Towards Social Context Summarization with Convolutional Neural Networks

Minh-Tien Nguyen[1 2], Duc-Vu Tran[1], Viet-Anh Phan[1], and Le-Minh Nguyen[1]

[1] Japan Advanced Institute of Science and Technology (JAIST), 1-1 Asahidai, Nomi, Ishikawa, 923-1292, Japan.
[2] Hung Yen University of Technology and Education, Hung Yen, Vietnam.
`{tiennm,vu.tran,anhphanviet,nguyenml}@jaist.ac.jp`

Abstract. This paper studies Web document summarization by exploiting social information. The motivation comes from the fact that social context of a Web document provides additional information to enrich the content of sentences. To take advantage of such information, we design a model based on Convolutional Neural Networks. Unlike traditional ones, our model enhances representation of sentences by mutually combining internal and social information. The model learns to rank sentences and user posts and then extracts top ranked ones as a summary. Experimental results on three datasets in two languages confirm the efficiency of our model in summarizing single documents.

Keywords: Summarization, Convolutional Neural Networks.

1 Introduction

In the context of social media, there exists a relationship between a document and its relevant information. For example, after reading the `Asiana Airlines Flight 214 crash` event, readers can write their comments on the event by using the Web interface or post tweets on their Twitter timeline. In the meantime, the content of the main document can be found in relevant articles published by different news providers. The combination of Web documents and their related information defines a summarization task which takes advantage of social information to extract high-quality summaries [23,3,21,17].

Social context The social context of a Web document d is $C_d = \langle S_d, R_d, UP_d \rangle$, where S_d is a set of sentences in d, R_d is relevant articles of d, and UP_d is a set of user posts, e.g. tweets or comments of d [23].

The task Given a document d and its context C_d, the task is to extract m important sentences from S_d and m representative user posts from UP_d [23,21,17]. The intuition is that user posts can enrich extracted sentences in providing new information which is not usually available in the main document.

This paper introduces a model based on Convolutional Neural Networks (CNN) [6] to build a summarizer, which exploits social context of Web documents to extract high-quality summaries. The motivation comes from the fact that social context provides additional valuable information for enriching the meaning of primary documents. To take advantage of such information, we employ CNN to learn hidden features of sentences and user posts. More precisely, our model captures n-grams and combines them to enhance hidden representation. To integrate the support of external information, we incorporate surface features into the hidden representation to create final vectors. Finally, our model learns to rank sentences and user posts and extracts top m ranked ones as a summary. This paper makes the following main contributions:

– It studies the task of summarizing Web documents by exploiting their social information with the use of CNN. To the best our knowledge, no existing methods address this task by using CNN.
– It provides a way to enhance the representation of sentences and user posts, which benefits the learning process. The enhancement is in two levels: enriching hidden representation learned by the model and integrating social information in the form of features.
– It investigates the influence of features, which reveals the contribution of each information channel in our model and releases two improved datasets including related articles of main documents.[3]

We apply our model to the task of sentence and highlight extraction on three datasets, in English and Vietnamese. Our model achieves competitive ROUGE-scores compared to advanced methods in summarizing Web documents.

2 Related Work

Web document summarization has recently been enriched by using social information [20,4,9,23]. Researchers integrate social information by using supervised methods to build dual wing factor graphs [23], or presenting features for modeling cross relationships between sentences and tweets [21]. By contrast, several unsupervised models have been also presented [3,22,7] such as building a cross-collection topic-aspect for a co-ranking method to extract sentences and tweets [3], creating heterogeneous graphs of random walks to summarize single documents [22], using integer linear programming [7], or matrix co-factorization [12].

The system of Nguyen et al. [17,15] is perhaps the most relevant work to our study. It integrates sentences, relevant social messages, and related articles in a unified model, which extracts features from three information channels. The authors use CRF and Ranking SVM to train summarizers to output scores used to select top-ranked sentences and user posts. In this work, we extend the work of [17,15] by adapting a different learning algorithm based on CNN. We also encode social context into our model by integrating sophisticated features adopted from [17,15] to enrich the representation of sentences and user posts.

3 Summarization with CNN and Social Context

This section shows our proposal to address the task in three steps: data preparation, sentence ranking and selection, and the settings of experiments.

[3] We provide main codes, features, and data at https://goo.gl/rC6C8n.

3.1 Data Preparation

Since DUC datasets lack social information, we used three other ones for social context summarization. *SoLSCSum* [14] is an English dataset collected from Yahoo News, containing sentences and comments, which were manually annotated. To validate our model in a non-English language, we used a Vietnamese dataset created for social context summarization named *VSoLSCSum* [13]. It was collected from several Vietnamese Web pages.[4]

Table 1: Statistical observation on the two datasets.

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

We also used *USAToday-CNN* [18] derived from [21] for news highlight extraction. The dataset contains 121 events collected from USAToday and CNN.

We followed [15] in exploiting relevant Web documents as third-party sources to enrich feature extraction. To do that, we used the provided dataset of SoLSCSum from [15]. It includes primary documents, relevant user posts, and related articles. For VSoLSCSum and USAToday-CNN, we used guideline from [15] to retrieve top 10 relevant articles from Google by searching the title of the main document (Table 1). Finally, each dataset contains three parts: main documents, relevant user posts (comments/tweets), and related articles.

3.2 Sentence Ranking

Our model receives documents and their social context for ranking. To do that, we adapted CNN for our summarization task because it has shown the ability to effectively learn n-gram sequences, which help to generate rich textual representation and to tackle sentences with variable lengths naturally. For ranking, our model combines two types of features: (i) latent textual features generated from CNN with customized pooling operation and (ii) surface features extracted from social context. The combination of the two types allows our model to learn the representation of sentences and user posts effectively. We next describe our model in two steps: convolution and polling, and social context integration.

Convolution and Pooling Our model adapts CNN with customized pooling operation for generating latent features, representing textual information from both sentences and user posts.

Convolution. Let h is the kernel size (window size) and $M \in \mathbb{R}^{N \times k}$ is an embedding matrix where N is the max sentence length and k is the dimension of word embedding, the convolution operation applies the kernel size h to each position i of M to produce non-linear transformation of the input [6].

$$c_i^h = f(\boldsymbol{W}^h \cdot v_i + b) \tag{1}$$

[4] We acknowledge [14,13] for sharing data partition of SoLSCSum and VSoLSCSum.

where $f()$ is a non-linear function, e.g. *tanh()*, $\boldsymbol{W}^h$ is the kernel weights, b is the bias term, and v_i is an input corresponding to a sub-region of size h. The feature map of a kernel with size h is $\boldsymbol{C}^h = [c_1^h, ..., c_{N-h+1}^h]$. To obtain rich representation, we adopted multi-window-size filters from Kim. (2014) [5] in order to formulate n-grams of an input sentence.

Pooling. The pooling is the second layer of a CNN-based model, which receives the output of the convolution. It includes down-sampling operation to modify the output of the convolution for retaining most important features from a feature map. For our purpose, we adapted the pooling operation in two phases: (i) max-over-time pooling and (ii) cross-window pooling. The first pooling outputs the most essential features $\widehat{c}^h$ from the feature map $\boldsymbol{C}^h$.

$$\widehat{c}^h = max\{\boldsymbol{C}^h\} \tag{2}$$

The second pooling phase applies a max-pooling function over representation from different window size filters. Our intuition is that the model can, across window sizes, obtain richer hidden representation which benefits the learning process. We denote this operation as cross-window max-pooling, which extracts rich and representative information across all window sizes. Let $\widehat{\boldsymbol{C}} = \{\widehat{c}^h\}$ be a set of all different important features of window sizes h, and $\widehat{\boldsymbol{C}}_1^k, ..., \widehat{\boldsymbol{C}}_N^k$ be all the size-k subsets of $\widehat{\boldsymbol{C}}$ where $|\widehat{\boldsymbol{C}}_i^k| = k$. The output of the second pooling is:

$$x_p = [max\{\widehat{\boldsymbol{C}}\}; max\{\widehat{\boldsymbol{C}}_1^k\}; ...; max\{\widehat{\boldsymbol{C}}_N^k\}] \tag{3}$$

The vector x_p is the concatenation of latent features extracted by concatenating all feature maps $\widehat{\boldsymbol{C}}$ and a subset of feature maps. Our pooling operation shares the idea with [1] which also applies pooling on different window sizes. However, their pooling operation only selects one value from all window sizes leading to least information kept. By contrast, our pooling operates not only on the all but also on the subsets of the windows sizes. This allows our model to enhance the hidden representation of sentences.

Social context incorporation As discussed, there exists relationships between Web documents and their related information, e.g. relevant social messages and articles. We, therefore, take advantage of such related information to enrich the hidden representation learned by CNN. To do that, we adopted features from [15]. Given a sentence in the main document, we extracted three types of features from three channels: local features, relevant user-posts features, and third-party features (related articles). Local features individually measure the importance of a sentence in the main document (or a comment/tweet), without considering relevant user posts or third-party sources. User-post features present relationships between sentences in a primary document and its user posts. Third-party features estimate the importance of a sentence (or a user post) by using relevant documents. Table 2 shows our features.

Suppose $\boldsymbol{x}_e$ is the concatenation of three vectors $\boldsymbol{x}_l$, $\boldsymbol{x}_s$, and $\boldsymbol{x}_t$ denoted for the three feature groups correspondingly, the final representation of a sentence

Table 2: Local (LF), social (SF), and third-party features (TPF).

Group	Feature	Description
LF	Position	The sentence position of x_i in a document (pos = 1, 2, or 3).
	Length	The number of terms appearing in s_i after removing stop words.
	Title-words	#common words in s_i and the title after removing stop words.
	#Stopwords	Number of stopwords in a sentence s_i.
	HIT-score	The HIT score of a sentence s_i.
	LSA-score	The LSA score of a sentence s_i.
	Cosine	Cosine similarity with N next and previous sentences (N=1,2,3).
	Them-words	Count #frequent words calculated by TF appearing in x_i.
	In-words	Whether x_i contains indicator words appearing in a dictionary.
	Up-words	Whether x_i contains upper case words e.g., proper names.
UPF	Max-cosine	Maximal Cosine of a sentence s_i with social information.
	Max-dist	Maximum distance of a sentence s_i with social information.
	Max-lexical	Maximal lexical score of s_i with social information.
	Max-W2V	Maximal W2V score of a sentence s_i with social information.
TPF	Voting	#sentences in relevant documents having Cosine $\geq$ a threshold regarding to s_i.
	Cluster-dist	The Euclidean distance from s_i to relevant documents.
	stp-TF	The score of s_i in relevant documents calculated by TF-IDF.
	aFrqScore	The average probability of frequent words in relevant documents regarding to s_i.
	frqScore	The probability of frequent words in supporting documents regarding to s_i.
	rFrqScore	The relative probability of frequent words in supporting documents regarding to s_i.

(or a user post) is the concatenation of $\boldsymbol{x}_p$ and $\boldsymbol{x}_e$.

$$\boldsymbol{x}_e = [\boldsymbol{x}_l; \boldsymbol{x}_s; \boldsymbol{x}_t] \tag{4}$$

$$\boldsymbol{\Theta} = [\boldsymbol{x}_p; \boldsymbol{x}_e] \tag{5}$$

By integrating indicators extracted from three channels, our model not only enriches vector representation but also incorporates social context into the summarization process. The final vector representation $\boldsymbol{\Theta}$ of a sentence s_i is fed to two MLP layers for regression. Also, note that we present sentences and user posts in a mutual support fashion. When modeling a user post, sentences and related articles are used as supporting information with the same feature set.

3.3 Sentence Selection

We separately trained two models, for sentences and user posts. We used scores generated from the models for ranking. Summaries are extracted by selecting top m ranked sentences and user posts.

3.4 Settings and Evaluation Metrics

Settings We used 10-fold cross-validation for SoLSCSum with $m = 6$ as the suggestion of [14]; 5-fold cross-validation for VSoLSCSum and USAToday-CNN datasets. We set $m = 6$ for VSoLSCSum as the same setting in [13] and $m = 4$ for USAToday-CNN because each Web document has 3-4 highlights. Stopwords and links were removed. Table 3 summarizes parameters used in our model.

Table 3: Settings of our model.

Parameter	Value
Batch size	50
Number of epochs	25
Word dimension (Word2Vec [10])[5]	25
Hidden size of two MLP layers	20, 1
Learning rate	1.0
Dropout rate	0.5
CNN kernels	$h \in \{1, 2, 3\}$
Subset size k of the second pooling	2
Optimizer	Adadelta
The loss function	cross-entropy

In training, 15% training data were randomly selected to form a validation set which helps our model to avoid over-fitting.

Evaluation metrics Gold-standard references in SoLSCSum and VSoLSCSum, and highlights in USAToday-CNN were used for evaluation by using ROUGE-N F-1 [8] implemented in ROUGE-1.5.5 (N=1, 2).[6]

4 Results and Discussion

This section reports our experiments in three aspects: ROUGE-scores, the analysis of feature contribution, and output observation.

4.1 ROUGE-scores Observation

CNN-based models We first compared our model to PriorSum, which also uses CNN for ranking sentences [1]. Table 4 reports our comparison. ROUGE-scores show two interesting points. Firstly, as our expectation, our model is consistently better than PriorSum. Improvements come from two factors. (i) Besides directly combining feature maps from tri-grams, we create a subset combination, which improves the representation of each input sentence. (ii) Human knowledge in the form of features enriches vector representation, which benefits in estimating the importance of each sentence and user post. For example, our model is better 2.6% of ROUGE-1 than PriorSum for sentence selection on USAToday-CNN. As mentioned, the small number of training examples on our datasets (see Table 1) challenges PriorSum and our model. Hence, adding features profits the estimation. For user post extraction, the trend is consistent. Our model also surpasses the basic one in all cases. Secondly, our model with features achieves better ROUGE-scores than the model without using features. It is understandable that deep learning learns representation from data; therefore, small training examples challenge such models. In this aspect, additional features from social context improve the performance of our model.

[6] The parameters are ‘‘`-c 95 -2 -1 -U -r 1000 -n 2 -w 1.2 -a -s -f B -m`"

Table 4: Our model vs. PriorSum. RG is ROUGE. Value means our model is significantly better with $p \leq 0.05$ using the pairwise t-test. F stands for features.

Method	**SoLSCSum**				**VSoLSCSum**				**USAToday-CNN**			
	Sentence		**Comment**		**Sentence**		**Comment**		**Sentence**		**Tweet**	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2
PSum	0.413	0.367	0.211	*0.157*	0.543	*0.447*	0.318	0.161	0.214	0.071	0.242	0.082
Ours	*0.424*	*0.380*	*0.213*	0.156	*0.550*	0.446	**0.430**	*0.253*	*0.234*	**0.088**	*0.246*	**0.089**
Ours-F	**0.428**	**0.386**	**0.233**	**0.177**	**0.554**	**0.451**	**0.430**	**0.290**	**0.240**	*0.087*	**0.250**	**0.089**

Our model vs. non-social context methods We next report the comparison between our model and non-social context methods. **Lead-m** selects top m sentences as a summary [11]. **LexRank** uses a stochastic graph-based method for computing relative importance of textual units for extractive summarization [2]. **SVM**[7] uses a RBF kernel to train a summarizer by using features in [23]. **CRF** employs a set of features in [19] for single document summarization. Table 5 shows ROUGE-scores of these methods.

Table 5: Our model vs. non-social context methods; * is supervised methods; **bold** is the best and *italic* is the second best. Lead-m is not used for user posts. Value means our model is significantly better with $p \leq 0.05$.

Method	**SoLSCSum**				**VSoLSCSum**				**USAToday-CNN**			
	Sentence		**Comment**		**Sentence**		**Comment**		**Sentence**		**Tweet**	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2
Lead-m	0.345	0.322	—	—	0.495	0.420	—	—	0.249	**0.106**	—	—
LRank	0.327	0.243	0.210	0.115	0.506	0.432	**0.348**	*0.198*	*0.251*	*0.092*	0.193	0.068
SVM*	0.325	0.263	0.152	0.089	0.497	0.440	0.374	0.212	**0.261**	**0.106**	0.221	*0.084*
CRF*	0.393	0.379	0.091	0.075	0.422	0.357	0.111	0.062	0.186	0.088	0.190	0.065
Ours*	*0.424*	*0.380*	*0.213*	0.156	*0.550*	0.446	**0.430**	*0.253*	0.234	0.088	*0.246*	**0.089**
Ours-F*	**0.428**	**0.386**	**0.233**	**0.177**	**0.554**	**0.451**	**0.430**	**0.290**	0.240	0.087	**0.250**	**0.089**

ROUGE-scores from Table 5 indicate a consistent trend with Table 4, in which our model is the best in almost all cases, except for sentence selection on USAToday-CNN. For example, our model is better than CRF, which is a very strong baseline on SoLSCSum, e.g. 0.428 vs. 0.393. It is similar to VSoLSCSum, in that two CNN-based methods significantly surpass baselines (values with text are significant with $p \leq 0.05$). This again confirms the efficiency of our model, which takes advantage of CNN for capturing internal features and exploits social context for providing additional useful information.

On USAToday-CNN, on the one hand, SVM is the best, followed by LexRank for sentence selection. It is possible to explain that features used by SVM are appropriate for this dataset. However, on other datasets, SVM does not prove to be efficient. Our model is in the third position even it uses many sophisticated features. This is because all our features are for extraction, but this dataset is of abstraction, which also challenges other advanced methods. For example, ROUGE-scores of RankBoost CCF [21] are lower than Lead-m (see Table 6). This also shows the limitation of CNN in capturing hidden features. On the

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

other hand, for tweet extraction, our methods are the best. The reason is that we enhance vector representation by using a combination of feature maps and integrating social context denoted by features.

Our model vs. social context methods We final challenged our model with advanced methods. **cc-TAM**[8] builds a cross-collection topic-aspect modeling for creating a bipartite graph used by co-ranking [3]. **HGRW** is a variation of LexRank named Heterogeneous Graph Random Walk [22]. **RankBoost (RB)** exploits set of local and cross features trained by RankBoost[9] for training two L2R models [21], for sentences and tweets. **SVMRank** is an extension of RB, that the authors added more advanced features [16]. **Voting** bases on three L2R models [15]. Table 6 shows their ROUGE-scores.

Table 6: CNN-based models vs. social context methods.

Method	**SoLSCSum**				**VSoLSCSum**				**USAToday-CNN**			
	Sentence		**Comment**		**Sentence**		**Comment**		**Sentence**		**Tweet**	
	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2	RG-1	RG-2
cc-TAM	0.306	0.238	0.054	0.022	0.488	0.377	0.301	0.167	0.261	0.074	*0.248*	0.071
HGRW	0.379	0.204	0.209	0.115	0.570	0.479	0.454	0.298	*0.279*	*0.098*	0.242	0.088
RB*	0.360	0.283	0.190	0.098	0.561	0.494	*0.471*	*0.308*	0.221	0.070	0.233	*0.091*
SVMR*	0.381	0.304	0.209	0.122	*0.572*	*0.521*	**0.482**	**0.319**	0.253	0.084	0.213	0.080
Voting	0.401	0.322	0.223	0.132	**0.576**	**0.523**	0.467	**0.319**	**0.287**	**0.114**	0.243	**0.095**
Ours*	*0.424*	*0.380*	*0.213*	0.156	0.550	0.446	0.430	0.253	0.234	*0.088*	0.246	0.089
Ours-F*	**0.428**	**0.386**	**0.233**	**0.177**	0.554	0.451	0.430	0.290	0.240	0.087	**0.250**	0.089

Results from Table 6 are similar to those in Table 5, in which our model produces promising results. For example, it is competitive on SoLSCSum, and on USAToday-CNN for tweet extraction. Among social context methods, HRGW shows its efficiency, in which it achieves good performance on three datasets. This is because HRGW extends LexRank to utilize the social information of a Web document in an appropriate form. cc-TAM is the second best of ROUGE-1 for tweet selection on USAToday-CNN. Results from unsupervised methods motivate that we can improve their quality to reach the performance of supervised learning ones by using social information in a suitable way. L2R-based models also obtain consistent results. As discussed, it is trained with sophisticated features leading to improvements compared to unsupervised learning models, e.g. cc-TAM. Our model is not the best on VSoLSCSum because of a possible reason that we can not extract some features from this dataset, e.g. indicator words, due to the limitation of tools in Vietnamese.

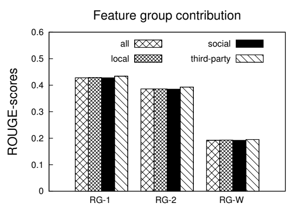
4.2 Feature Contribution

We observed the contribution of feature groups in our model. To do that, we ran the model with four settings, using: (i) all features, (ii) local features, (iii) user-post features, and (iv) using third-party features. It is possible to observe the

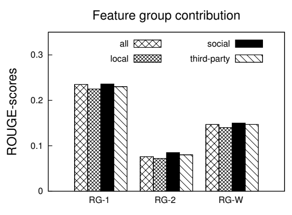
[8] We acknowledge Gao et al., for sharing two parts of code [3].

[9] https://people.cs.umass.edu/∼vdang/ranklib.html

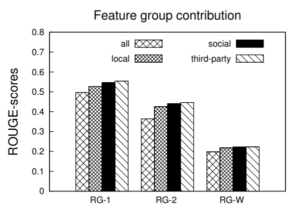
influence of each feature; however, due to training time, we leave this observation as a future task. We plot our observation on Figs. 1 and 2.



(a) SoLSCSum

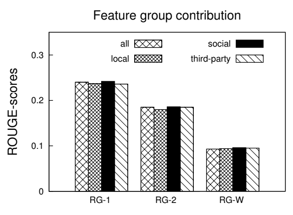


(b) USAToday-CNN

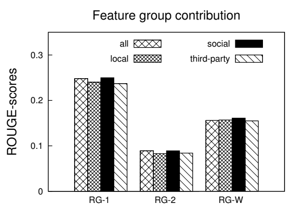


(c) VSoLSCSum

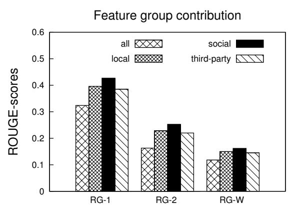
Fig. 1: Feature group contribution of sentence selection.



(a) SoLSCSum



(b) USAToday-CNN



(c) VSoLSCSum

Fig. 2: Feature group contribution of user post extraction.

The general trend on SoLSCSum and USAToday-CNN indicates that: (i) there are small margins among feature groups and (ii) using all and user-post features seems to be more efficient than other groups. This is because they include some good indicators, e.g. sentence length, which can support latent features extracted by CNN. By contrast, the tendency on VSoLSCSum is different, indicating that using all features obtains the worst results. The reason may come from the conflict when combining many features. ROUGE-scores also show that using user-post and third-party features may be appropriate for this dataset.

4.3 Output Observation

Table 7 shows outputs of CNN-based methods extracted from the document *"Seconds before crash, passengers knew they were too low"* (the *Asiana Airlines Flight 214 crash* event). We can observe that extracted sentences and tweets provide useful information for readers on this event. This shows the efficiency of two methods in extracting summaries. Their outputs also share common extracted sentences and tweets because they have similar behavior of extraction. This comes from the reason that they employ CNN for learning. On the other hand, they also extract different outputs, indicating that two models produce different summaries even their scores are quite similar.

For sentence selection, PriorSum extracts S2 and S3, which seem to be relevant to the highlights. By reading these sentences, we can partly guess the situation of the event. However, it selects S1, containing information in the past, which may support the main story but it does not directly relate to the event. This is similar to S4, which shows the opinion of Hayes-White. By contrast, outputs of our model may be closer to the highlights than those of PriorSum. For example, we know the total number of passengers (in S3) or what happened with flight recorders (in S2). They provide richer information than PriorSum.

Table 7: A summary example of the document 14th on USAToday-CNN.

Highlights
Flight recorders have been found, the NTSB says.
Asiana identifies the two 16-year-old girls killed in the crash.
182 people were hospitalized, while 123 were uninjured.
Passengers say the plane's rear struck the edge of the runway.
Outputs of PriorSum
Sentence selection
S1: In 1993, a crash near South Korea's Mokpo Airport killed 68 of the 116 people on board.
S2: The flight recorders from the plane have been recovered and are on the way to Washington, the NTSB said Sunday.
S3: The Boeing 737-500 went down in poor weather as the plane was attempting its third landing, the Aviation Safety Network said.
S4: "We're lucky there hasn't been a greater loss of life," San Francisco Fire Chief Joanne Hayes-White said.
Tweet extraction
T1: Seconds before crash, passengers knew they were too low - Asiana Airlines Flight 214 was seconds away from landing...
T2: NTSB says Asiana Airlines Flight 214 flight recorders have been found. - @CNN.
T3: That had to be scary! "@cnnbrk: Flight recorders recovered from San Francisco crash site, NTSB says.
T4: Seconds before crash, passengers knew they were too low Look these pics! A terrible plane accident in San Francisco.
Outputs of our model with features
Sentence selection
S1: Perhaps one of the reasons so many people survived Saturday's crash was because the Boeing 777 is built so that everybody can get off the plane within 90 seconds, even if half the doors are inoperable.
S2: The flight recorders from the plane have been recovered and are on the way to Washington, the NTSB said Sunday.
S3: Once the plane fell short of the runway, passengers found themselves on a roller coaster.
S4: Asiana Airlines Flight 214 was seconds away from landing when the passengers sensed something horribly amiss.
Tweet extraction
T1: NTSB says Asiana Airlines Flight 214 flight recorders have been found. - @CNN.
T2: That had to be scary! "@cnnbrk: Flight recorders recovered from San Francisco crash site, NTSB says.
T3: 2 teens killed in San Francisco plane crash; 182 hospitalized - CNN: ABC News2 teens killed in San Francisco.
T4: Seconds before crash, passengers knew they were too low - Asiana Airlines Flight 214 was seconds away from landing...

For tweet extraction, two methods output valuable tweets, which include important information. For example, T2 of PriorSum or T1 of our model completely match with the first highlight. Our model selects two very important tweets: T1 and T3. While T1 is similar to PriorSum, T3 provides critical information, which totally matches with the second and the third highlight. It includes the status of two teens (*"killed"*), the name of the event (*"San Francisco plane crash"*), and the number of victims (*"182 hospitalized"*). Extracted tweets from our model can cover almost important information from the highlights. Two methods also share some common tweets due to the use of CNN.

5 Conclusion

This paper presents a CNN-based model for summarizing Web documents by exploiting their social context. The model provides a way to enhance the representation of sentences and user posts by combining hidden and external features. The enhancement benefits the learning process to capture more important information. We carefully conduct experiments on three datasets in two languages, English and Vietnamese. Promising results confirm that our model can take advantage of social context to improve the quality of sentence ranking, which benefits to extract high-quality summaries. Applying the model to sentence and highlight extraction tasks of single documents shows that it can be viable alternative to extraction-based systems.

Future work will investigate features to reveal the role of each indicator. Our model should integrate LSTM to exploit the contextual sentence relations.

Acknowledgment

This work was supported by JSPS KAKENHI grant numbers JP15K16048 and JP15K12094, and Center for Research and Applications in Science and Technology, Hung Yen University of Technology and Education, under the grant number UTEHY.T026.P1718.04.

References

1. Ziqiang Cao, Furu Wei, Sujian Li, Wenjie Li, Ming Zhou, and Houfeng Wang. Learning summary prior representation for extractive summarization. In *ACL (2): 829-833*, 2015.
2. Gunes Erkan and Dragomir R. Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research, 22: 457-479*, 2004.
3. Wei Gao, Peng Li, and Kareem Darwish. Joint topic modeling for event summarization across news and social media streams. In *CIKM:1173-1182*, 2012.
4. Meishan Hu, Aixin Sun, and Ee-Peng Lim. Comments-oriented document summarization: Understanding document with readers' feedback. In *SIGIR: 291-298*, 2008.
5. Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP: 1746-1751*, 2014.
6. Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86(11), pp. 2278-2324. IEEE*, 1998.

7. Chen Li, Zhongyu Wei, Yang Liu, Yang Jin, and Fei Huang. Using relevant public posts to enhance news article summarization. In *COLING: 557-566*, 2016.
8. Chin-Yew Lin and Eduard H. Hovy. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *HLT-NAACL: 71-78*, 2003.
9. Yue Lu, ChengXiang Zhai, and Neel Sundaresan. Rated aspect summarization of short comments. In *WWW: 131-140*, 2009.
10. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS: 3111-3119*, 2013.
11. Ani Nenkova. Automatic text summarization of newswire: lessons learned from the document understanding conference. In *AAAI: 1436-1441*, 2005.
12. Minh-Tien Nguyen, Tran Viet Cuong, Nguyen Xuan Hoai, and Minh-Le Nguyen. Utilizing user posts to enrich web document summarization with matrix co-factorization. In *The Eight International Symposium on Information and Communication Technology (SoICT), pp. 70-77*, 2017.
13. Minh-Tien Nguyen, Viet Dac Lai, Phong-Khac Do, Duc-Vu Tran, and Minh-Le Nguyen. Vsolscsum: Building a vietnamese sentence-comment dataset for social context summarization. In *The 12th Workshop on Asian Language Resources: 38-48*, 2016.
14. Minh-Tien Nguyen, Chien-Xuan Tran, Duc-Vu Tran, and Minh-Le Nguyen. Solscsum: A linked sentence-comment dataset for social context summarization. In *CIKM: 2409-2412*, 2016.
15. Minh-Tien Nguyen, Duc-Vu Tran, and Le-Minh Nguyen. Social context summarization using user-generated content and third-party sources. *Knowledge-Based System, 144, pp. 51-64*, 2018.
16. Minh-Tien Nguyen, Duc-Vu Tran, Chien-Xuan Tran, and Minh-Le Nguyen. Learning to summarize web documents using social information. In *ICTAI: 619-626*, 2016.
17. Minh-Tien Nguyen, Duc-Vu Tran, Chien-Xuan Tran, and Minh-Le Nguyen. Summarizing web documents using sequence labeling with user-generated content and third-party sources. In *NLDB: 454-467*, 2017.
18. Minh-Tien Nguyen, Duc-Vu Tran, Xuan-Chien Tran, , and Le-Minh Nguyen. Exploiting user-generated content to enrich web document summarization. *International Journal on Artificial Intelligence Tools, 26(5), pp. 1-26*, 2017.
19. Dou Shen, Jian-Tao Sun, Hua Li, Qiang Yang, and Zheng Chen. Document summarization using conditional random fields. In *IJCAI: 2862-2867*, 2007.
20. Jian-Tao Sun, Dou Shen, Hua-Jun Zeng, Qiang Yang, Yuchang Lu, and Zheng Chen. Web-page summarization using clickthrough data. In *SIGIR: 194-201*, 2005.
21. Zhongyu Wei and Wei Gao. Utilizing microblogs for automatic news highlights extraction. In *COLING: 872-883*, 2014.
22. Zhongyu Wei and Wei Gao. Gibberish, assistant, or master?: Using tweets linking to news for extractive single-document summarization. In *SIGIR: 1003-1006*, 2015.
23. Zi Yang, Keke Cai, Jie Tang, Li Zhang, Zhong Su, and Juanzi Li. Social context summarization. In *SIGIR: 255-264*, 2011.