Social Context Summarization using User-generated Content and Third-party Sources[☆]

Minh-Tien Nguyen[a,b], Duc-Vu Tran[a], Minh-Le Nguyen[a,*]

[a]*School of Information Science,*
Japan Advanced Institute of Science and Technology (JAIST),
1-1 Asahidai, Nomi, Ishikawa, 923-1292, Japan.
[b]*Hung Yen University of Technology and Education (UTEHY), Hung Yen, Vietnam.*

Abstract

In the context of social media, users mutually share their interests of an event mentioned in a Web document. Its content can also be found in different news providers with a writing variation. This paper presents a framework which exploits the support of both user-generated content such as comments or tweets and third-party sources such as relevant documents retrieved from a search engine to generate a high-quality summarization. The summarization was formulated in two steps: sentence scoring and selection. The scoring is modeled as a learning to rank problem, which uses Ranking SVM to mutually exploits sentences, user-generated content, and third-party sources in the form of features to cover summary aspects. In the selection, the summarization is extracted by using score-based or voting method. For evaluation, three datasets containing Web documents, comments (or tweets), and relevant documents in two languages were taken as a case study. Promising results indicate that: (i) by integrating the user-generated content and third-party sources, our model obtains improvements of ROUGE-scores over state-of-the-art baselines and (ii) representing sentence extraction by learning to rank benefits the summarization.

[☆]This manuscript is an improved and extended version of the paper: Summarizing Web Documents using Sequence Labeling with User-generated Content and Third-party Sources, presented at 22th *International Conference on Natural Language & Information Systems* (NLDB) 2017, Liege, Belgium.

[*]Corresponding author: Minh-Le Nguyen

Email addresses: `tiennm@jaist.ac.jp` (Minh-Tien Nguyen), `vu.tran@jaist.ac.jp` (Duc-Vu Tran), `nguyenml@jaist.ac.jp` (Minh-Le Nguyen)

Keywords: Data Mining, Information Retrieval, Document Summarization, Social Context Summarization, Learning to Rank.

1. Introduction

Text summarization is a reduction process which extracts salient information within a document or a set of news articles to generate an essential subset of textual content for users usage (Luhn, 1958; Edmundson, 1969; Jones, 2007; Nenkova and McKeown, 2011). The outputs from a summary system benefit to assist with user's information needs. For example, search engines, e.g. Google or Bing show the snippets of relevant pages corresponding to a search query; or online news providers usually present an extracted summary corresponding each article for readers. The snippet information and extracted summary provide a quick view for readers in catching the main content of a document. Such beneficial use demands high-quality text summarization systems.

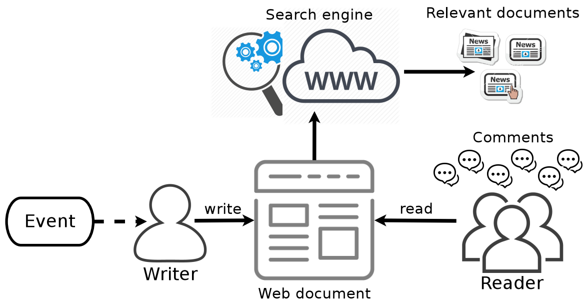


Figure 1: The schema of Web document, user-generated content, and relevant documents.

In the context of social media, readers can freely express their opinions in the form of user-generated content such as comments or tweets, one form of social information (Amitay and Paris, 2000; Delort et al., 2003; Sun et al., 2005; Hu et al., 2008; Lu et al., 2009; Yang et al., 2011; Hu et al., 2011; Wei and Gao, 2014; Nguyen and Nguyen, 2016), regarding an event mentioned in a Web document. For example, after reading a Web document describing the

Boston bombing event in USAToday[1], readers discuss the event by writing their comments on the website or posting tweets on their Twitter[2] timeline. After writing, other users can immediately update the news content. The content of this event can also be published by different news providers in a writing variation form. Such relevant Web documents can be found by using a search engine, e.g. Google search with a search query. Figure 1 shows this scheme. The combination of user-generated content and relevant documents not only reveals the opinions of readers but also reflect the content of the original document and describe the facts of an event. This inspires a novel summarization task which mutually exploits the user-generated content and relevant documents (third-party sources) to support sentences in extracting salient information.

Traditional extractive summarization methods (Luhn, 1958; Edmundson, 1969; Kupiec et al., 1995; Conroy and O'leary, 2001; Osborne, 2002; Yeh et al., 2005; Shen et al., 2007; Lin and Bilmes, 2011) focus on selecting important sentences in a document by using statistical or linguistic information. They treat each sentence individually and train a binary classifier to classify sentences, in which label 1 denotes summary sentences and 0 represents non-summary sentences. Although these methods have achieved promising results, they only consider inherent document information, e.g. sentence or word/phrase level while ignoring its social context, which can provide additional information from social users, e.g. the viewpoint of users who already involve an event.

Our objective is to propose a summary framework which automatically extracts important sentences and representative user-generated content of a Web document by incorporating its social context. In order to do that, the summarization was formulated as a learning to rank problem which mutually exploits the support of user-generated content and third-party sources to cover summary aspects in each sentence. This paper makes the following contributions:

- It formally models the relation of a Web document, user-generated con-

[1]http://www.usatoday.com

[2]http://twitter.com - a microblogging system

tent, and relevant articles from other news providers and puts them into a unified summary framework. The relation of a Web document and user-generated content is represented in the form of generation, in which readers usually generate their messages after reading a document. The relation of an original Web document and relevant news articles is presented as a phenomenon in which the same content of an event can be written in various ways and published by several news providers.

- It conducts a careful investigation to extract 14 new features represented in the form of three groups: local features, user-generated features, and third-party features. The investigation provides an overview of feature selection for summarization. The architecture of our model is straightforward to integrate any additional features from any relevant sources.
- It carefully observes several important aspects of the summarization such as feature contribution, relevant document contribution, the impact of training data size, and sentence position. This observation contributes to summarization literature.
- It proposes a unified framework[3] which utilizes 14 new features to extract summaries on three datasets in two languages.

In remaining sections, we first introduce related works. Next, we describe SoSVMRankSum, which exploits the mutual support from user-generated content and third-party sources to model of each sentence and comment (or tweet). This section also mentions our idea and the model. Subsequently, we describe data collection and preparation. After preparing the data, we illustrate the summary process to achieve our objective in two steps: sentence scoring and sentence selection. After generating the summarization, we show experimental results along with discussions and deep analyses. We finish by drawing important conclusions.

[3]http://150.65.242.101:9293

2. Literature Review

2.1. Extractive Summarization

Text summarization is a challenging task studied in a long history. (Luhn, 1958; Edmundson, 1969) summarized documents by using the significant components of sentences, such as high-frequency content words and sentence position. With the growth of artificial intelligence, many studies have applied machine learning to text summarization (Kupiec et al., 1995; Osborne, 2002; Yeh et al., 2005; Shen et al., 2007). They formulate the summarization as a binary classification and train a classifier to distinguish summary (label 1) or non-summary sentences (label 0). For example, (Yeh et al., 2005) combined classification and latent semantic analysis (LSA) to build summarizers, in which the score of a sentence is ranked by using features or the semantic matrix. The highest F-score is 0.49 with 30% compression. (Shen et al., 2007) exploited sequence aspect in a document and trained a sequence labeling classifier to extract summaries. This method achieves 0.483 in ROUGE-2[4] (Recall-Oriented Understudy for Gisting Evaluation: 1-gram or 2-gram based co-occurrence statistics of output summary and gold-standard references) and 0.419 in F-1 on DUC 2001 dataset.

Recently, (Lin and Bilmes, 2011) presented a set of submodular functions for document summarization. Each function guarantees two aspects: the representativeness and diversity. This method obtains the best results of Recall and F-score over DUC 2004 to 2007. (Woodsend and Lapata, 2010; Woodsend and Lapata, 2012) formulated summary sentences in the form of concepts and defined a set of constraints for generating highlights of a Web document. The authors represent the concepts as phrases extracted from a dependency tree. This method achieves 0.25 in both ROUGE-1 and F-score. There are also many studies which focus on extracting summary sentences in a document such as LexRank (Erkan and Radev, 2004), TextRank (Mihalcea and Tarau, 2004), or deep learning (Cao et al., 2015; Zhang et al., 2016; Nallapati et al., 2016).

[4]https://en.wikipedia.org/wiki/ROUGE_(metric)

2.2. Social Context Summarization

To the best our knowledge, (Amitay and Paris, 2000) were the first researchers who exploited the hyperlinks of a Web document for the summarization. They assume that the linked document reflects the content of original document. The authors build InCommonSense, including hypertext retrieval and description selection (using classification) to extract a paragraph from the linked document as the summarization. The average user voting (1 to 5) is 4.71 compared to 4.14 of AltaVisata-style and 4.13 of Google-style. (Delort et al., 2003) extended the hyperlink assumption by considering whole linked documents as the context instead of using paragraphs including hyperlinks as (Amitay and Paris, 2000). The authors proposed content and context summarization algorithms based on similarity measurements. The best result is around 0.45 in term of similarity in the content and context summarization.

(Sun et al., 2005) utilized click-through data retrieved from search engines to extract salient sentences in a Web document. The assumption of this method is that query keywords from users typed on search engines usually reflect the content of a Web document. Based on that, they present two summary algorithms using an adaptation of significant words (Luhn, 1958) and latent semantic analysis (Gong and Liu, 2001). The ROUGE-1 is around 0.55 and 0.20 on their datasets. This method, however, exists an issue that pages which Web users click on may be irrelevant to their interest. (Delort, 2006) extracted sentences by using the link from sentences to comments represented by clusters. This method achieves around 50% extracted matches corresponding to post summaries. (Hu et al., 2008) selected sentences that best represent topics discussed among readers in blog posts. This approach first derives salient words denoted in three graphs: topic, quotation, and mention from comments. Summary sentences are next generated by calculating the distance from each sentence to the graphs. The method obtains around 0.64 in ROUGE-1 and 0.60 in NDCG[5] on their dataset. (Lu et al., 2009) extracted phrases which help users for better

[5]https://en.wikipedia.org/wiki/Discounted_cumulative_gain

understanding the discussions of a target object in comments. The authors introduce a model containing three steps: aspect discovery and clustering, aspect rating prediction, and representative phrase extraction. It obtains 0.592 of precision and 0.637 of recall in the phrase extraction on their dataset.

Tweets from Twitter were widely used to support sentences in generating the summarization. (Yang et al., 2011) presented a dual wing factor graph model which uses Support Vector Machines (SVM) and Conditional Random Fields (CRF) as preliminary steps for incorporating tweets into the summarization. The authors use a ranking method, which approximates an objective function to select both summary sentences and tweets. The model achieves around 0.615 in ROUGE-1 and 0.500 in ROUGE-2 on five collected datasets. (Gao et al., 2012) introduced a cross-collection topic-aspect modeling (cc-TAM) as a preliminary step to generate a bipartite graph used by co-ranking to select sentences and tweets for multi-document summarization. The ROUGE-1 is 0.55 in the document and 0.67 in tweet selection. (Wei and Gao, 2014) integrated the human knowledge into the summary process by proposing local sentence features, local tweet features, and cross features used for a learning-to-rank model in a news highlight extraction task. The model selects top m sentences and tweets after ranking. The ROUGE-1 is 0.292 and 0.295 in document and tweet summarization, respectively. (Wei and Gao, 2015) proposed a variation of LexRank, which uses auxiliary tweets for building a heterogeneous graph random walk (HGRW) to summarize single documents. This method achieves 0.298 in ROUGE-1 when combining summary sentences and tweets.

The previous methods exist two issues: (i) unsupervised methods, e.g LexRank or HGRW are sensitive to data, and (ii) several methods only select sentences or social messages as the summarization. We address the two issues, in which, firstly, we present a framework which integrates the human knowledge of the summarization in the form of supervised learning fashion. Secondly, the summarization in our method contains both sentences and user-generate content instead of only selecting sentences. The selected comments or tweets help to enrich information which may not be available in sentences.

3. Definition

As mentioned, the goal of social context summarization is to generate the summarization of a Web document by using its user-generated content, e.g. comments and relevant documents from a search engine. In this section, we provide necessary notations and definitions used in this study.

User-generated content: is defined as comments or tweets generated from readers after reading a Web document. Formally, given a document d and a set of users $U = \{u_1, ..., u_n\}$, who read d, user-generated content of d is denoted by UG_d created by $d \xrightarrow[U]{posting} C$ or $d \xrightarrow[U]{posting} T$, where C or T is a set of comments or tweets, $\xrightarrow[U]{posting}$ presents that C or T is produced by U.

Third-party sources: are a set of relevant Web documents retrieved from a search engine by searching the title of the original document. Formally, given d is the original document with its title t, $R = \{r_1, ..., r_m\}$ is m news providers, third-party sources of d is TS_d created by $d \xrightarrow[searching(t)]{R} TS_d$. We only consider top ten searching Web pages returned from Google search as a global information because articles outside top ten are usually irrelevant.

Social context: of a Web document d is defined by C_d presented by $\langle S_d, UG_d, TS_d, U_d \rangle$ (Yang et al., 2011), where S_d is a set of sentences in document d, UG_d is a set of tweets or comments of d written by users U_d, and TS_d is relevant Web documents returned by a search engine by searching the title of d. We eliminate user relation because it is an implicit factor.

Social context summarization: is to select important sentences and representative user-generated content such as comments (or tweets) as the summarization by using C_d giving the original document d.

4. Learning to Summarize with Social Context

This section shows our proposal to extract summary sentences and comments or tweets. We first present our idea and summary framework. Next, we show data preparation and observation. Finally, we describe the process for achieving the objective, and show baselines and evaluation metric.

4.1. Summary Framework

Our framework in Figure 2 includes three components: data collection, sentence scoring, and sentence selection. The collection collects main documents along with their comments or tweets from readers and relevant articles from different news providers. The input of the framework is a combination of the three elements collected from the collector. The scoring generates the score of each sentence and comment/tweet based on a L2R algorithm, which exploits information from three channels: local sentences, social messages, and relevant articles. For example, when modeling a sentence, the framework first extracts a set of local features. In the same time, it collects a set of features from comments (the red line) and third-party sources (the blue line) to support the local features. When presenting a comment (or tweet), similarly, social features from sentences (the red line) and third-party features from relevant documents (the blue line) are utilized to support the local features of each comment. In this view, we also consider information from sentences as social features of comments. The selection takes the scores from the scoring and outputs a summary by using score-based or voting method. Due to the space limitation, in this paper, we describe the scoring and selection.

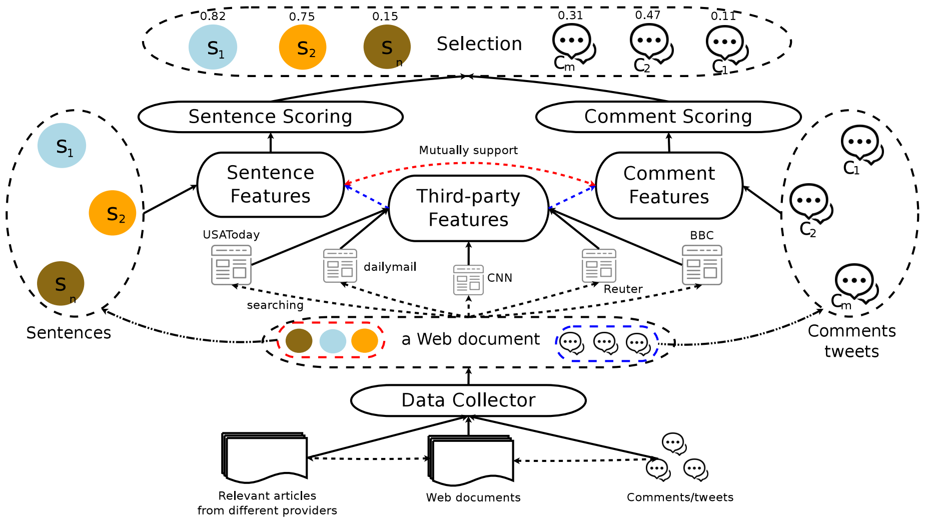


Figure 2: The overview of summarization model. The red and blue lines present the support of social context. Dashed lines from document to news providers denote the searching process.

The framework in Figure 2 is an extension of our model in (Nguyen et al., 2016b), in which we also formulate the summarization as a L2R problem. However, comparing to the model in (Nguyen et al., 2016b), the framework is refined with significant improvements. Firstly, we integrate relevant documents retrieved from a search engine. The additional documents allow to cover more summary aspects and lead to new novel features in Section 4.3.1 (third-party features). Secondly, we divide the summarization into two steps: sentence scoring and sentence selection, which allow to investigate alternative selection methods such as ranking or voting in Section 4.3.2. Compared to the original paper (Nguyen et al., 2017), we improve two significance: (i) we treat the limitation of CRF in summarizing comments by using Ranking SVM and (ii) we validate the framework on three datasets in two languages with deep analyses. Our approach shares the idea of using L2R with (Svore et al., 2007; Wei and Gao, 2014); however, we make two significant differences: (i) enriching social context by adding relevant Web documents returned from Google search and (ii) presenting new novel features. Our study distinguishes traditional methods (Luhn, 1958; Edmundson, 1969; Kupiec et al., 1995; Osborne, 2002; Yeh et al., 2005; Shen et al., 2007; Lin and Bilmes, 2011) in two points: (i) integrating social information and (ii) selecting both summary sentences and user-generated content as the summarization.

4.2. Data Preparation and Observation

This section shows the data preparation used for running our summary framework in three steps: collection, observation, and data segmentation.

4.2.1. Data Collection

DUC[6] datasets are well-known data for document summarization; these datasets, however, lack social information. Therefore, we prepared three datasets in two languages for social context summarization.

[6]http://duc.nist.gov/data.html

SoLSCSum: We used an English dataset created for social context summarization (Nguyen et al., 2016c). The dataset contains 157 open-domain news articles along with 3,462 sentences, 5,858 gold-standard references, and 25,633 comments collected from Yahoo News. The label of each sentence or comment was created based on majority voting between annotators. The Cohen's Kappa[7] between the annotators after validating is 0.5845 with 95% confidence interval.

To create third-party sources, we retrieved relevant Web documents by searching the titles of original documents using Google[8] search. Top ten Web pages appearing on the search result page were selected by removing duplicate Web documents. Unnecessary information, e.g. HTML tags was removed to obtain raw texts. We kept the order of relevant articles. Finally, we obtained 1,570 relevant documents corresponding to 157 original events.

USAToday-CNN: contains 121 events along with 455 highlights and 78.419 tweets derived from (Wei and Gao, 2014). The dataset (without labels) was used for news highlight extraction. To train supervised learning methods, e.g. SVM or CRF, we created label for each sentence and tweet. After removing near-duplicate tweets by using Simpson formula, we ask two annotators to give weak labels (0 and 1, only used for training, not for evaluation) for each sentence and tweet based on Cosine similarity suggestion and their content via an annotation page[9]. The Cosine similarity based on bag-of-words model is the maximal score of a sentence or tweet with the highlights of each document. Annotators make a decision based on both content reflection and Cosine suggestion. The selected sentences or tweets are no less than five and no more than 15 in total. Cohen's Kappa between annotators after validating is 0.617 with 95% confidence interval.

To create third-party information, Google search engine was again used to retrieve relevant articles of each original document via its title. Finally, we collected 1,210 relevant documents.

[7]http://graphpad.com/quickcalcs/kappa1.cfm
[8]https://www.google.com
[9]http://150.65.242.97/doc-sum-annotator/annotate

VSoLSCSum: To validate the performance of our model in non-English language, we used a Vietnamese dataset created for social context summarization (Nguyen et al., 2016a). The dataset consists of 141 open-domain articles along with 3,760 sentences, 2,448 gold-standard references, and 6,926 comments in 12 events. The agreement computed by Cohen's Kappa between the two annotators is 0.685 with 95% confidence interval.

To create the third-party sources, we used the same mechanism with SoLSC-Sum and USAToday-CNN dataset. In the creation, we removed duplicate documents, which have the same content with the original document. Finally, we obtained 1,410 relevant Web documents from Google search engine corresponding to 141 original documents.

4.2.2. Data Observation

Table 1 presents the statistics of the three datasets. It indicates that the number of user-generated content and relevant documents is large enough to support sentences.

Table 1: Statistical observation on the three datasets.

Dataset	**#doc**	**#sents**	**#comments/ tweets**	**#refs**	**#relevant docs**
SoLSCSum	157	3,462	25,633	5,858	1,570
USAToday-CNN	121	6,413	78,419	455	1,210
VSoLSCSum	141	3,760	6,926	2,448	1,410

We also observed word overlapping between sentences and comments (or tweets) on the three datasets. We consider each token as single word and compute the percentage of word overlapping. Note that we did not observe the computation with stopword removal on VSoLSCSum due to no formal stop word list in Vietnamese. Table 2 shows the observation.

The word overlapping observation indicates that: (i) there exists common words or phrases between sentences and comments, and (ii) readers tend to use words or phrases appearing in sentences to create their comments or tweets, e.g.

Table 2: Word overlapping over the three datasets; s: sentences, c: comments, and t: tweets.

Dataset	Observation	Sentences	Comments/ Tweets
SoLSCSum	% Token overlapping	s/c: 13.26	c/s: 42.05
	% Token overlapping (stopword removal)	s/c: 8.90	c/s: 31.21
USAToday-CNN	% Token overlapping	s/t: 22.24	t/s: 16.94
	% Token overlapping (stopword removal)	s/t: 15.61	t/s: 12.62
VSoLSCSum	% Token overlapping	s/c: 37.712	c/s: 44.820

31.21% of word overlapping. This observation suggests four hypotheses:

- *Representation*: summary sentences in a Web document contain important information.
- *Reflection*: representative tweets or comments written by readers reflect document content as well as summay sentences.
- *Generation*: readers tend to use words or phrases appearing in a document to create their social messages, e.g. tweets or comments.
- *Common topic*: sentences and social messages mention some common topics represented in the form of common words.

4.2.3. Data Segmentation

We conducted a pre-processing step to remove comments with fewer than five tokens since they are fairly short for summarization. In SoLSCSum, 10-fold cross validation was used with $m = 6$ (six summary sentences and six summary comments), the same setup with (Nguyen et al., 2016c). In USAToday-CNN dataset, 5-fold cross validation was used with $m = 4$, the same setup with (Wei and Gao, 2014; Nguyen and Nguyen, 2016). For VSoLSCSum, 5-fold cross validation with $m = 6$ was used with the same configuration of (Nguyen et al., 2016a). The summarization in SoLSCSum and USAToday-CNN was stemmed[10] using the method in (Porter, 2011). Note that summaries in VSoLSCSum were not stemmed because there is no stemming method in Vietnamese.

4.3. Summarization by Ranking SVM with Social Context

This section describes our process to generate the summarization by using the support from social context. We first introduce the sentence scoring with a basic model which uses local information. Next we present a novel model which is an extension of the basic one by incorporating the social context. Finally, we show a procedure of the sentence selection to generate the summarization.

4.3.1. Sentence Scoring

Basic Model with Local Information. As shown in our previous work (Nguyen et al., 2016b), Ranking SVM[11] (Joachims, 2006) has proved as an efficient method for summarization. We, therefore, adopt this method for sentence scoring. Ranking SVM applies the characteristics of SVM (Cortes and Vapnik, 1995) to perform pairwise classification. Given n training queries $\{q_i\}_{i=1}^{n}$, their associated document pairs $(x_u^{(i)}, x_v^{(i)})$ and the corresponding ground truth label $y_(u,v)^{(i)}$, SVM-Rank optimizes an objective function shown in Eq. (1):

$$\min \frac{1}{2} \|w\|^2 + \lambda \sum_{i=1}^{n} \sum_{u,v:y_{u,v}^{(i)}} \xi_{u,v}^{(i)} \tag{1}$$

$$\text{s.t. } w^T(x_u^i - x_v^{(i)}) \geq 1 - \xi_{u,v}^{(i)}, \text{ if } y_{u,v}^{(i)} = 1 \tag{2}$$

$$\xi_{u,v}^{(i)} \geq 0, \; i = 1, ..., n \tag{3}$$

where: $f(x) = w^T x$ is a linear scoring function, (x_u, x_v) is a pairwise and $\xi_{u,v}^{(i)}$ is the loss. The document pair-wise is sentence-sentence or comment-comment. The pair-wise order was determined by the label of each instance, i.e. 1 or 0. Next paragraphs describe features used to train the L2R models to generate the scores. Table 3 presents all features.

[10]http://snowball.tartarus.org/algorithms/porter/stemmer.html
[11]https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html

Table 3: The features used to train a L2R model for sentences. Note that they are also used to train a L2R model for comments (or tweets).

Feature group	Description
Local features	The position of the sentence
	The length of a sentence
	Log-likelihood of a sentence
	Thematic words in a sentence using a dictionary
	Indicator words in a sentence using a dictionary
	Uppercase words
	Cosine similarity of a sentence with previous and next sentences (N = 1, 2, 3)
	LSA score of a sentence
	HIT score of a sentence
	The number of common words with a sentence and title
	The number of stopwords in a sentence
	Local LDA score of a sentence
User-generated features	Maximum semantic similarity of a sentence with comments (or tweets)
	Auxiliary LDA score of a sentence with comments (or tweets)
	Maximum distance similarity of a sentence and comments (or tweets)
	Maximum lexical similarity of a sentence and comments (or tweets)
Third-party features	Cosine voting of a sentence with sentences in relevant documents
	Cluster distance of a sentence with relevant documents
	Sentence-thir-party term frequency
	The average probability of frequent terms
	The frequent term summing score
	The relative frequent term summing score

Basic features: a straightforward way to build a L2R model is to define features, which capture the characteristics of data. We started by using traditional features for sentence classification such as: sentence position, sentence length, log-likelihood. We also included basic features for sentence labeling from (Shen et al., 2007). The basic features include: thematic words, indicator words (count the number of indicator words using a dictionary), uppercase words, Cosine similarity with previous and next N ($N = 1, 2, 3$) sentences, LSA and HIT score. The remaining sections describe our new features used to improve the performance of summarization model. Note that we describe new features for sentences; however, the new features can also be applied to comments or tweets in the same mechanism.

New local features: our first effort is to define a set of features, which represent inherent information of each sentence. We denote them as local features, which capture three aspects: the similarity of a sentence (or comment/tweet) with a title, informativeness, and topical covering.

- *Sentence-title common words:* counts the number of words in a sentence s_i appearing in the title.
- *Function words:* An important sentence should contain content words rather than function ones, e.g. stop words. This feature counts the number of stop words in a sentence. Let l_{org} to be the length of original sentence and l_{rmw} is the length of the sentence after removing stop words, Eq. (4) computes this feature.

$$\textit{num-stopword}(s_i) = l_{org}(s_i) - l_{rmw}(s_i) \tag{4}$$

- *Local LDA score:* bases on a hypothesis that a summary sentence usually contains salient information represented in the form of topics. This feature was calculated in two steps: training and inference.

  Training: We first trained an LDA[12] (Blei et al., 2003) model on a large

[12]http://mallet.cs.umass.edu/topics.php

number of news articles. In this view, we considered the LDA model as a external source. We used more than 280.000 articles[13] from Daily-Mail[14] for training an LDA model in English and 4 million articles[15] from Baomoi[16] for training an LDA model in Vietnamese. Topic and word distribution parameter for training are $\alpha = \beta = 0.01$ with 1000 iterations; the topic number $k = 100$ is empirically set. More precisely, we divided each document into two parts: d_s (sentences) and d_c (comments or tweets), then we formed each part as a smaller single document, that is an input for inference. After training, we obtained a word-topic-weight matrix and document-topic distribution matrix used for the inference step.

Inference: Given a small single document d_s (or d_c), we first obtained top t closet topics of d_s. The closet topics of d_s are those that have the highest values in the document-topic distribution matrix. With each topic, we selected top w words, which have the highest weights in the word-topic-weight matrix. As a result, the number of topical words for each small single document is $(|t| \times |w|)$. In practice, we set $|t| = 5$ and $|w| = 5$, then the total number of topical words is 25. Given a set of topical words named $TF = \{w_1, ..., w_k\}$ inferred from d_s, Eq. (5) computes the local LDA score:

$$local\text{-}lda\text{-}score(s_i) = \frac{\sum_{j=1}^{k} weight(w_j)}{n} \text{ if } w_j \in s_i \tag{5}$$

where: $weight()$ returns the word weight of w_j, e.g. 0.45 in s_i (normalized in [0, 1]) ; n is the number of words in s_i.

Novel Model with Social Context Integration. The content of a document is not only mentioned in comments or tweets written by readers but also described in other Web documents from different news providers defined as third-party sources. For example, the Boston bombing event published by CNN can

[13]http://homepages.inf.ed.ac.uk/s1537177/resources.html
[14]http://www.dailymail.co.uk/home/index.html
[15]https://1drv.ms/u/s!ArAsY4TKffFIiCPoKbBXZpSrZioJ
[16]http://www.baomoi.com

be found at USAToday or BBC with a content variation. This observation motivates us to propose new features, which exploit the support from user-generated content and third-party sources. We generated 13 new features, in which each feature covers the characteristic of a summary sentence or comment/tweet that may match with gold-standard references.

User-generated Features: when modeling sentences, a set of features was extracted from user-generated content such as comments or tweets, which have an explicit relation with the original document (readers post these messages after reading the document). These features cover semantic similarity, topic, and textual entailment aspect.

- *Maximal semantic-based similarity*: This feature bases on the *generation* hypothesis, in which readers tend to use salient words in sentences to create their comments or tweets in a variation form. Table 4 shows the generation example taken from Yahoo News[17].

Table 4: An example of the generation behaviour. The first sentence is in the Web document and the second one is a comment generated from readers.

The 26-year-old man, identified as Usaamah Rahim, brandished a **knife** and advanced on **officers** working with the Joint Terrorism Task Force who initially tried to retreat before opening **fire**, Boston **Police** Superintendent William Evans told reporters.
If I had been one of the **police officers** I would have whispered 3 times "**drop the knife**" then quickly **fired** several **shots** at his sternum.

In this example, we can observe that: (i) the sentence can be seen as a summary because it contains essential information; (ii) the comment also reflects the event and includes the viewpoint of readers; and (iii) the comment and sentence share several common words (bold words), which are directly extracted from the sentence, e.g. *"knife"*, *"officers"*, or are derived in a variation, e.g. *"fired"*, *"shots"*. From these observation, we

[17]https://www.yahoo.com/news/boston-man-shot-police-target-terrorism-probe-officials-022407784.html?ref=gs

present the sentence-comment relation by semantic similarity.

To exploit the semantic aspect, we present the relation by using *Word2Vec* (Mikolov et al., 2013). The Word2Vec[18] takes a large dataset as an input and produces word vectors as the output. In training, the Word2Vec first generates a vocabulary from the dataset and maps each word into a high-dimensional vector space, in which each vector represents the meaning of a word with its context. The context of a word is the number of its surrounding words (usually called by window size). After training, we can calculate the distance between two words, e.g. Cosine similarity between two words: *"police"* and *"officer"*. In practice, we trained a Word2Vec model on 1 billion words from Google by using SkipGram model, the vector dimension = 300 with the window size (word context) = 7.

Given the Word2Vec model, Eq. (6) calculates the semantic similarity of a sentence with the user-generated content:

$$\textit{w2v-score}(s_i, UG_d) = \max_{j=1}^{m} (sentSim(s_i, c_j)) \tag{6}$$

where: m is the number of comments, $sentSim()$ returns the semantic similarity of s_i and c_j and is calculated by Eq. (7):

$$sentSim(s_i, c_j) = \frac{\sum_{w_i}^{N_s} \sum_{w_j}^{N_c} w2vSim(w_i, w_j)}{N_s + N_c} \tag{7}$$

where: N_s and N_c are the number of words in s_i and c_i after removing stop words; $w2vSim()$ returns the semantic similarity between two words and was computed by the *Word2Vec* model.

- *Auxiliary LDA score:* We formulate this feature in the view of *Local LDA score*, in which a summary should also cover topics discussed among readers. Given a set of topical words named $TF = \{w_1, ..., w_k\}$ inferred from d_c (the topical words were inferred from comments or tweets), the LDA score from social information is computed by Eq. (8), the same mechanism

[18]https://code.google.com/p/word2vec/

as Eq. (5).

$$aux\text{-}lda\text{-}score(s_i) = \frac{\sum_{j=1}^{k} weight(w_j)}{n} \text{ if } w_j \in s_i \tag{8}$$

where: n is the number of words in s_i, the word weight is derived from topical words on the social side. By combining the two LDA scores, our model states that a summary sentence should not only cover topics written by writers on the document side but also include topics discussed among readers on the social side.

- *Maximal distance-based similarity*: This feature tackles the *generation* hypothesis by formulating the sentence-comment (or tweet) relation in the form of recognizing textual entailment (RTE) (Dagan et al., 2010). The RTE is a task which decides whether the meaning of a text can be plausibly inferred from another text in the same context (Dagan et al., 2010). This feature treats a different aspect compared to the *semantic-based similarity*, in which it operates on word and lexical level instead of semantic level. We present a distance-based feature, which bases on a set of distance features derived from (Nguyen and Nguyen, 2016; Nguyen and Nguyen, 2017). Table 5 shows the features. The detail of each feature can be seen in (Nguyen and Nguyen, 2017).

Table 5: The features; S: a sentence, C: a comment or tweet.

Distance Features	**Lexical Features**
Manhattan distance	The longest common sub string of S and C
Euclidean distance	Inclusion-exclusion coefficient
Cosine similarity	% words of S in C
Word matching	% words of C in S
Dice coefficient	Word overlap coefficient
Jaccard coefficient	—
Jaro coefficient	—
Damerau-Levenshtein	—
Levenshtein distance	—

The distance feature states that a summary sentence and comment should be closer compared to non-summary ones. To compute the distance between two vectors, a similarity score, e.g. Cosine can be used; however, using a single measurement may not efficient enough to completely capture the similarity aspect of a sentence-comment pair. For example, a summary sentence and comment may not share common words due to content variation, which may negatively affect the Cosine calculation. We, therefore, consider the word and character level of a sentence-comment pair by using various distance features. For instance, the Manhattan distance covers pairs those share common words and the Levenshtein distance based on characters treats pairs; those are content variation in sharing common characters. Eq. (9) presents this features:

$$dist(s_i, UG_d) = \max_{j=1}^{m} (distSim(s_i, c_j)) \tag{9}$$

where: m is the number of comments in UG_d; $distSim()$ returns the distance similarity of s_i and c_j and is computed by Eq. (10):

$$distSim(s_i, c_j) = \frac{1}{F} \sum_{n=1}^{F} f_n(s_i, c_j) \tag{10}$$

where: F contains nine distance features in Table 5; $f_n()$ is a similarity function computed by each n^{th} feature.

- *Maximal lexical-based similarity*: This feature also shares the *generation* hypothesis with *distance-based similarity* but using common word aspect. It states that a summary sentence and comment should share common words. Eq. (11) computes the lexical similarity:

$$lex(s_i, UG_d) = \max_{j=1}^{m} (lexSim(s_i, c_j)) \tag{11}$$

This feature was modeled in the same mechanism with the distance feature but using five lexical features in Table 5. By using distance and lexical-based features, our model considers entailment aspect of a sentence-comment (or tweet) pair.

Third-party Features: An event in the original document is also mentioned in different Web documents from other news providers. We define the relation of the original and relevant documents is implicit because all the documents are created independently and social users do not involve the creation. Although the relation is implicit, information from third-party sources helps to reveal potential aspects of the original document. It is possible that third-party features can also be applied to comments; however, they were only used for relevant documents due to the implicit relation. We present new features capturing social voting, social distance, and the appearance of frequent words.

- *Voting:* The hypothesis of this feature is the *reflection* and *generation* observation, in which relevant documents should include salient terms in a summary sentence. Given a sentence s_i in the original document d, the voting counts Cosine similarity (greater a threshold) with sentences (stopword removal) in relevant documents and is shown in Eq. (12).

  $$n\_vote(s_i) = \frac{\sum_{j=1}^{m} cos(s_i, t_j)}{N_S} \tag{12}$$

  where: m is total sentences in relevant documents and N_S is the number of sentences in d. In practice, we empirically set the threshold $= 0.65$ when modeling sentences and 0.35 when modeling comments or tweets.

- *Cluster distance:* The hypothesis of this feature is that a summary sentence should be close to clusters represented by relevant documents. In this view, each relevant document is considered as a cluster. Given a sentence s_i (no stopwords) and a relevant document $rd_j \in D$ represented by a set of frequent terms, Eq. (13) defines this feature.

  $$c\text{-}dist(s_i, D) = \frac{\sum_{j=1}^{N_D} eucDist(s_i, rd_j)}{N_D} \tag{13}$$

  where: $eucDist()$ returns Euclidean distance of s_i and rd_j using bag-of-words model, N_D is the number of relevant documents. The frequency threshold equals 10 for both sentences and comments (tweets).

- *Sentence-third-party term frequency:* represents the relation of a sentence and other sentences in relevant documents in the form of term frequency.

This feature bases on a hypothesis that a sentence containing frequent words appearing in both original document and third-party sources is more important than other. Given a sentence s_i (stopword removal) in the original document d and N_{SD} is total sentences in relevant documents, Eq. (14) defines the term frequency feature.

$$stp\text{-}TF(s_i) = \frac{\sum_{j=1}^{|s_i|} TF(w_j) * IDF(w_j)}{|s_i|} \tag{14}$$

$$TF(w_j) = \text{the frequency of } w_j \text{ in } d \tag{15}$$

$$IDF(w_j) = log(\frac{N_{SD}}{DF(w_j)}) \tag{16}$$

where: $DF(w_j)$ is the number of sentences in D containing word w_j.

- *Frequent-terms probability:* The remaining features base on an assumption that if the most frequent words appear in relevant documents, the summarization should contain these terms because they are essential for both document and global level.

  A set of frequent terms was collected from the raw texts of relevant documents based on frequency. In practice, we set the frequency threshold = 5 for both sentences and comments (or tweets). If a term frequency is greater than a certain threshold, this term was considered to be frequent. Given a frequent term set $FT = \{w_1, ..., w_t\}$ and the original document d, three probability features: the average probability of frequent terms (aFrqScore), frequent term sum score (frqScore), and relative frequent term sum score (rFrqScore) were included (Svore et al., 2007). These features were calculated by Eqs. (17), (19) and (20).

$$aFrqScore(s_i) = \frac{\sum_{w \in s_i} p(w)}{|w \in s_i|} \tag{17}$$

where: $w \in FT$.

$$p(w) = \frac{count(w)}{|w \in d|} \tag{18}$$

where: $count(w)$ is the frequency of w in d, and $|w \in d|$ is total number of frequent words in d.

$$frqScore(s_i) = \sum_{w \in s_i} p(w) \tag{19}$$

$$rFrqScore(s_i) = \frac{\sum_{w \in s_i} p(w)}{|s_i|} \tag{20}$$

Note that our features were also used for comments or tweets in the same mechanism to train the summary model.

4.3.2. Sentence Selection

Score-based ranking: We first considered the extraction by directly using scores generated from the Ranking SVM model. After arranging sentences and comments (or tweets) in a decreased score oder, the summarization was generated by selecting top m sentences and comments (or tweets), which have the highest scores in Eq. (21).

$$S_r \leftarrow ranking(S); \quad T_r \leftarrow ranking(T) \tag{21}$$

where: *ranking()* returns a list of sentences (comment or tweets) in a decreased weight order. After ranking, top m ranked sentences and comments (or tweets) from S_r and T_r were extracted as the summarization.

Majority voting: We further investigated the extraction by using majority voting from L2R models. We run two additional L2R methods: RankBoost (Freund et al., 2003) (iteration = 300, metric is ERR@10), and Coordinate Ascent (CA) (Metzler and Croft, 2007) (random restart = 5, iteration = 25, performance tolerance = 0.001 with no-regularization) implemented in RankLib[19] with the same feature set of Ranking SVM. Final outputs were generated by using majority voting among the three L2R models.

[19]http://people.cs.umass.edu/∼vdang/ranklib.html

Table 6: An example of majority voting. R_k is rank position ($rank\_pos$) of the sentence by ranker $k = (1, 2, 3)$.

Sentences	R_1	R_2	R_3	Voted Rank	Sentence Length
a	1	1	1	3	20
b	2	3	4	9	21
c	4	2	3	9	22
d	3	4	2	9	23

Output ranked sentences: $a > d > c > d$

Assume that there exists three rankers, and each ranker outputs different orders of sentences, Table 6 presents an example of the majority voting. It contains four sentences with different rank positions corresponding to three rankers (R_1, R_2, R_3). It is possible to use scores generated from each rankers to compute the voted rank; however, the scores are computed by different mechanisms in various rages, which are inappropriate to be used in the majority voting. Therefore, the voted rank of a sentence is the sum of ranked position of the sentence generated by each ranker. Eq (22) shows the voting.

$$voted\_rank(s) = \sum_k \alpha_k * rank_k(s) \tag{22}$$

where: α_k is the weight of each ranker, which is set to 1 (it means that the three rankers are equal); k is the number of rankers; $rank_k(s)$ returns the rank position of sentence s using ranker k. The $rank_k(s)$ is defined in Eq. (23).

$$rank_k(s) = rank\_pos(s, k) \tag{23}$$

where: $rank\_pos(s, k)$ returns the ranked position of s in ranker k.

The voting algorithm first selects sentences, which have the highest voted rank. If two sentences have the same voted rank ($voted\_rank(s_i) = voted\_rank(s_j)$), the longer sentence is chosen. For example, in Table 6, sentence a is first selected. Sentences b, c, and d own the same voted rank, i.e. 9; however, our algorithm picks up d because it has the longest length. As a result, the final order of the sentences is $a > d > c > d$.

4.4. Statistical Analysis

This section first introduces baselines used to compare to our method. The baselines include methods, which use inherent information in a document or exploit both inherent and social information. Next we show evaluation metric used to compare our method against the baselines.

4.4.1. Baseline

We compared SoSVMRankSum to baselines in social context summarization. These methods are listed as the following:

- **SentenceLead:** chooses the first m sentences as the summarization (Nenkova, 2005). This method was not used in selecting tweets or comments.
- **LexRank:** was proposed by (Erkan and Radev, 2004). This method builds a stochastic graph-based method for computing relative importance of textual units in text summarization. LexRank considers extractive text summarization by relying on the concept of sentence salience to identify the most important sentences in a document, in which the salience is typically defined by terms. In this study, LexRank algorithm[20] was applied with the usage of tokenization and stemming[21].
- **SVM:** was proposed by (Cortes and Vapnik, 1995) and used by (Yang et al., 2011). The authors trained a binary classifier on training data and applied the classifier on testing data to create the summarization. The summarization was generated by selecting sentences or comments labeled by 1. In our study, LibSVM[22] was used with the RBF kernel. The features are sentence length, sentence position, average TF-IDF score of words in sentence, the number of common words with the title, and log-likelihood generated from document. The features were scaled in [-1, 1]; comments were weighted by 85% as the suggestion of (Nguyen et al., 2016c).

[20]https://code.google.com/p/louie-nlp/source/browse/trunk/louie-ml/src/main/java/org/louie/ml/lexrank/?r=10

[21]http://nlp.stanford.edu/software/corenlp.shtml

[22]http://www.csie.ntu.edu.tw/∼cjlin/libsvm/

- **SoRTESum:** was proposed by (Nguyen and Nguyen, 2016). The method uses one wing (sentences) or dual wing information (sentences and tweets). For example, this method only uses the support from the remaining sentences when calculating the score of a sentence. Similarly, the remaining tweets are also utilized to compute the score of a tweet. It contains two models: using inter information (SoRTESum Inter Wing) and dual wing information (SoRTESum Dual Wing)
- **CRF**: was proposed by (Lafferty et al., 2001) and used in (Shen et al., 2007) for single document summarization. This method formulates the summarization as a sequence labeling task, in which summary sentences are labeled by 1 and non-summary sentences are labeled by 0. A set of basic features in Section 4.3.1 were used to train the labeling model.
- **SVM Ranking**: was recently used in (Nguyen et al., 2016b) for summarization as well as ranking problem. It presents the summarization as a L2R problem and only using local information to train two summary models ($C = 3$ with the linear kernel) for sentences and comments. This method uses the same features in Section 4.3.1 as CRF.

4.4.2. Evaluation Metric

In USAToday-CNN dataset, highlights were used as gold-standard references. In SoLSCSum and VSoLSCSum, selected sentences and comments (those which were labeled by 1 in the annotation step) were used as gold-standard references. For evaluation, F-1 ROUGE-N[23] (Lin and Hovy, 2003)(N=1, 2) was employed, in which ROUGE-N was defined in Eq. (24):

$$ROUGE - N = \frac{\sum_{s \in S_{ref}} \sum_{gram_n \in s} Count_{match}(gram_n)}{\sum_{s \in S_{ref}} \sum_{gram_n \in s} Count(gram_n)} \tag{24}$$

where: n is the length of n-gram, $Count_{match}(gram_n)$ is the maximum number of n-grams co-occurring in a candidate summary and the reference summaries, $Count(gram_n)$ is the number of n-grams in the reference summaries.

[23]http://kavita-ganesan.com/content/rouge-2.0-documentation

5. Results and Discussion

In order to measure our success, in Section 5.1 we show comparison results of SoSVMRankSum with the baselines. The comparison answers two questions: (i) whether the performance of our approach can compare to other methods and (ii) whether our approach is efficient. Section 5.2 investigates the contribution of each feature in our model. We also observe the contribution of relevant documents, the relation between training data size and summary performance, and sentence position. We finally validate our hypotheses by deeply analyzing our model by a running example.

5.1. Experimental Results

We report results of Web document summarization using social context in Tables 7, 8, and 9 with ROUGE-1 and ROUGE-2, in which Tables 7 and 8 show summary performance on the two English datasets and Table 9 presents results on the Vietnamese dataset. Results in Tables 7, 8 indicate that our methods (SoSVMRankSum and voting) clearly outperforms baselines over two datasets except for ROUGE-2 of sentence selection in SoLSCSum. This supports our idea stated in Section 4.1. The performance of sentence selection is better than that in comment or tweet extraction because they are usually generated from document content (Yang et al., 2011; Wei and Gao, 2014; Nguyen and Nguyen, 2016), supporting the *reflection* observation.

The results of score-based ranking and SVM-Rank (basic features) indicate that integrating user-generated and third-party information improves the quality of our summary model, especially in comment or tweet extraction, i.e. 0.341 vs. 0.316 or 0.251 vs. 0.209 of ROUGE-1 in Tables 7 or 8. This shows the efficiency of our proposed features. In sentence selection, adding new features slightly increases summary performance, e.g. 0.427 vs. 0.405 or 0.295 vs. 0.275 of ROUGE-1 in Tables 7 and 8 because basic features are enough efficient to capture summary aspects in each sentence. On the other hand, in comment or tweet extraction, new features from social context boost the summary performance with a big gap compared to SVM-Rank. This is because some basic

Table 7: Summary performance on SoLSCSum; * is supervised methods; **bold** is the best value; *italic* is the second best. Methods with *So* use the support from social information.

Method	Document		Comment	
	ROUGE-1	ROUGE-2	ROUGE-1	ROUGE-2
Sentence Lead	0.365	0.322	—	—
LexRank	0.328	0.257	0.244	0.140
SVM*	0.293	0.239	0.141	0.074
SoRTESum (IW)	0.357	0.299	0.237	0.135
SoRTESum (DW)	0.362	0.302	0.206	0.113
CRF*	0.413	**0.391**	0.074	0.062
SVM-Rank*	0.405	0.339	0.316	0.172
Score-based ranking*	**0.427**	0.345	**0.341**	*0.175*
Voting	*0.418*	*0.351*	*0.321*	**0.178**

Table 8: Summary performance on USAToday-CNN dataset.

Method	Document		Tweet	
	ROUGE-1	ROUGE-2	ROUGE-1	ROUGE-2
Sentence Lead	0.249	0.096	—	—
LexRank	0.183	0.045	0.154	0.056
SVM*	0.262	0.088	0.216	0.073
SoRTESum (IW)	0.255	0.098	0.201	0.068
SoRTESum (DW)	0.254	0.096	0.209	0.074
CRF*	0.232	0.062	0.189	0.052
SVM-Rank*	0.275	0.096	0.209	0.060
Score-based ranking*	*0.295*	*0.107*	**0.251**	**0.081**
Voting	**0.298**	**0.111**	*0.246*	*0.079*

features, e.g. sentence position are inefficient in modeling a comment or tweet. By adding new features, our model can efficiently capture summary aspects. The voting obtains competitive results with score-based ranking, e.g. 0.178 vs.

0.175 in Table 7 or 0.298 vs. 0.295 in Table 8 because it exploits the support from other L2r models in the form of voting. In some cases, score-based ranking achieves better ROUGE-scores, e.g. 0.427 vs. 0.418. It is understandable that in some cases, two incorrect sentence positions can dominate the correct one, leading to a wrong selection. However, compared to the baselines in Tables 7 and 8, the voting still obtains sufficient improvements.

CRF obtains competitive results in document summarization, e.g. 0.391 of ROUGE-2 in Table 7; however, its performance is very poor for comment or tweet extraction, e.g. 0.074 vs. 0.341 or 0.189 vs. 0.251 of ROUGE-1 in Tables 7 and 8. This is because, in sentence selection, CRF exploits sequence aspect in sentences by using several features, e.g. Cosine with the previous and next sentences. On the other hand, CRF is limited for comment or tweet extraction. This is because firstly, the sequence aspect does not explicitly exist in comments or tweets (Figure 10b); therefore, it negatively affects sequence labeling process. Secondly, similar to SVM-Rank, some basic features, e.g. position are inefficient in modeling comments or tweets.

We extensively validated our method on VSoLSCSum to answer two questions: (i) how is the performance of our model against the baselines in Vietnamese and (ii) is the summary performance in non-English consistent with English language. In Table 9, our methods are the best. The comparison trend is consistent with results in Tables 7 and 8 and validate the efficiency of our model. This proves that our methods and features are not only efficient in English but also effective in non-English language, i.e. Vietnamese.

Results from Tables 7, 8, and 9 draw interesting observation. Firstly, score-based ranking, voting, and SVM-Rank obtain very competitive results over the three datasets. This indicates that formulating sentence selection by L2R benefits the summarization. Secondly, CRF with several basic features (Shen et al., 2007) (same features with SVM-Rank) can outperforms almost methods in document summarization; however, in comment or tweet extraction, CRF obtains very poor performance due to lack of sequence aspect. Another point is that SoRTESum (Nguyen and Nguyen, 2016) also obtains promising results even this

Table 9: Summary performance on VSoLSCSum.

Method	**Document**		**Comment**	
	ROUGE-1	ROUGE-2	ROUGE-1	ROUGE-2
Sentence Lead	0.437	0.393	—	—
LexRank	0.471	0.381	0.344	0.246
SVM*	0.505	0.438	0.324	0.181
SoRTESum (IW)	0.471	0.383	0.336	0.233
SoRTESum (DW)	0.486	0.427	0.296	0.203
CRF*	0.378	0.341	0.070	0.052
SVM-Rank*	0.528	0.485	0.343	0.250
Score-based ranking*	*0.548*	*0.500*	*0.362*	*0.271*
Voting	**0.552**	**0.504**	**0.383**	**0.291**

is an unsupervised learning method, showing the helpfulness of social information. LexRank is quite sensitive to data. For example, LexRank outperforms some methods in Table 9; however, its performance is very weak in Table 7 and 8. This is because the noise of social messages challenges the extraction because LexRank uses *IDF-modified-cosine similarity* (Erkan and Radev, 2004). Finally, Sentence Lead is a strong baseline (Nenkova, 2005) because it picks up some first sentences, which usually include important information of a document.

We compared our methods to state-of-the-art models reported by (Nguyen et al., 2016c; Nguyen et al., 2016a; Wei and Gao, 2014) on the three datasets. The results in Table 10 is consistent with the results in Tables 7, 8, and 9, in which our methods are the best. Ranking SVM with sophisticated features (Nguyen et al., 2016b) obtains very competitive results. For example, it is the best in ROUGE-1 of comment extraction on VSoLSCSum. It is understandable that it is also a supervised learning method and exploits sophisticated features from both sentences and comments/tweets. Interestingly, HGRW (Wei and Gao, 2015) performs comparably to other methods even though it is unsupervised. This shows the help of social information. The performance of cc-TAM (Gao et al., 2012) is quite poor because it is designed for multi-document summarization while all the three datasets are for single-document summarization.

Table 10: Our model vs. state-of-the-art methods on the three datasets.

Dataset	Method	Document		Comment	
		R-1	R-2	R-1	R-2
SoLSCSum	cc-TAM (Gao et al., 2012)	0.321	0.268	0.166	0.088
	HGRW (Wei and Gao, 2015)	0.377	0.321	0.248	0.145
	L2R CCF (Wei and Gao, 2014)	0.363	0.321	0.217	0.111
	SoSVMRank (Nguyen et al., 2016b)	0.425	0.323	**0.371**	0.158
	Score-based ranking*	**0.427**	*0.345*	*0.341*	*0.175*
	Voting	*0.418*	**0.351**	0.321	**0.178**
USAToday-CNN	cc-TAM (Gao et al., 2012)	0.261	0.074	*0.248*	0.071
	HGRW (Wei and Gao, 2015)	0.271	0.091	0.207	0.053
	L2R (CCF) (Wei and Gao, 2014)	0.251	0.069	0.238	0.076
	SoSVMRank (Nguyen et al., 2016b)	0.261	0.062	0.233	0.071
	Score-based ranking*	*0.295*	*0.107*	**0.251**	**0.081**
	Voting	**0.298**	**0.111**	0.246	*0.079*
VSoLSCSum	cc-TAM Gao et al., 2012	0.405	0.336	0.199	0.125
	HGRW Wei and Gao, 2015	0.514	0.438	0.362	0.265
	L2R CCF (Wei and Gao, 2014)	0.525	0.478	0.364	0.266
	SoSVMRank (Nguyen et al., 2016b)	0.534	0.489	*0.367*	0.268
	Score-based ranking*	*0.548*	*0.500*	0.362	*0.271*
	Voting	**0.552**	**0.504**	**0.383**	**0.291**

580 The results of SoSVMRank (Nguyen et al., 2016b) and our methods in Table 10 support our idea stated in Section 4.1, in which we extend the model in (Nguyen et al., 2016b) by adding relevant documents. The improvements come from three factors: adding an external information, using different features, and selecting summaries by voting.

585 *5.2. Feature Contribution Analysis*

This section investigates feature contribution in the summary model. We first observed feature weigh in training the model to show the role of each feature. We also compared ROGUE-scores of the model using different feature group. We finally analyzed our features with various L2R methods.

590 *5.2.1. Feature Weight Observation*

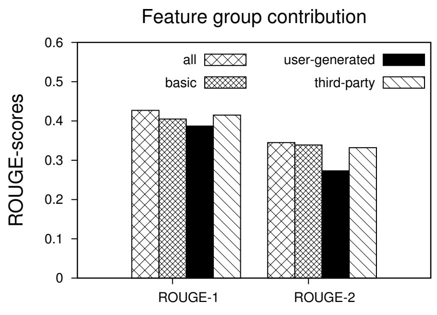
We investigated the contribution of all features by averaging feature weight generated from the model in each fold on SoLSCSum dataset. Table 11 presents top 15 effective features.

Table 11: Top 15 effective features on SoLSCSum dataset; **bold** denotes new features

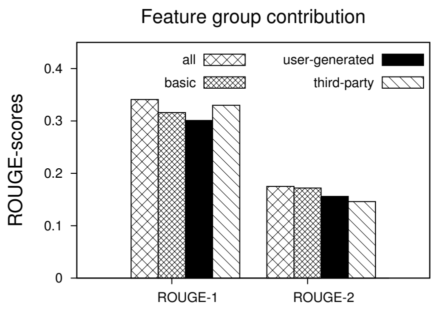
Document		**Comment**	
Feature	**Weight**	**Feature**	**Weight**
HIT-score	1.013	HIT-score	9.404
Cosine voting	0.703	**max-Cosine**	1.982
Cosine (N+1)	0.643	**max-W2V**	0.330
R-P-keyword	0.608	length	0.286
Cosine (N+3)	0.466	**P-keyword**	0.142
length	0.340	# **com-word-title**	0.132
Cosine (N-1)	0.322	# **stopword**	0.112
indicator word	0.311	thematic word	0.065
local-LDA	0.260	**local-LDA**	0.054
aux-LDA	0.260	**aux-LDA**	0.054
max-Cosine	0.256	indicator word	0.039
# **com-word-title**	0.081	log-likelihood	0.027
max-W2V	0.050	**STF-IDF**	0.006
# **stopword**	0.032	**R-P-keyword**	0.004
log-likelihood	0.032	**Cosine voting**	0.001

In sentence selection, HIT-score has the biggest feature weight. Our proposed features also play an important role by holding large feature weights, e.g. Cosine voting or R-P-keyword. Most of basic features are in top eight explaining the results in Table 7, in which CRF and SVM-Rank without new features obtain very competitive performance. Our new features appear in the rest. This supports the results in Table 7, in which adding new features slightly improves the summary performance, e.g. 0.421 vs. 0.405 of ROUGE-1. On the other hand, in comment extraction, our new features locate in top eight, e.g. max-Cosine or max-W2V. This explains that integrating the support from sentences and relevant documents boots the quality of summarization, e.g. 0.341 vs. 0.316 of ROUGE-1. Similar to sentence selection, HIT-score also obtains the highest weight. Table 11 also indicates that the contribution of each feature for sentence and comment extraction is different. This suggests that different features should be used for sentences or comments (tweets).

We also observed the contribution of feature groups. The observation was conducted by running SVM-Rank with four settings: using basic features, user-generated features, third-party features, and all features.



(a) Sentence selection



(b) Comment extraction

Figure 3: The contribution of feature groups

For document summarization in Figure 3a, summary performance indicates that training model with using only user-generated features outputs poor results in both ROUGE-1 and ROUGE-2. This is because these features modeling a sentence by using cross relations with comments, e.g. max-Cosine. They do not consider inherent characteristics of a sentence such as sentence length or

HIT-score. Model with basic features generates competitive results compared to running all features (see Table 7). As mentioned in Section 5.1, several summary aspects can be represented by basic features such as sentence length, thematic words, or HIT-score. Interestingly, the model using third-party features is also comparable to the model using all features, e.g. 0.427 vs. 0.415 of ROUGE-1 in Figure 3a. This is because relevant documents include salient phrases which can be captured by using statistical features. This also shows the efficiency of relevant documents in generating summary sentences (Svore et al., 2007).

For comment extraction in Figure 3b, the trend of summary performance in ROUGE-1 is similar to sentence selection in Figure 3a. For instance, the model using third-party features can be compared to the model using all features, e.g. 0.330 vs. 0.341. However, in ROUGE-2, the model with third-party features generates a poor result because the content mentioned in comments covers a wider range compared to sentences. For example, readers usually show their opinions along with the mention of an event content. As a result, third-party features are inefficient to capture summary aspects of each comment.

5.2.2. Summarization Observation with L2R Methods

We observed the performance with three L2R methods: RankBoost, Coordinate Ascent, and Ranking SVM on the three datasets using all features. Table 12 shows that Ranking SVM outperforms RankBoost and Coordinate Ascent in almost cases. For instance, Ranking SVM is the best on SoLSCSum and obtains competitive results on the two remaining datasets. The general trend in Table 12 also indicates that the gap among L2R methods is small; however, compared to ROUGE-scores of baselines in Tables 7, 8, and 9, the L2R methods in Table 12 obtains sufficient improvements. This suggests that modeling sentence selection in form of L2R benefits the summarization.

We further investigated the contribution of our features over the three L2R methods. This investigation answers a question that whether our features are efficient with other L2R methods. We run the three L2R methods in two settings: (i) using all features, including the new features and (ii) using the basic

Table 12: L2R methods on the three datasets.

Dataset	Method	Document		Comments/ tweets	
		R-1	R-2	R-1	R-2
SoLSCSum	RankBoost	0.414	0.344	0.285	0.158
	Coordinate Ascent	0.413	0.337	0.296	0.158
	SVMRank	0.427	0.345	0.341	0.175
USAToday-CNN	RankBoost	0.295	0.108	0.238	0.073
	Coordinate Ascent	0.294	0.106	0.246	0.078
	SVMRank	0.295	0.107	0.251	0.081
VSoLSCSum	RankBoost	0.540	489	0.365	0.274
	Coordinate Ascent	0.548	0.499	0.363	0.278
	SVMRank	0.548	0.500	0.362	0.271

features. Table 13 shows the contribution of our features. Due to the space limitation, we only show result on SoLSCSum dataset.

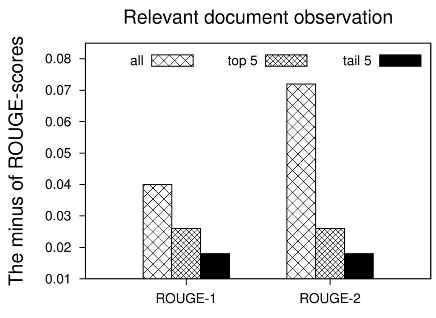
Table 13: The performance of L2R methods on SoLSCSum dataset.

Method	Document		Comment	
	ROUGE-1	ROUGE-2	ROUGE-1	ROUGE-2
RankBoost	**0.417**	0.342	0.283	0.153
RankBoost (with new features)	0.414	**0.344**	**0.285**	**0.158**
Coordinate Ascent	0.402	0.328	0.274	0.155
Coordinate Ascent (with new features)	**0.413**	**0.337**	**0.296**	**0.158**
SVM-Rank	0.405	0.339	0.316	0.172
SVM-Rank (with new features)	**0.421**	**0.345**	**0.341**	**0.175**

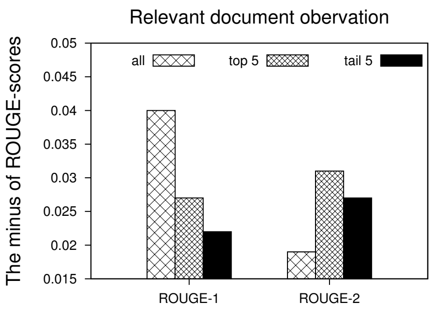
Results in Table 13 show that in the most cases, our features contribute to improve summary performance. The improvement in comment extraction is larger than that in sentence selection. This is because some basic features, e.g. sentence position are inefficient in modeling social messages. However, the gap between L2R methods is small compared to results in Table 7. Results from Table 13 indicate that: (i) SVM-Rank obtains the best performance compared to the two remaining ones and (ii) adding additional features from social context improves the ROUGE-scores of SVM-Rank as well as other L2R methods.

5.3. Relevant Document Analysis

We analyzed the relevant documents to answer a question that how do the relevant documents affect the summarization. We divided the relevant documents of SoLSCSum into two sets: top and tail five documents, and combined with original documents and their comments. The combination creates three settings: (i) using all relevant documents, (ii) using top five relevant documents, and (iii) using tail five relevant documents. We trained three L2R models under the three settings with all features. The ROUGE-scores of each model were normalized by the minus with the model only using user-generate features shown in Figures 3a and 3b.



(a) Sentence selection



(b) Comment extraction

Figure 4: The observation of relevant documents with ROUGE-scores.

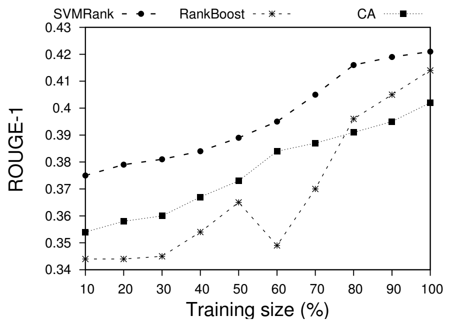
Figures 4a and 4b indicate that using all relevant documents improves the summary performance. It is understandable that all the third-party features are statistical; therefore, the large number of sentences benefits feature calculation. When cutting off the relevant documents, the summary performance reduces. For example, the ROUGE-scores of top 5 are lower than that of using all relevant documents, e.g. 0.026 vs. 0.04 in ROUGE-1 of sentence selection. The performance of top 5 is better than that of the tail 5 because the former usually includes documents which are more relevant than the latter. This is because a search engine usually returns irrelevant documents which include noise in the tail of research result page. This suggests that the more relevant documents are used, the more summary performance can be improved.

5.4. Training Data Observation

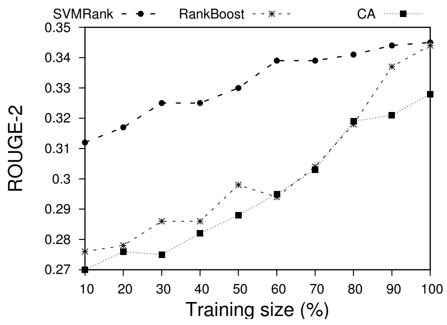
This sections presents our investigation of training data aspect. The investigation answers two questions: (i) how does training data-size affect the summarization and (ii) what is the relation of training data and training time.

5.4.1. Data Size and Performance

We conducted an observation of data-size in training the three L2R methods to reveal the relation of training data and summary performance. We used the data in each fold as the testing set and the remaining data as training set. In each fold, we randomly picked up a subset from the training set for learning. The subset ranges from 10% to 100% of the original training set. The performance at each training size is the average ROUGE-scores of 10-folds. Due to the space limitation, we only show the observation on SoLSCSum dataset.



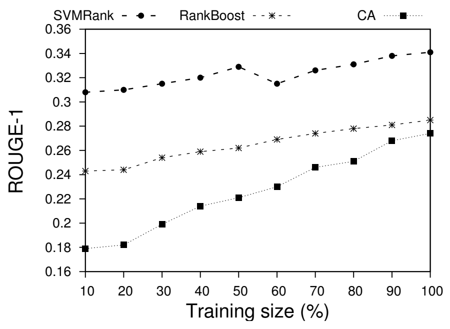
(a) ROUGE-1 of sentence selection



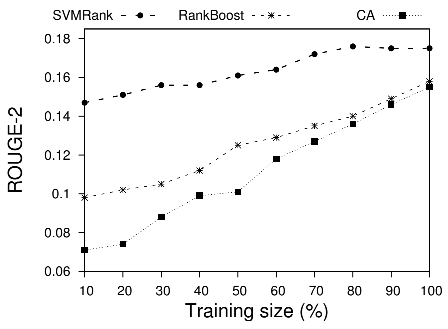
(b) ROUGE-2 of sentence selection

Figure 5: Training data-size in sentence selection.

The general trend in Figures 5a, 5b, 6a, and 6b indicates that adding training data improve the summary performance. It is understandable that putting



(a) ROUGE-1 of comment extraction



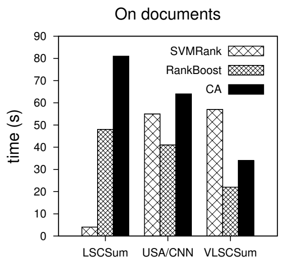
(b) ROUGE-2 of comment extraction

Figure 6: Training data-size in comment extraction.

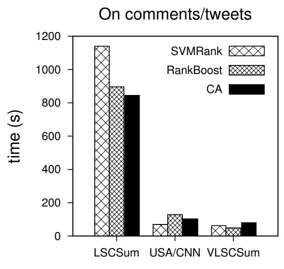
more training data allows L2R methods to learn more correctly data distribution. As a result, the methods can efficiently predict unseen data in the testing set. However, in some cases, adding more training data reduces the summary performance due to the noise of data, e.g. from 50% to 60% of RankBoost in Figure 5a. The trend also indicates that there is a big gap of Ranking SVM and the other methods during the training process, e.g. Figure 5b. It shows the efficiency of Ranking SVM for the summarization.

5.4.2. Training Time Analysis

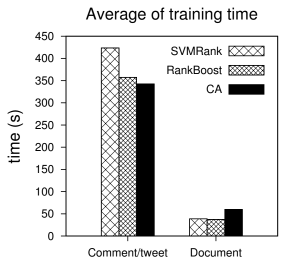
We analyzed time aspect in training L2R methods. Figure 7 presents the training time when using 100% training data on the three datasets.



(a) Sentence selection



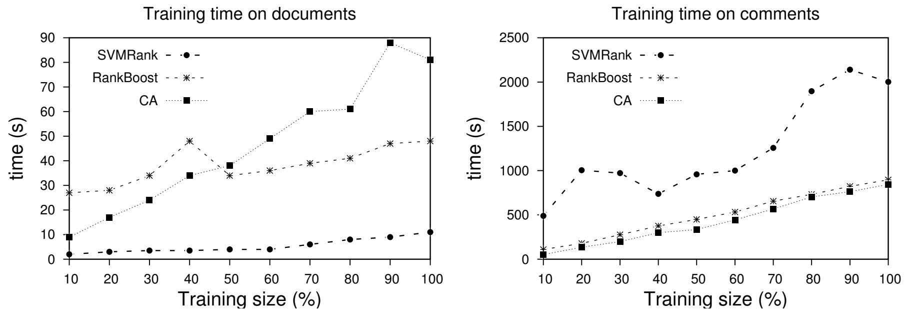
(b) Comment extraction



(c) Average

Figure 7: Training time on the three datasets. CA is Coordinate Ascent.

The trend in Figures 7a and 7b shows that training a model for sentences takes a shorter time compared to training for comments or tweets. For example, Coordinate Ascent spends 80 seconds for training with sentences (Figure 7a) while it needs more than 800 seconds to complete when training with comments (Figure 7b). The training time in Figure 7c supports this observation. In Figure 7c, there is no difference when training L2R models with sentences because the number of sentence is small. In comments, the training time dramatically increases, in which Ranking SVM needs more time than RankBoost and Coordinate Ascent.



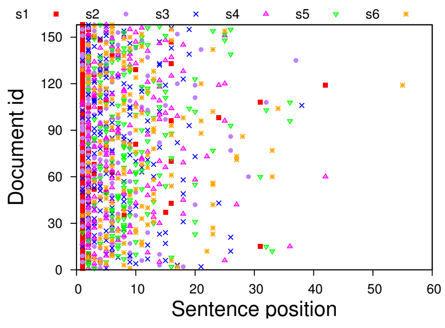
(a) Sentence selection (b) Comment extraction

Figure 8: Training time with different data size.

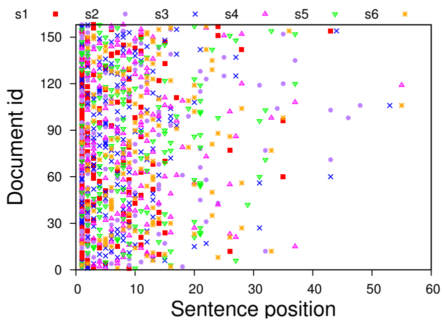
We zoomed in the training time of the L2R methods on SoLSCSum. The training time in each data point was observed. The general trend in Figures 8a and 8b is consistent with results in Figures 7a and 7b, in which Ranking SVM is the fastest in training with sentences; however, in comments, it takes more time compared to the other ones. It is understandable that in practice, Ranking SVM based on SVM spends a lot of time to find out optimal hyper-planes to separate the training data.

5.5. Sentence Position Observation

We also investigated sentence position of output summaries generated from CRF and our method on SoLSCSum dataset. This investigation reveals the role of position in a strong and weak method. We matched extracted sentences and comments from the two methods again the original documents to obtain their positions. Figures 9 and 10 shows the position observation.

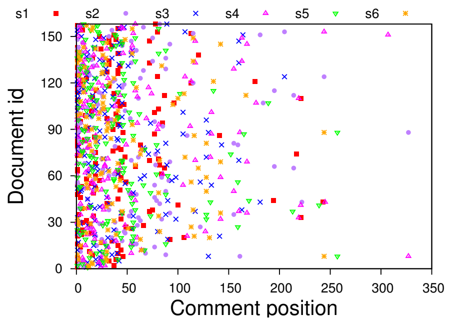


(a) Score-based ranking

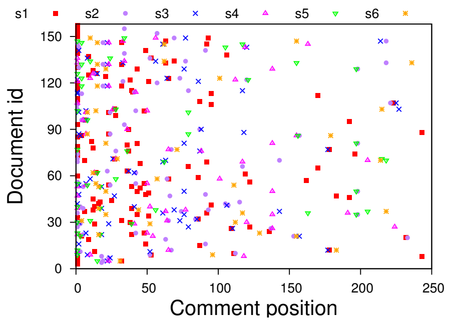


(b) CRF

Figure 9: Sentence selection.



(a) Score-based ranking



(b) CRF

Figure 10: Comment extraction.

For sentence selection in Figures 9a and 9b, we observe that CRF and our method output a similar position distribution, in which the most summary sentences locate within the first 10 sentences. This explains that score-based ranking slightly outperforms CRF (see Tables 7, 8, and 9). There are also some outlier points, e.g. 55^{th} in Figures 9a or 9b because some documents contain a larger number of sentences and comments. On the other hand, for comment extraction in Figures 10a and 10b, the trend is inconsistent. The comment distribution generated from our method is dense whereas the output of CRF is so sparse. This is because the lack of sequence aspect in comments limits CRF and explains that its performance is very poor on comments or tweets. By using new features from sentences (as social information for comments) and third-party relevant documents, we aim to address this problem.

Data observation from Figures 9 and 10 suggests three important points. Firstly, summary sentences appear in the first part of a Web document (usually in top 10). This supports the results of Lead-m method in Tables 7, 8, and 9 because Sentence Lead generates the summarization which completely matches with data distribution in sentences. However, our method still outperforms Lead-m because score-based ranking integrates social context and formulates the summarization in a learning to rank task. Secondly, Figures 10a and 10b indicate that representative comments appear in a wider range compared to sentences; therefore, Sentence Lead is inefficient to summarize comments. Finally, sentence position is an important feature in text summarization.

5.6. Hypothesis Analysis

We deeply analyzed our hypotheses in stated Section 4.2.2 by running an example implemented from (Nguyen and Nguyen, 2016). The example contains two sentences and tweets shown in Table 14, in which S_1 and T_2 are summary sentences and S_2 and T_1 are non-summary sentences.

Table 14: An example of the Boston Bombing (24) in the USAToday-CNN dataset.

Sentences	**Tweets**
[S1] Police have identified Tamerlan Tsarnaev as the dead Boston bombing suspect	[T1] Who is Tamerlan Tsarnaev, 26, the man ID&#39 as the dead #BostonBombing
[S2] The brothers had been living together on Norfolk Street in Cambridge	[T2] Before his death Tamerlan Tsarnaev called an uncle andasked for his forgiveness. Said he is married and has a baby

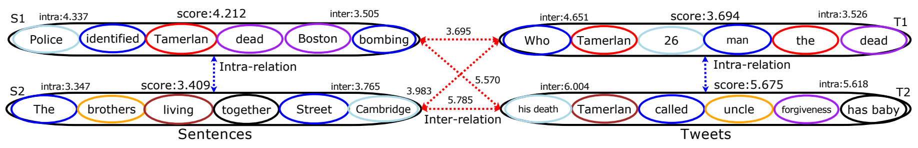


Figure 11: A running example generated by SoRTESum Dual-Wing.

Figure 11 indicates that the summary sentences, i.e. S_1 and T_2 receive higher scores compared to non-summary sentences, i.e. S_2 and T_1. This validates our idea stated in Section 4.1. In addition, $T1$ and $T2$ contain important information of the Boston bombing event, e.g. the name of terrorist. This supports the *representation* and *reflection* hypothesis. We also observe that sentences and tweets share common words, e.g. *"Tamerlan"*, *"bombing"*, *"dead"* supporting the *generation* and *common topic* hypothesis.

5.7. Error Analysis

We deeply observed the output from our model on SoLSCSum dataset. We show three extracted sentences and comments instead of six due to space limitation. In Table 15, score-based ranking selects two correct sentences and comments (denoted by [+]) which clearly mention the event of Boston man shot by police. This is because summary sentences and comments contain important words *"Boston"*, *"police"* and *"arrest"*, which appear frequently in comments and relevant documents; therefore, our features, e.g. max-lexical similarity, social voting, or sentence-third-party term frequency can efficiently capture these sentences. In addition, max-W2V score feature can represent a sentence-comment pair containing words *"police"* and *"arrest"* by using semantic similarity. This leads to the improvement in Table 7.

On the other hand, non-summary sentences (denoted by [-]), e.g. S3 and C3 also including salient words challenge our method. For example, S3 contains *"Evans"*, *"officers"*, and *"weapons"*; therefore, our features such as local-LDA, max-Cosine, max-W2V or Cosine-voting are inefficient in this case. In addition, the similarity of sentence length in S3 and C3 also challenges our method. However, these sentences are still relevant to the event.

Score-based ranking generates C1 and C2 which perfectly reflects the content of original document. While S1 and S2 show a complete story of the event, C1 and C2 provide a shorter summary, which helps people to rapidly understand the story of this event. In this case, summary comments can be represented as the first layer of the summarization, which provides a snapshot and summary

Table 15: An example of 121th document showing three extracted sentences and comments.

Summary	
Sentences	**Comments**
[+]S1: The 26-year-old man, identified as Usaamah Rahim, brandished a knife and advanced on officers working with the Joint Terrorism Task Force who initially tried to retreat before opening fire, Boston Police Superintendent William Evans told reporters	[+]C1: "Boston Police and State Police made an arrest this evening in Everett"
[+]S2: Boston Police said in a statement on their website that "as part of this ongoing investigation, Boston Police and State Police made an arrest this evening in Everett	[+]C2: Boston man shot by police was target of terrorism probe
[-]S3: Evans said officers had approached the man in a strip-mall parking lot without weapons drawn and opened fire only after he repeatedly advanced on them, leaving them in fear for their lives	[-]C3: War on BS is what the American people need to adopt when they hear those escape paths being uttered by anyone..

sentences can be seen as the second layer, which people can zoom in the story of an event. C3 shows the opinion of readers after reading the document. It can support sentences to provide a perspective viewpoint of the event.

6. Conclusion

This paper presents a novel L2R-based summary framework which exploits the social context of a Web document to generate a high-quality summarization. Our framework presents sentences, user posts, and relevant documents in a mutual reinforcement fashion. This paper concludes that integrating social context and formulating a sentence selection by a L2R task with sophisticated features benefit the summarization. In the first aspect, social context supports local information for better capturing summary aspects in summary sentences and comments (or tweets). The social context not only comes from user posts but also is from sentences due to the mutual reinforcement support. In the second aspect, combining features allows to integrate human-knowledge into the summary process. Our methods are extensively evaluated on three datasets in two languages: English and Vietnamese. Experimental results show that SoSVMRankSum achieves improvements over state-of-the-art baselines and our features are efficient for single-document summarization.

For future direction, an obvious step is to investigate how the our framework works to other domains and text genres. The framework is straightforward to integrate any additional features. e.g tree edit distance. Finally, we suggest that our problem should also be represented in a deeper model, e.g. LSTM or CNN to enrich semantics aspect.

Acknowledgment

This work was supported by JSPS KAKENHI Grant number 15K16048, JSPS KAKENHI Grant Number JP15K12094, and CREST, JST. We would like to thank Chien-Xuan Tran for building the Web interface.

References

Amitay, E., & Paris, C. (2000). Automatically summarising web sites: is there a way around it? In *Proceedings of the Ninth International Conference on Information and Knowledge Management (CIKM)*, (pp. 173–179). ACM.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of machine Learning research*, *3*(Jan), 993–1022.

Cao, Z., Wei, F., Dong, L., Li, S., & Zhou, M. (2015, February). Ranking with Recursive Neural Networks and Its Application to Multi-Document Summarization. In *AAAI*, (pp. 2153–2159).

Conroy, J. M., & O'leary, D. P. (2001, September). Text summarization via hidden markov models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (pp. 406–407). ACM.

Cortes, C., & Vapnik, V. (1995). Support-Vector Networks. *Machine Learning*, *20*(3), 273–297.

Cao, Z., Qin, T., Liu, T. Y., Tsai, M. F., & Li, H. (2007, June). Learning to Rank: from Pairwise Approach to Listwise Approach. In *Proceedings of the 24th International Conference on Machine Learning (ICML)*, (pp. 129–136). ACM.

Dagan, I., Dolan, B., Magnini, B., & Roth, D. (2010). Recognizing textual entailment: Rational, evaluation and approaches - Erratum. *Natural Language Engineering*, *16*(1), 105–105.

Delort, J. Y., Bouchon-Meunier, B., & Rifqi, M. (2003). Enhanced Web Document Summarization Using Hyperlinks. In *Proceedings of the Fourteenth ACM Conference on Hypertext and Hypermedia*, (pp. 208–215). ACM.

Delort, J. Y. (2006). Identifying commented passages of documents using implicit hyperlinks. In *Proceedings of the Seventeenth Conference on Hypertext and Hypermedia*, (pp. 89–98). ACM.

Edmundson, H. P. (1969). New Methods in Automatic Extracting. *Journal of the Association for Computing Machinery (JACM)*, *16*(2), 264–285.

Erkan, G., & Radev, D. R. (2004). Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, *22*, 457–479.

Freund, Y., Iyer, R., Schapire, R. E., & Singer, Y. (2003). An Efficient Boosting Algorithm for Combining Preferences. *Journal of Machine Learning Research*, *4(Nov)*, 933–969.

Gao, W., Li, P., & Darwish, K. (2012). Joint Topic Modeling for Event Summarization across News and Social Media Streams. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management (CIKM)*, (pp. 1173–1182). ACM.

Gong, Y., & Liu, X. (2001). Generic Text Summarization using Relevant Measure and Latent Semantic Analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (pp. 19–25). ACM.

Hu, M., Sun, A., & Lim, E. P. (2008). Comments-Oriented Document Summarization: Understanding Document with Readers' Feedback. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (pp. 291–298). ACM.

Hu, P., Sun, C., Wu, L., Ji, D. H., & Teng, C. (2011). Social Summarization via Automatically Discovered Social Context. In *IJCNLP*, (pp. 483–490).

Joachims, T. (2006, August). Training linear SVMs in Linear Time. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (pp. 217–226). ACM.

Jones, K. S. (2007). Automatic summarising: The state of the art. *Information Processing & Management, 43(6)*, 1449–1481.

Kupiec, J., Pedersen, J., & Chen, F. (1995). A Trainable Document Summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (pp. 68–73). ACM.

Lafferty, J., McCallum, A., & Pereira, F. (2001, June). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the eighteenth international conference on machine learning, (ICML)*, (Volume 1, pp. 282–289).

Lin, H., & Bilmes, J. (2011). A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, (Volume 1, pp. 510–520). Association for Computational Linguistics.

Lin, C. Y., & Hovy, E. (2003). Automatic Evaluation of Summaries Using N-gram Co-occurrence Statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology* (Volume 1, pp. 71–78). Association for Computational Linguistics.

Lu, Y., Zhai, C., & Sundaresan, N. (2009). Rated aspect summarization of short comments. In *Proceedings of the 18th International Conference on World Wide Web (WWW)*, (pp. 131–140). ACM.

Luhn, H. P. (1958). The Automatic Creation of Literature Abstracts. *IBM Journal of Research Development*, *2*(2), 159–165.

McCallum, A. K. (2002) Mallet: A Machine Learning for Language Toolkit.. *http://mallet.cs.umass.edu*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality. In *Advances in neural information processing systems (NIPS)*, (pp. 3111–3119).

Nallapati, R., Zhai, F., & Zhou, B. (2016) SummaRuNNer: A Recurrent Neural Network based Sequence Model for Extractive Summarization of Documents. In *AAAI 2017*.

Nenkova, A. (2005). Automatic text summarization of newswire: lessons learned from the document understanding conference. In *AAAI*, (Vol.5, pp. 1436–1441).

Nenkova, A., & McKeown, K. (2011). Automatic summarization. In *Foundations and Trends in Information Retrieval, 5(23)*, 103–233).

Nguyen, M. T., Tran, V. D., Tran, C. X., & Nguyen, M. L. (2017). Summarizing Web Documents using Sequence Labeling with User-generated Content and Third-party Sources. In *Proceedings of NLDB*. Springer International Publishing.

Nguyen, M. T., & Nguyen, M. L. (2017). Intra-relation or inter-relation?: Exploiting social information for Web document summarization. In *Expert Systems with Applications, 76*, 71–84.

Nguyen, M. T., Lai, V. D., Do, P. K., Tran, D. V., & Nguyen, M. L. (2016). VSoLSCSum: Building a Vietnamese Sentence-Comment Dataset for Social Context Summarization. In *Proceedings of the 12th Workshop on Asian Language Resources (ALR)*, (pp. 2409–2412). Association for Computational Linguistics.

Nguyen, M. T., Tran, V. D., Tran, C. X., & Nguyen, M. L. (2016). Learning to Summarize Web Documents using Social Information. In *Proceedings of ICTAI*, (pp. 619-626). IEEE.

Nguyen, M. T., Tran, C. X., Tran, D. V., & Nguyen, M. L. (2016). Solscsum: A linked sentence-comment dataset for social context summarization. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM)*, (pp. 2409–2412). ACM.

Nguyen, M. T., & Nguyen, M. L. (2016). SoRTESum: A Social Context Framework for Single-Document Summarization. In *European Conference on Information Retrieval (ECIR)*, (pp. 3–14). Springer International Publishing.

Metzler, D., & Croft, W. B. (2007). Linear feature-based models for information retrieval. *Information Retrieval, 10(3)*, 257–274).

Mihalcea, R., & Tarau, P. (2004, July). TextRank: Bringing order into texts. Association for Computational Linguistics.

Osborne, M. (2002). Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization* (Volume 4, pp. 1–8). Association for Computational Linguistics.

Porter, M. F. (2011). Snowball: A language for stemming algorithms.

Shen, D., Sun, J. T., Li, H., Yang, Q., & Chen, Z. (2007). Document Summarization Using Conditional Random Fields. In *IJCAI*, (Vol.7, pp. 2862–2867).

Svore, K. M., Vanderwende, L., & Burges, C. J. (2007, June). Enhancing Single-Document Summarization by Combining RankNet and Third-Party Sources. In *EMNLP-CoNLL*, (pp. 448–457).

Sun, J. T., Shen, D., Zeng, H. J., Yang, Q., Lu, Y., & Chen, Z. (2005). Web-page summarization using clickthrough data. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (pp. 194–201). ACM.

Wei, Z., & Gao, W. (2014). Utilizing Microblogs for Automatic News Highlights Extraction. In *COLING*, (pp. 872–883). Association for Computational Linguistics.

Wei, Z., & Gao, W. (2015). Gibberish, Assistant, or Master?: Using Tweets Linking to News for Extractive Single-Document Summarization. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, (pp. 1003–1006). ACM.

Woodsend, K., & Lapata, M. (2010, July). Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL)*, (pp. 565–574). Association for Computational Linguistics.

Woodsend, K., & Lapata, M. (2012, July). Multiple aspect summarization using integer linear programming. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, (pp. 233–243). Association for Computational Linguistics.

Yang, Z., Cai, K., Tang, J., Zhang, L., Su, Z., & Li, J. (2011). Social Context Summarization. In *Proceedings of the 34th International ACM SIGIR Confer-*

ence on Research and Development in Information Retrieval, (pp. 255–264). ACM.

Yeh, J. Y., Ke, H. R., Yang, W. P., & Meng, I. H. (2005). Text summarization using a trainable summarizer and latent semantic analysis. *Information Processing & Management*, *41*(1), 75–95.

Zhang, Y., Er, M. J., Zhao, R., & Pratama, M. (2016). Multiview Convolutional Neural Networks for Multidocument Extractive Summarization. *IEEE Transactions on Cybernetics*.