	local	low	1344

Table 3: Characteristics of the stories chosen for evaluation.

As we have already discussed, *KeyGraph*[10], *MetroMap*[11] and *Story Forest*[12] are structured summarization methods which are the most similar to our work both in objectives and techniques. We

[9] https://github.com/gevra/hashtagger_plus_offline.
[10] http://keygraph.codeplex.com
[11] https://github.com/snap-stanford/MetroMaps
[12] https://github.com/BangLiu/StoryForest

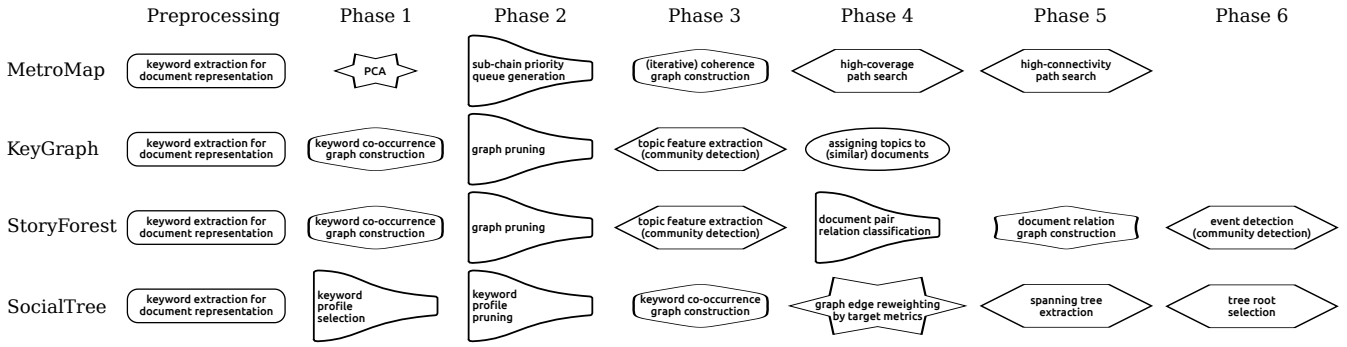

Figure 4: Processing phases of structured summarization algorithms (similar processing steps are shown in identical shapes).

also evaluate a word cloud with 25 keywords, as a weak summarization baseline for all the methods. The available implementation of *MetroMap* implements a variant of the algorithm described in the original paper. All the algorithm parameters were set to the recommended values in the original papers and were kept constant in all the experiments. The code for *Story Forest* was kindly provided by the authors, but despite our best efforts, we could not get it to produce any summaries.

Since our dataset does not contain ground truth to directly determine the precision and recall of story profile construction, we use the following criteria to evaluate structured summaries, based on the design goals previously described in the paper,:

- efficiency (runtime), scalability and ability to operate for both big and small stories,
- coverage (number of documents reachable from the summary),
- compactness (number of items in the summary),
- informativeness (ease of understanding/navigating the summary).

Similar to prior work [13], the **coverage** is defined as the percentage of documents in $\mathbb{D}$ reachable through profile tags, as *SocialTree* tags are seen as substories and the documents are already linked to these tags. We do not report on per-topic recall, since the documents in our dataset are already annotated with social tags representing the relevant story aspects. Similar to *MetroMap* and *KeyGraph*, the coverage of a *SocialTree* does not depend on the structure of the summary. **Compactness** is the size of the summary, i.e., the number of nodes in the summary graph. **Informativeness** is a metric capturing the ease of understanding, the ease of navigation and the diversity/redundancy of the substories. In this work we do not define a quantitative measure for informativeness, and only use it to discuss the helpfulness of summaries in our user study.

5.1 Quantitative Evaluation

In addition to comparing our method to the state-of-the-art, our goal is to find out whether the social tags allow better summaries than keyword-based document representations. To answer the first research question (RQ1), we compare our method to its two keyword-based equivalents with (i) keywords based on nouns and named entities and (ii) TF-IDF keywords used in other methods. We call these methods *KeywordTree* and *TFIDFTree* respectively. For the keyword-based trees, we use our algorithm with top 5 keywords with the highest TF-IDF scores instead of the assigned tags. The only parameter setting difference was $min\_cooc\_freq = 2$ due to a higher cardinality of the representation space.

Story	Method	Nodes	Clusters / Lines / Trees	Coverage (%)	Runtime (seconds)
impeachment (236 articles)	KeyGraph	100	8	26	7.1
	MetroMap	21	5	**97**	10.4
	TFIDFTree	24	2	89	5.4
	KeywordTree	18	1	89	4.2
	SocialTree	**14**	1	93	**3.6**
Pope Francis visit (141 articles)	KeyGraph	62	9	45	**1.3**
	MetroMap	11	3	69	4.3
	TFIDFTree	19	2	**72**	3.7
	KeywordTree	**8**	1	71	2.1
	SocialTree	10	1	62	2.9
plane crash (979 articles)	KeyGraph	443	30	30	23.6
	MetroMap	**34**	5	16	38.6
	TFIDFTree	73	6	67	25.0
	KeywordTree	48	1	78	**17.2**
	SocialTree	59	1	**82**	21.9
earthquake (573 articles)	KeyGraph	411	17	28	25.6
	MetroMap	33	5	25	19.1
	TFIDFTree	43	9	77	10.3
	KeywordTree	23	1	88	6.4
	SocialTree	**21**	1	**89**	**6.3**
hurricane (274 articles)	KeyGraph	381	14	46	53.2
	MetroMap	26	5	37	12.7
	TFIDFTree	58	3	**84**	11.4
	KeywordTree	30	3	83	7.0
	SocialTree	**23**	1	78	**5.4**
Brazil (1344 articles)	KeyGraph	506	30	28	28.2
	MetroMap	43	5	45	86.4
	TFIDFTree	100	5	75	33.4
	KeywordTree	38	1	**82**	17.4
	SocialTree	**19**	1	78	**9.6**

Table 4: Cross-story method comparison. Word clouds for all the stories had identical size of 25 terms (nodes), achieved at least 98% coverage and ran in less than two seconds. Coverage is the percentage of relevant documents $\mathbb{D}$ which are linked to at least a single substory/topic in the summary. The number of trees in our approach, the number of clusters in *KeyGraph* and the number of "metro lines" in *MetroMap* serve as a proxy for achievable compactness and the ease of navigation. Runtimes (seconds) are the averages over 10 runs and had <10% coefficient of variation.

We use runtime, summary compactness and document coverage to quantitatively compare the studied methods. Similar to related work, we report the summary size and the coverage, but do not report on precision due to the absence of a ground truth for summaries. Instead, the informativeness of summaries is assessed in the qualitative evaluation.

The experiment results are presented in Table 4. It can be seen that our algorithm's runtimes across different stories and different document representations are in accordance with its estimated

linear complexity. With the exception of the smallest story about "Pope Francis visit", our algorithm is consistently faster than *KeyGraph* and *MetroMap*. Moreover, the latter methods perform inconsistently across different story types. *MetroMap* struggles with complex stories and has a reasonable coverage and compactness for simpler stories ("Pope Francis visit" and "impeachment"). We found *KeyGraph* to be very sensitive to parameters. Its summaries are the biggest for all the stories and never achieve 50% coverage. Compared to state-of-the-art, our methods consistently produced more compact summaries without compromising coverage. The only exception was the "plane crash" story, where the *MetroMap* summary, although compact, had only 16% coverage. *SocialTree*'s advantage over *KeywordTree* was more evident for the very large, yet relatively low-complexity "Brazil" story, where the compactness was achieved thanks to the low redundancy of the social representation. Among the hierarchical summaries, *TFIDFTree* was always bigger in size. The experiments show that social augmentation of document representations can be advantageous across all story types, e.g., "plane crash" and "hurricane" consist of a dozens of small substories, whereas "Brazil" and "earthquake" have fewer and bigger substories.

Our proposed method, *SocialTree*, outperforms the state-of-the-art w.r.t. to compactness, coverage and runtime, and is much more efficient. To evaluate the informativeness of these summaries, we performed a user study on the same set of stories.

5.2 Qualitative Evaluation via a User Study

We conducted a user study to answer the second research question (RQ2) and to assess the informativeness of *SocialTree* by comparing it to state-of-the-art methods. All summaries except for *Wordcloud* are interactive, allow zooming and have tooltips on hovering for nodes (showing document coverage, time period, etc.) and edges (showing document coverage overlap of nodes). We enlisted 12 reviewers including postgraduate students, academic staff and software developers, to blindly evaluate the summaries generated by different methods. The reviewers were given instructions on how to navigate each summary prior to taking part in the study. Each story was introduced by a set of relevant Wikipedia articles to allow the reviewer to verify his/her knowledge on the story if needed. Providing more elaborate abstracts for the stories was unnecessary for the following reasons: (i) we want to evaluate the summaries w.r.t. the coverage in the respective document collection and not w.r.t. the complete knowledge, (ii) the stories like "plane crash" are too big for a human to remember everything even if the person is aware of all the aspects of the story, (iii) in the context of comparing multiple methods, the reviewers can assess the relative informativeness of each summary. Each individual story was summarized by each of the six methods, and all summaries were reviewed by all the reviewers. Each method was assessed 72 times across 6 stories and by all 12 reviewers for each of the 5 assessment statements. As a result, each method has received 360 assessments. For each summary, the reviewers were asked to assess their agreement with the following statements on a Likert scale (*strongly disagree*, *disagree*, *neither agree nor disagree*, *agree*, *strongly agree*):

1. The story focus (query) can be easily inferred from the summary.
2. The summary correctly highlights the key entities of the story.
3. The key temporal events of the story stand out in the summary.
4. The paths/links in the summary are logically coherent[13].
5. The parent nodes correctly express the most relevant common concepts shared by their child nodes[13].

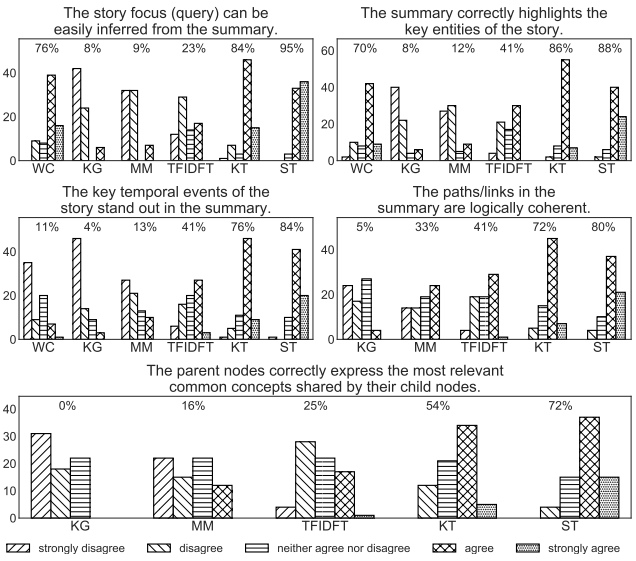

Figure 5: Distribution of user study reviews on a Likert scale for each of the methods. The method names on horizontal axis are abbreviated as follows: Wordcloud↦WC, KeyGraph↦KG, MetroMap↦MM, TFIDFTree↦TFIDFT, KeywordTree↦KT, SocialTree↦ST. The number above each method barplot is the percentage of reviewers who agree or strongly agree with the statement. Right skew in the distribution is better.

The reviews are summarized in Figure 5 from which we can draw the following conclusions. For the stories in this experiment, the reviewers found the links between the expected story focus and the summaries produced by *KeyGraph* and *MetroMap* to be weak. The story focus was not easily inferable from the *TFIDFTree* summaries, which can be explained by the absence of the main keywords in TF-IDF representations, as reported by the reviewers. The TF-IDF representation is not suitable for co-occurrence-based methods, as it often does not include the common aspects in query-retrieved document collections. The *Wordcloud*, *KewordTree* and moreover *SocialTree* presented a descriptive information of the story. An interesting finding is that despite *Wordcloud* not having a temporal dimension and linkage information, the reviewers found it to highlight the events better than the state-of-the-art methods do. The reviewers showed > 70% agreement (answered *agree* or *strongly agree*) with all the statements referring to *SocialTree*.

Also, for each of the stories, the reviewers were asked to select the summary that they found the most informative. The reviewer responses are summarized in Figure 6 and show that 67% of reviewers found *SocialTree* to be the most informative. Interestingly, the reviews preferring *KeywordTree* over *SocialTree* came from multiple stories. The reviewer preference between these methods was not related to the compactness or to the characteristics of the story.

[13]This question was omitted for Wordcloud which does not have any links.

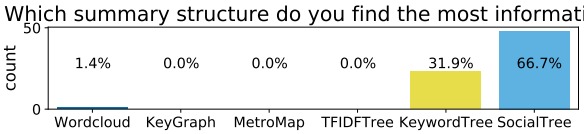

Figure 6: Number of times a summary was selected by a reviewer as the most informative for the given story over 72 assessments (across 6 stories and 12 reviewers).

The user study allowed us to conclude that our non-parametric method produces informative news story summaries where the state-of-the-art methods fail. The results show that inclusion of hashtags in the document representation enables our algorithm to achieve a clear advantage w.r.t. keywords-only summaries. At the same time, we identified types of stories where the socially augmented summaries may be at a disadvantage. The results of the user study demonstrate that the informativeness of summaries produced by our method comes both from the novel document representation and from the hierarchical structure. We attribute the efficiency of our system to the scalable profile construction prior to the structure extraction step, as well as to operating in a dense representation space. Our algorithm's ability to cope with noise can be attributed to the robust profile construction and the subsequent structure pruning steps. At the same time, we explain the weak performance of *KeyGraph* and *MetroMap* by their need of parameter tuning for each different story.

5.3 Bias in the Document Collection

The experiments presented in this paper were conducted on a multi-source dataset. Uneven coverage across multiple news sources may result in an importance boost of certain substories and alter the extracted summaries. We were also interested in the effect of geographical and topical bias of a news source on the resulting summaries. For this purpose we have tested our method on the NIST TREC 2018 News Track collection[14] of The Washington Post articles, which offers an American perspective on news. The summaries produced with our method on The Washington Post corpus had only minor differences from the corresponding story summaries from our original dataset. The results obtained from a different news corpus further support our claim about the robustness of our method to document wording and demonstrate the advantage of our algorithm's non-parametric nature. These summaries are not presented in the paper due to space limitations, but the readers are encouraged to explore them on the paper's GitHub page.

6 DISCUSSION AND FUTURE WORK

Our experiments have shown that state-of-the-art methods for structured summarization are not fit for purpose. In addition to having poor scalability, high sensitivity to keyword selection and requiring extensive parameter tuning, the summaries produced by these methods had poor informativeness even compared to a word cloud. We presented an algorithm for hierarchical summary extraction from news articles, that addresses the weaknesses of the existing methods. It leverages the connections of social tags, making the summaries less sensitive to the keyword extraction algorithm and at the same time minimizing the loss in document coverage. Our method organizes story aspects in a hierarchy, yielding compact and comprehensive news summaries. The results of the user study demonstrate that the informativeness of summaries produced by our method comes both from the novel document representation and from the hierarchical structure. Another strength of our method is its non-parametric nature, which enables consistent performance across stories of different complexity and popularity.

In the future, we are planning to (i) extend the experiments to different datasets including one with human-tagged news articles (e.g., The New York Times editor-annotated news), (ii) assess the algorithm robustness to the changes in the input data (e.g., for input subsampling or newcoming articles), (iii) test our method on very long range (e.g., decades long) news stories, since no suitable method exists for such stories. We plan to test our algorithm on domains beyond news, including blogs, recipes and scientific papers. We also think that our method can be extended to other hierarchical structures, e.g., core-periphery networks instead of trees.

The data, the code and interactive summaries are available at https://gevra.github.io/socialtree/.

[14] https://trec.nist.gov/data/wapost/

ACKNOWLEDGMENTS

This work was funded by Science Foundation Ireland (SFI) under grant number 12/RC/2289_P2. The authors would like to thank Eileen Culloty and the reviewers for their helpful feedback.

REFERENCES

[1] Amr Ahmed, Qirong Ho, Choon H Teo, Jacob Eisenstein, Eric P Xing, and Alex J Smola. 2011. Online inference for the infinite topic-cluster model: Storylines from streaming text. In *International Conference on Artificial Intelligence and Statistics*.
[2] Ronaldo Florence, Bruno Nogueira, and Ricardo Marcacini. 2017. Constrained Hierarchical Clustering for News Events. In *Proceedings of the 21st International Database Engineering & Applications Symposium*. ACM, 49–56.
[3] Linton C Freeman. 1977. A set of measures of centrality based on betweenness. *Sociometry* (1977), 35–41.
[4] Lei Hou, Juanzi Li, Zhichun Wang, Jie Tang, Peng Zhang, Ruibing Yang, and Qian Zheng. 2015. NewsMiner: Multifaceted news analysis for event search. *Knowledge-Based Systems* 76 (2015), 17–29.
[5] Ting Hua, Xuchao Zhang, Wei Wang, Chang-Tien Lu, and Naren Ramakrishnan. 2016. Automatical Storyline Generation with Help from Twitter. In *CIKM*.
[6] Evangelos Kanoulas, Keshi Dai, Virgil Pavlu, and Javed A Aslam. 2010. Score distribution models: assumptions, intuition, and robustness to score manipulation. In *SIGIR*. ACM, 242–249.
[7] Evangelos Kanoulas, Virgil Pavlu, Keshi Dai, and Javed Aslam. 2009. Modeling the score distributions of relevant and non-relevant documents. *Advances in Information Retrieval Theory* (2009), 152–163.
[8] Andrew Kehoe and Matt Gee. 2011. Social Tagging: A new perspective on textual "aboutness". *Studies in Variation, Contacts and Change in English* 6 (2011).
[9] Joseph B Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical society* 7, 1 (1956), 48–50.
[10] Gregor Leban, Blaz Fortuna, and Marko Grobelnik. 2016. Using News Articles for Real-time Cross-Lingual Event Detection and Filtering. In *ECIR*.
[11] Bang Liu, Di Niu, Kunfeng Lai, Linglong Kong, and Yu Xu. 2017. Growing Story Forest Online from Massive Breaking News. (2017).
[12] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. 2008. *Introduction to information retrieval*. Vol. 1. Cambridge university press Cambridge.
[13] Hassan Sayyadi and Louiqa Raschid. 2013. A graph analytical approach for topic detection. *ACM Transactions on Internet Technology (TOIT)* 13, 2 (2013), 4.
[14] Dafna Shahaf, Carlos Guestrin, and Eric Horvitz. 2012. Trains of thought: Generating information maps. In *WWW*.
[15] Bichen Shi, Georgiana Ifrim, and Neil Hurley. 2016. Learning-to-rank for real-time high-precision hashtag recommendation for streaming news. In *WWW*.
[16] Siliang Tang, Fei Wu, Si Li, Weiming Lu, Zhongfei Zhang, and Yueting Zhuang. 2015. Sketch the storyline with charcoal: a non-parametric approach. In *IJCAI*.
[17] Mohammed Javeed Zaki. 2000. Scalable algorithms for association mining. *IEEE transactions on knowledge and data engineering* 12, 3 (2000), 372–390.
[18] Deyu Zhou, Haiyang Xu, Xin-Yu Dai, and Yulan He. 2016. Unsupervised Storyline Extraction from News Articles.. In *IJCAI*. 3014–3021.