From Neural Sentence Summarization to Headline Generation: A Coarse-to-Fine Approach

Jiwei Tan and **Xiaojun Wan** and **Jianguo Xiao**
Institute of Computer Science and Technology, Peking University
The MOE Key Laboratory of Computational Linguistics, Peking University
{tanjiwei, wanxiaojun, xiaojianguo}@pku.edu.cn

Abstract

Headline generation is a task of abstractive text summarization, and previously suffers from the immaturity of natural language generation techniques. Recent success of neural sentence summarization models shows the capacity of generating informative, fluent headlines conditioned on selected recapitulative sentences. In this paper, we investigate the extension of sentence summarization models to the document headline generation task. The challenge is that extending the sentence summarization model to consider more document information will mostly confuse the model and hurt the performance. In this paper, we propose a coarse-to-fine approach, which first identifies the important sentences of a document using document summarization techniques, and then exploits a multi-sentence summarization model with hierarchical attention to leverage the important sentences for headline generation. Experimental results on a large real dataset demonstrate the proposed approach significantly improves the performance of neural sentence summarization models on the headline generation task.

1 Introduction

Text summarization is a task of producing a condensed version of text while preserving its meaning. When the original text is a document and the target text is a sentence, the task is specialized to headline generation [Dorr *et al.*, 2003]. Headline generation is useful in several scenarios, like compressing text for mobile device users [Chen, 2010], generating content table [Erbs *et al.*, 2013], and also has the potential for more advanced artificial intelligence applications like machine writing. The task is challenging for its request of short, fluent and informative text generation.

Previous headline generation methods can be generally categorized to extractive methods and abstractive methods. Extractive methods treat sentences from the original document as the candidates, and exploit sentence compression techniques to produce the headline. Abstractive methods may treat phrases, concepts or events as candidates, and exploit sentence synthesis techniques to generate the headline. Fully abstractive methods even do not choose candidates from the original document, but generate a sentence from scratch using natural language generation techniques based on understanding the document. As headlines are highly condensed, extractive methods usually suffer from generating less informative headlines. Therefore abstractive methods are essentially more appropriate for the task. However, it is more difficult for abstractive methods to ensure the grammaticality of the generated headlines, due to the difficulty and immaturity of natural language generation techniques.

Recent success of neural sequence-to-sequence (seq2seq) models provides an effective way for text generation. Impressive progress has been achieved on tasks like machine translation, image captioning, and sentence summarization. Rush *et al.* [2015] explore training a sentence summarization model on headlines and lead (first) sentences of news articles, which shows the capacity of seq2seq models to generate informative, fluent headlines conditioned on recapitulative sentences. Rush *et al.* [2015] assume that the lead sentences of most news articles will be the summary of the important information, and eliminate those articles whose first sentences are not significantly overlapped with the headlines. Extensive later studies follow this assumption and concentrate on further improving the sentence summarization model.

In this paper, we investigate whether the neural sentence summarization model is applicable to the headline generation task. It turns out that lead sentences are not always sufficient for headline generation. Alfonseca *et al.* [2013] also reveal that the most important information is usually distributed across several sentences in an article. Therefore we hope to improve the sentence summarization model to leverage the content of the whole document. However, this extension is not straightforward, and runs into two problems. First is that tackling the document-level input sequence remains a major challenge for seq2seq models. Moreover, introducing more information will probably confuse the sentence summarization model and result in degraded performance. How to distinguish the important information of the input has not been considered in sentence summarization models.

In this paper we investigate the extension of neural sentence summarization model to the headline generation task. We propose a coarse-to-fine approach, which first identifies the most important sentences of the document, and then generates the headline according to the important sentences. We propose a multi-sentence summarization model with hierarchical attention, and show that a recurrent layer built on the representations of multiple summaries, has the ability to control the generation of a more appropriate headline. We conduct experiments on the New York Times news corpus. Different from sentence summarization works we do not filter the dataset by using the word overlap between the headlines and the lead sentences, which ensures the dataset to be more realistic. We compare with various baselines, and show that our approach significantly improves the lead-sentence based summarization models on the headline generation task.

We organize the paper as follows. Section 2 introduces related work. Section 3 describes preliminaries. In Section 4 we introduce our approach, and in Section 5 we show the experimental results. Finally Section 6 concludes this paper.

2 Related Work

Headline generation methods can be generally categorized to extractive methods and abstractive methods. Extractive methods usually explore compressing the document sentences to produce the headline. Dorr *et al.* [2003] propose the Hedge Trimmer algorithm to compress a sentence by making use of handcrafted linguistically-based rules. Alfonseca *et al.* [2013] introduce a multi-sentence compression model because they find that the important information is usually distributed across several sentences in an article. Recently Colmenares *et al.* [2015] propose a sequence-prediction technique which models headline generation as a discrete optimization task in a feature-rich space.

As for abstractive methods, Banko *et al.* [2000] apply count-based noisy-channel machine translation model for content selection and surface realization. Soricut and Marcu [2007] present a new paradigm based on direct transformation of relevant textual information into well-formed textual output. Woodsend *et al.* [2010] propose a quasi-synchronous grammar approach to produce legible headlines. Xu *et al.* [2010] utilize Wikipedia to extract features to select keywords, and then employ keyword clustering method to construct a headline. Sun *et al.* [2015] propose an event-driven model to extract structural events, and use a multi-sentence compression algorithm to fuse the events. In summary, most of these abstractive methods still extract words from the document, and synthesize them to generate the headline.

Recently, Rush *et al.* [2015] propose a sentence summarization model, which is a neural encoder-decoder framework and trained on large amount of news headlines and selected recapitulative sentences. Extensive later works follow their strategy and investigate to improve the sentence summarization model. Chopra *et al.* [2016] employ RNN as the encoder, and incorporate the position information of words, which shows significant improvement. Nallapati *et al.* [2016] introduce several effective mechanisms to decrease the vocabulary size and model key words with a feature-rich encoder. Different from these efforts, we study in another view of adapting the sentence summarization model to the headline generation task.

There are some recent studies [Shen *et al.*, 2016; Yu *et al.*, 2016; Takase *et al.*, 2016] titled as neural headline generation. However, these works still only use the lead sentences as the input, and therefore are different from our work.

3 Preliminaries

In this section we introduce the encoder-decoder framework of seq2seq models. The encoder is used for encoding the input text into a representation vector and the decoder is used to generate the output according to the input representation.

3.1 Sequence-to-sequence Framework

Denote the input text $\boldsymbol{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_M\}$ as a sequence of M words, and each word $\mathbf{x}_t$ is a one-hot vector of size $|V|$ from the word vocabulary V. Seq2seq model takes $\boldsymbol{X}$ as input, and generates the output $\boldsymbol{Y}$ from $\boldsymbol{X}$: $\boldsymbol{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$. The goal is to find $\hat{\boldsymbol{Y}}$ so as to maximize the conditional probability of $\boldsymbol{Y}$ given $\boldsymbol{X}$, as: $\hat{\boldsymbol{Y}} = \arg\max_{\boldsymbol{Y}} P(\boldsymbol{Y}|\boldsymbol{X};\theta)$ where θ indicates the parameters for the model to learn.

Seq2seq model processes the input and produces the output sequentially. The model processes input sequence $\boldsymbol{X}$ into a low-dimensional vector representation $\mathbf{c}$ with an encoder. Then the model generates output $\boldsymbol{Y}$ word by word with a decoder, with each generated word $\mathbf{y}_t$ conditioned on input representation $\mathbf{c}$ and previously generated words $\{\mathbf{y}_1, \ldots, \mathbf{y}_{t-1}\}$, as:

$$P(\boldsymbol{Y}|\boldsymbol{X};\theta) = \prod_{t=1}^{N} P\left(\mathbf{y}_t | \{\mathbf{y}_1, \ldots, \mathbf{y}_{t-1}\}, \mathbf{c};\theta\right) \quad (1)$$

3.2 Encoder

In the seq2seq model the encoder is used to process the input sequence into its vector representation $\mathbf{c}$. Recurrent Neural Network (RNN) [Rumelhart *et al.*, 1988] is natural to be the encoder for text inputs due to its ability to deal with variable length of input. The idea of RNN is to perform the same task for every element of a sequence, with the output being depended on the previous computations. Specifically, as the encoder, RNN gets hidden state $\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1})$ for each input word $\mathbf{x}_t$ in the input sequence $\boldsymbol{X}$, where f indicates the function of RNN unit. Then $\mathbf{c} = g\left(\{\mathbf{h}_1, \ldots, \mathbf{h}_M\}\right)$, where g is a function to calculate $\mathbf{c}$ from hidden states. A typical instance of g is $g\left(\{\mathbf{h}_1, \ldots, \mathbf{h}_M\}\right) = \mathbf{h}_M$. In this paper we use Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] as the encoder and also the decoder. We use the LSTM structure defined in [Graves, 2013].

3.3 Decoder with Attention Mechanism

The decoder is used to generate the output sequence given the input representation $\mathbf{c}$. The decoder generates one word every step based on the input representation and previously generated words. RNN is also mostly used as the decoder. The way to generate word $\mathbf{y}_t$ is:

$$\mathbf{y}_t = \arg\max_{\mathbf{y}'} P\left(\mathbf{y}' | \{\mathbf{y}_1, \ldots, \mathbf{y}_{t-1}\}, \mathbf{c};\theta\right) \quad (2)$$

In primitive decoder models, $\mathbf{c}$ is same for generating all the output words, which requires $\mathbf{c}$ to be a good representation for the whole input sequence. Attention mechanism

[Bahdanau *et al.*, 2014] is introduced to allow the decoder to pay different attention to different parts of input when generating different words. Many successful applications show the effectiveness of attention mechanism. Attention mechanism sets different $\mathbf{c}_t$ when generating different word $\mathbf{y}_t$, as $\mathbf{c}_t = \sum_{i=1}^{M} \alpha_i^t \mathbf{h}_i$. α_i^t indicates how much the i-th word $\mathbf{x}_i$ from the source input contributes to generating the t-th word, and is usually computed by:

$$\alpha_i^t = \frac{\exp\left(\mathbf{h}_i \cdot \mathbf{h}_{y_t}\right)}{\sum_{j=1}^{M} \exp\left(\mathbf{h}_j \cdot \mathbf{h}_{y_t}\right)} \tag{3}$$

where $\mathbf{h}_{y_t}$ represents the hidden state of the decoder when generating word $\mathbf{y}_t$.

4 Our Approach

4.1 Overview

In this study we aim to investigate neural generation models on the headline generation task. A straightforward strategy is to view headline generation as a document summarization task, and treat the document as the input to seq2seq models. Unfortunately, a document is too long to be the input to a sentence summarization model. Too long input sequence will make the model too hard to train. A common solution to document-level input in seq2seq models is utilizing hierarchical framework like [Li *et al.*, 2015]. However, generating a headline conditioned on the whole document will mostly confuse the model and get poor results.

Inspired by the investigation of various architectures, in this paper, we propose a coarse-to-fine framework to tackle the challenge. The approach first identifies multiple important sentences of the document using well-studied document summarization techniques, and then exploits a multi-sentence summarization model to further abstract these document summaries to a headline.

4.2 Document Summarization

We utilize extractive summarization methods to select the informative parts of a document from the less informative ones. We use seven diverse sentence-based extractive summarization methods to generate seven summaries of a document respectively. The use of multiple summaries better covers the important information of a document, so that our approach will not heavily rely on the performance of one specific summarization method. The document summaries are limited to maximum length of 50 words, with typically one or two sentences. The document summarization methods are:

- **Lead** is the baseline which selects the lead several sentences. The baseline has shown extremely strong performance in the single news document summarization task.
- **Luhn** [Luhn, 1958] is an early method where statistical information derived from word frequency and distribution is used to compute a relative measure of significance, first for individual words and then for sentences.
- **LSA** [Steinberger and Jezek, 2004] is a method that applies Singular Value Decomposition (SVD) to find principal and mutually orthogonal dimensions of sentence vectors, and then picks a representative sentence from each of the dimensions.
- **LexRank** [Erkan and Radev, 2004] is a graph-based method inspired by the PageRank algorithm [Page *et al.*, 1999], which computes sentence importance based on the concept of eigenvector centrality in a graph representation of sentences.
- **TextRank** [Mihalcea and Tarau, 2004] is also a graph-based method inspired by the PageRank algorithm. The difference is that LexRank and TextRank use different methods to calculate the similarity of two sentences.
- **SumBasic** [Nenkova and Vanderwende, 2005] is a simple and effective method and often used as a baseline in the literature. Each sentence is assigned a score reflecting how many high-frequency words it contains.
- **KL-Sum** [Haghighi and Vanderwende, 2009] aims to find a set of summary sentences which closely match the document set unigram distribution.

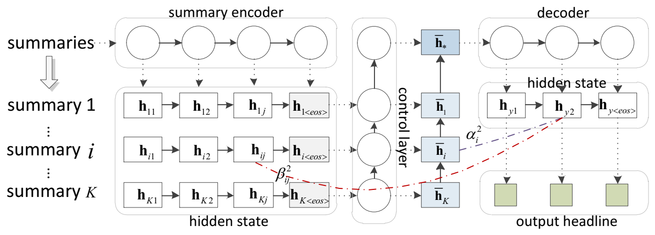

Figure 1: The framework of summary-to-headline generation.

4.3 Multi-sentence Summarization

Provided with multiple summaries, our multi-sentence summarization model aims to generate the headline according to these summaries. We employ a seq2seq framework as shown in Figure 1. A summary encoder is used to encode each summary to a summary representation. Afterwards, a control layer is added on the top of summary representations, to merge these summaries to the representation of the important information of the whole document, which is further provided to a decoder to generate the headline accordingly. We introduce a hierarchical attention strategy. Summary-level attention is introduced as the control function to distinguish the contribution of different summaries. The word-level attention is similar to the effective attention mechanism used in sentence summarization models, which enables the decoder to align to important original words during the generation.

Summary Encoder

The summary encoder is similar to that used in sentence summarization models. The summary encoder is used to accept each document summary (e.g. $summary\ i$) of a document as input, and get its hidden representation sequence (e.g. $\{\mathbf{h}_{i1}, \mathbf{h}_{i2}, \ldots, \mathbf{h}_{i<eos>}\}$) as output. Each summary is appended with an "<eos>" token to indicate the end of the summary. The last hidden state after the summary encoder receives "<eos>" (e.g. $\mathbf{h}_{i<eos>}$) is treated as the representation of the summary.

Control Layer

Provided with the representations of multiple summaries, we hope to generate the document headline according to the summary representations. We introduce another LSTM layer on top of the summary representations as the control layer, to identify the important information and control the generation of headline from multiple summaries. The control layer receives all the summary representations $\{\mathbf{h}_{i<eos>}\}$ and produces $\bar{\mathbf{h}}_*$, which is further provided to the decoder to generate the headline accordingly. The benefit of using LSTM as the control layer is that it allows to keep the mapped representation $\bar{\mathbf{h}}_i$ for every summary representation $\mathbf{h}_{i<eos>}$, and therefore it enables a hierarchical attention mechanism in the decoder, which is introduced as follows.

Decoder

The decoder accepts the merged representation $\bar{\mathbf{h}}_*$ as input, but is able to access all the hidden states of words in all summaries. When generating a word at step t, the decoder takes all the words of document summaries to form the context vector $\mathbf{c}_t$, by assigning summary-level attention and word-level attention, as:

$$\mathbf{c}_t = f\left(\{\mathbf{h}_{ij}\}\right) = \sum_{i=1}^{K} \sum_{j} \alpha_i^t \beta_{ij}^t \mathbf{h}_{ij} \tag{4}$$

where i indicates the i-th document summary and j indicates the j-th word of document summary i. α is the summary-level attention weight and β is the word-level attention weight. The summary-level attention controls which summary is important, as learned by the control layer. The summary-level attention weight α_i^t for summary i when generating word $\mathbf{y}_t$ is calculated by:

$$\alpha_i^t = \frac{\exp\left(\bar{\mathbf{h}}_i \cdot \mathbf{h}_{y_t}\right)}{\sum_{s=1}^{K} \exp\left(\bar{\mathbf{h}}_s \cdot \mathbf{h}_{y_t}\right)} \tag{5}$$

where K is the number of document summaries. When generating word $\mathbf{y}_t$, the word-level attention weight β_{ij}^t for input word j of summary i is calculated by:

$$\beta_{ij}^t = \frac{\exp\left(\mathbf{h}_{ij} \cdot \mathbf{h}_{y_t}\right)}{\sum_{w} \exp\left(\mathbf{h}_{iw} \cdot \mathbf{h}_{y_t}\right)} \tag{6}$$

The attention weight determines how much attention should be paid to that input word. α determines how much attention should be paid to a document summary, and β determines how much attention should be distributed over the words in a summary. The weights are used to compute a weighted average of the hidden states $\{\mathbf{h}_{ij}\}$ of all the input words. The weighted average, referred to as the context $\mathbf{c}_t$, is then used at the step of decoding $\mathbf{y}_t$, so that the headline generation is based on the important information of the whole document.

4.4 Model Learning

The loss function $\mathcal{L}$ is the negative log likelihood of generating headlines over the training set $\mathcal{D}$:

$$\mathcal{L} = \sum_{(\boldsymbol{X},\boldsymbol{Y})\in\mathcal{D}} -\log P(\boldsymbol{Y}|\boldsymbol{X};\theta) \tag{7}$$

where $P(\boldsymbol{Y}|\boldsymbol{X};\theta)$ is defined by Eq. (1). $\mathcal{D}$ is the set of document summaries and corresponding headlines. We use the RMSProp adaptive gradient method to optimize the model parameters θ.

5 Experiments

5.1 Dataset

Previous sentence summarization models are evaluated on news articles from the English Gigaword corpus[1], and only the lead sentences which have significant overlap with the headlines are selected. However, lead sentences are not guaranteed to be the adequate source for the headline generation task. For instance, the articles from the New York Times (NYT) corpus, which take up a large proportion of the Gigaword corpus, follow less the journalistic convention to put most important information in the first sentence. Statistics of the 1.4 million NYT articles show that with the average headline length of 7.9 words, there are on average only 2.9 overlapping words between the headline and the first sentence. While for the filtered Gigaword dataset used in sentence summarization models, the average headline length is 8.3 words, and there are on average 4.6 overlapping words. The significant difference of word overlap indicates that for the headline generation task, more document content should be considered. Experimental results also support the observation. In this paper, we conduct experiments on the 1.4 million NYT articles. We leave out the articles whose headlines have less than 3 words or more than 15 words, and whose texts have less than 20 words or more that 2000 words. There are 1.03 million articles left. We randomly sample 2000 articles to form the test set and the other articles are used as the training data.

5.2 Implementation

For the document-level summarization we use a Python toolkit *sumy*[2] which has the implementation of the introduced summarization methods. The summaries and headlines are then lowercased, and tokenized to separate punctuations from words. All digit characters are replaced with the "#" symbol similar to [Rush *et al.*, 2015]. We keep the 40,000 most frequently occurring words and other words are replaced with the "<unk>" symbol. The processed summaries-headline pairs then serve as the input for the neural generation model.

We implement the multi-sentence summarization model with theano[3]. For the summary encoder we use three hidden layers of LSTM, and for the control layer we use one layer of LSTM, and each layer has 512 hidden units. We use pre-trained GloVe vectors[4] for the initialization of word vectors and the word vectors will be further trained in the model. The dimension of word vectors is 100. Other parameters are randomly initialized in the range [-0.1,0.1]. The learning rate of RMSProp is 0.01 and the decay and momentum are both 0.9. We use a batch size of 64 samples, and process

[1]https://catalog.ldc.upenn.edu/LDC2012T21
[2]https://github.com/miso-belica/sumy
[3]https://github.com/Theano/Theano
[4]http://nlp.stanford.edu/projects/glove

30,016 samples an epoch. We adopt an effective simple attention mechanism [Lopyrev, 2015] which splits the hidden vector into two sets: the first 472 dimensions for decoding words and the last 40 dimensions for computing the attention weight. We adopt an early stopping strategy, which stops training if the performance no longer improves on held-out training data in 20 epochs. Convergence is reached at about 400 epochs. We run the model on a GTX-1080 GPU card, and it takes about one day for every 100 epochs.

5.3 Evaluation

To study the extension of sentence summarization models to the headline generation task, we compare our approach with various variants of seq2seq models. The variants are implemented by replacing our approach **summ-hieratt** with different inputs and model frameworks, as:

- **lead-flat-att** is a typical RNN-based attentional sentence summarization model which generates the headline according to the lead sentence only.
- **doc-hierenc** is to take the whole document as input using a hierarchical encoder structure, which first encodes sentences to sentence representations and then encodes sentence representations to a document representation.
- **doc-hierenc-att** is to add sentence-level attention to **doc-hierenc**, and it can be viewed as the adaption of the hierarchical attention model in [Li *et al.*, 2015].
- **doc-hierenc-hieratt** is to add both sentence-level and word-level attention to **doc-hierenc**. It is similar to our model except it uses the whole document as the input.
- **summ-flat-att** is a baseline to concatenate the multiple summaries of a document as the input and apply the sentence summarization model.

We also experiment to verify applying existing methods on the headline generation task, including lead-sentence based summarization model and neural abstractive document summarization model. Therefore we have three more baselines for comparison, namely:

- **PREFIX** is to use the first sentence as the headline.
- **ABS** is the attention bag-of-words encoder based sentence summarization model in [Rush *et al.*, 2015].
- **doc-Distraction** is a state-of-the-art neural abstractive document summarization model proposed in [Chen *et al.*, 2016].

We adopt the widely used ROUGE [Lin, 2004] toolkit for evaluation, and report recall results on three metrics of Rouge-1, Rouge-2 and Rouge-L. Results are shown in Table 1.

As shown in Table 1, on the headline generation task, extending sentence summarization models to consider more document information is not straightforward. Introducing more document content usually confuses the model, and the results decline compared with using lead sentences only. The results of *doc-hierenc* show that extending sentence summarization to document summarization for the headline generation task will get very poor results, even with attention mechanism (*doc-hierenc-att*). Results of *doc-hierenc-hieratt* show that using word-level attention is effective. However, even the state-of-the-art document summarization model (*doc-Distraction*) fails to generate headlines conditioned the whole document as the input. These results reflect that it is hard for seq2seq models to discover important information from too much input content of a document. Our coarse-to-fine approach provides an effective solution to the problem. Document summarization techniques first help to identify the important information of a document at coarse level. Then, with hierarchical attention, the multi-sentence summarization model is able to generate appropriate headlines according to the document summaries. The significant better results of *summ-hieratt* over *summ-flat-att* demonstrate the effectiveness of the hierarchical attention framework.

To further verify the effectiveness of the proposed approach, we also test the model on the DUC-2004 Task-1. The dataset of the task consists of 500 news articles from the New York Times and Associated Press Wire services, and each article is paired with 4 different human-generated reference summaries. We train our model on the same Gigaword dataset used in [Rush *et al.*, 2015; Chopra *et al.*, 2016]. We compare with the performance of sentence summarization models. *RAS-Elman* is the best model in [Chopra *et al.*, 2016], with an elaborately tuned sophisticated RNN encoder incorporating word position information. Results are shown in Table 2. The results indicate that our approach is able to leverage the extra important information of the documents. Without using any other effective techniques, our multi-sentence summarization model is able to significantly improve the sentence summarization baseline model *lead-flat-att*, and achieve comparable results to recent sophisticated sentence summarization models.

Method	Rouge-1	Rouge-2	Rouge-L
PREFIX	17.85	4.94	16.28
ABS	25.82	7.03	23.07
lead-flat-att	26.57	6.84	23.18
doc-hierenc	15.91	2.68	14.19
doc-hierenc-att	17.26	3.76	15.35
doc-hierenc-hieratt	18.57	4.25	16.15
doc-Distraction	14.53	4.49	13.83
summ-flat-att	22.78	4.91	19.48
summ-hieratt	**29.60**	**8.17**	**26.05**

Table 1: Results of Rouge recall at 10 words on the NYT test data. Two-tailed t-tests demonstrate the improvements of *summ-hieratt* to other approaches are all statistically significant ($p < 0.01$).

Method	Rouge-1	Rouge-2	Rouge-L
PREFIX	22.43	6.49	19.65
ABS	26.55	7.06	22.05
RAS-Elman	**28.97**	8.26	24.06
lead-flat-att	26.73	7.50	22.63
summ-hieratt	28.68	**8.27**	**24.39**

Table 2: Results of Rouge recall at 75 bytes on the DUC-2004 Task-1 test data. Two-tailed t-tests demonstrate the improvement of *summ-hieratt* to *lead-flat-att* is statistically significant ($p < 0.01$).

G: # plays define giants ' weird victory over eagles
L: rams ' a to score in the of title game
O: giants defeat eagles , ##-## in victory over eagles

S1: football games are won in focal moments , sequences almost stolen from the game 's routine _ the stab of hand on ball , a <unk> to block a kick , a deflected pass that leads to a <unk> and touchdown . [Lead; LexRank]
S2: about four minutes later , peter 's colleague at tackle , keith hamilton , knocked the ball from the hands of eagles running back duce staley on the philadelphia #-yard line . [Luhn]
S3: strahan , who acknowledged he would have considered the game over had philadelphia converted the field goal peter blocked , and who called staley 's fumble ' ' a miracle from god , ' ' said he watched the deflection of pederson 's pass spin through the air in wonder . [LSA]
S4: the last act was giants defensive end michael strahan _ head back , legs churning to carry his ### pounds and a pound of pigskin _ running ## yards with an overtime interception return for a touchdown , giving the giants a ##-## victory over the eagles and salvaging a season at [TextRank; KL-Sum]
S5: the giants had three such moments sunday against the eagles . [SumBasic]

Figure 2: An example of headline generation from NYT test data. **G** is the true headline, **L** is the output of *lead-flat-att*, **O** is the output of our approach *summ-hieratt*. **S1** to **S5** are the document-level summaries, and each summarization method is indicated in "[]" at the end.

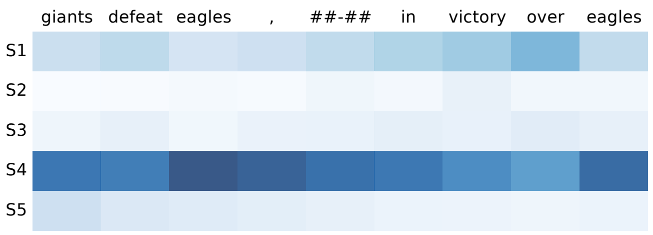

Figure 3: Heat map of the distribution of summary-level attention weights when generating every word for the example in Figure 2. Darker color indicates higher weight.

5.4 Case Study

An example from the NYT test data is shown in Figure 2. As we can see from summary **S1**, even the names of the two teams do not occur in the lead sentence. The sentence summarization baseline *lead-flat-att* results in a poor output because the lead sentence is not the good source for headline generation. The most informative summary is **S4**, which is extracted by both the *TextRank* and *KL-Sum* methods. In Figure 3 we show the heat map of summary-level attention weights when our model generates the example. As seen from Figure 3, the most informative summary **S4** is assigned with the highest attention. From the input of multiple document summaries, the proposed multi-sentence summarization model is able to identify the more important information and generate an appropriate headline to describe the document.

We show two more examples to compare the proposed approach with the lead-sentence based baseline model *lead-flat-att* in Figure 4. Due to limited space we only show the summary generated by the *Lead* method, which is indicated as **S1**. The first example shows that, when the lead sentence covers the main information of the document, both *lead-flat-att* and *summ-hieratt* are able to generate good headlines. However, *summ-hieratt* is able to generate the more accurate word "surge" by utilizing more information of the document,

G: japanese bonds rise after u.s. treasuries surge overnight
L: japanese bonds rise as u.s. treasuries rebound
O: japanese bonds rise as u.s. treasuries surge
S1: japanese bonds rose for a second day after u.s. treasury bonds , the bellwether for global debt markets , posted their biggest gain in seven weeks overnight .

G: trained eyes see new planets
L: evidence of planets may be matter on the holes
O: two worlds are circling the planet of a new light
S1: a little more of the universe has been pried out of the shadows . two groups of astronomers have taken the first pictures of what they say - - and other astronomers agree - - are most likely planets going around other stars .

Figure 4: Headlines generated by *lead-flat-att* and *summ-hieratt* for two examples from the NYT test data. **S1** indicates the summary extracted by the *Lead* method.

while *lead-flat-att* can only generate the word "rebound" because there is no "surge" in the lead sentence. The second is an example when both *lead-flat-att* and *summ-hieratt* get poor results. As we see the true headline is not the common style of news headlines that describes the story briefly. The human written headline is highly abstracted, with the author's understanding for the whole article. This is one of the difficulties for practical headline generation models. In this case, due to the inadequate information of the lead sentence, considering the important content of the whole document for headline generation is necessary.

6 Conclusion and Future Work

In this paper we investigate the application of neural sequence-to-sequence models to the headline generation task. It turns out that the lead sentences are not always sufficient for generating document headlines, and unfortunately, introducing more document content will usually confuse the model. To tackle this challenge, we propose a coarse-to-fine approach, which first leverages the well-studied document summarization techniques, and then improves sentence summarization model to generate the headline from multiple summaries using a hierarchical attention model. Experimental results demonstrate the proposed approach significantly improves the performance of neural sentence summarization models on the headline generation task. Our work is an investigation towards neural abstractive headline generation, while fully end-to-end method based on understanding the whole document remains a challenge for future work.

Acknowledgments

This work was supported by 863 Program of China (2015AA 015403), NSFC (61331011), and Key Laboratory of Science, Technology and Standard in Press Industry (Key Laboratory of Intelligent Press Media Technology). Xiaojun Wan is the corresponding author.

References

[Alfonseca *et al.*, 2013] Enrique Alfonseca, Daniele Pighin, and Guillermo Garrido. HEADY: news headline abstraction through event pattern clustering. In *ACL 2013*, pages 1243–1253, 2013.

[Bahdanau *et al.*, 2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[Banko *et al.*, 2000] Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. Headline generation based on statistical translation. In *ACL*, pages 318–325, 2000.

[Chen *et al.*, 2016] Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, and Hui Jiang. Distraction-based neural networks for document summarization. In *IJCAI-16*, pages 2754–2760, 2016.

[Chen, 2010] Chih-Ming Chen. Intelligent location-based mobile news service system with automatic news summarization. *Expert Syst. Appl.*, 37(9):6651–6662, 2010.

[Chopra *et al.*, 2016] Sumit Chopra, Michael Auli, and Alexander M Rush. Abstractive sentence summarization with attentive recurrent neural networks. In *NAACL*, 2016.

[Colmenares *et al.*, 2015] Carlos A. Colmenares, Marina Litvak, Amin Mantrach, and Fabrizio Silvestri. HEADS: headline generation as sequence prediction using an abstract feature-rich space. In *NAACL*, 2015.

[Dorr *et al.*, 2003] Bonnie Dorr, David Zajic, and Richard Schwartz. Hedge trimmer: A parse-and-trim approach to headline generation. In *HLT-NAACL 03 on Text summarization workshop*, pages 1–8, 2003.

[Erbs *et al.*, 2013] Nicolai Erbs, Iryna Gurevych, and Torsten Zesch. Hierarchy identification for automatically generating table-of-contents. In *RANLP 2013*, pages 252–260, 2013.

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

[Graves, 2013] Alex Graves. Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*, 2013.

[Haghighi and Vanderwende, 2009] Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *NAACL*, 2009.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[Li *et al.*, 2015] Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. A hierarchical neural autoencoder for paragraphs and documents. In *ACL 2015*, pages 1106–1115, 2015.

[Lin, 2004] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out ACL-04 workshop*, volume 8. Barcelona, Spain, 2004.

[Lopyrev, 2015] Konstantin Lopyrev. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712*, 2015.

[Luhn, 1958] Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, 2(2):159–165, 1958.

[Mihalcea and Tarau, 2004] Rada Mihalcea and Paul Tarau. Textrank: Bringing order into texts. In *EMNLP*, 2004.

[Nallapati *et al.*, 2016] Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *CoNLL 2016*, pages 280–290, 2016.

[Nenkova and Vanderwende, 2005] Ani Nenkova and Lucy Vanderwende. The impact of frequency on summarization. *Microsoft Research, Redmond, Washington, Tech. Rep. MSR-TR-2005-101*, 2005.

[Page *et al.*, 1999] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web, 1999.

[Rumelhart *et al.*, 1988] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5:3, 1988.

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

[Shen *et al.*, 2016] Shiqi Shen, Zhiyuan Liu, Maosong Sun, et al. Neural headline generation with minimum risk training. *arXiv preprint arXiv:1604.01904*, 2016.

[Soricut and Marcu, 2007] Radu Soricut and Daniel Marcu. Abstractive headline generation using widl-expressions. *Inf. Process. Manage.*, 43(6):1536–1548, 2007.

[Steinberger and Jezek, 2004] Josef Steinberger and Karel Jezek. Text summarization and singular value decomposition. In *ADVIS 2004*, pages 245–254, 2004.

[Sun *et al.*, 2015] Rui Sun, Yue Zhang, Meishan Zhang, and Dong-Hong Ji. Event-driven headline generation. In *ACL 2015*, pages 462–472, 2015.

[Takase *et al.*, 2016] Sho Takase, Jun Suzuki, Naoaki Okazaki, Tsutomu Hirao, and Masaaki Nagata. Neural headline generation on abstract meaning representation. In *EMNLP*, pages 1054–1059, 2016.

[Woodsend *et al.*, 2010] Kristian Woodsend, Yansong Feng, and Mirella Lapata. Title generation with quasi-synchronous grammar. In *EMNLP*, pages 513–523, 2010.

[Xu *et al.*, 2010] Songhua Xu, Shaohui Yang, and Francis Chi-Moon Lau. Keyword extraction and headline generation using novel word features. In *AAAI*, 2010.

[Yu *et al.*, 2016] Lang-Chi Yu, Hung-yi Lee, and Lin-Shan Lee. Abstractive headline generation for spoken content by attentive recurrent neural networks with ASR error modeling. In *SLT*, pages 151–157, 2016.