Improving Multi-Document Summarization via Text Classification

Ziqiang Cao[1,2] **Wenjie Li**[1,2] **Sujian Li**[3] **Furu Wei**[4]
[1]Department of Computing, The Hong Kong Polytechnic University, Hong Kong
[2]Hong Kong Polytechnic University Shenzhen Research Institute, China
[3]Key Laboratory of Computational Linguistics, Peking University, MOE, China
[4]Microsoft Research, Beijing, China
{cszqcao, cswjli}@comp.polyu.edu.hk
lisujian@pku.edu.cn
fuwei@microsoft.com

Abstract

Developed so far, multi-document summarization has reached its bottleneck due to the lack of sufficient training data and diverse categories of documents. Text classification just makes up for these deficiencies. In this paper, we propose a novel summarization system called TCSum, which leverages plentiful text classification data to improve the performance of multi-document summarization. TCSum projects documents onto distributed representations which act as a bridge between text classification and summarization. It also utilizes the classification results to produce summaries of different styles. Extensive experiments on DUC generic multi-document summarization datasets show that, TCSum can achieve the state-of-the-art performance without using any hand-crafted features and has the capability to catch the variations of summary styles with respect to different text categories.

Introduction

The increasing online information has necessitated the development of effective automatic multi-document summarization systems. Through long-term research, the learning-based summarization approaches have grown to become dominant in the literature. By far, a prominent issue that hinders the further improvement of supervised approaches is the lack of sufficient human summaries used for training (**?**). For instance, the widely-used DUC[1] generic multi-document summarization benchmark datasets contain less than 400 human reference summaries in total. Writing summaries is an extremely labor-intensive and time-consuming process. Because of the limitation of training data, a learning-based summarization system is often forced to heavily rely on well-designed features. Simple models like Support Vector Regression can achieve the state-of-the-art performance with extensive linguistic and statistical features (**?**). To break through the bottleneck of insufficient summarization training data, taking advantage of other rich data sources might be a good idea worth considering.

Meanwhile, existing summarization approaches basically apply a uniform model to generate summaries for the documents in different text categories. However, according to what we observe, summary styles in different categories can vary to a large degree. Take the two common categories in DUC datasets, i.e., Natural Disaster and Biography as an example. To summarize a natural disaster like a hurricane, people tend to present its moving path and the loss it brings. By contrast, a biography summary is expected to include the personal profile and the main contributions of the person. Apparently, summaries should focus on different aspects of the topics which belong to the corresponding categories. When the document category is given, (**?**) finds that the introduction of category-specific language models largely promotes the summarization performance. The experiments of (**?**) also show that a summarization model with good overall performance still produces low-quality summaries in certain document sets. The summary style issue previously mentioned may partly explain these phenomena and suggest a possible way to improve the summarization performance.

Compared with summarization, the text classification datasets are much richer. Note that both summarization and text classification require models to understand the semantics of documents. Better text representations learned by classification data can help to train more effective summarization models. Moreover, if we know the category of a document, we will have a chance to explore more proper summary styles. To this end, we propose a novel summarization system called TCSum, which leverages text classification data to improve the performance of summarization. Since distributed representations of documents have demonstrated advantages in both summarization (e.g., (**?**)) and text classification (e.g., (**?**)), TCSum projects all documents onto the distributed representations that are shared by the two tasks. Then, for text classification, the document embeddings are followed by a classifier to learn their association with the categories. For summarization, the document embeddings are transformed to match the "meaning" of the reference summaries. To make the transformed embeddings also hold the information of summary styles, we utilize the classification result and develop a category-specific transformation process. Our model adopts the recent hot topic of neural network based transfer learning (e.g., from syntactic parsing to discourse parsing (**?**)). It is also noted that our model is totally data-driven, i.e., all the abstract features are

[1]http://duc.nist.gov/

learned automatically.

We verify the effectiveness of TCSum on DUC generic summarization benchmark datasets. TCSum is able to compete with state-of-the-art summarization systems which usually heavily depends on hand-crafted features. We also observe that TCSum indeed catches the variations of the summary styles among different text categories.

The contributions of this paper are listed as follows:

- We leverage text classification datasets to learn better document representations for summarization.
- We explore the variations of summary styles with respect to different text categories.
- We develop a competitive summarization system which does not need any hand-crafted features.

Method

Let D denote a document which is composed of a set of sentences $\{s_i|i \in [1, N]\}$. For text classification, we use C to stand for the entire set of categories. We assume D belongs to one of C, i.e., $c_D \in [1, |C|]$ where c_D represents the actual category for of the document D. The **text classification model** is trained to predict a category for D. For supervised sentence ranking required by learning-based summarization, each sentence holds a saliency score, usually measured with respect to the human summaries (hereafter the reference summaries). The **summarization model** is expected to learn how to rank sentences in accord with the actual sentence saliency.

In this section, we describe how our summarization system, called TCSum, ranks the sentences with the help of text classification. The overall framework of TCSum is illustrated in Fig. 1. At first, a **text classification model** is trained using a convolutional neural network. This model projects a document onto the distributed representation, and adds a softmax classifier to predict the category of the document. The **summarization model** shares the same projection process to generate document embeddings given that the semantic analysis and understanding of documents are essential for both classification and summarization. Afterwards, it transforms the document embedding to the summary embedding and tries to maximize the match to the "meaning" of the reference summaries. To make the transformed summary embedding sensitive to the different summary styles, TCSum learns category-specific transformation matrices according to the predicted categories. Finally, the sentences are ranked according to their saliency scores calculated based on the similarity between the sentence embedding and the summary embedding. The rest of this section describes the details of our model.

Text Classification Model

Convolutional Neural Networks (CNNs) can learn the abstract representations of N-grams effectively and tackle the sentences with variable lengths naturally. Models using CNNs have achieved excellent performance both in text classification (**?**) and summarization (**?**). In this paper, we

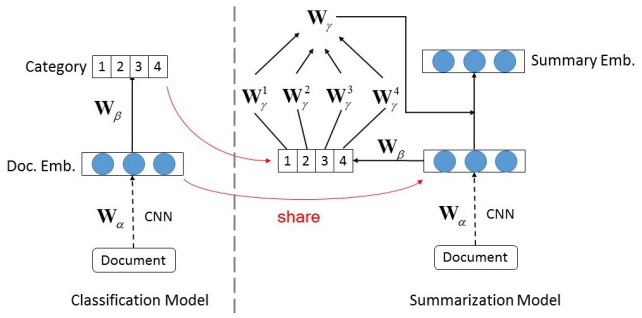

Figure 1: Overview of TCSum.

develop a simple CNN-based classification model. Specifically, we use a CNN to project a sentence s onto its distributed representation $\mathbf{v}(s) \in \mathbb{R}^m$, i.e.,

$$s \xrightarrow{\text{CNN}} \mathbf{v}(s) \tag{1}$$

A basic CNN contains a convolution operation on the top of word embeddings, which is followed by a pooling operation. Let $\mathbf{v}(w_i) \in \mathbb{R}^k$ refer to the k-dimensional word embedding corresponding to the i_{th} word in the sentence. Let $\mathbf{v}(w_i : w_{i+j})$ be the concatenation of word embeddings $[\mathbf{v}(w_i), \cdots, \mathbf{v}(w_{i+j})]$. A convolution operation involves a filter $\mathbf{W}_\alpha \in \mathbb{R}^{m \times hk}$, which is applied to a window of h words to produce the abstract features $\mathbf{g}_i^h \in \mathbb{R}^m$,

$$\mathbf{g}_i^h = f(\mathbf{W}_\alpha \times \mathbf{v}(w_i : w_{i+j})), \tag{2}$$

where $f(\cdot)$ is a non-linear function and the use of $tanh$ is the common practice. To make it simple, the bias term is left out. This filter is applied to each possible window of words in the sentence to produce a feature map. Subsequently, a pooling operation is applied over the feature map to obtain the final features $\hat{\mathbf{g}}^h \in \mathbb{R}^m$ of the filter. Here we use the max-over-time pooling (**?**).

$$\hat{\mathbf{g}}^h = \max\{\mathbf{g}_1^h, \mathbf{g}_2^h, \cdots\} \tag{3}$$

The primary purpose of this pooling is to capture the most important features in a feature map. $\hat{\mathbf{g}}^h$ is the output of the CNN, i.e., the embedding of a sentence.

Then a document is represented by the average pooling of its sentence embeddings, just like (**?**),

$$\mathbf{v}(D) = \frac{1}{|D|} \sum\nolimits_{s \in D} \mathbf{v}(s) \tag{4}$$

To learn the association between the document embedding and the categories, the document embedding is followed by a softmax classifier:

$$\mathbf{v}_C(D) = \text{softmax}(\mathbf{W}_\beta \times \mathbf{v}(D)), \tag{5}$$

where $\mathbf{W}_\beta \in \mathbb{R}^{|C| \times m}$ is the weight matrix, and $\mathbf{v}_C(D) \in \mathbb{R}^{|C|}$ is the predicted probability distribution over the categories.

Summarization Model

As previously mentioned, the summarization model in TCSum, shares the same convolution and pooling operations with the classification model when generating the document embedding $\mathbf{v}(D)$. Then, TCSum transforms $\mathbf{v}(D)$ to match the "meaning" of the reference summary, i.e.,

$$\mathbf{v}_S(D) = \tanh(\mathbf{W}_\gamma \times \mathbf{v}(D)), \tag{6}$$

where $\mathbf{v}_S(D) \in \mathbb{R}^m$ is the transformed embedding called summary embedding, and $\mathbf{W}_\gamma \in \mathbb{R}^{m \times m}$ is the transformation matrix. Note that we define the same dimension for both document and summary embeddings. This setting simplifies the sentence ranking process, which is explained later.

We would also like the summary embedding to hold the information of summary styles. Inspired by the work of (**?**), we develop the category-specific transformation matrix $\mathbf{W}_\gamma$ according to the predicted category. We introduce $|C|$ sub-matrices $(\mathbf{W}_\gamma^1, \cdots, \mathbf{W}_\lambda^{|C|})$, with each directly corresponding to one text category. Based on the predicted category derived from Eq. 5, the transformation matrix $\mathbf{W}_\gamma$ is computed as the weighted sum of these sub-matrices.

$$\mathbf{W}_\gamma = \sum\nolimits_{i=1}^{|C|} \mathbf{v}_C^i(D)\mathbf{W}_\gamma^i \tag{7}$$

In this way, $\mathbf{W}_\gamma$ is automatically biased to the sub-matrix of the predicted text category.

The summary embedding $\mathbf{v}_S(D)$ is expected to match the "meaning" of the reference summaries. It should have the ability to properly judge the sentence saliency, which is consistent with the reference summaries. Following (**?**), we use the cosine similarity between the summary embedding $\mathbf{v}_S(D)$ and a sentence embedding $\mathbf{v}(s)$ to predict the sentence saliency r_s.

$$r_s = \frac{\mathbf{v}(s) \bullet \mathbf{v}_S^T(D)}{||\mathbf{v}(s)|| \bullet ||\mathbf{v}_S(D)||} \tag{8}$$

That is why both document and summary embeddings are of the same dimensionality.

Training

We use the pre-trained word embeddings and do not update them to avoid over-fitting. Thus, there are three types of weight matrices in our models, i.e., $\mathbf{W}_\alpha$, $\mathbf{W}_\beta$ and the transformation sub-matrices $(\mathbf{W}_\gamma^1, \cdots, \mathbf{W}_\lambda^{|C|})$. Since the text classification dataset is much larger than the summarization dataset, $\mathbf{W}_\alpha$ and $\mathbf{W}_\beta$ are learned from the classification data only. Yet, the transformation matrices have to be trained with the summarization data.

For text classification, we adopt the cross entropy as the cost function, i.e.,

$$\varepsilon_C(D) = \sum\nolimits_{i=1}^{|C|} 1\{c_D == i\} \ln \mathbf{v}_C^i(D), \tag{9}$$

where $1\{c_D == i\}$ equals 1 iff the actual category is i. Under this cost function, the gradient of softmax is similar to a linear function, which fastens the training process.

For summarization, we apply the pairwise ranking strategy (**?**) to tune the weights. Specifically, each time we randomly select a sentence with a high actual saliency score and the other one with a low actual saliency score. They are denoted as s^+ and s^-, respectively. By Eq. 8, we obtain their predicted saliency scores. With the pairwise ranking criterion, TCSum should give s^+ a higher score in comparison with s^-. Therefore the cost function is defined as follows:

$$\varepsilon_S(D) = \max(0, \Omega - r_{s^+} + r_{s^-}), \tag{10}$$

where Ω is a margin threshold.

With the above two cost functions, we apply the diagonal variant of AdaGrad with mini-batches (**?**) to update model parameters. AdaGrad adapts the learning rate for different parameters at different steps. Thus it is less sensitive to initial parameters than the stochastic gradient descent.

Experiments

Datasets

Summarization The most commonly used evaluation corpora for summarization are the ones published by the Document Understanding Conferences (DUC) and Text Analytics Conferences (TAC[2]). In this work, we focus on the generic multi-document summarization task, which was carried out in DUC 2001, 2002 and 2004. The documents are all from the news domain and a collection of documents related to the same topic are grouped together into a cluster. Each cluster is accompanied by 2 to 4 reference summaries written by human experts. Our summarization model compiles the documents in a cluster into *a single document*. Table 1 shows the size of the three datasets and the summary length limitation for each task. The DUC datasets come from a wide range of categories, and we manually categorize the DUC documents into 11 categories, i.e., Biography, Culture, Business, Health, Politics, Law, Society, Natural Disaster, Science, Sports and International. The category distribution of DUC 2002 is illustrated in Fig. 2. Among these categories, Natural Disaster, Politics and Biography account for 60% of the documents.

Dataset	Cluster #	Doc. #	Ref. #	Limitation
DUC 01	30	309	60	100 words
DUC 02	59	567	116	100 words
DUC 04	50	500	200	665 bytes

Table 1: Statistics of the summarization datasets.

Text Classification In order to benefit from text classification, we need to have a classification dataset large enough to cover all the 11 categories discovered in the DUC datasets. We build such a dataset from the New York Times (NYT) Annotated Corpus [3]. The NYT corpus contains over 1.8 million articles published and annotated by the New York Times. Notably, the New York Times is also an important data provider for DUC. The NYT documents have rich metadata. We utilize three types of metadata (Types Of Material, Taxonomic Classifiers and Online Descriptors) to pick out the documents within those 11 categories. We notice that the

[2] `http://www.nist.gov/tac/` from 2007 $\sim$ now

[3] https://catalog.ldc.upenn.edu/LDC2008T19

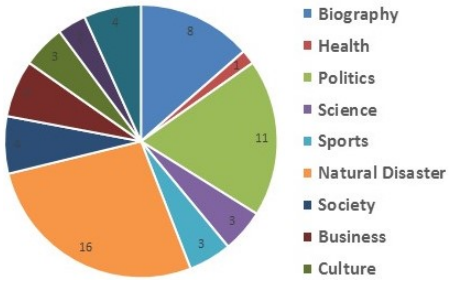

Figure 2: Category distribution on DUC 2002.

numbers of documents in different categories are extremely imbalanced. For example, the category of Business contains more than 140,000 documents, while there are only 3,200 documents in the category of Natural Disaster. Therefore, we conduct a sampling process to ensure that each category contains 3000-5000 documents. This classification dataset is about 30 times larger than the summarization dataset.

The cross validation shows that the learned classification model of TCSum achieves over 85% accuracy on this dataset. Since classification is not the focus of this paper, here we ignore the detailed performance evaluation of our classification model.

Evaluation Metric for Summarization

For evaluation, we use ROUGE[4] (**?**), which has been regarded as a standard automatic evaluation metric since 2004. ROUGE measures summary quality by counting overlapping units such as N-grams, word sequences and word pairs between the candidate summary and the reference summary. Following the common practice, we take ROUGE-1 and ROUGE-2 recall scores as the main metrics for comparison. ROUGE-1 and ROUGE-2 measure the uni-gram and bi-gram similarities, respectively. During training, the actual saliency of a sentence (Eq. 10) is also evaluated by ROUGE-2.

Model Settings

For CNN, we introduce a word embedding set trained on a large English news corpus (10^{10} tokens) using word2vec (**?**). The dimension of word embeddings is set to 50, as in many previous papers (e.g., (**?**)). We also set the dimension of sentence and document embeddings equivalent the dimension of word embeddings, and the window size h to 2, to be consistent with ROUGE-2 evaluation. We empirically set the margin threshold of pairwise ranking $\Omega = 0.1$. The initial learning rate is 0.1 and batch size is 128.

A summary is obliged to offer both informative and non-redundant content. While TCSum focuses on sentence ranking, it employs a simple greedy algorithm, similar to our previous work (**?**), to select summary sentences.

[4]ROUGE-1.5.5 with options: -n 2 -m -u -c 95 -x -r 1000 -f A -p 0.5 -t 0. The parameter of length constraint is "-l 100" for DUC 2001/2002, and "-b 665" for DUC 2004.

Baseline Methods

We compare TCSum with the best peer systems participating DUC evaluations, which are named as "**Peer**" plus their IDs. In addition, we include **R2N2** (**?**)[5], a state-of-the-art supervised summarization model based on neural networks. It applies the recursive neural network to learn the combination of hand-crafted features. Notably, R2N2 still heavily depends on hand-crafted features. By contrast, TCSum is fully data-driven, i.e., features are all learned automatically.

We implement a widely-used learning-based summarization method Support Vector Regression (**SVR**) (**?**). It extracts a number of manually-compiled features from a sentence, such as TF (the frequency of a word in the cluster), CF(the number of documents containing this word in the cluster) and NUMBER (whether the sentence contains a number), etc. We also design three neural network based baselines, named as **NoTC**, **SingleT** and **EmSim**. The first two are used to verify the value of text classification NoTC does not use any classification data and just applies the summarization model of TCSum. It is designed to check whether the summarization model can work alone. SingleT ignores the predicted text category and uses a single transformation matrix. It explores the effect of summary styles. The last one, EmSim, aims to test whether or not we need to learn the summary embedding. It just uses the cosine similarity between a sentence embedding $\mathbf{v}(s)$ and the document embedding $\mathbf{v}(D)$ to rank sentences. EmSim is an unsupervised summarization model and similar to (**?**). All these baselines employ the same sentence selection process as our model.

Summarization Performance

We conduct three-fold validation. The model is trained on two years' data and tested on the remaining year's. The ROUGE scores of the models being compared are presented in Table 2. We draw lines in this table to distinguish the models with and without hand-crafted features.

As can be seen, among the models completely dependent on automatically learned features, TCSum achieves highest performance on all the three datasets. The poor performance of EmSim denotes that we could not directly use the document embeddings learned from text classification to measure the sentence saliency for summarization. Note that even NoTC achieves competitive performance with SVR. Thus summarization models without hand-crafted features are doable. Meanwhile, SingleT greatly outperforms NoTC. It verifies that text classification can indeed help a summarization model to learn better document representations. Although TCSum does not always greatly surpass SingleT in terms of ROUGEs, we will show in the next section that it usually captures different summary styles.

Compared with other models, TCSum largely outperforms SVR and peer systems most of the time, and it is always superior to the state-of-the-art method R2N2. Considering TCSum is not supplemented with any hand-crafted features, its performance is very promising. After taking a

[5]Although R2N2 can use integer linear programming to select better sentences, here we just consider the result of greedy selection for a fair comparison.

Year	Model	ROUGE-1	ROUGE-2
2001	Peer T	33.03	**7.86**
	SVR	29.78	6.01
	R2N2	35.88	7.64
	NoTC	33.45	6.07
	EmSim	24.66	2.67
	SingleT	35.22	7.42
	TCSum	**36.45**	7.66
2002	Peer 26	35.15	7.64
	SVR	31.56	6.78
	R2N2	36.84	8.52
	NoTC	34.02	7.39
	EmSim	29.46	5.28
	SingleT	36.54	8.44
	TCSum	**36.90**	**8.61**
2004	Peer 65	37.88	9.18
	SVR	36.18	9.34
	R2N2	38.16	9.52
	NoTC	35.66	8.66
	EmSim	30.80	5.07
	SingleT	37.94	9.46
	TCSum	**38.27**	**9.66**

Table 2: ROUGE scores (%) of different methods.

closer look at the feature weights learned by SVR, we find the most important feature to measure sentence saliency is CF. Since we treat the documents in a topic cluster as a single document, this feature is lost in our current summarization model. It may be an important aspect that impedes the more excellent performance of TCSum.

Discussion on Summary Style Learning

We examine the ability of TCSum to learn summary styles in two ways. At first, we speculate that similar transformation matrices tend to generate summaries with similar styles. Therefore, we calculate the similarity among the transformation matrices $(\mathbf{W}_\gamma^1, \cdots, \mathbf{W}_\lambda^{|C|})$. Here we flatten each matrix into a vector and use the cosine similarity to measure the similarity. The scores of different transformation matrices are presented in Fig. 3. For ease of reference, we only show the results of three common categories on DUCs, i.e., Biography, Politics and Natural Disaster. As can be seen, the similarity relations of these three categories vary greatly, which matches the intuition that the large difference of the summary styles exists among these categories. For Biography, we find its transformation matrix is similar to 4 categories'. They are Business, Culture, Politics and International Relation. One possible reason is that summaries in Biography necessarily tell the career-related information of a person. Since DUC prefers choosing biographies about artists, businessmen and politicians, it is reasonable the summary style for Biography to be associated with these categories. By contrast, Natural Disaster does not present obvious similarity to any other category. We observe that summaries in Natural Disaster often contain a series of times, sites and numbers, while other categories seldom need so many details. For Politics, we find it is similar to International Relationship and Law. The former is understandable since we may use a number of terms of politics when describing international relationships. The latter may be caused by the news content. Many documents in this category are concerned with political scandals which often lead to lawsuits. Interestingly, there is an obvious negative similarity between Politics and Culture. The wordings in Politics are often thought to be serious while the documents in Culture are usually related to entertainment.

We also inspect the style change of the summaries generated according to different categories. To this end, we manually assign a category to a document cluster and then calculate the sentence saliency based on our summarization model. The salient sentences with respect to different categories are shown in Table 3. Due to the limit of space, we only display the top ranked summary sentences with the styles of three common text categories.

- "D097" is about a hurricane (Natural Disaster).
- "D066" introduces the founder of Wall-Mart (Biography).
- "D076" describes the resignation of a prime minister (Politics).

As can be seen, the salient sentences calculated by the correct categories can properly represent the main idea of the document cluster. Although "D097" and "D066" are not related to Politics, sentences selected by the corresponding transformation matrix still contain many terms of politics. It is also shown that the three Biography sentences contain either the words describing the careers (killer, mayor, founder) or the evaluative words (better, boldly). The career is a part of personal profile, and the description of main contributions of a person usually involves the evaluative words. Therefore, the corresponding transformation matrix seems to well catch the two types of needs for Biography summaries. We read the documents in "D066" and "D076" carefully, and find there is no sentence exactly matching Natural Disaster. Thus it is not surprising that the sentences selected by Natural Disaster in these two clusters are somewhat strange. However, we can see both sentences contain the date and site information. This is absolutely consistent with the style that a summary of Natural Disaster is expected to have. Moreover, both the money value and the word "bombing" can be used to describe the loss of a disaster. It appears that, the transformation matrix for Natural Disaster still works well even on a topic other than Natural Disaster, with "due diligence" to complete its own task.

Related Work

Work on extractive summarization spans a large range of approaches. Starting from unsupervised methods, one of the widely known approaches is Maximum Marginal Relevance (MMR) (**?**). It used a greedy approach to select sentences and considered the trade-off between saliency and redundancy. Good results could be achieved by reformulating it as an Integer Linear Programming (ILP) problem which was able to find the global optimal solution (**?**; **?**). Graph-based models such as Manifold (**?**) played an important role in extractive summarization because of its ability to reflect various sentence relationships. In contrast to these unsupervised methods, there are also many successful learning-based

Cluster	Category	Sentence
D097	Natural Disaster	The **storm**, packing **winds of up to 135 mph**, raged into **Charleston Thursday night**.
	Biography	"This is a dangerous, **killer** hurricane, the likes of which few people who have lived all their lives in Charleston have experienced," warned **Mayor** Joseph P. Riley Jr.
	Politics	**Gov.** Joe Frank Harris **declared** a **state of emergency** in six counties.
D066	Biography	Sam Walton, **founder** of the Wal-Mart chain of discount supermarkets who died of cancer in April, negotiated these pitfalls much **better** than most.
	Natural Disaster	By **1991** the chain's sales had risen to nearly **Dollars 44bn**, making it the world's largest retailer in terms of revenues, and the Walton family probably **America**'s richest.
	Politics	Bud is a senior **vice president** and board **member** of Wal-Mart.
D076	Politics	Flamboyant former **Defense Minister** Hazeltine's challenge to **Prime Minister** Margaret Thatcher for **leadership** of the **Conservative Party** has caused a **political** sensation in Britain.
	Biography	In the Persian Gulf crisis, she **boldly** joined with George Bush in sending troops to the Middle East.
	Natural Disaster	Among Western allies, she was alone at Ronald Reagan's side in **1986** in supporting the U.S. **bombing** of **Libya**.

Table 3: Salient sentences selected by different categories. Sentences in the correct categories are displayed first.

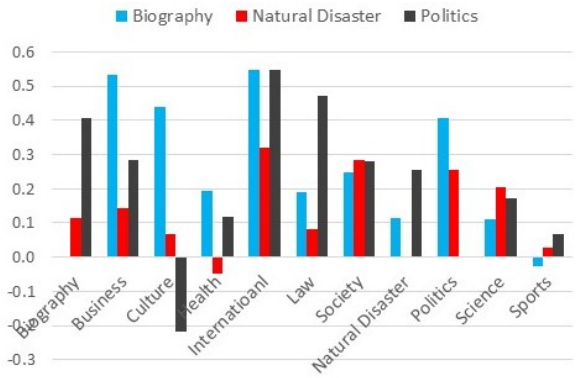

Figure 3: Similarity among the transformation matrices (we set the self similarity scores to 0).

summarization approaches. Different classifiers have been explored, including Conditional Random Field (**?**), Support Vector Regression (**?**) and Logistic Regression (**?**), etc. Recently, the application of deep neural network techniques has attracted more and more interest in the summarization research. (**?**) used unsupervised auto-encoders to represent both manual and system summaries for summary evaluation. Their method , however, did not surpass ROUGE. (**?**; **?**) tried to use neural networks to complement sentence ranking features. Although the models achieved the state-of-the-art performance, they still relied on hand-crafted features. A few researches explored to directly measure similarity based on distributed representations. (**?**) trained a language model based on convolutional neural networks to project sentences onto distributed representations. (**?**) treated single document summarization as a sequence labeling task and modeled it by recurrent neural networks. Others like (**?**) simply used the sum of trained word embeddings to represent sentences or documents. In addition to extractive summarization, deep learning technologies have also been applied to compressive and abstractive summarization (**?**; **?**).

Conclusion and Future Work

In this paper, we propose a novel summarization system called TCSum, which leverages text classification to improve the performance of summarization. Extensive experiments on DUC generic summarization benchmark datasets show that TCSum achieves the state-of-the-art performance, even without using any hand-crafted features. We also observe that TCSum indeed catches the variations of summary styles among different text categories. We believe our model can be used to other summarization tasks including query-focused summarization and guided summarization. In addition, we plan to let the model distinguish documents in a topic cluster, which is better adapted to the multi-document summarization.

Acknowledgments

The work described in this paper was supported by Research Grants Council of Hong Kong (PolyU 152094/14E), National Natural Science Foundation of China (61272291, 61672445) and The Hong Kong Polytechnic University (G-YBP6, 4-BCB5, B-Q46C). The correspondence authors of this paper are Wenjie Li and Sujian Li.

Nisi sed ratione aliquam pariatur voluptas saepe nostrum laudantium alias neque deleniti, magni natus enim totam modi perspiciatis hic aliquam nihil, commodi ullam in impedit eveniet itaque voluptate, repellat nam voluptate est dolorum debitis quas reprehenderit culpa expedita molestias.Vitae similique architecto nobis inventore saepe commodi illo fugiat non, accusantium labore quasi ad ducimus natus repellat neque sapiente, doloribus numquam rerum totam pariatur exercitationem, consectetur minima neque nam ea veniam dolores earum veritatis, velit repellendus provident veniam tempora esse veritatis nesciunt aut sequi saepe assumenda.Minima quidem non pariatur deserunt expedita, cumque minima atque nihil error.Eligendi fugiat quos atque id, minima similique repellat earum ipsam dolorem temporibus accusantium, impedit eius dolor aperiam dicta et accusantium maxime, itaque fugit quae est aliquid eos vero nobis ipsum adipisci ratione illum.Necessitatibus illo temporibus earum nobis iste minima