Sentence Relations for Extractive Summarization with Deep Neural Networks

Article *in* ACM Transactions on Information Systems · April 2018

DOI: 10.1145/3200864

7 authors, including:

Pengjie Ren
Shandong University
94 PUBLICATIONS **1,598** CITATIONS

SEE PROFILE

Jun Ma
Shandong University
150 PUBLICATIONS **2,610** CITATIONS

SEE PROFILE

Sentence Relations for Extractive Summarization with Deep Neural Networks

PENGJIE REN and ZHUMIN CHEN, Shandong University
ZHAOCHUN REN, Data Science Lab, JD.com
FURU WEI, Microsoft Research Asia
LIQIANG NIE and JUN MA, Shandong University
MAARTEN DE RIJKE, University of Amsterdam

Sentence regression is a type of extractive summarization that achieves state-of-the-art performance and is commonly used in practical systems. The most challenging task within the sentence regression framework is to identify discriminative features to represent each sentence. In this article, we study the use of sentence relations, e.g., Contextual Sentence Relations (CSR), Title Sentence Relations (TSR), and Query Sentence Relations (QSR), so as to improve the performance of sentence regression. CSR, TSR, and QSR refer to the relations between a main body sentence and its local context, its document title, and a given query, respectively.

We propose a deep neural network model, Sentence Relation-based Summarization (SRSum), that consists of five sub-models, PriorSum, CSRSum, TSRSum, QSRSum, and SFSum. PriorSum encodes the latent semantic meaning of a sentence using a bi-gram convolutional neural network. SFSum encodes the surface information of a sentence, e.g., sentence length, sentence position, and so on. CSRSum, TSRSum, and QSRSum are three sentence relation sub-models corresponding to CSR, TSR, and QSR, respectively. CSRSum evaluates the ability of each sentence to summarize its local contexts. Specifically, CSRSum applies a CSR-based word-level and sentence-level attention mechanism to simulate the context-aware reading of a human reader, where words and sentences that have anaphoric relations or local summarization abilities are easily remembered and paid attention to. TSRSum evaluates the semantic closeness of each sentence with respect to its title, which usually reflects the main ideas of a document. TSRSum applies a TSR-based attention mechanism to simulate people's

This paper is a substantially extended version of Ren et al. (2017). The additions are three-fold. First, we propose a new summarization model by incorporating two sub-models that consider title-sentence relations and query-sentence relations. Second, the new model can simulate people's reading ability with the main idea (title) and reading intention (query) in mind by introducing two attention mechanisms. Third, more than half of the experiments reported in the paper were not in Ren et al. (2017) and all involved tables and figures are either new additions to the article or report new results.

This article is supported by the Natural Science Foundation of China (No. 61672324, No. 61672322), the Natural Science Foundation of Shandong province (No. 2016ZRE27468), the Fundamental Research Funds of Shandong University, Ahold Delhaize, Amsterdam Data Science, the Bloomberg Research Grant program, Elsevier, the European Community's Seventh Framework Programme (No. FP7/2007-2013) under grant agreement nr 312827 (VOX-Pol), the Google Faculty Research Awards program, the Microsoft Research Ph.D. program, the Netherlands Institute for Sound and Vision, the Netherlands Organisation for Scientific Research (NWO) under Projects No. CI-14-25, No. 652.002.001, No. 612.001.551, No. 652.001.003, and Yandex. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

Authors' addresses: P. Ren, Shandong University, Jinan, China; email: jay.ren@outlook.com; Z. Chen, Shandong University, Jinan, China; email: chenzhumin@sdu.edu.cn; Z. Ren, Data Science Lab, JD.com, Beijing, China; email: renzhaochun@jd.com; F. Wei, Microsoft Research Asia, Beijing, China; email: fuwei@microsoft.com; L. Nie, Shandong University, Jinan, China; email: nieliqiang@gmail.com; J. Ma, Shandong University, Jinan, China; email: majun@sdu.edu.cn; M. de Rijke, University of Amsterdam, Amsterdam, The Netherlands; email: derijke@uva.nl.

https://doi.org/10.1145/3200864

reading ability with the main idea (title) in mind. QSRSum evaluates the relevance of each sentence with given queries for the query-focused summarization. QSRSum applies a QSR-based attention mechanism to simulate the attentive reading of a human reader with some queries in mind. The mechanism can recognize which parts of the given queries are more likely answered by a sentence under consideration. Finally as a whole, SRSum automatically learns useful latent features by jointly learning representations of query sentences, content sentences, and title sentences as well as their relations.

We conduct extensive experiments on six benchmark datasets, including generic multi-document summarization and query-focused multi-document summarization. On both tasks, SRSum achieves comparable or superior performance compared with state-of-the-art approaches in terms of multiple ROUGE metrics.

CCS Concepts: • **Information systems → Content analysis and feature selection**; • **Computing methodologies → Information extraction**;

Additional Key Words and Phrases: Extractive summarization, sentence relations, neural network, attentive pooling

ACM Reference format:
Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Liqiang Nie, Jun Ma, and Maarten de Rijke. 2018. Sentence Relations for Extractive Summarization with Deep Neural Networks. *ACM Trans. Inf. Syst.* 36, 4, Article 39 (April 2018), 32 pages.
https://doi.org/10.1145/3200864

1 INTRODUCTION

Extractive summarization aims to generate a short text summary for a long document or a set of documents by selecting salient sentences in the document(s) (Over and Yen 2004). In recent years, sentence regression has emerged as an extractive summarization framework that achieves state-of-the-art performance (Cao et al. 2015b; Wan et al. 2015); it has been widely used in practical systems (Hong and Nenkova 2014; Hu and Wan 2013; Ren et al. 2016a; Wan and Zhang 2014). There are two major components in sentence regression: *sentence scoring* and *sentence selection*. The former scores a sentence to measure its importance, and the latter chooses sentences to generate a summary by considering both sentence saliency and redundancy.

Sentence scoring has been extensively investigated in extractive summarization. Many approaches (Cao et al. 2015b; Ouyang et al. 2007) directly measure the salience of sentences whereas others (Gillick and Favre 2009; Li et al. 2013) first score words (or bi-grams) and then combine these scores to score sentences. Traditional scoring methods incorporate feature engineering as a necessary but labor-intensive task. In Table 1, we list the scores achieved by *t-SR* (Ren et al. 2016a), a traditional feature engineering-based sentence regression method for extractive summarization that achieves state-of-the-art performance. We list an upper bound on the performance of sentence regression, which is obtained by scoring sentences against human written summaries. There is a sizable gap in performance between *t-SR* and the upper bound. We hypothesize that the reason for this is that none of *t-SR*'s features tries to encode semantic information.

Recent neural network-based methods for *abstractive* summarization have addressed this matter (Chopra et al. 2016; Nallapati et al. 2016; Rush et al. 2015). Extracting semantic features via neural networks has received increased attention, also for extractive summarization (Cao et al. 2015a, 2015b; Cheng and Lapata 2016) . PriorSum (Cao et al. 2015b) is a recent example. It uses Convolutional Neural Networks (CNN) to encode each sentence into a vector representation and uses that vector as a prior that quantifies to which degree a sentence is appropriate for inclusion in summary, without consideration of its context. They also propose a ranking framework upon recursive neural networks (RNNs) (namely, R2N2) (Cao et al. 2015a). R2N2 uses RNNs to encode

Table 1. Multi-Document Summarization

Dataset	Approach	ROUGE-1	ROUGE-2
DUC 2001	*t-SR*	34.82	7.76
	Upper bound	40.82	14.76
DUC 2002	*t-SR*	37.33	8.98
	Upper bound	43.78	15.97
DUC 2004	*t-SR*	37.74	9.60
	Upper bound	41.75	13.73
DUC 2005	*t-SR*	38.40	7.60
	Upper bound	45.34	14.11
DUC 2006	*t-SR*	40.35	9.09
	Upper bound	49.17	17.42
DUC 2007	*t-SR*	42.42	11.20
	Upper bound	50.83	19.30

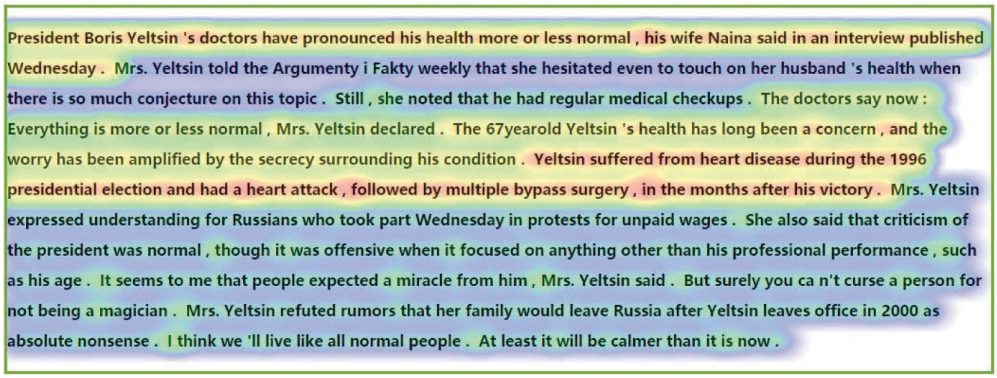
ROUGE (%) of sentence regression (with greedy-based sentence selection). Upper bounds are determined by scoring sentences against human written summaries.

each sentence into a vector representation by considering its syntactic structure reflected by its parse tree. Both PriorSum and R2N2 show great improvements over traditional feature engineering methods, which proves that latent semantic features learned by neural networks are effective. Importantly, most methods, including PriorSum and R2N2, extract latent features from stand-alone sentences without considering sentence relations.

While understanding the meaning of a sentence is important to generate a good summary, the meaning of a sentence is not independent of the meaning of other sentences and sometimes it is incomplete without considering its relations with other sentences. This statement is hardly controversial since each sentence usually only expresses one view or states one fact, which may be hard to grasp without knowing the background reflected in the related sentences. Therefore, we argue that sentence saliency depends both on its own meaning and on relations with other sentences. Christensen et al. (2013) demonstrate the importance of considering discourse relations among sentences in multi-document summarization. Yasunaga et al. (2017) propose a multi-document summarization system that exploits the representational power of deep neural networks and the sentence relation information encoded in graph representations of document clusters. However, they do not specify different sentence relations and model the general relations between each pair of sentences. In this article, we study three types of relations between sentences, *contextual sentence relations* (CSR), *title sentence relations* (TSR), and *query sentence relations* (QSR), to improve the performance of sentence regression. CSR refers to the relation between a main body sentence and its local contexts, i.e., its surrounding sentences. Important sentences are usually those that can summarize their local contexts. Figure 1(a) illustrates a general-to-specific paragraph structure, where the first sentence (with the highest color depth) is a general summary of the event that is explained in detail by the following sentences. Figure 1(b) illustrates a specific-to-general paragraph structure, where the last sentence (again, with the highest color depth) is a conclusion or reason of the event described by its preceding sentences. Figure 1(c) illustrates a specific-to-general-to-specific paragraph structure, where the most important sentence (in the center of the snippet, with the highest color depth) is a connecting link between the preceding and the following context. So it summarizes both its preceding and following sentences.

(a) General to specific.

(b) Specific to general.



(c) Specific to general to specific.

Fig. 1. Sentence contexts in different instances from the DUC 2004 dataset. The color depth represents the importance of the sentence in terms of ROUGE-2 based on human written summaries. (Best viewed in color.)

TSR refers to the relation between a title sentence and a main body sentence. Usually, the title sentence reflects the key idea of the whole document or at least contains the phrases or words that are key to the document topic. As a result, the main body sentences that have close relations with the title sentence are usually important and should be included in a summary.

QSR refers to the relation between a query sentence and a main body sentence. For query-focused summarization, whether a sentence should be included in the final summary depends not

only on its importance but also its relevance to the given query. The main body sentences that have close relations with the document topic might not necessarily answer the given query. In this case, they should be excluded from the final summary.

We propose a hybrid neural summarization model, namely *sentence relation-based summarization* (SRSum), to automatically learn sentence relation features from data. SRSum consists of five sub-models:

— PriorSum: a sentence meaning sub-model that is encoded by a bi-gram convolutional neural network (CNN);
— SFSum: a sentence surface sub-model that encodes the surface information, e.g., the sentence length and the sentence position;
— CSRSum: a sentence relation sub-models corresponding to CSR;
— TSRSum: a sentence relation sub-models corresponding to TSR; and
— QSRSum: a sentence relation sub-models corresponding to QSR.

CSRSum evaluates the ability of a sentence to summarize its local context. It applies a two-level attention mechanism (word level and sentence level) to attend differentially to more and less important content when constructing sentence/context representations, which simulates the context-aware reading of human behavior, where words and sentences that have anaphoric relations or local summarization abilities are easily remembered. Specifically, we first leverage sentence relations using a CNN with word-level attentive pooling to construct sentence representations. Then, we leverage contextual relations using a RNN with sentence-level attentive pooling to construct context representations. With its two-level attention mechanism, CSRSum can pay attention to more important content (words and sentences) in the surrounding context of a given sentence. Finally, CSRSum calculates the CSR relation scores as the sentence's capacity to summarize its contexts.

TSRSum evaluates the semantic closeness of each sentence with respect to its title, which reflects the main ideas of a document. TSRSum first uses a CNN to construct main body sentence representations. Then it uses a TSR-based attention mechanism when constructing title representations by assigning more weights to more relevant words, which simulates people's reading ability with the main idea (title) in mind. Compared with main body sentences, we usually adopt different syntax rules to write titles to make them concise. Thus we assume that they belong to two different spaces and use a bilinear matching mechanism to compute the TSR relation score between the title representation and main body sentence representation.

QSRSum evaluates the relevance of each sentence with given queries for the query-focused summarization. QSRSum first uses a CNN to construct main body sentence and query representations. Then it uses a QSR-based attention mechanism to assign more weights to more relevant queries with respect to the main body sentences, which simulates the attentive reading of a human reader with some queries in mind. Since the main body sentences are written by the article authors while the queries are given by the readers, we assume that they belong to two different spaces and use a bilinear matching mechanism to compute the QSR relation score between the query representation and main body sentence representation. Finally, SRSum automatically learns useful latent features by jointly learning representations of query sentences, content sentences, and title sentences as well as the their relations.

We conduct extensive experiments on the DUC 2001, 2002, 2004 multi-document summarization datasets and the DUC 2005, 2006, 2007 query-focused multi-document summarization datasets. Our experimental results demonstrate that SRSum achieves comparable or superior performance with state-of-the-art approaches in terms of multiple ROUGE metrics.

To sum up, the main contributions in this article are listed as follows:

- We propose a neural model, SRSum, which consists of five sub-models, PriorSum, CSRSum, QSRSum, TSRSum, and SFSum to take a sentence's meaning, three types of sentence relations, as well as surface information into consideration for extractive summarization.
- We fuse contextual relations with a two-level attention mechanism in CSRSum. With the mechanism, CSRSum can learn to pay attention to important content (words and sentences) in the surrounding sentences of a given sentence to simulate human context-aware reading.
- We apply the attention mechanism in QSRSum to construct the query representation, which simulates the human reading behavior with some queries in mind. With the mechanism, QSRSum can model which part of the query is answered by the current sentence.
- We apply the attention mechanism in TSRSum to construct the title representation. With the mechanism, TSRSum can better evaluate the TSR relation scores by focusing on more relevant words in the title to the current sentence.
- We carry out extensive experiments and analyses on six benchmark datasets. The results indicate that SRSum can significantly improve the performance of extractive summarization by modeling the three sentence relations.

2 RELATED WORK

We group related work on extractive summarization in three categories, which we discuss below.

2.1 Unsupervised Techniques

In early studies on extractive summarization, Luhn (1958) proposes the use of frequency thresholds to identify descriptive terms in a document to be summarized, a simple representation of the document's topic. The descriptive terms in his approach exclude the most frequent words in the document, which are likely to be determiners, prepositions, or domain-specific terms, as well as those occurring only a few times. Dunning (1993) proposes a statistical version of Luhn's idea, which applies the likelihood ratio test for the identification of composite terms and for the determination of domain-specific terms that are highly descriptive of the input. Later, Lin and Hovy (2000) refer to such terms as "topic signatures" in the summarization literature.

In the early 2000s, unsupervised sentence scoring methods become popular (Lin and Hovy 2002; Radev et al. 2000). Centroid-based and Maximum Marginal Relevance (MMR)-based approaches are prominent examples. Centroid-based methods use sentence centrality to indicate importance (Mihalcea 2004). Radev et al. (2000, 2004) model cluster centroids in their summarization system, MEAD. LexRank (or TextRank) computes sentence importance based on eigenvector centrality in a graph of sentence similarities (Erkan and Radev 2004; Mihalcea and Tarau 2004). Wan (2008, 2011), Wan and Xiao (2009), and Wan and Yang (2008) propose several centroid-based approaches for summarization.

MMR-based methods consider a linear trade-off between relevance and redundancy (Carbonell and Goldstein 1998). Goldstein et al. (2000) extend MMR to support extractive summarization by incorporating additional information about the document set and relations between the documents. McDonald (2007) achieves good results by reformulating MMR as a knapsack packing problem and solving it using ILP. Later, Lin and Bilmes (2010, 2011) propose a variant of the MMR framework that maximizes an objective function that considers the linear trade-off between coverage and redundancy terms.

Unlike our approach to summarization, these unsupervised methods do not need human-written summaries to train a model; their performances are usually limited.

2.2 Traditional Machine-Learning-Based Techniques

Machine-learning techniques have been used to obtain better estimations of sentence importance. Kupiec et al. (1995) train a Naive Bayes classifier to decide whether to include a sentence in the summary. Barzilay et al. (2002) propose a methodology for studying the properties of ordering information in the news genre and describe experiments done on a corpus of multiple acceptable orderings. They implement a strategy for ordering information that combines constraints from chronological order of events and topical relatedness. Lapata (2003) propose an approach to information ordering that is particularly suited for text-to-text generation. They describe a model that learns constraints on sentence order from a corpus of domain-specific texts and an algorithm that yields the most likely order among several alternatives. Li et al. (2007) evaluate sentence importance with support vector regression, after which a rule-based method is applied to remove redundant phrases. Gillick and Favre (2009) evaluate bi-gram importance and use the scores to evaluate sentence importance and redundancy with a linear combination. Bollegala et al. (2010) present a bottom-up approach to arrange sentences extracted for multi-document summarization. To capture the association and order of two textual segments (e.g., sentences), they define four criteria: chronology, topical-closeness, precedence, and succession. These criteria are integrated into a criterion by a supervised learning approach. Lin and Bilmes (2010) propose a structural SVM learning approach to learn the weights of feature combinations using the MMR-like submodularity function proposed by Lin and Bilmes (2010, 2011). Lin and Bilmes (2012) introduce a method to learn a mixture of submodular "shells" in a large-margin setting. A submodular shell is an abstract submodular function that can be instantiated with a ground set and a set of parameters to produce a submodular function. A mixture of such shells can then also be instantiated to produce a more complex submodular function. They provide a risk bound guarantee when learning in a large-margin structured prediction setting using a projected subgradient method when only approximate submodular optimization is possible (such as with submodular function maximization). Their method is also used for image collection summarization (Tschiatschek et al. 2014).

Yan and Wan (2015) propose the Deep Dependency Sub-Structure (DDSS) and topic-sensitive Multi-Task Learning (MTL) model. Given a document set, they parse all sentences into deep dependency structures with a Head-Driven Phrase Structure Grammar parser and mine the frequent DDSSs after semantic normalization. They then employ MTL to learn the importance of these frequent DDSSs. Hu and Wan (2015) propose a system (namely PPSGen) to automatically generate presentation slides by selecting and aligning key phrases and sentences.

The methods listed above all rely on human-engineered features. Unlike our work with SRSum, most of the features used with traditional machine-learning-based techniques are surface features that do not take contextual relations into account.

2.3 DeepLearning-Based Techniques

Deep-learning techniques have attracted considerable attention in the summarization literature, e.g., abstractive summarization (Bing et al. 2015; Chopra et al. 2016; Nallapati et al. 2016), sentence summarization (Filippova et al. 2015; Hu et al. 2015; Rush et al. 2015), and extractive summarization (Cao et al. 2015a, 2015b; Cheng and Lapata 2016). We focus on the use of deep-learning techniques for extractive summarization.

Kågebäck et al. (2014) and Kobayashi et al. (2015) use the sum of trained word embeddings to represent sentences or documents. They formalize the summarization task as the problem of maximizing a submodular function based on the similarities of the embeddings. Yin and Pei (2015) propose CNNLM, a model based on CNNs, to project sentences into dense distributed representations, then model sentence redundancy by cosine similarity. Cao et al. (2015b) propose the concept of

summary prior to define how much a sentence is appropriate to be selected into summary without consideration of its context. They develop a summarization system called PriorSum, which applies enhanced CNNs to capture the summary prior features derived from length-variable phrases. In other work, the authors develop a ranking framework based on RNNs (R2N2) to rank sentences for multi-document summarization. R2N2 formulates the ranking task as a hierarchical regression process that simultaneously measures the salience of a sentence and its constituents (e.g., phrases) in the parse tree (Cao et al. 2015a). Cheng and Lapata (2016) treat single document summarization as a sequence labeling task and model it with recurrent neural networks. Their model is composed of a hierarchical document encoder and an attention-based extractor; the encoder derives the meaning representation of a document based on its sentences and their constituent words while the extractor adopts a variant of neural attention to extract sentences or words. Cao et al. (2016) propose a system called AttSum for query-focused multi-document summarization that applies an attention mechanism to simulate the attentive reading of human behavior when a query is given.

A growing number of publications on extractive summarization focus on deep-learning techniques. To the best of our knowledge, we are the first to consider CSRs under sentence regression framework for extractive summarization. Our own previous publication (Ren et al. 2017) is an exception: in that paper we improve the performance of generic multi-document summarization by modeling the CSRs in the surrounding contexts of a given sentence. We build on that paper by expanding it with two other sub-models, TSRSum and QSRSum, corresponding to the TSR relations and QSR relations, respectively. With these two sub-models, the new model can simulate people's reading ability with the main idea (title) and reading intention (query) in mind.

3 METHOD

3.1 Overview

We follow the sentence regression-based approach to summarization. Thus, there are two phases in our method to generate a summary: *sentence scoring* and *sentence selection*. In the sentence scoring phase, we learn a scoring function $f(S_t \mid \theta)$ for each sentence S_t to fit the ground truth ROUGE-2 score:[1]

$$f(S_t \mid \theta) \sim \text{ROUGE-2}(S_t \mid S_{ref}), \tag{1}$$

where θ are the parameters; ROUGE-2$(S_t \mid S_{ref})$ is the ground truth score of S_t in terms of ROUGE-2 based on human written summaries S_{ref} (Lin 2004). In Section 3.2, we detail how we model $f(S_t \mid \theta)$.

During the sentence selection phase, we select a subset of sentences as the summary Ψ subject to a given length constraint l, i.e.,

$$\Psi^* = \arg\max_{\Psi \subseteq D} \sum_{S_t \in \Psi} f(S_t \mid \theta) \text{ such that } \sum_{S_t \in \Psi} |S_t| \leq l \text{ and } r(\Psi) \text{ holds}, \tag{2}$$

where D is the set of sentences from one or more documents that belong to the same topic; $|S_t|$ is the length of S_t in words or bytes; $r(\Psi)$ is a constraint function to avoid redundancy in the final summary. In this article, a sentence is considered non-redundant if it contains more new bigrams compared to the current summary content. Details of the sentence selection algorithm are explained in Section 3.8.

[1] http://www.berouge.com/Pages/default.aspx.

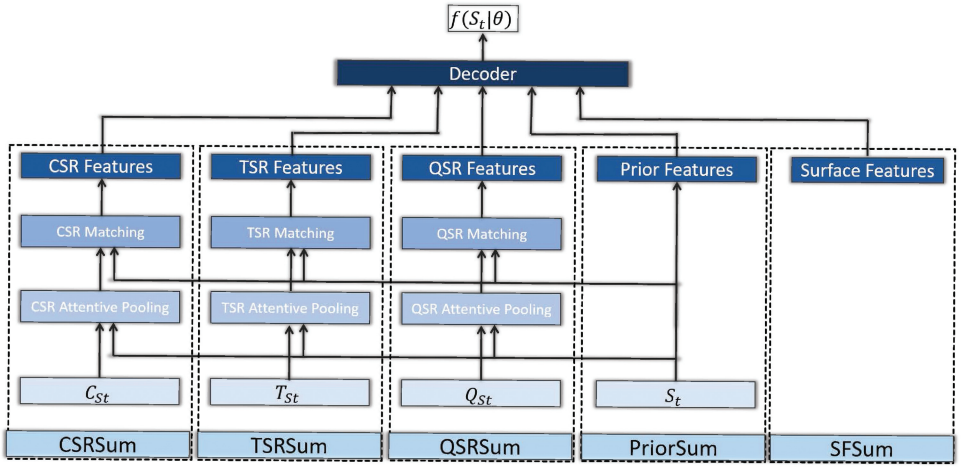


Fig. 2. Overview of SRSum.

3.2 Sentence Scoring

For sentence scoring, we propose a neural model, SRSum. The architecture of SRSum is shown in Figure 2. SRSum contains five submodels, CSRSum, TSRSum, QSRSum, PriorSum, and SFSum. CSRSum encodes the CSRs. TSRSum encodes the TSRs. QSRSum encodes the QSRs. PriorSum encodes the prior features that represent the meaning of the sentence. SFSum enhances SRSum by appending a small number of effective surface features. We concatenate these features and then apply a Multilayer Perceptron (MLP) (Gardner and Dorling 1998; Ruck et al. 1990) as the decoder to transform the features (the outputs of the five submodels) into a single value as the final salience score to S_t, as shown in Equation (3):

$$f(S_t \mid \theta) = \text{MLP}\left(\begin{bmatrix} f^{CSR}(S_t) \\ f^{TSR}(S_t) \\ f^{QSR}(S_t) \\ f^{Prior}(S_t) \\ f^{SF}(S_t) \end{bmatrix}\right). \tag{3}$$

Here, $f^{CSR}(S_t)$, $f^{TSR}(S_t)$, $f^{QSR}(S_t)$, $f^{Prior}(S_t)$, and $f^{SF}(S_t)$ are the outputs of CSRSum, TSRSum, QSRSum, PriorSum, and SFSum, respectively. θ are the parameters of the neural network. We use a three-hidden-layer MLP with tanh activation function.

As with existing studies (Cao et al. 2015a, 2015b; Ren et al. 2016a), we use the standard mean square error as the loss function to train SRSum:

$$\begin{aligned} L(\theta) &= \frac{1}{|C| \cdot |D|} \sum_{D \in C} \sum_{S_t \in D} Err(S_t), \\ Err(S_t) &= \Big(f(S_t \mid \theta) - \text{ROUGE-2}(S_t \mid S_{ref})\Big)^2, \end{aligned} \tag{4}$$

where C is the set of all documents.

In the next five subsections, we describe the five components (PriorSum, CSRSum, TSRSum, QSRSum, and SFSum) in detail.

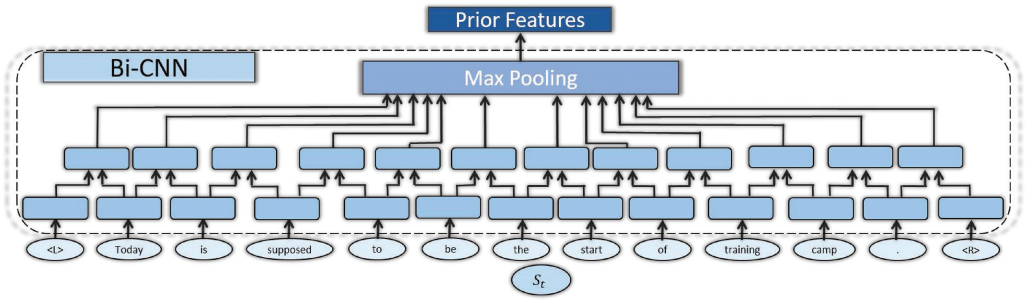


Fig. 3. Architecture of PriorSum.

3.3 PriorSum

Since we conduct regression with respect to ROUGE-2, which is computed as the bi-gram overlap between the system-generated summary and the human-written summary, we use Bi-CNN (Cao et al. 2015b) to model each sentence, as shown in Figure 3. We first concatenate adjacent words into bi-grams:

$$\mathrm{bi}^{Prior}(i, i+1) = \begin{bmatrix} \mathrm{w}_i \\ \mathrm{w}_{i+1} \end{bmatrix}, \tag{5}$$

where w_i is the word embedding for the i-th word of a sentence. The index of w_i starts from 0 and ends with sentence length +1 because we add two padding words "<L>" (Left) and "<R>" (Right) to each sentence. After that, we perform convolutions on the bi-grams with a filter matrix:

$$\mathrm{h}^{Prior}(i, i+1) = f(\mathrm{W}_{Prior}^{T} \cdot \mathrm{bi}^{Prior}(i, i+1) + b^{Prior}), \tag{6}$$

where $\mathrm{W}_{Prior} \in \mathbb{R}^{2|\mathrm{w}_i| \times |\mathrm{w}_i|}$ is the filter matrix; b^{Prior} is the bias; and $f(\cdot)$ is the activation function. We use the $\tanh(\cdot)$ function in our experiments.

Then we perform element-wise max pooling over the bi-gram representations $H_{bi}^{Prior}(S_t) = \{\mathrm{h}^{Prior}(i, i+1) \mid 0 \le i \le |S_t|\}$ to obtain the representation $\mathrm{h}(S_t)$ of sentence S_t:

$$\mathrm{h}(S_t) = \max_{\mathrm{h}^{Prior}(i,i+1) \in H_{bi}^{Prior}(S_t)} \mathrm{h}^{Prior}(i, i+1). \tag{7}$$

The function max chooses the maximum value of each dimension of the vectors in $H_{bi}^{Prior}(S_t)$.

The output of PriorSum, $\mathrm{h}(S_t)$, is regarded as the prior features ($f^{Prior}(S_t) = \mathrm{h}(S_t)$) of S_t that represent its "meaning" or prior capability as a summary sentence (Cao et al. 2015b).

3.4 CSRSum

CSRSum is based on the intuition that important sentences are usually those that can summarize their local contexts. Given a sentence S_t, we assume that its preceding context sentence sequence is $C_{pc}^{CSR} = \{S_{t-m}, \ldots, S_{t-c}, \ldots, S_{t-1} \mid 1 \le c \le m\}$ and that its following context sentence sequence is $C_{fc}^{CSR} = \{S_{t+1}, \ldots, S_{t+c}, \ldots, S_{t+n} \mid 1 \le c \le n\}$. Settings of m and n are discussed in Section 4 below. We write $f^{CSR}(S_t) = [f_{pc}^{CSR}(\mathrm{h}(S_t), \mathrm{h}_{pc}^{CSR}(S_t)), f_{fc}^{CSR}(\mathrm{h}(S_t), \mathrm{h}_{fc}^{CSR}(S_t))]$ for the output of CSRSum; it represents the CSR features, as shown in Figure 4. The purpose of $f_{pc}^{CSR}(\mathrm{h}(S_t), \mathrm{h}_{pc}^{CSR}(S_t))$ is to estimate the ability of S_t to summarize its preceding context:

$$f_{pc}^{CSR}(\mathrm{h}(S_t), \mathrm{h}_{pc}^{CSR}(S_t)) = \cos(\mathrm{h}(S_t), \mathrm{h}_{pc}^{CSR}(S_t)). \tag{8}$$

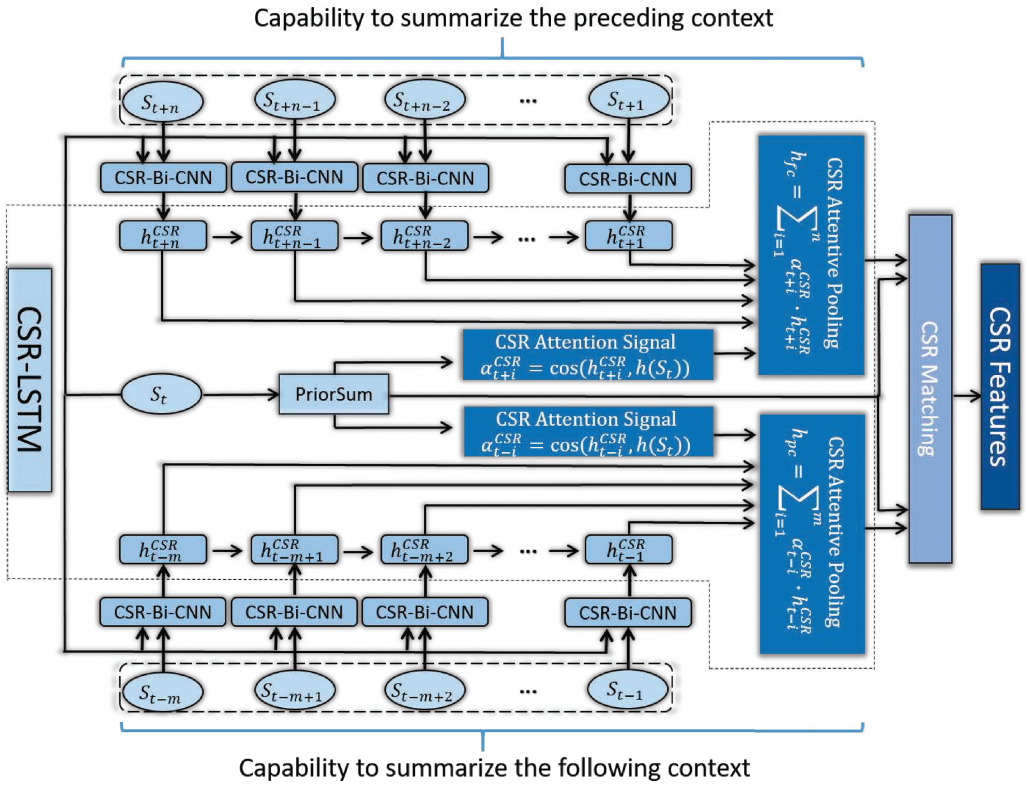


Fig. 4. Architecture of CSRSum.

Similarly, $f_{fc}^{CSR}(\mathrm{h}(S_t), \mathrm{h}_{fc}^{CSR}(S_t))$ estimates the ability of S_t to summarize its following context:

$$f_{fc}^{CSR}(\mathrm{h}(S_t), \mathrm{h}_{fc}^{CSR}(S_t)) = \cos(\mathrm{h}(S_t), \mathrm{h}_{fc}^{CSR}(S_t)), \tag{9}$$

where $\mathrm{h}(S_t)$ is the sentence model of S_t (the outputs of PriorSum); cos indicates the cosine similarity. The reason that we employ cos here and also in some following equations is that cos does not introduce additional parameters and is frequently used to model semantic similarities. Besides, we found that changing cos to parametric similarities (i.e., linear layer with activation function) does not improve the results. $\mathrm{h}_{pc}^{CSR}(S_t)$ and $\mathrm{h}_{fc}^{CSR}(S_t)$ are the context models of C_{pc}^{CSR} and C_{fc}^{CSR} as described next.

CSR Attentive Context Modeling: $h_{pc}^{CSR}(S_t)$ and $h_{fc}^{CSR}(S_t)$. We use Recurrent Neural Networks with a Long Short-Term Memory (LSTM) unit to model the context, which have been successfully applied to many sequence modeling tasks (Filippova et al. 2015; Nallapati et al. 2016; Rush et al. 2015). There are many variations of LSTMs that differ in their connectivity structure and activation functions. We employ the LSTM architecture presented in Graves et al. (2013).

$$\mathrm{LSTM} : \mathrm{h}_{t-1}^{CSR}, \mathrm{v}^{CSR}(S_t), c_{t-1} \rightarrow c_t, \mathrm{h}_t^{CSR} \tag{10}$$

h_t^{CSR} is the hidden state with respect to the t-th time step input $\mathrm{v}^{CSR}(S_t)$ (defined next in Equation (13)); c_t is the memory cell vector of the t-th time step; and sigm and tanh are applied element-wise. LSTMs have a complicated dynamics that allows them to easily "memorize" information for

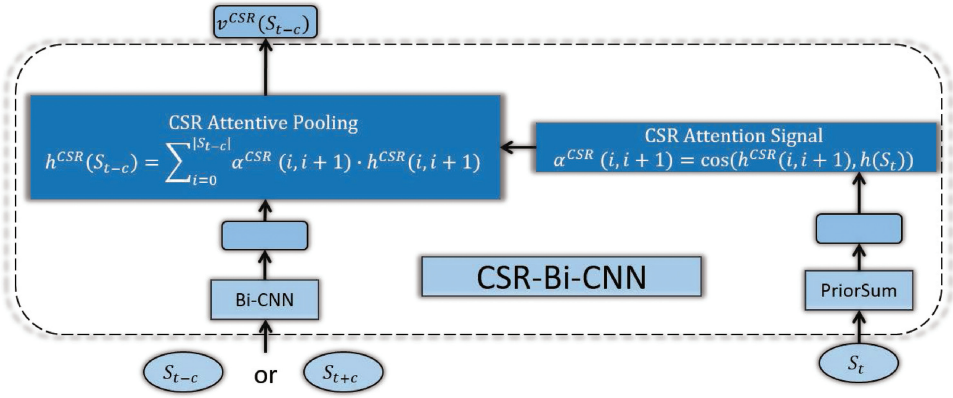


Fig. 5. Architecture of CSR-Bi-CNN.

an extended number of time steps. The "long term" memory is stored in a vector of memory cells c_t. An LSTM can decide to overwrite the memory cell, retrieve it, or keep it for the next time step.

Given a sentence S_t, we recurrently apply the LSTM unit to its preceding context sentence sequence C_{pc}^{CSR} and following context sentence sequence C_{fc}^{CSR}. For each timestamp t, S_t is fed into the LSTM unit and a corresponding vector representation h_t^{CSR} is generated. Then, we have $H_{pc}^{CSR} = \{\mathrm{h}_{t-m}^{CSR}, \ldots, \mathrm{h}_{t-1}^{CSR}\}$ for C_{pc}^{CSR} and $H_{fc}^{CSR} = \{\mathrm{h}_{t+1}^{CSR}, \ldots, \mathrm{h}_{t+n}^{CSR}\}$ for C_{fc}^{CSR}. Finally, we encode H_{pc}^{CSR} and H_{fc}^{CSR} into vector representations with an LSTM (CSR-LSTM) that can attend differentially to more and less important sentences, as shown in Figure 4. The formula for S_t's preceding context is

$$\mathrm{h}_{pc}^{CSR}(S_t) = \sum_{i=1}^{m} \alpha_{t-i}^{CSR} \cdot \mathrm{h}_{t-i}^{CSR}, \tag{11}$$

where α_{t-i}^{CSR} is the attention weight for the hidden context state h_{t-i}^{CSR}. The formula for S_t's following context is similar.

Unlike most existing attention mechanisms, where the last hidden state of an LSTM is used to learn the attention weights (Chopra et al. 2016; Rush et al. 2015), here CSR-LSTM applies CSR relations to model attention weights:

$$\begin{bmatrix} \alpha_{t-m}^{CSR} \\ \vdots \\ \alpha_{t-i}^{CSR} \\ \vdots \\ \alpha_{t-1}^{CSR} \end{bmatrix} = \mathrm{softmax}\left(\begin{bmatrix} \cos(\mathrm{h}_{t-m}^{CSR}, \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}_{t-i}^{CSR}, \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}_{t-1}^{CSR}, \mathrm{h}(S_t)) \end{bmatrix} \right). \tag{12}$$

CSR Attentive Sentence Modeling: $v^{CSR}(S_{t-c})$. To selectively encode the more important bi-grams in the surrounding contexts of a sentence into the representation of the sentence, a CSR attentive Convolutional Neural Network (CSR-Bi-CNN) is applied, as shown in Figure 5. The difference with Bi-CNN is that we jointly learn a bi-gram weight $\alpha^{CSR}(i, i+1)$ when conducting pooling:

$$\mathrm{v}^{CSR}(S_{t-c}) = \sum_{i=0}^{|S_{t-c}|} \alpha^{CSR}(i, i+1) \cdot \mathrm{h}^{CSR}(i, i+1). \tag{13}$$

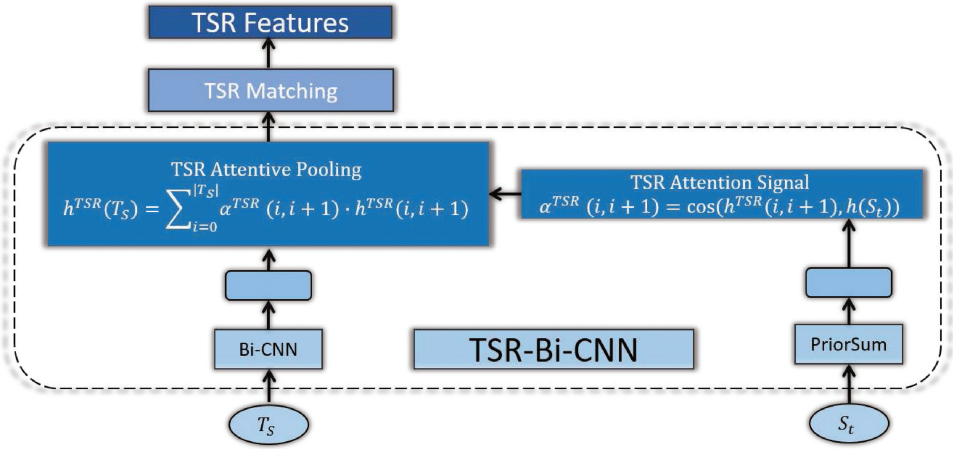


Fig. 6. Architecture of TSRSum.

Here, S_t is the sentence to conduct regression on; S_{t-c} is S_t's context sentence; and $\alpha^{CSR}(i, i+1)$ is the attention signal for the bi-gram vector $\mathrm{h}^{CSR}(i, i+1)$.

Unlike existing attentive pooling techniques (dos Santos et al. 2016; Yin et al. 2016), we use CSR relations to learn the pooling weights in Equation (14):

$$\begin{bmatrix} \alpha^{CSR}(0,1) \\ \vdots \\ \alpha^{CSR}(i,i+1) \\ \vdots \\ \alpha^{CSR}(|S_{t-c}|, |S_{t-c}+1|) \end{bmatrix} = \mathrm{softmax}\left(\begin{bmatrix} \cos(\mathrm{h}^{CSR}(0,1), \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}^{CSR}(i,i+1), \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}^{CSR}(|S_{t-c}|, |S_{t-c}+1|), \mathrm{h}(S_t)) \end{bmatrix} \right). \tag{14}$$

We use the softmax function to normalize the weights.

3.5 TSRSum

Titles usually reflect the key idea of the full document or at least they contain the phrases or words that are key to the document topics. As a result, if a sentence S_t is closely related to the title sentence, then we have enough confidence that S_t is important and should be included in the summary. TSRSum is meant to model this, as shown in Figure 6. Unlike CSRSum, where cosine similarity is applied to compute the matching of two main body sentences, here we use a bilinear scheme to calculate the matching of the title sentences and main body sentences in Equation (15), because we assume that article authors usually adopt different syntax styles to write titles to make them concise:

$$f^{TSR}(S_t) = \mathrm{h}^{TSR}(T_S)^T \mathrm{W}_{TSR}^{bilinear} \mathrm{h}(S_t), \tag{15}$$

where $\mathrm{W}_{TSR}^{bilinear} \in \mathbb{R}^{|\mathrm{h}^{TSR}(T_S)| \times |\mathrm{h}(S_t)|}$ is the bilinear matching matrix; $\mathrm{h}(S_t)$ is the sentence model of S_t (the output of PriorSum); and $\mathrm{h}^{TSR}(T_S)$ is the sentence model of the title sentence T_S.

TSR Attentive Sentence Modeling: $\mathrm{h}^{TSR}(T_S)$. To get $\mathrm{h}^{TSR}(T_S)$, TSR attentive Convolutional Neural Network (TSR-Bi-CNN) is applied, as shown in Figure 6. First, we adopt a similar Bi-CNN as described in Section 3.3 to get each bi-gram representation $\mathrm{h}^{TSR}(i, i+1)$. Then, the title sentence representation $\mathrm{h}^{TSR}(T_S)$ is obtained by leveraging TSR-based attentive pooling in Equation (16); note that the Bi-CNN part of TSRSum shares the same parameters as PriorSum because the

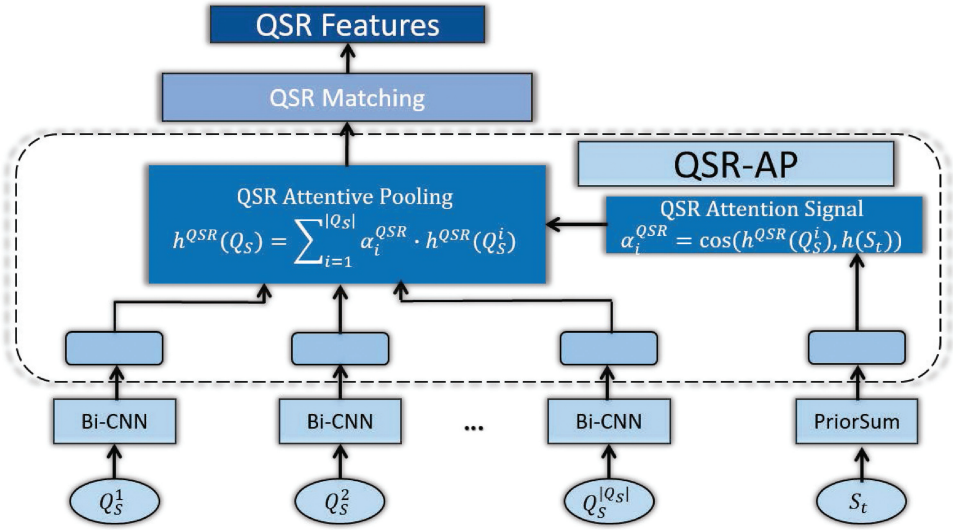


Fig. 7. Architecture of QSRSum.

number of title sentences is much smaller than the number of main body sentences and is not enough to train a good model:

$$\mathrm{h}_{TSR}(T_S) = \sum_{i=0}^{|T_S|} \alpha^{TSR}(i, i+1) \cdot \mathrm{h}^{TSR}(i, i+1), \tag{16}$$

where $\alpha^{TSR}(i, i+1)$ is the attention weight defined as follows:

$$\begin{bmatrix} \alpha^{TSR}(0,1) \\ \vdots \\ \alpha^{TSR}(i, i+1) \\ \vdots \\ \alpha^{TSR}(|S_{t-c}|, |S_{t-c}+1|) \end{bmatrix} = \mathrm{softmax}\left(\begin{bmatrix} \cos(\mathrm{h}^{TSR}(0,1), \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}^{TSR}(i, i+1), \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}^{TSR}(|S_{t-c}|, |S_{t-c}+1|), \mathrm{h}(S_t)) \end{bmatrix} \right). \tag{17}$$

The TSR-based attention used here is meant to identify those bi-grams in the title sentences that are closely related to S_t.

3.6 QSRSum

For query-focused summarization, whether a sentence should be included in the final summary depends not only on its importance but also on its relevance to a given query. The main body sentences that are closely related to the document topic might not necessarily answer the given query. On the contrary, a sentence that does not reflect the core idea of the article might properly answer the given query. QSRSum is proposed to model this, as shown in Figure 7. The main body sentences are written by the article authors while the queries are given by the readers. We assume that they belong to different spaces and use a bilinear matching mechanism to compute the QSR relation score between the query representation and main body sentence representation in Equation (18):

$$f^{QSR}(S_t) = \mathrm{h}^{QSR}(Q_S)^T \mathrm{W}_{QSR}^{bilinear} \mathrm{h}(S_t), \tag{18}$$

Table 2. Basic Surface Features Used in This Article

Feature	Description
$f_{len}(S_t) = \frac{1}{len(S_t)}$	$len(S_t)$ means the length of S_t
$f_{pos}(S_t) = \frac{1}{pos(S_t)}$	$pos(S_t)$ means the position of S_t in its document
$f_{tf}(S_t) = \frac{\sum_{w \in S_t} TF(w)}{f_{len}(S_t)}$	Average term frequency. $TF(w)$ is the term frequency of word w
$f_{df}(S_t) = \frac{\sum_{w \in S_t} DF(w)}{f_{len}(S_t)}$	Average document frequency. $DF(w)$ is the document frequency of word w

where $\mathrm{W}_{QSR}^{bilinear} \in \mathbb{R}^{|\mathrm{h}^{QSR}(Q_S)| \times |\mathrm{h}(S_t)|}$ is the bilinear matching matrix; $\mathrm{h}(S_t)$ is the sentence model of S_t (the outputs of PriorSum); $\mathrm{h}^{QSR}(Q_S)$ is the query sentence model of the queries $Q_S = \{Q_S^1, Q_S^2, \ldots, Q_S^{|Q_S|}\}$.

$\mathrm{h}^{QSR}(Q_S)$ is computed in Equation (19):

$$\mathrm{h}^{QSR}(Q_S) = \sum_{i=1}^{|Q_S|} \alpha_i^{QSR} \cdot \mathrm{h}^{QSR}(Q_S^i), \tag{19}$$

where $\mathrm{h}^{QSR}(Q_S^i)$ is gotten with the Bi-CNN in Figure 3; α_i^{QSR} is the attention signal where we use QSR-based attention to highlight more relevant queries in Equation (20).

$$\begin{bmatrix} \alpha_1^{QSR} \\ \vdots \\ \alpha_i^{QSR} \\ \vdots \\ \alpha_{|Q_S|}^{QSR} \end{bmatrix} = \mathrm{softmax}\left(\begin{bmatrix} \cos(\mathrm{h}^{QSR}(Q_S^1), \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}^{QSR}(Q_S^i), \mathrm{h}(S_t)) \\ \vdots \\ \cos(\mathrm{h}^{QSR}(Q_S^{|Q_S|}), \mathrm{h}(S_t)) \end{bmatrix} \right). \tag{20}$$

QSR-based attention uses the main body sentence to attend to the query sentences as opposed to attentive readers which use the query sentence to attend to sentence words (Cao et al. 2016). In other words, QSR-based attention tries to predict which query sentence is answered by the main body sentence while attentive readers try to predict which words (or parts) in the given sentence mostly answer the query.

3.7 SFSum

Even though PriorSum, CSRSum, TSRSum, and QSRSum are effective in modeling sentence meanings and relations, they cannot encode some important surface information, e.g., sentence length, sentence position and so on. These surface features are commonly used in feature engineering approaches and are indispensable for extractive summarization. To this end, we adopt four basic surface features listed in Table 2: $f_{len}(S_t)$ is the sentence length of S_t, $f_{pos}(S_t)$ is the sentence position of S_t; $f_{tf}(S_t)$ is the average term frequency of the terms in S_t, and $f_{df}(S_t)$ is the average document frequency of the terms in S_t. These features are concatenated together with CSR features, TSR features, QSR features, and prior features to decode the sentence importance score.

Table 3. Statistics of the DUC Datasets

	Clusters	Sentences	Length limit	Avg. queries per cluster	Avg. query length
2001	30	11,295	100 words	–	–
2002	59	15,878	100 words	–	–
2004	50	13,070	665 bytes	–	–
2005	50	45,931	250 words	2.18	13.55
2006	59	34,560	250 words	2.10	12.53
2007	30	24,282	250 words	1.62	15.11

3.8 Sentence Selection

There are two branches of commonly used algorithms for sentence selection, namely Greedy and Integer Linear Programming (ILP). Greedy is a little less promising than ILP because it greedily maximizes a function, which ILP exactly maximizes. However, it offers a nice trade-off between performance and computational cost. Besides, since the objective (Equation (2)) is submodular, maximizing it with Greedy has a mathematical guarantee on optimality (Lin and Bilmes 2010, 2011; Nemhauser et al. 1978). Thus, we use Greedy as the sentence selection algorithm. The algorithm starts with the sentence of the highest score. In each step, a new sentence S_t is added to the summary Ψ if it satisfies the following two conditions:

It has the highest score among the remaining sentences; and (21)

$$\frac{\textit{bi-gram-overlap}(S_t, \Psi)}{f_{len}(S_t)} \leq 1 - \lambda, \text{where } \textit{bi-gram-overlap}(S_t, \Psi) \text{ is the count of bi-gram} \quad (22)$$

overlap between sentence S_t and the current summary Ψ.

The algorithm terminates when the length constraint is reached. We skip the sentence if it does not satisfy the bi-gram overlap. Settings of λ are discussed in Section 7.4 below.

4 EXPERIMENTAL SETUP

The two main research questions we aim to answer are: Does modeling CSRs, TSRs and QSRs improve the performance of extractive summarization? And does it help to improve the performance of query-focused extractive summarization?

We list the datasets used in Section 4.1, implementation details of our model, SRSum, in Section 4.2, baselines in Section 4.3, and metrics in Section 4.4.

4.1 Datasets and Evaluation Metrics

For evaluation we use well-known corpora made available by the Document Understanding Conference (DUC).[2] The documents are all from the news domain and are grouped into various thematic clusters. Table 3 shows the size of the six datasets and the maximum length limit of summaries for each task. Each cluster contains 2 to 4 summaries written by professional experts. The DUC 2001, 2002, and 2004 datasets are for multi-document summarization. The DUC 2005, 2006, and 2007 datasets are for query-focused multi-document summarization. Most clusters contain more than one query sentence. The average number of query sentences per cluster of DUC 2007 are less, because there are more than one queries in one sentence. For example, "What is the scope of operations of Amnesty International and what are the international reactions to its activities?" For each document cluster, we concatenate all articles and split them into sentences using the tool

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

provided with the DUC 2003 dataset. We follow standard practice and train our models on two years of data and test on the third (Cao et al. 2015a).

No titles are available for the DUC 2001, 2002, and 2004 datasets. As a primary approximation titles, we use the first sentence as the title. This allows us to use TSR also for these datasets.

4.2 Implementation Details

During training, we first give a score to each sentence as ground truth based on human-written summaries using the official ROUGE evaluation tool. Then we train our model through conducting regression to the scores. This is the standard and commonly used training procedure for sentence regression methods (Cao et al. 2015a; Li et al. 2007). Stanford CoreNLP[3] is used to tokenize the sentences. The 50 dimensional GloVe[4] vectors are used to initialize the word embeddings. The hidden sizes of CNN and LSTM are the same with the word embeddings. The hidden sizes of the MLP layers are 100, 50, and 1. Increasing the number and dimension size of layers has little influence in the performance according to our experiments. We replace a word that is not contained in the GloVe vocabulary as "<U>" (Unknown). The word embeddings are fine-tuned during training. We also tried the 100, 200, and 300 dimensional word embeddings and found that they do not improve the results. Before feeding the word embeddings into the neural models, we perform the dropout operation that sets a random subset of its argument to zero with drop ratio $p = 0.5$. The dropout layer acts as a regularization method to reduce overfitting during training (Srivastava et al. 2014). To learn the weights of our model, we apply the diagonal variant of AdaGrad (Duchi et al. 2011) with mini-batches, whose size we set to 20. For the parameters m and n that represent the number of context sentences, we use values ranging from 1 to 10 on the DUC 2001 dataset. Generally, larger values will result in better performance, but the training speed slows down greatly when $m, n > 4$. As a trade-off, we set $m, n = 4$ in our experiments when compared with other methods. The best settings of the parameter λ are decided by presenting the ROUGE-2 performance with λ ranging from 0 to 0.9 with a step size of 0.05. The toolkit to reproduce the experimental results of SRSum is available on Github.[5]

4.3 Baselines and Approaches Used for Comparison

We first consider the generic multi-document summarization task. We list the methods compared against SRSum in Table 4. LexRank, ClusterHITS, ClusterCMRW are centroid-based methods; of these, ClusterHITS achieves the best ROUGE-1 score on DUC 2001. Lin is an MMR-based method. REGSUM, Ur, Sr, U+Sr, and SF are feature engineering-based methods with different features. R2N2 uses an RNN to encode each sentence into a vector based on its parse tree; then it performs sentence regression combined with 23 features. GA and ILP are greedy and ILP-based sentence selection algorithms, respectively. PriorSum uses a CNN to encode each sentence into a feature vector and then performs sentence regression combined with surface features.

We list the methods against which SRSum is compared for the query-focused multi-document summarization task in Table 5. LEAD simply selects the leading sentences to form a summary; it is often used as an official baseline of this task (Cao et al. 2016). QUERY_SIM ranks sentences according to their TF-IDF cosine similarity to the query. MultiMR is a graph-based manifold ranking method. SVR is a feature engineering-based method. ISOLATION contains two parts: sentence saliency is modeled as the cosine similarity between a sentence embedding and the document embedding and query relevance is modeled as the TF-IDF cosine similarity between a sentence and the

[3] http://stanfordnlp.github.io/CoreNLP/.
[4] http://nlp.stanford.edu/projects/glove/.
[5] https://github.com/PengjieRen/LibSum.

Table 4. Methods Considered for Comparison on the Multi-Document Summarization Task in Section 6.1

Acronym	Gloss	Reference
SRSum	PriorSum + CSRSum + TSRSum + SFSum.	Section 3
Unsupervised methods		
LexRank	Centroid-based method	Erkan and Radev (2004)
ClusterHITS	Centroid-based method	Wan and Yang (2008)
ClusterCMRW	Centroid-based method	Wan and Yang (2008)
Lin	Maximal marginal relevance method	Lin and Bilmes (2011)
Feature engineering-based methods		
REGSUM	Regression word saliency estimation	Cao et al. (2015a)
Ur	REGSUM with different features	Cao et al. (2015a)
Sr	SVR with 23 defined features	Cao et al. (2015a)
U+Sr	Combination of Ur and Sr	Cao et al. (2015a)
Deep-learning-based methods		
R2N2_GA	RNN with greedy sentence regression	Cao et al. (2015a)
R2N2_ILP	RNN with ILP sentence regression	Cao et al. (2015a)
PriorSum	REGSUM with different features	Cao et al. (2015b)

Table 5. Methods Considered for Comparison on the Query-Focused Multi-Document Summarization Task in Section 6.2

Acronym	Gloss	Reference
SRSum	The proposed model in this article	Section 3
Unsupervised methods		
LEAD	Select the leading sentences	Wasson (1998)
QUERY_SIM	TF-IDF cosine similarity	Cao et al. (2016)
MultiMR	Graph-based manifold ranking method	Wan and Xiao (2009)
Feature engineering-based methods		
SVR	SVR with hand-crafted features	Ouyang et al. (2011)
Deep-learning-based methods		
ISOLATION	Embedding and TF-IDF cosine similarity	Cao et al. (2016)
DocEmb	Embedding distributions-based summarization	Kobayashi et al. (2015)
AttSum	Neural attention summarization	Cao et al. (2016)
VAEs-A	Variational Auto-Encoders-based summarization	Li et al. (2017)

query. DocEmb summarizes by asymptotically estimating the KL-divergence based on document embedding distributions. AttSum learns distributed representations for sentences and the documents; it applies an attention mechanism to simulate human reading behavior. VAEs-A contains two components: latent semantic modeling and salience estimation. For latent semantic modeling, a neural generative model called Variational Auto-Encoders is employed to describe the observed sentences and the corresponding latent semantic representations. For salience estimation, VAEs-A considers the reconstruction for latent semantic space and observed term vector space. Finally, ILP is used to select the sentences based on the salience estimation.

(a) Ground truth. (b) SFSum.

(c) SRSum-SFSum. (d) SRSum.

Fig. 8. Visualization of sentence scoring. The depth of the color corresponds to the importance of the sentence given by groundtruth or models. The boxed characters S_i indicate sentence start. (Best viewed in color.)

4.4 Metrics and Significance Testing

The ROUGE metrics are the official metrics of the DUC extractive summarization tasks (Rankel et al. 2013). We use the official ROUGE tool[6] (Lin 2004) to evaluate the performance of the baselines as well as our approaches. The length constraint is "-l 100" for DUC 2001/2002, "-b 665" for DUC 2004, and "-l 250" for DUC 2005/2006/2007. We take ROUGE-2 recall as the main metric for comparison because Owczarzak et al. (2012) show its effectiveness for evaluating automatic summarization systems. We impose constraints on the length of generated summaries according to the official summary length limits.

For significance testing, we use a two-tailed paired Student's t-test with $p < 0.05$.

5 CASE STUDIES

Before reporting on the outcomes aimed at answering our research questions, we present two examples to illustrate our methods at work. From top-left to bottom-right, Figure 8 shows the ground truth, SFSum, SRSum-SFSum (that is, SRSUm minus the SFSum component), and SRSum,

[6]ROUGE-1.5.5 with options: -n 2 -m -u -c 95 -x -r 1000 -f A -p 0.5 -t 0.

respectively. The depth of the color corresponds to the importance of the sentence given by ground truth or models. We can see that SFSum cannot properly distinguish the different levels of importance of different sentences. It wrongly estimates which of the two is more important, the third sentence or the fourth. SRSum-SFSum is better than SFSum, however its ability to distinguish different degrees of importance (compared to the ground truth) is still limited, due to a lack of important surface factors encoded by SFSum. Finally, SRSum can better approach the ground truth, which means that combining sentence relation features with surface features improves the performance of sentence regression.

As a second example, we visualize the learned the attention signals of the CSR, TSR, and QSR relations, as shown in Figure 9. Figure 9(a) illustrates the CSR word-level attentive pooling. We can see that S_t helps to pick up the more important words of in the surrounding context sentence S_{t+1} when modeling S_{t+1} into a vector representation. Figure 9(b) illustrates the CSR sentence-level attentive pooling. As shown, the context sentences S_{t+1} to S_{t+5} are treated differently according to their relevance to S_t. The more relevant sentences have more effect on the final results. Figure 9(c) illustrates the TSR attentive pooling. The two different main body sentences S_t highlight different parts of the title sentence T_S. Figure 9(d) illustrates the QSR attentive pooling. The three main body sentences S_t answer different sub-queries of the given query Q_S. As shown, the learned signals correctly pay attention to the corresponding sub-queries.

6 RESULTS

In Section 6.1, we compare SRSum with several state-of-the-art methods on the DUC 2001, 2002, and 2004 multi-document summarization datasets. We further evaluate the effectiveness of SRSum on the DUC 2005, 2006, and 2007 query-focused multi-document summarization datasets in Section 6.2. Here, we show that modeling CSR, TSR, and QSR relations is also useful for query-focused summarization. We follow with further analyses of the results in Section 7.

6.1 Generic Multi-Document Summarization

The ROUGE scores of the methods listed in Table 4 on the DUC 2001, 2002, and 2004 datasets are presented in Table 6. For each metric, the best performance per dataset is indicated in bold face. Generally, SRSum achieves the best performance in terms of both ROUGE-1 and ROUGE-2 on all three datasets. Although ClusterHITS achieves higher ROUGE-1 scores on DUC 2001, its ROUGE-2 scores are much lower. In contrast, SRSum works quite stably across datasets. ClusterCMRW gets higher ROUGE-1 scores on DUC 2002 and its ROUGE-2 score is comparable with R2N2_GA, but SRSum improves over ClusterCMRW by over 1.6 percentage points (%pts) in terms of ROUGE-2.

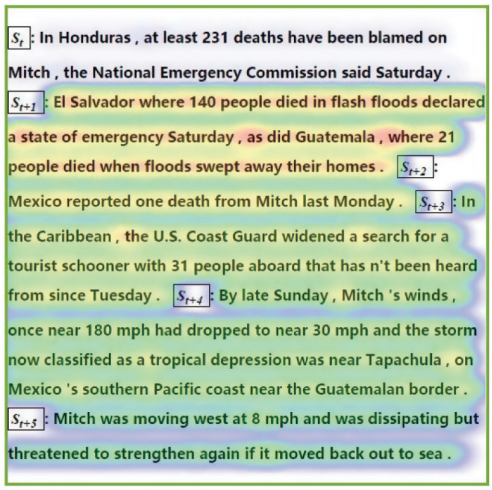
SRSum is much effective than deep learning models, R2N2_GA, R2N2_ILP, and PriorSum. Specifically, SRSum improves over PriorSum, the best method, in terms of ROUGE-2 by 1%pt on DUC 2001, 2002, and over 0.5%pt on DUC 2004. The improvements in terms of ROUGE-2 achieved on the three benchmark datasets are considered big (McDonald 2007; Rankel et al. 2013). Note that SRSum only uses four basic surface features while R2N2_GA, R2N2_ILP, and PriorSum are combinations of neural models and dozens of hand-crafted features. The neural parts of R2N2_GA, R2N2_ILP, and PriorSum model the stand-alone sentence, while SRSum further considers the CSR and TSR relations.

The main insight is that SRSum captures complementary factors by considering CSR and TSR relations that existing neural models or hand-crafted features do not capture, which we will analyze in detail in Section 7.

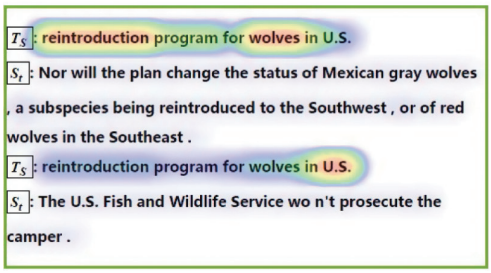
6.2 Query-Focused Multi-Document Summarization

Next, we consider the performance of SRSum on the query-focused multi-document summarization task.

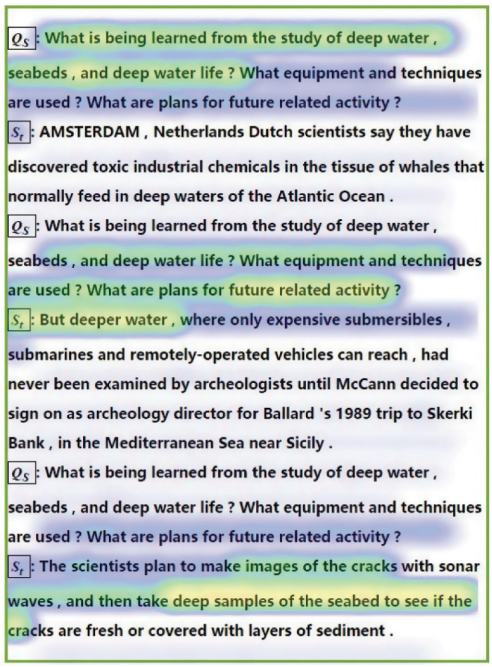
(a) CSR Word-level attention. The color depth corresponds to the attention signal $\alpha^{CSR}(i, i+1)$ (Equation 14).



(b) CSR Sentence-level attention. Color depth corresponds to the attention signal α_{t-i}^{CSR} (Equation 12).



(c) TSR attention. Color depth corresponds to the attention signal $\alpha^{TSR}(i, i+1)$ (Equation 17).



(d) QSR attention. Color depth corresponds to the attention signal α_i^{QSR} (Equation 20).

Fig. 9. Visualization of CSR, TSR, and QSR attention mechanisms. (Best viewed in color.)

The results on the query-focused multi-document summarization task on the DUC 2005, 2006, and 2007 datasets are presented in Table 7. Generally, SRSum achieves the best performance on all three datasets. SRSum is more effective than traditional methods like LEAD, QUERY_SIM, MultiMR, and SVR, which shows the great potential of deep learning techniques for this task. SRSum

Table 6. Multi-document Summarization

	Approach	ROUGE-1	ROUGE-2
DUC 2001	Peer T	33.03	7.86
	ClusterHITS*	**37.42**	6.81
	LexRank	33.43	6.09
	Ur*	34.28	6.66
	Sr*	34.06	6.65
	U+Sr*	33.98	6.54
	R2N2_GA*	35.88	7.64
	R2N2_ILP*	36.91	7.87
	PriorSum*	35.98	7.89
	SFSum+PriorSum	35.79	7.97
	SRSum	36.04†	**8.44**†
DUC 2002	Peer 26	35.15	7.64
	ClusterCMRW*	38.55	8.65
	LexRank	35.29	7.54
	Ur*	34.16	7.66
	Sr*	34.23	7.81
	U+Sr*	35.13	8.02
	R2N2_GA*	36.84	8.52
	R2N2_ILP*	37.96	8.88
	PriorSum*	36.63	8.97
	SFSum+PriorSum	37.47	9.01
	SRSum	**38.93**†	**10.29**†
DUC 2004	Peer 65	37.88	9.18
	REGSUM*	38.57	9.75
	LexRank	37.87	8.88
	Lin*	39.35	–
	Ur*	37.22	9.15
	Sr*	36.72	9.10
	U+Sr*	37.62	9.31
	R2N2_GA*	38.16	9.52
	R2N2_ILP*	38.78	9.86
	PriorSum*	38.91	10.07
	SFSum+PriorSum	38.13	9.97
	SRSum	**39.29**†	**10.70**†

(ROUGE results (%) on DUC 2001, 2002, 2004 datasets. Per dataset, significant improvements over the underlined methods are marked with † (t-test, $p < .05$). Peer T, 26, 65 are the best performing participants at DUC 2001, 2002, 2004, respectively. Scores of the methods marked with * are taken from the corresponding references listed in Table 4. Note that PriorSum* listed in this table is the one proposed in Cao et al. (2015b). Different from ours, they combine unigram, bigram, and trigram CNN as well as three surface features. Also note that a part of SRSum's improvement also comes from tuning SFSum+PriorSum, so we also list the results of SFSum+PriorSum for comparison.)

Table 7. Query-Focused Multi-Document Summarization

	System	ROUGE-1	ROUGE-2
DUC 2005	Peer 15	37.52	7.25
	LEAD*	29.71	4.69
	QUERY_SIM*	32.95	5.91
	SVR	36.91	7.04
	MultiMR*	35.58	6.81
	DocEmb*	30.59	4.69
	ISOLATION*	35.72	6.79
	AttSum*	37.01	6.99
	SFSum+PriorSum	37.86	7.39
	SRSum	**39.83**†	**8.57**†
DUC 2006	Peer 24	41.11	9.56
	LEAD*	32.61	5.71
	QUERY_SIM*	35.52	7.10
	SVR	39.24	8.87
	MultiMR*	38.57	7.75
	DocEmb*	32.77	5.61
	ISOLATION*	40.58	8.96
	AttSum*	40.90	9.40
	VAEs-A*	39.60	8.90
	SFSum+PriorSum	40.30	9.13
	SRSum	**42.82**†	**10.46**†
DUC 2007	Peer 15	44.51	12.45
	LEAD*	36.14	8.12
	QUERY_SIM*	36.32	7.94
	SVR	43.42	11.10
	MultiMR*	41.59	9.34
	DocEmb*	33.88	6.46
	ISOLATION*	42.76	10.79
	AttSum*	43.92	11.55
	VAEs-A*	42.10	11.00
	SFSum+PriorSum	42.98	11.29
	SRSum	**45.01**†	**12.80**†

(ROUGE Results (%) on DUC 2005, 2006, 2007 datasets. Per dataset, significant improvements over the underlined methods are marked with † (t-test, $p < .05$). Peer 15, 24, 15 are the best performing participants at DUC 2005, 2006, 2007, respectively. Scores of the methods marked with * are taken from the corresponding references listed in Table 5. Note that a part of SRSum's improvement also comes from tuning SFSum+PriorSum, so we also list the results of SFSum+PriorSum for comparison.)

outperforms embedding-based methods like ISOLATION and DocEmb, because ISOLATION and DocEmb do not have carefully designed network architectures. Instead, they simply benefit from the distributed representations of words that have a limited capability. Though AttSum and VAEs-A have carefully designed network architectures, it merely considers a stand-alone sentence and its relevance to the given query and neglects the CSR and TSR relations. Besides the factors considered in AttSum, SRSum further considers the CSR and TSR relations, which leads to the observed improvements over AttSum.

7 ANALYSIS

Having answered our main research questions in the previous section, we now analyze our experimental results and the impact of our modeling choices. In Section 7.1, we analyze the effectiveness of the five components of our model; in Section 7.2, we analyze the effectiveness of SRSum compared to each of the surface features in SFSum; in Section 7.3, we replace our neural model with a feature engineering method to analyze the effectiveness of our neural model; in Section 7.4, we explore different settings of the threshold parameter λ in the greedy algorithm (sentence selection phase, Section 3.8) to determine the sensitivity of our method; and we analyze our attention mechanisms.

7.1 Effectiveness of Different Components

We analyze the effectiveness of the five components, SFSum, PriorSum, CSRSum, TSRSum, and QSRSum of our model on the six DUC datasets by removing each component in turn. The results are listed in Table 8. Generally, removing any component will decrease the ROUGE scores. SFSum is the most effective component as it contains essential surface features (i.e., the sentence length, the sentence position, and the word frequency), which are commonly recognized as indispensable in the news summarization literature (Cao et al. 2015a; Li et al. 2007). These features are not considered by the other components. Each of the other components is relatively less effective because their effects overlap. They all encode the sentence importance from the semantic perspective. PriorSum is also very important as it encodes the meaning of sentences, which proves to be an essential factor for summarization. What we want to emphasize from the results in Table 8 is that CSRSum, TSRSum, and QSRSum are also very useful, which means that the CSR, TSR, and QSR relations indeed exist and are helpful factors for both summarization tasks.

We also found that the decreases of removing the five components vary with different datasets. CSRSum is more effective on the DUC 2002, 2004, and 2007 datasets, while TSRSum is more effective on the DUC 2005, 2006, and 2007 datasets. The reason that TSRSum is less effective or simply not useful on the DUC 2001 and 2004 datasets is because no titles are available on these datasets. Instead, we regard the first sentence as the title, which is mostly reasonable but still has many exceptions. QSRSum is essential on the DUC 2005, 2006, and 2007 datasets as it is the only component that considers the query relevance, which is indispensable on the query-focused summarization task.

7.2 SRSum vs. the Surface Features

Pearson correlation coefficients can reflect the effectiveness of the feature to some extent. We examine correlations with the ground truth of the surface features in Table 2 and of SRSum, as shown in Table 9. SRSum achieves higher correlation scores with the ground truth than the surface features ($f_{len}(S_t)$, $f_{pos}(S_t)$, $f_{tf}(S_t)$, and $f_{df}(S_t)$). The results also show that $f_{len}(S_t)$ and $f_{pos}(S_t)$ are important features for extractive summarization, confirming lessons reported in the literature (Cao et al. 2015a, 2015b; Wan and Zhang 2014).

Table 8. Effectiveness of Different Components

		SRSum without the sub-model		The sub-model alone	
		ROUGE-1 decrease	ROUGE-2 decrease	ROUGE-1	ROUGE-2
DUC2001	SFSum	1.01	0.79	34.82	7.76
	PriorSum	0.97	0.85	31.68	5.21
	CSRSum	0.65	0.76	32.47	7.45
	TSRSum	0.5	−0.31	31.34	5.04
DUC2002	SFSum	2.43	1.52	37.33	8.98
	PriorSum	1.01	0.70	33.27	6.61
	CSRSum	0.71	0.59	36.87	9.01
	TSRSum	0.11	0.26	31.31	5.65
DUC2004	SFSum	1.28	0.92	37.74	9.60
	PriorSum	0.91	0.77	35.02	7.37
	CSRSum	1.02	0.65	37.03	8.97
	TSRSum	−0.24	0.10	34.96	7.53
DUC2005	SFSum	2.07	1.23	37.07	6.81
	PriorSum	0.44	0.72	35.05	5.58
	CSRSum	0.6	0.41	35.80	6.09
	TSRSum	0.82	0.57	35.21	5.62
	QSRSum	1.05	0.78	36.76	6.11
DUC2006	SFSum	2.82	1.43	39.47	8.60
	PriorSum	0.76	0.53	36.86	7.16
	CSRSum	1.08	0.56	38.97	8.15
	TSRSum	0.87	0.42	37.51	7.33
	QSRSum	1.43	0.69	39.48	8.24
DUC2007	SFSum	2.53	1.76	42.28	11.15
	PriorSum	0.87	0.69	40.37	8.69
	CSRSum	0.87	0.51	40.59	10.93
	TSRSum	0.58	0.39	40.81	9.38
	QSRSum	1.41	0.80	41.33	9.54

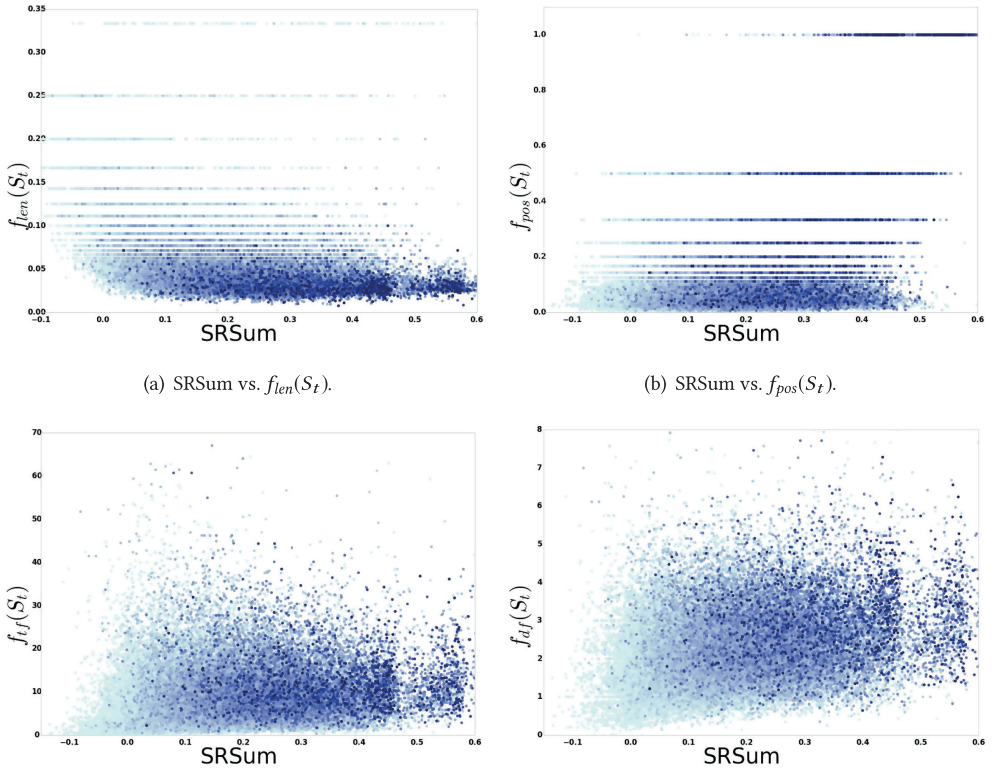
(All models in this table are retrained in the same procedure as we train SRSum. For example, if we remove a component from SRSum, then the left model is considered as a new model and retrained. Though CSRSum, TSRSum, and QSRSum are all useful, we would like to emphasize that SFSum and PriorSum are more essential parts than them.)

Pearson correlation coefficients only reflect linear correlations. Hence, we further visualize the relation between the feature space of SRSum and the surface features $f_{len}(S_t)$, $f_{pos}(S_t)$, $f_{tf}(S_t)$, and $f_{df}(S_t)$, as shown in Figure 10, by plotting SRSum scores against the feature values.[7] The color depth reflects the importance of a sentence according to the ground truth. Low SRSum scores

[7]The scores for SRSum range from −1 to 1 as its activation function is tanh.

Table 9. Pearson Correlation Coefficients of Surface Features and SRSum

Dataset (DUC)	2001	2002	2004	2005	2006	2007
$f_{len}(S_t)$	0.31	0.31	0.37	0.35	0.25	0.33
$f_{pos}(S_t)$	0.28	0.31	0.40	0.23	0.20	0.37
$f_{tf}(S_t)$	0.14	0.20	0.24	0.19	0.32	0.39
$f_{df}(S_t)$	0.19	0.23	0.38	0.25	0.37	0.24
SRSum	0.48	0.46	0.54	0.41	0.45	0.49



(a) SRSum vs. $f_{len}(S_t)$.

(b) SRSum vs. $f_{pos}(S_t)$.

(c) SRSum vs. $f_{tf}(S_t)$.

(d) SRSum vs. $f_{df}(S_t)$.

Fig. 10. SRSum scores vs. surface feature scores. Each point represents a sentence. The color depth reflects the importance of the sentence according to the ground truth. (Best viewed in color.)

mostly correspond to sentences with low ROUGE-2 scores, which means that SRSum is able to effectively identify useless sentences. Also, high SRSum scores mostly correspond to sentences with high ROUGE-2 scores, which means that SRSum can distinguish the most important sentences effectively. Obviously, this ability to identify the most important sentences is extremely useful, as a summary is usually short, containing just a few very important sentences; we should also note that this ability still leaves room for improvement as there are low-scoring and high-scoring sentences mixed together.

Table 10. CSR, TSR, and QSR Implementations with Hand-Crafted Features

Feature	Description
$f_{c1}(S_t, C_S) = \cos(TF(S_t), TF(C_S))$	Cosine of TF vectors of S_t and its contexts C_S
$f_{c2}(S_t, C_S) = \cos(emb(S_t), emb(C_S))$	Cosine of average embedding vectors of S_t and C_S
$f_{t1}(S_t, T_S) = \cos(TF(S_t), TF(T_S))$	Cosine of TF vectors of sentence S_t and title T_S
$f_{t2}(S_t, T_S) = \cos(emb(S_t), emb(T_S))$	Cosine of average embedding vectors of S_t and T_S
$f_{q1}(S_t, Q_S) = \cos(TF(S_t), TF(Q_S))$	Cosine of TF vectors of sentence S_t and query Q_S
$f_{q2}(S_t, Q_S) = \cos(emb(S_t), emb(Q_S))$	Cosine of average embedding vectors of S_t and Q_S

Table 11. Deep Learning vs. Feature Engineering

	Approach	ROUGE-1	ROUGE-2
DUC 2001	SFSum	34.82	7.76
	$f_{c1,c2,t1,t2}$+SFSum	35.44	8.10
	SRSum	36.04	8.44
DUC 2002	SFSum	37.33	8.98
	$f_{c1,c2,t1,t2}$+SFSum	37.01	9.19
	SRSum	39.29	10.70
DUC 2004	SFSum	37.74	9.60
	$f_{c1,c2,t1,t2}$+SFSum	37.97	9.78
	SRSum	39.03	10.57
DUC 2005	$f_{c1,c2,t1,t2,q1,q2}$+SFSum	39.75	8.21
	SRSum	39.83	8.57
DUC 2006	$f_{c1,c2,t1,t2,q1,q2}$+SFSum	41.45	9.57
	SRSum	42.82	10.46
DUC 2007	$f_{c1,c2,t1,t2,q1,q2}$+SFSum	44.29	11.73
	SRSum	45.01	12.80

7.3 Choice of Learning Framework

In principle, the core modeling idea underlying SRSum, namely CSRSum, to exploit contextual relations, TSRSum to exploit title relations, and QSRSum to exploit query relations, can be implemented differently, with hand-crafted features that capture these relations. To verify the effectiveness of our deep learning implementation, we compare SRSum with some hand-crafted features. Each of the three relations is encoded with two features; see Table 10. These features should be the first features that come to mind if we implement CSR, TSR, and QSR using a feature engineering approach.

From Table 11 we see that although these hand-crafted features all improve SFSum, they are much less effective than SRSum. The reasons are at least threefold. First, SRSum models sentences with relation-based attentive pooling Bi-CNN, which takes advantage of Bi-CNN and uses relation enhanced attentive pooling to learn to attend to more important words, while $f_{c1}(S_t, C_S)$, $f_{t1}(S_t, T_S)$, and $f_{q1}(S_t, Q_S)$ use a sparse TF representation and $f_{c2}(S_t, C_S)$, $f_{t2}(S_t, T_S)$, and $f_{q2}(S_t, Q_S)$ use a simple average word embedding representation to model sentences. Second,

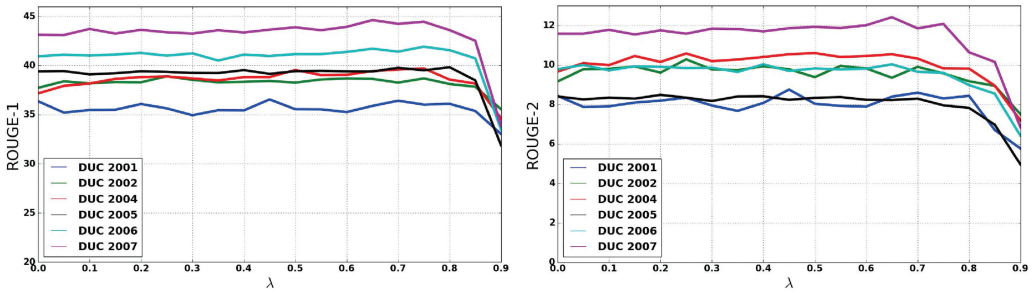


Fig. 11. Sensitivity to the parameter λ of SRSum during sentence selection.

SRSum leverages LSTMs to encode context sentences and relation-enhanced attentive pooling to learn to attend to the more important sentences, which gives SRSum the ability to selectively remember useful factors from the sequence of context sentences, while these hand-crafted features simply average sentence representations. Third, the word embeddings of SRSum are fine-tuned during training, which makes them better suited for this task.

In sum, while the core modeling ideas underlying SRSum could be realized differently, using a feature engineering-based approach, our deep-learning approach proves to be more effective.

7.4 Threshold Parameter λ

Recall from Section 3.8 that after giving a salience score to each sentence, we greedily select the sentence with the highest score for inclusion in the final summary until the length constraint is reached. During the process, a parameter λ is used to avoid redundant sentences by discarding sentences whose bi-gram overlap with already selected sentences is larger than $1 - \lambda$; see Equation (22). To investigate the sensitivity of our choice of λ, we examine the performance of SRSum with the threshold parameter λ ranging from 0 to 0.9 with a step size of 0.05. The results are shown in Figure 11, where we plot performance in terms of ROUGE-1 and ROUGE-2 against λ. Generally, the performance of SRSum is not sensitive to the setting of λ for values less than 0.8, with the best performance achieved around 0.65 to 0.75. When λ is over 0.8, the performance drops sharply. This is because some important sentences with little redundancy are dropped from the summary.

7.5 Attention

We have illustrated our CSR-, TSR-, and QSR-based attention mechanisms with some examples in Figure 9. Here, we analyze the performance of these attention mechanisms in Table 12, using the same data as in Section 6.1. Generally, without CSR-, TSR-, and QSR-based attentions, SRSum drops around 0.3%pt–0.5%pt in terms of ROUGE-2. The decreases differ between the six datasets, with larger decreases on the DUC 2001, 2002, 2005, and 2006 datasets and smaller decreases on the DUC 2004 and 2007 datasets. Interestingly, the different types of attention, CSR, TSR, and QSR, all yield comparable improvements over SRSum without attention. But their contribution is complementary as removing all the attention mechanisms leads to further decreases compared with removing only one of them, on all metrics and datasets, demonstrating the need for all three types of attention.

8 CONCLUSIONS & FUTURE WORK

This article presents a novel neural network model, SRSum, to automatically learn features contained in sentences and in its CSR, TSR, and QSR relations between sentences. We have conducted

Table 12. Analyzing Attention Mechanisms on the Multi-Document Summarization Task (%)

	Removed	ROUGE-1 decrease	ROUGE-2 decrease
DUC 2001	CSR attention	0.71	0.35
	TSR attention	0.33	0.19
	CSR+TSR attention	0.77	0.38
DUC 2002	CSR attention	0.63	0.31
	TSR attention	0.29	0.11
	CSR+TSR attention	0.65	0.34
DUC 2004	CSR attention	0.37	0.27
	TSR attention	0.21	0.09
	CSR+TSR attention	0.39	0.30
DUC 2005	CSR attention	0.57	0.36
	TSR attention	0.49	0.35
	QSR attention	0.51	0.33
	CSR+TSR+QSR attention	0.61	0.41
DUC 2006	CSR attention	0.62	0.34
	TSR attention	0.51	0.32
	QSR attention	0.67	0.42
	CSR+TSR+QSR attention	0.71	0.45
DUC 2007	CSR attention	0.28	0.23
	TSR attention	0.24	0.21
	QSR attention	0.49	0.31
	CSR+TSR+QSR attention	0.50	0.32

(All models in this table are retrained in the same procedure as we train SRSum. For example, if we remove the CSR attention from SRSum, then the left model is considered as a new model and retrained.)

extensive experiments on the DUC multi-document summarization datasets and query-focused multi-document summarization datasets. On all datasets, SRSum outperforms the baselines and achieves the best published performance on the DUC 2001, 2002, and 2004 datasets.

Based on our experimental results and subsequent analyses, we conclude that (1) to generate a better summary, leveraging sentence relations is necessary, mostly because sentence relations can influence the understanding of sentence meaning or even the article structure, both of which are important factors to consider when people write a summary; (2) our CSR, TSR, and QSR-based attention mechanisms show potential in being able to mimic aspects of human reading behaviors, i.e., the abilities to be context-aware, to be aware of the main idea of an article, and to be aware of answer sentences.

Despite the improvements of our summarization model over existing methods, it also has limitations. First, the model has a complex neural network architecture. Proper training of such complex networks is often a time-consuming task. Second, even though we carry out experiments to evaluate the effectiveness of each component (SFSum, PriorSum, CSRSum, TSRSum, QSRSum) and show some examples, it is still hard to give an intuitive explanation of how the learned latent features work, due to the characteristics of deep learning.

We believe our work can be advanced and extended in several directions: SRSum can be enriched by introducing a mechanism to explicitly model fine-grained sentence relations, such as parallelism relations, progressive relations, inductive reasoning relations and deductive reasoning relations (Song et al. 2016). Variants of SRSum can be also extended to other tasks, such as abstractive summarization or summarization on social text streams (Ren et al. 2013, 2016b).

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments.

REFERENCES

Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *J. Artif. Intell. Res.* 17, 1 (Aug. 2002), 35–55.

Lidong Bing, Piji Li, Yi Liao, Wai Lam, Weiwei Guo, and Rebecca J. Passonneau. 2015. Abstractive multi-document summarization via phrase selection and merging. In *ACL*. 1587–1597.

Danushka Bollegala, Naoaki Okazaki, and Mitsuru Ishizuka. 2010. A bottom-up approach to sentence ordering for multidocument summarization. *Inform. Process. Manag.* 46, 1 (2010), 89–109.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2016. AttSum: Joint learning of focusing and summarization with neural attention. In *COLING*. 547–556.

Ziqiang Cao, Furu Wei, Li Dong, Sujian Li, and Ming Zhou. 2015a. Ranking with recursive neural networks and its application to multi-document summarization. In *AAAI*. 2153–2159.

Ziqiang Cao, Furu Wei, Sujian Li, Wenjie Li, Ming Zhou, and Houfeng Wang. 2015b. Learning summary prior representation for extractive summarization. In *ACL*. 829–833.

Jaime Carbonell and Jade Goldstein. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*. 335–336.

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. In *ACL*. 484–494.

Sumit Chopra, Michael Auli, and Alexander M. Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL-HLT*. 93–98.

Janara Christensen, [No Value] Mausam, Stephen Soderland, and Oren Etzioni. 2013. Towards coherent multi-document summarization. In *HLT-NAACL*. Association for Computational Linguistics, 1163–1173.

Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv:1602.03609* (2016).

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.* 12 (2011), 2121–2159.

Ted Dunning. 1993. Accurate methods for the statistics of surprise and coincidence. *Comput. Linguis.* 19, 1 (1993), 61–74.

Günes Erkan and Dragomir R. Radev. 2004. LexRank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Intell. Res.* 22, 1 (2004), 457–479.

Katja Filippova, Enrique Alfonseca, Carlos A. Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with LSTMs. In *EMNLP*. 360–368.

Matthew W. Gardner and Stephen R. Dorling. 1998. Artificial neural networks (the multilayer perceptron) – A review of applications in the atmospheric sciences. *Atmospheric Environ.* 32, 14 (1998), 2627–2636.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *ILP-NLP*. 10–18.

Jade Goldstein, Vibhu Mittal, Jaime Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *NAACL-ANLP-AutoSum*. 40–48.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. 2013. Speech recognition with deep recurrent neural networks. In *ICASSP*. 6645–6649.

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *EACL*. 712–721.

Baotian Hu, Qingcai Chen, and Fangze Zhu. 2015. LCSTS: A large scale Chinese short text summarization dataset. In *EMNLP*. 1967–1972.

Yue Hu and Xiaojun Wan. 2013. PPSGen: Learning to generate presentation slides for academic papers. In *IJCAI*. 2099–2105.

Yue Hu and Xiaojun Wan. 2015. PPSGen: Learning-based presentation slides generation for academic papers. *IEEETrans. Knowl. Data Eng.* 27, 4 (2015), 1085–1097.

Mikael Kågebäck, Olof Mogren, Nina Tahmasebi, and Devdatt Dubhashi. 2014. Extractive summarization using continuous vector space models. In *CVSC@EACL*. 31–39.

Hayato Kobayashi, Masaki Noguchi, and Taichi Yatsuka. 2015. Summarization based on embedding distributions. In *EMNLP*. 1984–1989.

Julian Kupiec, Jan Pedersen, and Francine Chen. 1995. A trainable document summarizer. In *SIGIR*. 68–73.

Mirella Lapata. 2003. Probabilistic text structuring: Experiments with sentence ordering. In *ACL*. Association for Computational Linguistics, 545–552.

Chen Li, Xian Qian, and Yang Liu. 2013. Using supervised bigram-based ILP for extractive summarization. In *ACL*. 1004–1013.

Piji Li, Zihao Wang, Wai Lam, Zhaochun Ren, and Lidong Bing. 2017. Salience estimation via variational auto-encoders for multi-document summarization. In *AAAI*. 3497–3503.

Sujian Li, You Ouyang, Wei Wang, and Bin Sun. 2007. Multi-document summarization using support vector regression. In *DUC*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. 74–81.

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *COLING*. 495–501.

Chin-Yew Lin and Eduard Hovy. 2002. From single to multi-document summarization: A prototype system and its evaluation. In *ACL*. 457–464.

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL-HLT*. 912–920.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *HLT*. 510–520.

Hui Lin and Jeff Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. In *UAI*. 479–490.

Hans Peter Luhn. 1958. The automatic creation of literature abstracts. *IBM J. Res. Dev.* 2, 2 (1958), 159–165.

Ryan McDonald. 2007. A study of global inference algorithms in multi-document summarization. In *ECIR*. 557–564.

Rada Mihalcea. 2004. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In *ACL*. Article No. 20.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into texts. In *EMNLP*. 404–411.

Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *CoNLL*.

George L. Nemhauser, Laurence A. Wolsey, and Marshall L. Fisher. 1978. An analysis of approximations for maximizing submodular set functions—I. *Math. Prog. Seri. B* 14, 1 (1978), 265–294.

You Ouyang, Sujian Li, and Wenjie Li. 2007. Developing learning strategies for topic-based summarization. In *CIKM*. 79–86.

You Ouyang, Wenjie Li, Sujian Li, and Qin Lu. 2011. Applying regression models to query-focused multi-document summarization. *Inform. Process. Manag.* 47, 2 (2011), 227–237.

Paul Over and James Yen. 2004. Introduction to DUC-2001: An intrinsic evaluation of generic news text summarization systems. In *DUC*.

Karolina Owczarzak, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2012. An assessment of the accuracy of automatic evaluation in summarization. In *NAACL-HLT*. 1–9.

Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. 2000. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies. In *NAACL-ANLP-AutoSum'00 Proceedings of the 2000 NAACL-ANLP Workshop on Automatic Summarization*. 21–30.

Dragomir R. Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. 2004. Centroid-based summarization of multiple documents. *Inform. Process. Manag.* 40, 6 (2004), 919–938.

Peter A. Rankel, John M. Conroy, Hoa Trang Dang, and Ani Nenkova. 2013. A decade of automatic content evaluation of news summaries: Reassessing the state of the art. In *ACL*. 131–136.

Pengjie Ren, Zhumin Chen, Zhaochun Ren, Furu Wei, Jun Ma, and Maarten de Rijke. 2017. Leveraging contextual sentence relations for extractive summarization using a neural attention model. In *SIGIR*. ACM, 95–104.

Pengjie Ren, Furu Wei, Zhumin Chen, Jun Ma, and Ming Zhou. 2016a. A redundancy-aware sentence regression framework for extractive summarization. In *COLING*. 33–43.

Zhaochun Ren, Oana Inel, Lora Aroyo, and Maarten de Rijke. 2016b. Time-aware multi-viewpoint summarization of multilingual social text streams. In *CIKM*. 387–396.

Zhaochun Ren, Shangsong Liang, Edgar Meij, and Maarten de Rijke. 2013. Personalized time-aware tweets summarization. In *SIGIR*. 513–522.

Dennis W. Ruck, Steven K. Rogers, Matthew Kabrisky, Mark E. Oxley, and Bruce W. Suter. 1990. The multilayer perceptron as an approximation to a Bayes optimal discriminant function. *IEEE Trans. Neural Netw.* 1, 4 (1990), 296–298.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*. 379–389.

Wei Song, Tong Liu, Ruiji Fu, Lizhen Liu, Hanshi Wang, and Ting Liu. 2016. Learning to identify sentence parallelism in student essays. In *COLING*. 794–803.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* 15, 1 (2014), 1929–1958.

Sebastian Tschiatschek, Rishabh Iyer, Haochen Wei, and Jeff Bilmes. 2014. Learning mixtures of submodular functions for image collection summarization. In *NIPS*. 1413–1421.

Xiaojun Wan. 2008. An exploration of document impact on graph-based multi-document summarization. In *EMNLP*. 755–784.

Xiaojun Wan. 2011. Using bilingual information for cross-language document summarization. In *NAACL-HLT*. 1546–1555.

Xiaojun Wan, Ziqiang Cao, Furu Wei, Sujian Li, and Ming Zhou. 2015. Multi-document summarization via discriminative summary reranking. *arXiv:1507.02062* (2015).

Xiaojun Wan and Jianguo Xiao. 2009. Graph-based multi-modality learning for topic-focused multi-document summarization. In *IJCAI*. 1586–1591.

Xiaojun Wan and Jianwu Yang. 2008. Multi-document summarization using cluster-based link analysis. In *SIGIR*. 299–306.

Xiaojun Wan and Jianmin Zhang. 2014. CTSUM: Extracting more certain summaries for news articles. In *SIGIR*. 787–796.

Mark Wasson. 1998. Using leading text for news summaries: Evaluation results and implications for commercial summarization applications. In *ACL*. 1364–1368.

Su Yan and Xiaojun Wan. 2015. Deep dependency substructure-based learning for multidocument summarization. *ACM Trans. Inform. Syst.* 34, 1 (2015), 3.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir R. Radev. 2017. Graph-based neural multi-document summarization. In *CoNLL*. ACL, 452–462.

Wenpeng Yin and Yulong Pei. 2015. Optimizing sentence modeling and selection for document summarization. In *IJCAI*. 1383–1389.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-based convolutional neural network for modeling sentence pairs. *Trans. Association for Comput. Linguist.* 4, 1 (2016), 259–272.

Received October 2017; revised March 2018; accepted March 2018