

Dependency graph for short text extraction and summarization

Nigel Franciscus, Xuguang Ren & Bela Stantic

To cite this article: Nigel Franciscus, Xuguang Ren & Bela Stantic (2019): Dependency graph for short text extraction and summarization, Journal of Information and Telecommunication, DOI: [10.1080/24751839.2019.1598771](https://doi.org/10.1080/24751839.2019.1598771)

To link to this article: <https://doi.org/10.1080/24751839.2019.1598771>



© 2019 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group



Published online: 05 Apr 2019.



Submit your article to this journal [↗](#)



Article views: 541




View related articles [↗](#)



View Crossmark data [↗](#)



Dependency graph for short text extraction and summarization

Nigel Franciscus , Xuguang Ren and Bela Stantic

Institute for Integrated and Intelligent Systems, Gold Coast, Australia

ABSTRACT

A sheer amount of text generated from microblogs and social media brings huge opportunities to the text mining applications. Many techniques such as sentiment analysis and opinion mining are proven effective to deliver insights from documents. However, most of these textual data are in the form of short and fragmented texts which are difficult to visually extract due to the sparsity issue and the context in the content is often unknown. Naive while widely used models, term frequency and the bag-of-words never considered the semantic relationship between the words, making the results relatively difficult to interpret. A well-known technique in text mining like topic model may provide a general 'at glance' understanding but can be difficult to interpret or to understand. One alternative is to aggregate words in a semantical order and generates an output of human-understandable sentences. In this paper, we address this direction by proposing the belief graph data model that joins short texts by inducing the part-of-speech tagging to maintain the order and to preserve the context of the content. Extensive experiments showed that our approach improves the overall qualitative evaluation of text understanding compared to the previous state of the art text mining techniques.

ARTICLE HISTORY

Received 27 June 2018

Accepted 20 March 2019

KEYWORDS

Graph-of-word; text representation; short text; part of speech

1. Introduction

In the era of big data, a tremendous amount of fragmented short texts is growing at a scale that makes them impossible for the human to visually extract. The information overload is creating redundant and duplicates textual data, making it difficult to quickly understand information. As an illustration, in [Figure 1](#) we can quickly scan the news headlines but it will take some time to understand the underlying event. For example, on the left side, in the Great Barrier Reef news, we might want to know why does it struggle and what was the cause of the funding; and on the right side, we might want to know what is the overall summarization of public opinion. Summarizing the details of the content can save users a significant amount of time. Many data mining applications have been developed to help understand and to summarize these collections of texts, for example, topic modelling (Blei, Ng, & Jordan, 2003) and text summarization (Mihalcea & Tarau, 2004).

CONTACT Nigel Franciscus  n.franciscus@griffith.edu.au, nigel.franciscus@griffithuni.edu.au

© 2019 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group

This is an Open Access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

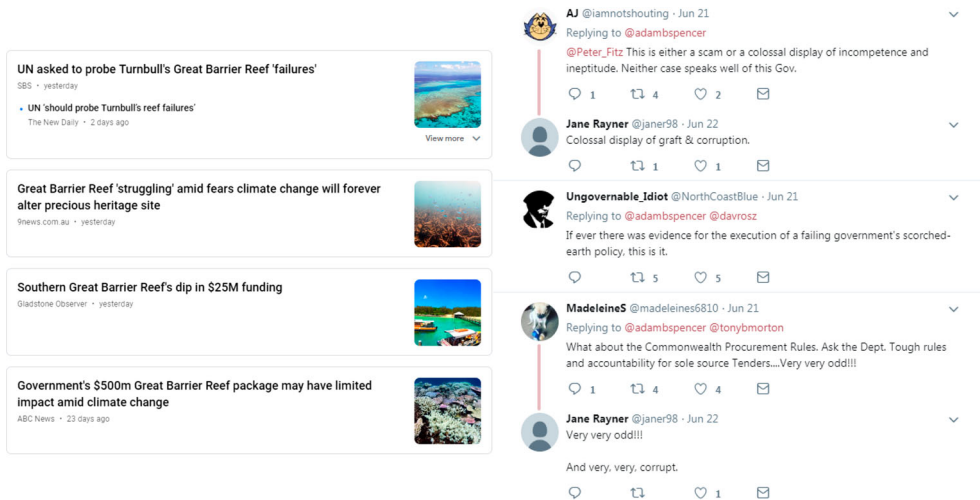


Figure 1. Snippet of Google News and comments on Twitter.

However, these techniques emphasize the generalization which failed to capture the underlying context to concrete a profound summary.

A naive while widely used model is term-frequency or unigram which naturally presents hot keywords, where the topic is represented by frequent keywords. By only providing some keywords, unigram is far too limited. Bi-gram extends the unigram by capturing the relationship of two words based on the co-occurrence. Later, n-gram complements the words identification into a phrase with part-of-speech tagging. However, the n-gram is modelled in the bag-of-words form, which does not consider the word order in the sentence. One option is to organize those keywords according to the grammatical order and generate an output similar to the human-understandable sentences. This is a challenging task since it combines part-of-speech processing and data mining for summarization. In this paper, we conduct the empirical evaluation of this direction by using the *belief graph* model.

In order to build the belief graph, we are combining the text mining with graph linguistic approach, which has been the focus of multiple lines of research (see e.g. Mihalcea & Tarau, 2004; Sayyadi & Raschid, 2013; Scaiella, Ferragina, Marino, & Ciaramita, 2012; Zhang, Wang, Xu, & Hu, 2013; Zuo, Zhao, & Xu, 2016). At the same time, there has been growing interest in adopting dependency parses tree (De Marneffe, MacCartney, & Manning, 2006) for a range of Natural Language Processing tasks, from machine translation (Deutch, Frost, & Gilad, 2017; Li & Jagadish, 2014) to question answering (Amsterdamer, Kukliansky, & Milo, 2015). Here, we specify the dependency parses tree simply as the dependency tree. Dependency tree parses the grammatical relation to capture dependencies between words such as predicate-argument. We will use the term dependency tree and tree interchangeably.

In this paper, we extend our previous work (Franciscus, Ren, & Stantic, 2018a) and present a scheme to build the belief graph by utilizing the graph dependency architecture. Our objective is by using the dependency tree, we are able to store the intermediate result of text processing in a graph database which further enables extensive graph queries such

as centrality and clustering. The basic idea of our architecture is to simplify the process by utilizing state of the art language processing to enable graph computation over a large amount of text. To be specific, we provide the following contributions:

- We improve the proposed reversed dependency to convert raw short text into a graph model by preserving grammatical properties and then reuse these properties to generate the synopsis of text.
- We designed a practical schema and storage structure to build belief graph by using NoSQL databases, MongoDB and Neo4j. The recent update in the database version improves the writing performance significantly.
- We performed comprehensive experiments to demonstrate the system performance and practical usage of graph queries in comparison with other well-known text mining and text summarization techniques.

The rest of the paper is organised as follows: in Section 2 we present some related works; in Section 3, we present the details of belief graph definition and construction; in Section 4, we provide the experiment results; in Section 5 we discuss our interpretation of the experiments and finally in Section 6 we conclude the paper and indicate the future work.

2. Related work

In this section, present some related work in-line with the purpose of our contribution which can be categorized into several classifications.

- (1) *Natural Language Dependency Tree* The rising of NLP processing with the dependency tree has captured the interest of research communities with the availability of NLP toolkit (Manning et al., 2014). Two recent prominent works utilizing the natural language to interpret (Li & Jagadish, 2014) and answering (Deutch et al., 2017) query from relational database. Another work also uses dependency tree for crowd-sourcing platform (Amsterdamer et al., 2015). Our work is similar to these concepts, however, we are targeting directly into text mining without involving SQL queries and focusing on graph approach.
- (2) *Topic Modelling* Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is a well-known topic modelling technique which can be used to discover the latent information from the document. Several variances of LDA have been proposed to alleviate short and sparse text processing. Our work differs in the way we preserve the word order in contrast to using bag-of-words assumption which does not consider word order. Our work also uses a graph instead of a sampling method to obtain keywords. Another strategy is to restrict the document-topic distribution so that each short text is sampled from a single topic (single topic per document), this is known as Dirichlet Multinomial Mixture (DMM) or mixture of unigrams model (Yin & Wang, 2014). Given the limited content in a short text, this strategy is reasonable and it temporarily alleviates the data sparsity problem. The third strategy is to explicitly incorporate word co-occurrence information. For instance, modelling word co-occurrence patterns (Yan, Guo, Lan, & Cheng, 2013) and using soft-clustering mechanism for further word co-occurrence augmentation (Quan, Kit, Ge, & Pan, 2015).

- (3) *Graph-based model* Graph model offers a simple, effective and interactive representation based on the relation between each vertex. Each vertex can be treated as an entity of sentences, words or characters with the edges as the label or property of the relationship between two vertices. The fundamental of the graph is based on the idea of KeyGraph (Ohsawa, Benson, & Yachida, 1998) which converts text into a term graph based on co-occurrence relations between terms. The main strength of graph representation lies in the connection between the entity, enable users to query the relationship between keywords. Further, graph-based models explicitly consider word co-occurrence as one of the main parameters to determine the score of a keyword (Sayyadi & Raschid, 2013; Zhang et al., 2013). Contrast to previous work, we focus specifically on the dependency tree to capture the grammatical relationship between keywords.
- (4) *Word Embedding*. Recently, word embedding growth is intensified with the development of vector representation of global words (Mikolov, Sutskever, Chen, Corrado, & Dean, 2013; Pennington, Socher, & Manning, 2014). Word Embedding can capture semantically similar words and word analogy (e.g. woman + king – man = queen) based on the position of words in the sentence. Words are mapped into a large dimension vector space using the skip-gram or the continuous bag-of-words (CBOW) model. However, it often requires a large number of corpus (millions number of words) to achieve a satisfactory performance since the word context is heavily relying on the surrounding words (Grave, Mikolov, Joulin, & Bojanowski, 2017; Joulin et al., 2016). Our work focuses specifically on the part of speech to explicitly capture the context based on the word order.

3. Belief graph

In this section, we present the definitions and characteristics of two classes of knowledge base model, knowledge graph and belief graph. The belief graph follows knowledge graph intuition for information linking. Before we describe the construction of the belief graph, we outline its predecessor knowledge graph and contrast the differences between the two of them.

3.1. Knowledge graph vs. Belief graph

Knowledge graph has been the focus of research since the introduction of the Google Knowledge Graph in 2012.¹ In practice, knowledge graph has been known as a complementary for many applications. For example, assisting question answering (Chen, Fisch, Weston, & Bordes, 2017), providing auxiliary information for information retrieval (Li, Wang, Zhang, Sun, & Ma, 2016) and inferring new knowledge from its own collection (Song, Wu, & Dong, 2016). However, it does not have a valid definition and therefore prone to multi-interpretations. Here, we compose the definition and characteristics of a knowledge graph.

Definition 3.1 A knowledge graph G is a directed labelled graph (V, E, L) , where V is a set of nodes, and $E \subseteq V \times V$ is a set of edges. Each node $v \in V$ represents an entity with label $L(v)$ and each edge $e \in E$ represents a relationship $L(e)$ between the two entities.

Characteristics:

- *Atomicity*. Each statement has a single interpretation which refers to a subject–object relationship over a specific domain. For example, Shakespeare died on 23 April 1616 in Warwickshire, England. It has an exact and precise meaning.
- *Factoid Entities*. The content is generated from a human-curated knowledge base which contains a well-known real-world fact. It covers the broad domain of human-life aspect. For example, Wikipedia and Freebase.
- *Triples*. The subject–object relationships in a common knowledge graph are often presented in the triplets form such as ‘is-a’ relationship (e.g. Apple is-a Fruit). Since a knowledge graph carries a lot of information, nodes and edges content may have a name, type, and attribute represented as the label. For example:

$$Shakespeare_{(Person)} \xrightarrow[23April1616]{died} Warwickshire_{(county)}, England_{(country)}$$

On the other hand, although belief graph follows a similar schema with knowledge graph, it has significant differences in its content and source. It does not represent the grounded facts such as real-world entities and it is more focusing on the word-to-word relationship. Belief graph can be used to summarize unstructured and unlabelled text which further can be extended into topic modelling, text summarization, and text visualization. We present the definition and characteristics of belief graph as follows:

Definition 3.2 A belief graph G is a directed labelled graph (V, E, L, P) , where V is a set of nodes, and $E \subseteq V \times V$ is a set of edges. Each node $v \in V$ represents an entity with label $L(v)$ and has properties $P(v) = id, t$ where id is the identifier and t is the grammatical property type. Each edge $e \in E$ represents a relationship label $L(e)$ between two entities and has a property $P(e) = f$ where f is the co-occurrence frequency. A belief graph is formed from multiple words-to-words relations derived from sentences.

Characteristics:

- *Ambiguous*. The network can be constructed from any text regardless of the length or size of the document, thus each connection in the belief graph is prone to multi-interpretation. Unlike the triplets in the knowledge graph, belief graph takes into account all possible relationships which will be ranked according to their importance.
- *Non-Factoid Entities*, generated from a collection of unstructured and unlabelled opinion-based text where the ground-truth identity is unknown. For example, social media and product review data. Thus, the summary generated can be a fact or public opinions.
- *Ordered*. Since belief graph is using the dependency parser, it automatically preserves the word order of linguistic form including active and passive voice by using the grammatical information. Thus, it is relatively straightforward to identify the subject–object relationship. For example,

$$LeoDicaprio_{(nsubj)} \xrightarrow[isa(n)]{} Environmentalist_{(noun)}.$$

3.2. Building belief graph

In this section, we give an overview of our belief graph system as shown in [Figure 2](#). It can be divided into several inter-related components: (i) *Initial preprocessing*, where we collect the raw textual data and store them into MongoDB. (ii) *Dependency tree*, where we map a collection of text to the dependency parser and translate them into a grammatical structured tree. (iii) *Dependency tree join*, where we merge each dependency tree into a belief graph. As a case study, we demonstrate our belief graph system by using Twitter and Amazon product review data sources and using NoSQL database platforms.

3.2.1. Initial preprocessing

An efficient storage architecture is essential to manage raw data and handling preprocessing. Our initial preprocessing is primarily built based on the precomputing architecture (Franciscus, Ren, & Stantic, 2018b), where we classify unstructured data into manageable periodic content which can be drilled down or rolled up to support time-slicing query. However, in this case, we use the seed terms as the parameter to classify the collection. Hence, we index the collection level according to filtered key terms. Unlike normal text preprocessing, we do not necessarily have to tokenize or stem the words. Firstly, due to the process of Part-Of-Speech (POS) tagging, the system needs to capture the original sentence including punctuations, and secondly, when we capture the grammatical relation the system requires the connection between words which cannot be stem into their root. However, the basic preprocessing such as ASCII character formatting is required to remove uninterpreted emoticons or symbols.

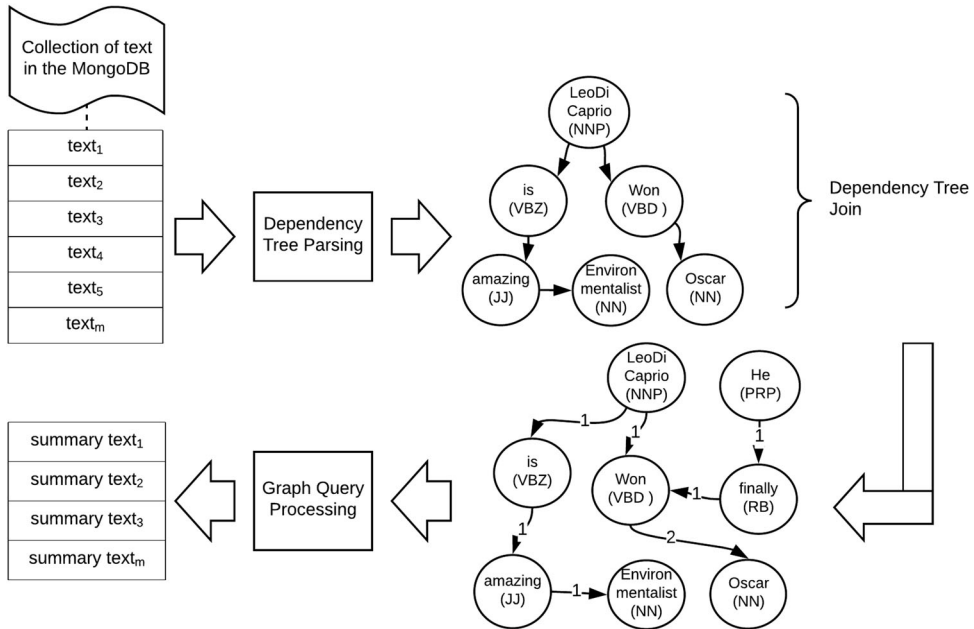


Figure 2. System overview of the graph dependency analytic engine.

3.2.2. Dependency tree

The human linguistic structure still plays a major role in determining the context of any given set of words. In this instance, we borrow the model of the linguistic structure from the Penn TreeBank which is normally used to structure English sentences. Specifically, the dependency tree is a linguistic tree generated from the typed dependency parses of English sentences from the phrase structure parses. In order to capture inherent relations occurring in corpus texts, we use the noun phrase (NP) relations which are included in the set of grammatical relations used (De Marneffe et al., 2006). In here, typed dependencies and phrase structures have a different way of representing the structure of sentences, while a phrase structure parse produces nesting of multi-word constituents, a dependency parse produces dependencies between individual words. A typed dependency parses further labels the dependencies with grammatical relations, such as subject or indirect object. The tree structure can be found in the original paper (De Marneffe et al., 2006).

We use dependency tree as the words segmentation process which is a process to identify the meaning of a word beyond the co-occurrence with other words. During the parsing stage, the system seeks to understand the meaning of different word relationships according to the sentence flow. We define the dependency tree as the key concept of natural language from Stanford NLP parser. In particular, the dependency tree is considered as:

Definition 3.3 A dependency tree $T = (V, E, L)$ is a node-labelled tree where labels consist of (1) Part of Speech tagging: the syntactic role of the word, and (2) Relationship (REL): the grammatical relationship between words according to their associates in the dependency tree.

Each vertex ($v \in V$) represents word (term) while each edge ($e \in E$) represents a grammatical structure in a sentence. Each vertex has a POS-tagging type t property. This property distinguishes each word according to its classification. For example, in Figure 3 on the sentence 'Marvel comic has Superman and Batman', the word *Marvel*, *Superman*, *Batman* belong to PROPER NOUN (NNP) family while *comic* belong to NOUN (NN) family.

Another example, given two sentences 'LeoDiCaprio is an amazing environmentalist. He finally won an oscar' in Figure 3, the parser will indicate subject-object when predicate (VBZ and VBD) exist. In this way, we can virtually answer that *LeoDiCaprio* is an *Environmentalist* through the NN indicator while he also *Won* an *Oscar* via predicate-object relationship. Once we obtain the property value for each vertex, we build the belief graph as an extension of the dependency tree. Belief graph can be seen as the network of dependency tree collection. We discuss how to join the dependency tree further in the next section.

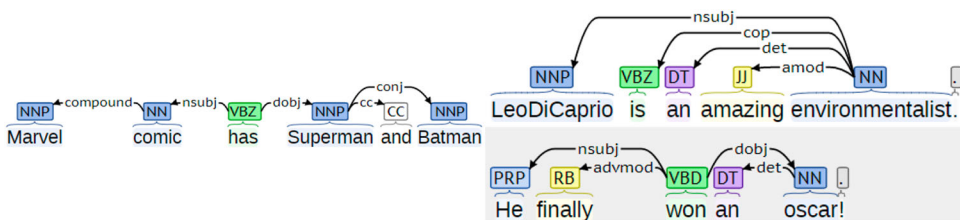


Figure 3. Dependency parses structure.

3.2.3. Dependency tree join

Each sentence will form a dependency tree after the mapping process. Each node in the tree represents a word/term with its POS-tag type. We join a collection of dependency trees into a graph where the edges will record the co-occurrence between two nodes. In this process, we emphasize the relationship (bi-gram frequency) instead of a single node(term) frequency. Therefore, a reliable estimation of the terms co-occurrence requires a large number of corpus. In the short text environment, for example, tweet, the more the number of tweets the better. Note that a word may have a different POS-tag type resulting in multiple nodes with the same word. We keep the duplication of words as part of identification. For example, in [Figure 4](#) we take the previous illustration ‘LeoDiCaprio is an amazing environmentalist. He finally won an oscar’ as the first sentence and ‘LeoDicaprio won an Oscar’ as the second sentence. We merge each tree into a graph by prioritizing the co-occurrence pair of nodes. In this way, we can capture the significance of both terms and phrases.

3.2.4. Graph query processing

Joining the dependency trees will result in a large directed network of words. Each word formed chain properties with a pointer to the next word. To filter the importance of the network, we use the off-shelf graph betweenness centrality algorithm as it shows a good performance in choosing the important topic ([Sayyadi & Raschid, 2013](#)). Betweenness centrality measures the value of a vertex based on the shortest path. Formally, recall that a geodesic path is not necessarily unique and the geodesic paths between a pair of vertices do not need to be node-independent, which means they may pass through some of the same vertices. Let $n_{a,b}^i$ be the number of geodesic paths from a to b that pass through i and let $n_{a,b}$ be the total number of geodesic paths from a to b . Then the betweenness centrality of vertex i is:

$$b_i = \sum_{a,b} w_{a,b}^i = \sum_{a,b} \frac{n_{a,b}^i}{n_{a,b}}$$

where by convention the weight ratio $w_{a,b}^i = 0$ if $n_{a,b} = 0$. The main intuition is that important terms are likely to be in the centre of the graph. Thus, the core of the topic can be automatically generated from the centrality measurement. For example, [Figure 5](#) shows how betweenness centrality able to generate keywords such as ‘gold’, ‘I’m’, and

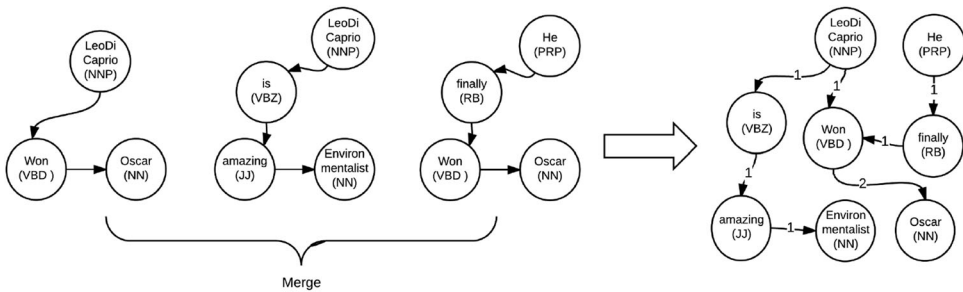


Figure 4. Merging multiple dependency trees.

4.1. Preliminary results

The results of constructing documents into the dependency tree are given in Figure 6. We report the time to label each document before the merging process for different document size. As we can see the time cost of converting documents grows linear with the size. We record approximately between 40 and 100 ms to convert each document depending on the number of words per document. Note that the time can be significantly improved with a better computer specification and bigger memory. Since Twitter has a maximum 140 characters allowance, the processing time appears to be stable. While on the other hand, Amazon review varies a lot in the number of words and tend to be double than Twitter which indicates a significant discrepancy.

Figure 7 depicts the results of loading dependency trees into the graph database. The time is measured as the writing time in the database, we noted a two times improvement in Neo4j version 3.4. We use transaction schema which writes per sequential word-relationship-word as we explained in Section 3.2.3. As shown, the writing time increases gradually as the size is doubling. This is due to the fact that the database has to match existing word-relationship-word to increment the frequency property in the relationship, and one word may have different property type which leads to the increased matching time.

The overall computation time from constructing to loading into the database takes seconds to minutes to process due to the rich POS-tagging graph properties. Stanford dependency parser credited for more than half of the overall processing time. Note that at this point we did not consider any predefined indexing technique prior to loading into the database. However, we consider this process as an offline task which can be computed as a background task. The graph analytical querying within Neo4j is real-time regardless of the size of the document.

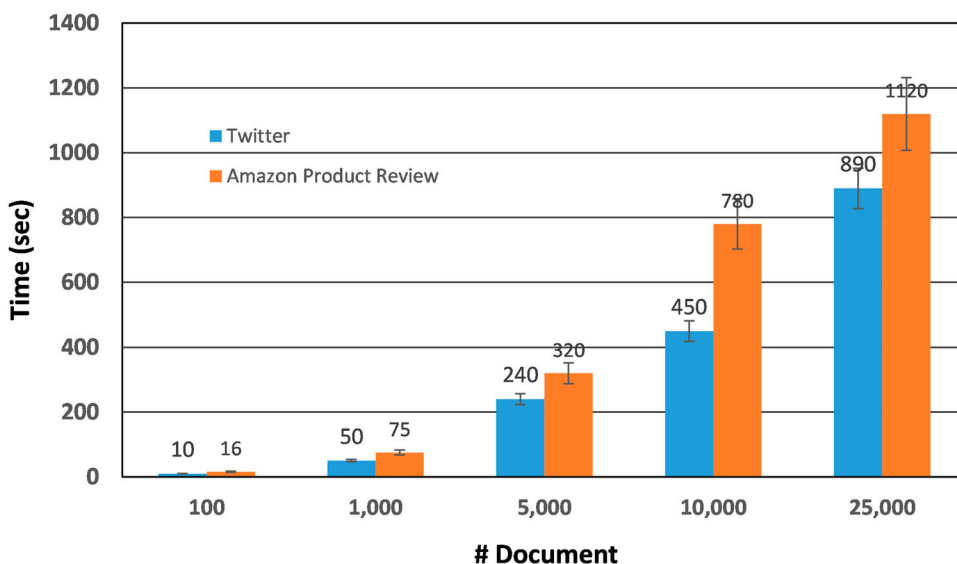


Figure 6. Dependency tree conversion time.

4.2. Baseline methods

Since clustering can be seen as the summarization method (Liu, Chen, & Tseng, 2015), we consider four other approaches to validate the performance of the proposed methods. K-Means (Lloyd, 1982) and DBScan (Ester, Kriegel, Sander, & Xu, 1996) are the most popular and the most straightforward to implement using the tf-idf vectorizer. Specifically, for K-Means and DBScan, we use the scikit-learn³ implementation and we choose the number of clusters with the best silhouette score after several iterations. We also consider LDA (Blei et al., 2003) with Gibbs Sampling as the standard representation of topic modeling. LDA is implemented using Gensim library⁴ with tf-idf document matrix and a variation of topic number, we found that the optimum number falls between 10 and 5 topics. LexRank (Erkan & Radev, 2004), and TextRank (Mihalcea & Tarau, 2004) are implemented using the Sumy summarizer⁵ with the default parameter since fine-tuning does not return a substantial improvement. It is worth noting that these approaches cannot be directly compared for the targeted problem. However, they can be seen as alternatives to achieve the same purpose.

4.3. Quantitative evaluation

For the quantitative evaluation, we use ROUGE metrics (Lin & Och, 2004) as the intrinsic evaluation. ROUGE metric has been a standard to evaluate text summarization quality. The common approach is that given some gold standards of human evaluation, the more words that overlap with the automated summary, the higher the ROUGE score is. ROUGE metrics is usually expressed as ROUGE-N metric:

$$\text{ROUGE} - N = \frac{\sum_{s \in M} \sum_{n\text{-grams} \in s} \text{match}(n\text{-gram})}{\sum_{s \in M} \sum_{n\text{-grams} \in s} \text{count}(n\text{-gram})}$$

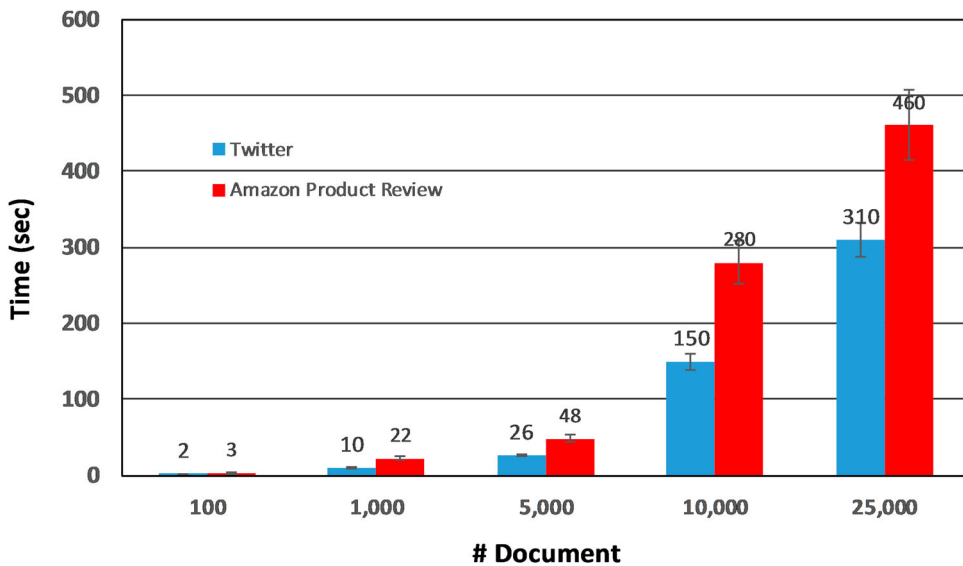


Figure 7. Neo4j loading time.

where M is the set of manual summaries, n is the length of n -grams, $\text{count}(n\text{-gram})$ is the number of n -grams in the manual summary, and $\text{watch}(n\text{-gram})$ is the number of *co-occurring n -grams* between the manual and generated summaries. Table 1 indicates the performance of ROUGE 1-gram and 2-grams on several benchmarks.

We sampled one thousand tweets with the seed keywords filtering (e.g. tweets mentioning ‘#Australia’ or ‘Gold Coast’) to obtain a better relevancy and a thousand Amazon product reviews in electronic goods. We manually generated three human gold standard evaluations for both sample datasets. The number of the gold standard summary is set to match the number of appropriate topics. The number of topics identified is usually less than the number of clusters generated. In the experiment, we maintained the number of topics to maximum ten. We report the average of ROUGE-1 and ROUGE-2 scores according to the gold standard for ten iterations on each topic. Note that it can be difficult to generate a gold standard summary for the product review since the human judgement is prone to subjectivity towards the aspect-specific. We generate the gold standard according to the deeper perspective (Ly, Sugiyama, Lin, & Kan, 2011), for example, the ‘good’ and ‘bad’ of a product.

4.4. Qualitative evaluation

For the qualitative evaluation, we judge the quality of the summarization based on human interpretation. A better approach is to present the results to someone who does not have any background on the topic and then ask her whether she can understand what is the summarization about. Once we translate the dependency trees into the dependency graph, we perform the summarizing task by considering several parameters when we queried the graph:

- *Frequency Pattern.* We set a certain threshold to get the most frequent words relation pattern. This method is based on the relationship that co-occurs between words since the edges preserve the frequency properties. Within a specific threshold, we can filter the graph to generate a summary of text. Note that the co-occurrence frequency is generated based on the word-to-word relationship in contrast to the bag-of-words model.
- *Core Nominals.* The other way is by filtering the core nominals property to get all the main object. Core nominals usually belong to ‘nsubj’, ‘obj’, or ‘iobj’. Once we get this,

Table 1. ROUGE metrics evaluation for text summarization on several approaches.

Dataset	Methods	Rouge-1	Rouge-2
Twitter	K-Means	20.07	5.06
	DBScan	21.43	5.75
	LDA	25.03	7.23
	LexRank	25.38	7.91
	TextRank	26.14	8.59
	BeliefGraph	26.50	9.83
Amazon product review	K-Means	25.21	4.21
	DBScan	25.74	5.02
	LDA	36.18	8.86
	LexRank	35.44	8.23
	TextRank	36.62	9.19
	BeliefGraph	37.46	9.67

we can further use the frequency threshold filtering as the combination to get stronger summarization. In our observation, this combination benefits the product review dataset as the content will be toward the aspect-specific (products).

Both methods have the trade-off between each other. The limitation of the first method is that it may eliminate the main subject or object (e.g. nsubj, obj) which will not generate strong interpretations, while the second may produce excessive terms. Frequency pattern often favours larger dataset (graph) while considering core nominals is better when the dataset is small. We balance the result by taking each subset of each method by tuning the predetermined threshold, for example:

```
MATCH (n) <-[ r ] - (m)
WHERE r.frequency >= 3
AND n.tag = "NN" OR n.tag = "NNS"
RETURN n;
```

In Table 2, we present the summaries of sampled topics from K-Means, DBScan, LDA, LexRank, TextRank, and Belief Graph. For approaches other than the belief graph, we aggregate the short texts into a single large document before each run. Based on Table 1, we can see that belief graph has more interpretable and more cohesive structure. Some keywords have the supporting word preposition such as ‘for the’ → audition generated from the graph. Due to space constraints, we only list keywords that are closely related. Belief graph does not use stopwords to retain the complete structure of the sentence. We will discuss the results further in the next section.

5. Discussion

We examined the ability of belief graph to summarize short text based on carefully tuned graph queries. The whole process consists of part of speech tagging and graph query

Table 2. Extractive evaluation on the text summarization quality.

Approaches	Categories	Summary
K-Means	Event	#theiAwards, #GetReady, #innovation, #tech, up, awards
	TV-Show	#MKR, kitchen, 2016, show, win, #channel10
	Place	#Seaworld, Gold, Coast, Beach, Helensvale, Dreamworld
DBScan	Event	#theiAwards, #tech, #innovation, up, nominate, opportunity
	TV-Show	#MKR, kitchen, show, 2016, win, you
	Place	Beach, Helensvale, Coast, go, Gold, Dreamworld
LDA	Event	Paradise, Surfers, QLD, #theiAwards, project, today
	TV-Show	nominate, #MKR, thought, 2016, ready, audition
	Place	Coast, Gold, summer, day, Dreamworld, sunshine
LexRank	Event	Paradise, Surfers, project, 2016, get, today
	TV-Show	kitchen, #MKR, TV, thought, ready, audition
	Place	Gold, Coast, summer, XXXX, Dreamworld, Q1
TextRank	Event	Paradise, Surfers, QLD, #theiAwards, project, today
	TV-Show	prepare, #MKR, get, come, ready, 2016
	Place	sunshine, Gold, coast, day, #Seaworld, beach
BeliefGraph	Event	nominate, your, project, today, #theiAwards, #innovation
	TV-Show	get, ready, (for the)audition, #MKR, 2016, (are)coming
	Place	in, Surfers, Paradise, Beach, Gold, Coast

execution. Based on the experiments, we showed that belief graph achieves a satisfactory result in terms of interpretability. By leveraging the natural order of words using part of speech tagging, we showed that it is possible to generate the summary from words relationship. Most state of the art techniques such as topic models neglect the importance of word order which often hindered the overall understanding and interpretability. Moreover, the bag-of-words model suffers from sparsity issue when the texts are short. Belief graph, on the other hand, can handle normal or short texts.

The overall process to construct the belief graph takes two sequential procedures. At first, we translate the dependency tree from the Stanford CoreNLP part of speech tagging. This process takes a considerable amount of time when the number of document is above a thousand. In the experiment, we used the original library without further modification which accounts for more than half of the process time. One plausible future direction is to reduce the labelling time to make the online process possible. The second procedure is to load the dependency tree into the graph database. We chose Neo4j due to its popularity for easier and faster implementation. The database also provides off-shelf graph algorithms making it more favourable in the production. The writing process gains two times improvement compared to the older version. Since we do not consider distributed implementation, the writing reaches its bottleneck at ten thousand documents. Overall, it takes less than a minute for a thousand documents.

On the quantitative evaluation, based on the ROUGE metrics results, belief graph achieved better results in the overall summarization. When the gold standard includes the common stop-words (e.g. word preposition), the belief graph gets a bonus score in the ROUGE metrics. Thus, a slightly noticeable improvement in the metrics. Most statistical techniques emphasize the word frequency for the keywords extraction while both LexRank and TextRank apply sentence extraction. Summarizing short text, especially tweets based on the word frequency and sentence ranking do not work well. Gold standard summarization for short text often takes word by word instead of sentence by sentence. Therefore, sentence ranking is not appropriate to fit ROUGE evaluation for short text.

Overall, since the metrics only consider the overlapping words between the generated summary and the gold standards, it is difficult to get the exact match of the words. The metrics do not acknowledge the word or context similarity. We argue that probability soft scoring is more appropriate to judge the relevancy. This explains why for the most techniques, the ROUGE metrics are very low.

In terms of the quality of summarization, we demonstrated that by maintaining the natural structure of human language makes the summarization process easier to understand. This is especially true when the user does not have background knowledge of the content. We observed that the word prepositions are helpful in mapping the relationship between words that appear in the final summarization. One drawback is when the grammatical structure of texts is constantly out of order. For other techniques, the quality of summarization is still somehow depending on the term frequency. When the frequency of words is low, the word choice in the generated summary tends to be unstable.

Finally, the belief graph offers a simple and straightforward technique to generate a short summary to quickly understand a document. In contrast to the previous techniques developed, our method embeds part of speech tagging in the directed graph to maintain the grammatical structure. Since the granularity of each vertex is based on words level, our

proposed method works for both normal and short text as it utilizes the word relationships instead of word frequency.

6. Conclusion

In this work, we presented the belief graph model construction from raw texts. Within the model, we proposed the implementation of the dependency tree from Stanford dependency parser to build a dependency graph. Based on our model we are able to detect important keywords and summarize text-based their word-to-word relationship. Through the performance and practical study in our experiments and using real-world social media post dataset, we showed the effectiveness of the belief graph, especially in mining short text. Additionally, there is a need to improve the current tagging system as it is not yet effective in predicting the grammar for short text, specifically for social media content. Looking into indexing of the tree merging prior loading into the database, which can improve the processing time, is another avenue for further improvement. At this point, the system is only suitable for the offline task due to the long overall processing time. Moreover, short text similarity measurement is important in the aggregation process to avoid duplication, which in the end will improve the efficiency of the overall process.

Notes

1. <https://googleblog.blogspot.co.at/2012/05/introducing-knowledge-graph-things-not.html>.
2. <https://github.com/neo4j-contrib/neo4j-apoc-procedures/releases>.
3. <https://scikit-learn.org/stable/modules/clustering.html>.
4. <https://radimrehurek.com/gensim/>.
5. <https://pypi.org/project/sumy/>.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Nigel Franciscus is a Ph.D. candidate at the School of ICT at Griffith University. He completed his Bachelor of Information Technology (Honours) at Griffith University in the area of Big Data Analysis. He has several years of experience working in a range of faculties as well as tertiary-level teaching expertise in Data Management & Analysis. His research interests include Text and Data Mining, NoSQL Databases, and Natural Language Processing.

Xuguang Ren received his Ph.D. degree from Griffith University. After that he worked in Griffith University as a research fellow. Now he is working in Inception Institute of Artificial Intelligence as a research scientist. His research mainly focuses on graph database and graph data mining. Now he also has been undertaking some research in artificial intelligence, especially deep learning.

Professor **Bela Stantic** is internationally recognized in the field of efficient management and analytics of complex data, such as that found in Big Data, spatiotemporal, and high dimensional data. He was invited to give many Keynotes and invited talks at highly ranked international conferences and prestigious institutions. He successfully applied his research interdisciplinary and has published more than 130 journal and conference publications, which in turn helped to attract external funding over several millions dollars. He is a founder and Director of "Big Data and Smart Analytics" Lab

within the Institute of Integrated and Intelligent Systems at Griffith University. Professor Stantic is also Head of the School of Information and Communication Technology at Griffith University, Brisbane, Australia.

ORCID

Nigel Franciscus  <http://orcid.org/0000-0002-8321-0040>

References

- Amsterdamer, Y., Kukliansky, A., & Milo, T. (2015). A natural language interface for querying general and individual knowledge. *Proceedings of the VLDB Endowment*, 8(12), 1430–1441.
- Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(Jan), 993–1022.
- Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. *Proceedings of the 55th annual meeting of the association for computational linguistics, ACL 2017*, Vancouver, Canada, July 30 – August 4, Volume 1: Long papers (pp. 1870–1879). Association for Computational Linguistics.
- De Marneffe, M.-C., MacCartney, B., & Manning, C. D. (2006). Generating typed dependency parses from phrase structure parses. *Proceedings of LREC*, Genoa, Italy (Vol. 6, pp. 449–454). European Language Resources Association (ELRA).
- Deutch, D., Frost, N., & Gilad, A. (2017). Provenance for natural language queries. *Proceedings of the VLDB Endowment*, 10(5), 577–588.
- Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22, 457–479.
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd*, Portland, OR (Vol 96, pp. 226–231). AAAI Press.
- Franciscus, N., Ren, X., & Stantic, B. (2018a). Beyond word-cloud: A graph model derived from beliefs. *Asian conference on intelligent information and database systems*, Dong Hoi, Vietnam (pp. 81–90). Springer.
- Franciscus, N., Ren, X., & Stantic, B. (2018b). Precomputing architecture for flexible and efficient big data analytics. *Vietnam Journal of Computer Science*, 5(2), 133–142.
- Grave, E., Mikolov, T., Joulin, A., & Bojanowski, P. (2017). Bag of tricks for efficient text classification. *Proceedings of the 15th conference of the european chapter of the association for computational linguistics*, Valencia, Spain (pp. 427–431). Association for Computational Linguistics.
- He, R., & McAuley, J. (2016). Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. *Proceedings of the 25th international conference on world wide web*, Montreal, Canada (pp. 507–517). ACM.
- Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., & Mikolov, T. (2016). Fasttext. zip: Compressing text classification models. *arXiv preprint arXiv:1612.03651*.
- Li, F., & Jagadish, H. V. (2014). Constructing an interactive natural language interface for relational databases. *Proceedings of the VLDB Endowment*, 8(1), 73–84.
- Li, C., Wang, H., Zhang, Z., Sun, A., & Ma, Z. (2016). Topic modeling for short texts with auxiliary word embeddings. *39th international ACM SIGIR conference on research and development in information retrieval*, Pisa, Italy (pp. 165–174). ACM.
- Lin, C.-Y., & Och, F. J. (2004). Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. *Proceedings of the 42nd annual meeting on association for computational linguistics*, Barcelona, Spain (pp. 605). ACL.
- Liu, C.-Y., Chen, M.-S., & Tseng, C.-Y. (2015). Incrests: Towards real-time incremental short text summarization on comment streams from social network services. *IEEE Transactions on Knowledge and Data Engineering*, 27(11), 2986–3000.
- Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2), 129–137.

- Ly, D. K., Sugiyama, K., Lin, Z., & Kan, M.-Y. (2011). Product review summarization from a deeper perspective. *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*, Ottawa, ON, Canada (pp. 311–314). ACM.
- Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., & McClosky, D. (2014). The stanford corenlp natural language processing toolkit. *ACL (system demonstrations)*, Baltimore, MD, USA (pp. 55–60). The Association for Computer Linguistics.
- Mihalcea, R., & Tarau, P. (2004). Texttrank: Bringing order into text. *EMNLP*, Barcelona, Spain (Vol. 4, pp. 404–411). ACL.
- Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, Lake Tahoe, Nevada, USA (pp. 3111–3119). Curran Associates.
- Ohsawa, Y., Benson, N. E., & Yachida, M. (1998). Keygraph: Automatic indexing by co-occurrence graph based on building construction metaphor. *Research and technology advances in digital libraries*, Santa Barbara, CA (pp. 12–18). IEEE Computer Society.
- Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global vectors for word representation. *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, Doha, Qatar (pp. 1532–1543). ACL.
- Quan, X., Kit, C., Ge, Y., & Pan, S. J. (2015). Short and sparse text topic modeling via self-aggregation. *Proceedings of the 24th international conference on artificial intelligence*, Buenos Aires, Argentina (pp. 2270–2276). AAAI Press.
- Sayyadi, H., & Raschid, L. (2013). A graph analytical approach for topic detection. *ACM Transactions on Internet Technology (TOIT)*, 13(2), 4.
- Scaiella, U., Ferragina, P., Marino, A., & Ciaramita, M. (2012). Topical clustering of search results. *Proceedings of the fifth ACM international conference on Web search and data mining*, Seattle, WA, USA (pp. 223–232). ACM.
- Song, Q., Wu, Y., & Dong, X. L. (2016). Mining summaries for knowledge graph search. *2016 IEEE 16th international conference on data mining (ICDM)*, Barcelona, Spain (pp. 1215–1220). IEEE Computer Society.
- Yan, X., Guo, J., Lan, Y., & Cheng, X. (2013). A biterm topic model for short texts. *Proceedings of the 22nd international conference on World Wide Web*, Rio de Janeiro, Brazil (pp. 1445–1456). International World Wide Web Conferences Steering Committee / ACM.
- Yin, J., & Wang, J. (2014). A dirichlet multinomial mixture model-based approach for short text clustering. *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA (pp. 233–242). ACM.
- Zhang, C., Wang, H., Xu, F., & Hu, X. (2013). Ideagraph plus: A topic-based algorithm for perceiving unnoticed events. *2013 IEEE 13th international conference on data mining workshops (ICDMW)*, Dallas, TX, USA (pp. 735–741). IEEE Computer Society.
- Zuo, Y., Zhao, J., & Xu, K. (2016). Word network topic model: A simple but general solution for short and imbalanced texts. *Knowledge and Information Systems*, 48(2), 379–398.