Experiment on Using Topic Sentence for Neural News Headline Generation

Jan Wira Gotama Putra[†] Hayato Kobayashi[‡§] Nobuyuki Shimizu[‡]
[†]Tokyo Institute of Technology [‡]Yahoo Japan Corporation [§]RIKEN AIP

gotama.w.aa@m.titech.ac.jp, {hakobaya, nobushim}@yahoo-corp.jp

1 Introduction

News headline generation is one variant of summarization tasks, which is introduced in DUC-2003 and DUC-2004 (Task 1) [15]. In this study, we are interested in news headline generation using abstractive approach. The generated headline possibly includes words not seen in the source document [11]. Headline generation can be cast as a task of mapping an input sequence of words into a target sequence of words using an encoder-decoder model (Section 2) [2].

Past studies on neural headline generation mostly used the first sentence-headline pair as an input-output for the encoder-decoder model [2, 5, 9, 11]. They emphasized on how to incorporate linguistics information or architectural strategy of the encoder-decoder model. However, Tan et al. in [13] questioned the effectiveness of using the first sentence as the input because information in the text is distributed across sentences [1]. They used the full-document or sentences extracted from statistical ranking techniques as the input for the encoder-decoder instead and showed improvement in performance. This study aims to investigate the use of the topic sentence (Section 3) as another type of input.

There are two key questions addressed in this work: (1) whether the topic sentence is more useful than the first sentence for headline generation or not, and (2) whether the topic sentence is helpful in addition to the first sentence for headline generation or not.

2 Encoder-Decoder Model

Encoder-decoder model maps a sequence of input into a sequence of output [4, 12]. Let us denote $\mathbf{x} = (x_1, \ldots, x_N)$ as a sequence of N input words. A word x_t is typically represented as a one-hot vector encoding or as a corresponding embedding vector $\mathbf{e}_t$. The headline generation task objective is to find the best sequence of M words $\mathbf{y} = (y_1, \ldots, y_M)$, $M < N$; for given input sequence $\mathbf{x}$. It means modeling the conditional probability $p(\mathbf{y} \mid \mathbf{x}, \theta)$ of input-output pair, where θ acts as learning parameters [11]. The conditional probability can be factored into Equation 1 as follows:

$$p(\mathbf{y} \mid \mathbf{x}, \theta) = \prod_{t=1}^{M} p(y_t \mid \{y_1, \ldots, y_{t-1}\}, \mathbf{x}, \theta), \tag{1}$$

where M is the length of the output. This formulation can be naturally cast as an encoder-decoder model.

An encoder represents an input sequence of words into a single vector representation $\mathbf{c}$. Past studies proposed to use recurrent neural network (RNN) encoder as shown in Equation 2 as follows [2]:

$$\begin{aligned} \mathbf{h}_t &= f(\mathbf{h}_{t-1}, \mathbf{e}_t) \\ &= f(\mathbf{U}_\mathrm{h}\mathbf{h}_{t-1} + \mathbf{U}_\mathrm{e}\mathbf{e}_t) \end{aligned} \tag{2}$$

where f is a non-linear activation function; $\mathbf{U}_\mathrm{h}$ and $\mathbf{U}_\mathrm{e}$ denote weight matrices. We also may use Bidirectional RNN to substitute RNN. The input representation $\mathbf{c}$ is obtained as the weighted sum of the hidden states $q(\{\mathbf{h}_1, \cdots, \mathbf{h}_N\})$, where q is a non-linear activation function. We also can simply use the last hidden state $\mathbf{h}_N$ as $\mathbf{c}$.

Decoder decodes the input representation $\mathbf{c}$ to generate the sequence of output words $\mathbf{y}$ (generated headline). RNN is also used for the decoder where a hidden state $\mathbf{h}'_t$ represents conditional probability distribution of words to select the current output y_t, typically represented in a vector form $\mathbf{e}'_t$. $\mathbf{h}'_t$ is computed in Equation 3 as follows:

$$\begin{aligned} \mathbf{h}'_t &= f'(\mathbf{h}'_{t-1}, \mathbf{e}'_{t-1}, \mathbf{c}) \\ &= f'(\mathbf{U}'_\mathrm{h}\mathbf{h}'_{t-1} + \mathbf{U}'_\mathrm{e}\mathbf{e}'_{t-1} + \mathbf{U}'_\mathrm{c}\mathbf{c}) \end{aligned} \tag{3}$$

where f' is a non-linear activation function; $\mathbf{U}'_\mathrm{h}$, $\mathbf{U}'_\mathrm{e}$, $\mathbf{U}'_\mathrm{c}$ represent weight matrices. In practice, we can replace the activation function using variant of neural network choices for example, using long short-term memory network. Attention mechanism also can be employed to denote which word the decoder should focus on a particular decoding step [3]. The encoder-decoder model is trained to tune the learning parameters θ governing the conditional probability by minimizing the negative log likelihood of the conditional probability, over a set of training data. It is typically done using stochastic gradient descent method. Once the model is trained, the encoder-decoder generates a headline $\mathbf{y}^*$ for a new input sequence $\mathbf{x}$ maximizing the conditional probability using beam search algorithm.

3 Related Work

Past studies on neural headline generation mostly used the first sentence-headline pair as an input-output for the encoder-decoder model [2, 5, 9, 11]. They gave more emphasize on how to incorporate linguistics information or architectural strategy of the encoder-decoder model. For example, Rush et al. [11] focused on how the encoder-decoder model can take into account word context. In addition to the word context, Chopra et al. [5] tried to incorporate information of the position of input words as well. Nallapati et al. [9] tried to incorporate linguistic and structural information when representing the input sequence. On the other hand, Ayana et al. [2] worked on comparing network architectures and training strategies.

As explained, previously mentioned studies used the first sentence of news as the input sequence. However, Tan et al. [13] questioned the effectiveness of using the first sentence as the input because information in the text is distributed across sentences [1]. Tan et al. [13] used the full-document or sentences extracted from statistical ranking techniques (e.g. LexRank) for the encoder-decoder. Though, they found that using a long input sequence (the full-document) possibly degrades the performance of the encoder-decoder.

We consider using the topic sentence as a middle-ground between the presented problems. Topic sentence is defined as the key information of news [14] as follows:

Topic sentence contains the core elements ⟨*subject*, *verb*, *object*⟩ and at least one subordinate element ***time*** or ***location***.

The core elements of functional information of a sentence are ⟨*subject*, *verb* (*predicate*), *object*⟩ triples [6]. In addition to these core elements, *time* and *location* are also important because they provide an additional factual information in news [6]. We hypothesize that incorporating topic sentence is likely to provide a better generalization of the encoder-decoder model than only using the first sentence. The generalization means allowing the model to predict the headline of unseen news. In contrast with statistical ranking techniques, the topic sentence considers 5W1H[1] information of news (indirectly).

[1]what, who, where, whom, when, how

4 Experimental Setting

4.1 Dataset

We use the annotated Gigaword dataset [10] (around 10 million documents) for our experiment. The annotation is only used for tokenization and sentence splitting during preprocessing step. Preprocessing also includes replacing digits with "#", and replacing low-frequency words with "⟨unk⟩", following the setting by Rush et al. [11][2]. First sentence, topic sentence, and original (reference) headline are extracted from each document.

We extract the earliest sentence containing ⟨*subject*, *verb*, *object*⟩ and at least one subordinate element *time* or *location* as the topic sentence. It follows the rationale of inverted pyramid structure of news that earlier sentences contain more general information than the later sentences. We analyze sentences using dependency parser and named entity tagger in spaCy[3] for a realistic setting. DATE and TIME named entity tags are used for recognizing *time* information, and GPE (i.e., countries, cities, states) and LOC (non-GPE locations) named entity tags are used for *location*[4]. We only extract one topic sentence to keep the length of input as minimum as possible to avoid vanishing gradient problem. In case there is no sentence satisfying the requirements for topic sentence, we use the first sentence as the topic sentence. When the topic sentence is the same as the first sentence, we only use the first sentence when feeding both first and topic sentences.

The dataset is split into training, validation, and testing data using the documents split provided by Rush et al. [11]. In the original script, the document whose headline consists of more than three and fewer than 50 words is included. The first sentence also should contain 10–100 words and at least one word in common with the corresponding headline. We add an additional filter that the topic sentence in a document should follow the same rule as the first sentence.

Data	# docs	not found	found-1	found-2-*
Train	2,755,324	5.54%	73.43%	21.06%
Valid	139,284	5.69%	72.76%	21.58%
Test	134,432	5.90%	72.91%	21.19%

Table 1: Filtered gigaword dataset.

The remaining documents after the filtering can be seen in Table 1. "# docs" column denotes the number of documents for the corresponding set. "not found" column denotes the perucentage of cases when there is no sentence satisfying the topic sentence extraction

[2]`https://github.com/facebookarchive/NAMAS`

[3]`https://spacy.io/`

[4]`https://spacy.io/usage/linguistic-features#section-named-entities`

rule. "found-1" denotes the percentage of cases when the first sentence satisfies the topic sentence requirement. "found-2-*" denotes the percentage of cases when the topic sentence is not the first sentence of the document (e.g., 2nd, 3rd, or 4th sentence). Headline contains eight tokens on average. While most of the topic sentences are the first sentence, the rest 21% topic can be meaningful for analysis compared to using only the first sentence directly as the input. In most cases, there is a sentence satisfying the topic sentence elements. It proves that using the topic sentence for headline generation is possible in a real-world setting. We use ROUGE (ROUGE-1, ROUGE-2, and ROUGE-L) to measure the performance of models [8].

4.2 Architectural Choice

We train the encoder-decoder model using three variants of input: (1) **first sentence**, (2) **topic sentence**, and (3) **both first and topic sentences** in which each is aligned with the headline. We use the default encoder-decoder implementation in OpenNMT using its default parameter setting [7][5]. This architecture is regarded as the standard sequence to sequence architecture. The reason for using this architecture is to investigate the consequence of using different input types, rather than architectural choice in neural headline generation task. When we feed both first and topic sentences, we place the first sentence earlier and separate them by dot ("."). To avoid the bias of parameter initialization, we train five models for each input-output pair and present the average performance in Section 5.

5 Experimental Result and Discussion

This experiment aims to confirm which input type is the best for the neural headline generation. In this section, we describe the result of evaluation of our models on the Gigaword test set (134K). We feed each model using topic, first and the combination of both sentences as the input during testing. Detail of the result is shown in Table 2 whose column denotes the corresponding testing input type. `OF`, `OT`, and `OTF` refer to our models trained on first, topic, and both first and topic sentences respectively. All models produce seven tokens on average. `OT` performs the best when fed using topic sentence, `OTF` performs the best when fed using the first sentence, while `OTF` and `OT` perform comparably when fed using the combination of first and topic sentences as input during testing. On the other hand, `OF` remains as the worst model across types of input.

We test the difference of performance between pairs of models using two-tailed t-test (two-sample unequal variance), and consider it statistically significant at $p < 0.05$. The difference of performance between models is significant when fed using topic sentence as input. `OTF` significantly outperforms `OT` when fed using first sentence. However, it does not significantly outperform `OF` on this input. Similarly, `OF` also does not significantly outperform `OT` when fed using the first sentence as input. When fed using both first and topic sentences, both `OTF` and `OT` significantly outperform `OF`, while the difference between `OTF` and `OT` is not significant. Based on this fact, we argue that the topic sentence can enhance the model performance across types of input compared when only using the first sentence as input during training. It means, the model trained using topic sentence has a better generalization than model trained using the first sentence. This result is interesting in the sense that the performance is improved on the test set only by using different training input type.

Another interesting thing to note is the copy rate. Copy rate denotes how much the model use the words found in the input data as the headline words (computed using recall). `OT` and `OF` show a relatively similar copy rate across types of input while present different performance scores. `OT` and `OF` perform comparably when fed using the first sentence despite `OT` is trained using the topic sentence (the difference is not statistically significant at $p < 0.05$). However, `OT` outperforms `OF` on other types of input. Simply said, copying words sourced from the topic sentence does a good job for headline generation.

Readers might wonder why the performance trend between `OT` and `OTF` is not consistent between types of input. `OTF` outperforms `OT` when fed using the first sentence during testing but not in other types of input. We infer that there is a possibility of output words sourced from the topic sentence (in training data) despite fed using the unseen first sentence during testing (or vice versa). We infer that `OTF` probably associates words in the first sentence with the words in the topic sentence during training while `OT` cannot do so. When we feed both models using the topic sentence or both first and topic sentences, `OTF` uses words in both first and topic sentence while `OT` only uses words from the topic sentence resulting in lower `OTF`'s performance score. When we test using the first sentence, `OTF` probably copies words from the topic sentence as the result of association capability, resulting in higher performance score. This explanation (although relatively weak) also supports our previous argument that copying words of the topic sentence is helpful in this task.

Generally, the performance scores are higher when models are fed using only the first sentence than other types of input during testing. The first sentence possi-

[5]https://github.com/OpenNMT/OpenNMT-py

Model	Topic				First				First and topic			
	R-1	R-2	R-L	CR	R-1	R-2	R-L	CR	R-1	R-2	R-L	CR
`OF`	29.45	12.06	26.97	0.72	40.83	20.32	37.97	0.81	23.26	7.90	20.89	0.69
`OT`	**33.73**	**14.37**	**30.77**	**0.71**	40.72	19.68	37.76	0.80	**26.69**	**8.98**	**23.69**	**0.71**
`OTF`	32.00	13.03	29.11	0.76	**41.47**	**20.49**	**38.46**	**0.83**	26.49	8.91	23.45	0.75

Table 2: Evaluation result on full Gigaword test set. R denotes full-length *F1* ROUGE score while CR denotes copy rate score.

bly contains a more specific information to the document which is useful for generating headline as it should express a particular input, while topic sentence is useful for generalization purpose.

Some readers might argue that the performance of `OTF` is higher than `OF` and `OT` when fed using the first sentence only because more information is fed during training for `OTF`. In that case, the performance of `OTF` when fed using other types of input during testing should be better than other models as well. In fact, this is not the case in which `OTF` is beaten by `OT` on other types of input. It means training encoder-decoder model using a longer input does not guarantee improvement in performance across types of input. We have to provide an "optimal" input-output pair to train the neural network. In this experiment, it is relatively difficult to conclude which one is more optimal between only using the topic sentence or using both first and topic sentences as input for the training, as there is no trend observed. We are only sure that using the topic sentence as opposed/in addition to the first sentence provides a better generalization.

6 Conclusion and Future Work

In this work, we experiment on incorporating topic sentence which is well studied in the past, and give an empirical proof that the topic sentence is useful for headline generation task. We train the encoder-decoder model using: (1) first sentence, (2) topic sentence, or (3) both first and topic sentences-headline pair. We find that the model trained using the topic sentence has a better generalization compared to the model trained using the first sentence. Training the model using both the first and topic sentences increases the performance even further in a certain case. This fact proves that the topic sentence is useful for news headline generation task.

As future work, we will assess the difference of using topic sentence as opposed to other sentence selection/ranking methods. In addition, it will be interesting to investigate whether using/adding other types of subset of the full news document is able to improve the performance, moreover to automatically decide the optimal subset of text as input for neural headline generation.

References

[1] Enrique Alfonseca, Daniele Pighin, and Guillermo Garrido. Heady: News headline abstraction through event pattern clustering. In *Proceedings of ACL*, pp. 1243–1253, 2013.

[2] Ayana, Shi-Qi Shen, Yan-Kai Lin, Cun-Chao Tu, Yu Zhao, Zhi-Yuan Liu, and Mao-Song Sun. Recent advances on neural headline generation. *Journal of Computer Science and Technology*, Vol. 32, No. 4, pp. 768–784, 2017.

[3] Dzmitry Bahdanau, KyungHyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *Proceedings of ICLR*, 2015.

[4] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of EMNLP*, pp. 1724–1734, 2014.

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

[6] Pierre-Etienne Genest and Guy Lapalme. Framework for abstractive summarization using text-to-text generation. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pp. 64–73, 2011.

[7] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL, System Demonstrations*, pp. 67–72, 2017.

[8] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, 2004.

[9] Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, and aglar Gülehre and Bing Xiang. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *CoNLL*, 2016.

[10] Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. Annotated gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*, pp. 95–100, 2012.

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

[12] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*, pp. 3104–3112, 2014.

[13] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. From neural sentence summarization to headline generation: A coarse-to-fine approach. In *Proceedings of IJCAI*, pp. 4109–4115, 2017.

[14] Wei Wang. Chinese news event 5w1h semantic elements extraction for event ontology population. In *Proceedings of WWW*, pp. 197–202, 2012.

[15] David Zajic, Bonnie J. Dorr, and Richard Schwartz. Bbn/umd at duc-2004: Topiary. In *Proceedings of NAACL Workshop on Document Understanding*, pp. 112–119, 2004.