Coherent Comment Generation for Chinese Articles with a Graph-to-Sequence Model

Wei Li[1], Jingjing Xu[1], Yancheng He[2], Shengli Yan[2], Yunfang Wu[1], Xu Sun[1,3]
[1]MOE Key Lab of Computational Linguistics, School of EECS, Peking University
[2]Platform & Content Group, Tencent
[3]Deep Learning Lab, Beijing Institute of Big Data Research, Peking University
{liweitj47, jingjingxu}@pku.edu.cn
{collinhe, victoryyan}@tencent.com
{wuyf, xusun}@pku.edu.cn

Abstract

Automatic article commenting is helpful in encouraging user engagement and interaction on online news platforms. However, the news documents are usually too long for traditional encoder-decoder based models, which often results in general and irrelevant comments. In this paper, we propose to generate comments with a graph-to-sequence model that models the input news as a topic interaction graph. By organizing the article into graph structure, our model can better understand the internal structure of the article and the connection between topics, which makes it better able to understand the story. We collect and release a large scale news-comment corpus from a popular Chinese online news platform Tencent Kuaibao.[1] Extensive experiment results show that our model can generate much more coherent and informative comments compared with several strong baseline models.[2]

1 Introduction

Online news platform is now a popular way for people to get information, where users also make comments or read comments made by others, making the comments very valuable resource to attract user attention and encourage interactions among users (Park et al., 2016). The ability to automatically generate comments is desirable for online news platforms, especially comments that can encourage user engagement and interactions, serving as one form of intelligent chatbot (Shum et al., 2018). Important as the comment generation task is, it is still relatively new. Qin et al. (2018) proposed the problem of automatic article comment generation, which is to generate comments given the title and content of the article (An example is shown in Table 1). They only proposed the task, but did not propose a specially designed solution to the problem other than sequence-to-sequence paradigm (Sutskever et al., 2014). Ma et al. (2018) proposed a retrieval based model that uses variational topic model to find comments that are related to the news in an unsupervised fashion. Lin et al. (2018) proposed to refer to the retrieved comments during generation, which is a combination of retrieval and generation based model. Pure generation based model remains challenging, yet is a more direct way to solve the problem. Additionally, when the article is very different from the historical ones, there may not be appropriate comments to refer to. In this work, we would like to explore a generation model that better exploits the news content to solve the problem.

Different from the scenarios where sequence-to-sequence models achieve great success like machine translation (Bahdanau et al., 2014) and summarization (See et al., 2017), comment generation has several nontrivial challenges:

- The news articles can be very long, which makes it intractable for classic sequence-to-sequence models. On the contrary, although the title is a very important information resource, it can be too short to provide sufficient information.

- The title of the news sometimes uses hyperbolic expressions that are semantically different from the content of the article. For example, the title shown in the example (Table 1) provides no valuable information other than "Marvel movie", which is far from enough to generate coherent comments.

- Users focus on different aspects (topics) of the news when making comments, which

[1]https://kuaibao.qq.com/

[2]Code for the paper is available at https://github.com/lancopku/Graph-to-seq-comment-generation

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4843–4852
Florence, Italy, July 28 - August 2, 2019.

Title
这部影片被称为“十年来最搞笑漫威电影”，你看了吗?
Have you seen the movie intitled as “the most hilarious Marvel movie”?
Content
点击“IPTV4K超高清”订阅，精彩内容等你共享《复仇者联盟3：无限战争》中的巅峰一役，将战火燃遍了整个宇宙...作为接档《复联3》的漫威电影，《蚁人2》的故事爆笑中带着温情，无疑成为了现阶段抚平漫威粉心中伤痛的一味良药...看过《复联3》的漫威粉们，心中都有同一个疑问：在几乎整个复仇者联盟都参与到无限战争的关键时刻，蚁人究竟去哪儿了？...
Click on the “IPTV4K ultra HD” to subscribe, fantastic contents are waiting for you to share. The battle in “Avengers: Infinity War” has spread the flames of war throughout the universe ... As the continuation Marvel movie to “Avengers 3”, the hilarious and warm “Ant-Man and the Wasp” is no doubt a good dose to heal the fans of Marvel at the time. ... Fans of the Marvel who have watched “Avengers 3” all have a doubt about where Ant-Man is when all other Avengers have been involved in the infinity war.
Comment
只有我觉得那个头盔像蚁人的头盔吗?
Am I the only one that thinks the helmet similar to the helmet of Ant-Man?

Table 1: An example of news article comment generation task, which is to generate new comments given the title and content of the news. Because the article is too long, only the first sentence and three fragments with topic words (blue) are shown. Note that the title and the first sentence of the news are very different from traditional news, which can not summarize the content of the article.

makes the content of the comments very diverse. For example, comments can be about the plots in “*Avengers*”, “*Ant-Man*” or other characters in *Marvel* movies.

Based on the above observations, we propose a graph-to-sequence model that generates comments based on a graph constructed out of content of the article and the title. We propose to represent the long document as a topic interaction graph, which decomposes the text into several topic centered clusters of texts, each of which representing a key aspect (topic) of the article. Each cluster together with the topic form a vertex in the graph. The edges between vertices are calculated based on the semantic relation between the vertices. Compared with the hierarchical structure (Yang et al., 2016), which is designed for long articles, our graph based model is better able to understand the connection between different topics of the news. Our model jointly models the title and the content of the article by combining the title into the graph as a special vertex, which is helpful to get the main point of the article.

We conduct extensive experiments on the news comments collected from Tencent Kuaibao news, which is a popular Chinese online news platform. We use three metrics consulting to Qin et al. (2018) to evaluate the generated comments. Experiment results show that our model can generate more coherent and informative comments compared with the baseline models.

We conclude the contributions as follows:

- We propose to represent the article with a topic interaction graph, which organizes the sentences of the article into several topic centered vertices.
- We propose a graph-to-sequence model that generates comments based on the topic interaction graph.
- We collect and release a large scale (200,000) article-comment corpus that contains title, content and the comments of the news articles.

2 Related Work

The Graph Neural Networks (GNN) model has attracted growing attention recently, which is good at modeling graph structure data. GNN is not only applied in structural scenarios, where the data are naturally performed in graph structure, such as social network prediction systems (Hamilton et al., 2017; Kipf and Welling, 2016), recommender systems (van den Berg et al., 2017; Ying et al., 2018), and knowledge graphs (Hamaguchi et al., 2017), but also non-structural scenarios where the relational structure is not explicit including image classification (Kampffmeyer et al., 2018; Wang et al., 2018), text, etc. In this paper, we explore to use GNN to model non-structural article text.

Some recent researches are devoted to applying GNN in the text classification task, which involves modeling long documents as graphs. Peng et al. (2018) proposed to convert a document into a word co-occurrence graph, which is then used as the input to the convolutional layers. Yao et al. (2018) proposed to organize the words and documents into one unified graph. Edges between words are calculated with point-wise mutual information (PMI), edges between word and document are calculated with TF-IDF. Then a spectral

Algorithm 1 Graph Construction

Require: The title $title$ and article text D, weight calculation function λ

Segment $title$ and D into words

Do named entity recognition and keyword detection and get the keywords κ

for sentence s **do**

  if s contains $k \in \kappa$ **then**

    Assign s to vertex v_k

  else

    Assign s to vertex v_{empty}

  end if

end for

for vertex v_i and v_j **do**

  Calculate edge weight: $w_{ij} = \lambda(v_i, v_j)$

end for

based graph convolutional networks (GCN) is applied to classify the documents. Liu et al. (2018) proposed a siamese GCN model in the text matching task by modelling two documents into one interaction graph. Zhang et al. (2018) adopted a similar strategy but used GCN to match the article with a short query. These works are inspiring to our work, however, they are only designed for the classification task, which are different from generation tasks.

There are also some previous work dedicated to use GNN in the generation tasks. Xu et al. (2018a,b) proposed to use graph based model to encode SQL queries in the SQL-to-Text task. Beck et al. (2018) and Song et al. (2018) proposed to solve the AMR-to-Text problem with graph neural networks. Zhao et al. (2018) proposed to facilitate neural machine translation by fusing the dependency between words into the traditional sequence-to-sequence framework. Although these work apply GNN as the encoder, they are meant to take advantage of the information that are already in the form of graph (SQL query, AMR graph, dependency graph) and the input text is relatively short, while our work tries to model long text documents as graphs, which is more challenging.

3 Graph-to-Sequence Model

In this section, we introduce the proposed graph-to-sequence model (shown in Figure 1). Our model follows the Encoder-Decoder framework. The encoder is bound to encode the article text presented as an interaction graph into a set of hidden vectors, based on which the decoder generates the comment sequence.

3.1 Graph Construction

In this section, we introduce how to construct the topic interaction graph from a news article. Algorithm 1 shows the construction process. Different from traditional news, the articles from online news platforms contain much noise. Many sentences of the articles are even irrelevant to the main topic of the news. For example, "谢谢大家点开这篇文章" (*Thanks for opening this article*). Therefore, we extract the keywords of the article which serve as the topics of the news. These keywords are the most important words to understand the story of the article, most of which are named entities. Since keyword detection is not the main point of this paper, we do not go into the details of the extraction process.

Given a news article D, we first do word segmentation and named entity recognition on the news articles with off-the-shelf tools such as Stanford CoreNLP.[3] Since the named entities alone can be insufficient to cover the main focuses of the document, we further apply keyword extraction algorithms like TextRank (Mihalcea and Tarau, 2004) to obtain additional keywords.

After we get the keywords κ of the news, we associate each sentence of the documents to its corresponding keywords. We adopt a simple strategy that assigns a sentence s to the keyword k if k appears in the sentence. Note that one sentence can be associated with multiple keywords, which implicitly indicates connection between the two topics. Sentences that do not contain any of the keywords are put into a special vertex called "*Empty*". Because the title of the article is crucial to understand the news, we also add a special vertex called "*Title*" that contains the title sentence of the article.

The sentences together with the keyword k they belong to form a vertex v_k in the interaction graph. The words of the sentences are concatenated together. The words within each vertex represent one aspect of the article. There can be many ways to construct the edges between vertices denoted as λ in Algorithm 1. In this paper, we propose to adopt a structure based method. If vertices v_i and v_j share at least one sentence, we add an edge e_{ij} between them, the weight of which is calculated by the number of shared sentences. The intuition behind this design is that the more sentences co-mention two keywords together, the closer these

[3] https://stanfordnlp.github.io/CoreNLP

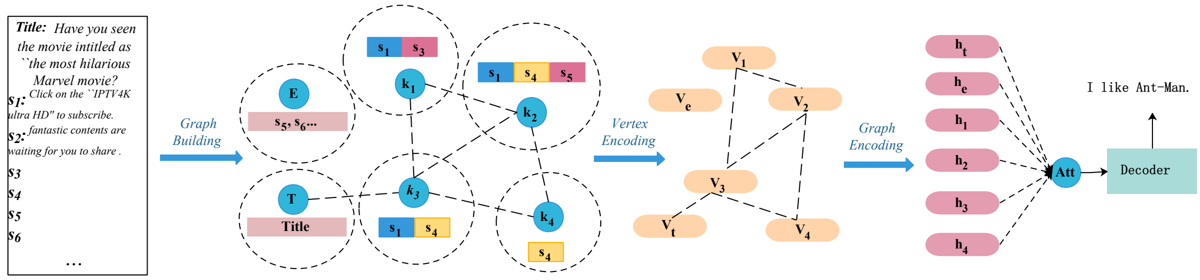

Figure 1: A brief illustration of our proposed graph-to-sequence model. A vertex in the interaction graph consists of a topic word k_i and the sentences containing k_i. If a sentence contains no topic word, it is archived to a special "Empty" vertex. Each vertex is first encoded into a hidden vector v_i by the vertex encoder. Then the whole graph is fed into the graph encoder and get the final vertex representation h_i encoded with structure information. A RNN decoder with attention mechanism is adopted to generate comment words.

two keywords are. One can also use content based method such as tf-idf similarity between the content of v_i and v_j.

3.2 Vertex Encoder

To encode each vertex in the graph into one hidden vector v, we propose to use a multi-head self-attention (Vaswani et al., 2017) based vertex encoder.

The vertex encoder consists of two modules, the first one is an embedding module, the second one is a self-attention module. For the i-th word w_i in the word sequence, we first look up the word embedding of the words e_i. Note that the keywords and regular words in the article share the same embedding table. By "regular words" we mean words other than keywords. To represent the position information of each word, a positional embedding p_i is added to the word. The keyword k of the vertex is put in the front of the word sequence. Therefore, the positional embedding of all the inserted keywords share the same embedding p_0, which indicates the special role of the keyword. Both the word embedding and positional embedding are set to be learn-able vectors. The final embedding ϵ_i of word w_i is the sum of the original word embedding e_i and positional embedding p_i,

$$\epsilon_i = e_i + p_i$$

Then we feed ϵ_i to the self-attention module and get the hidden vector a_i of each word. This module is to model the interaction between the words so that each hidden vector in this layer contains the context information of the vertex. The self-attention module contains multiple layers of multi-head self-attention. The hidden vector of each layer is calculated by Equation (1)-(3), where Q, K, V represent query vector, key vector and value vectors respectively. In our case, Q, K, V all represent the same vectors. For the first layer, they are ϵ. For the following layers, they are the hidden vectors calculated by the previous layer. W^o, W_i^Q, W_i^K, W_i^V are all learnable matrices,

$$Attention(Q, K, V) = softmax(QK^T)V \quad (1)$$

$$MultiHead(Q, K, V) = [head_1; \cdots; head_h]W^o \quad (2)$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

Since the keyword k is the most important information in the vertex, we use the hidden vector of the inserted keyword a_0 in the last layer as the vector that represents the whole vertex.

3.3 Graph Encoder

After we get the hidden vector of each vertex v_i in the graph, we feed them to a graph encoder to make use of the graph structure of the constructed topic interaction graph. We propose to use spectral based graph convolutional model (GCN). Spectral approaches work with a spectral representation of the graphs (Zhou et al., 2018). We choose this architecture because GCN can both model the content of the vertex and make use of the structure information of the graph.

We use an implementation of GCN model similar to the work of Kipf and Welling (2016). Denote the adjacency matrix of the interaction graph as $A \in R^{N \times N}$, where $A_{ij} = w_{ij}$ (defined in Section 3.1). We add an edge that points to the node itself (Equation 5). D is a diagonal matrix where $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$,

$$H^{l+1} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^l W^l) \quad (4)$$

$$\tilde{A} = A + I_N \quad (5)$$

where I_N is the identity matrix, $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}$ is the normalized symmetric adjacency matrix, W^l is a learnable weight matrix. To avoid the over-smoothing problem of GCN, we add residual connections between layers,

$$g^{l+1} = H^{l+1} + H^l \tag{6}$$
$$g^{out} = tanh(W_o g^K) \tag{7}$$

We add one feed forward layer to the final output of the GCN. g^K is the output of the last layer of GCN.

Since the title of the news is still an important information, we use the hidden output of the title vertex of the graph encoder as the initial state t_0 of the decoder. One can also use other pooling method such as max pooling or mean pooling.

3.4 Decoder

For the decoder, we adopt the recurrent neural network (RNN) decoder with attention mechanism (Bahdanau et al., 2014). Given the initial state t_0 and the output of the GCN $\langle g_0, g_1, \cdots, g_n \rangle$, the decoder is bound to generate a sequence of comment tokens $y_1, y_2, \cdots, y_m$. At each decoding step, a context vector c_i is calculated by doing attention on the outputs of the GCN,

$$t_i = RNN(t_{i-1}, e_{i-1}) \tag{8}$$
$$c_i = \sum \alpha_j \times g_j \tag{9}$$
$$\alpha_j = \frac{exp(\delta(t_i, g_j)}{\sum exp(\delta(t_i, g_k))} \tag{10}$$

where δ is the attention function.

Since the topic words (name of the vertices) κ are important information for the article and may appear in the comment, we adopt copy mechanism (Gu et al., 2016) by merging the predicted word token probability distribution with the attention distribution. The probability p_{copy} of copying from the topic words is dynamically calculated with the decoding hidden state t_i and the context vector c_i,

$$y_i = softmax(W_o(tanh(W([t_i; c_i]) + b))) \tag{11}$$
$$p_{copy} = \sigma(W_{copy}[t_i; c_i]) \tag{12}$$
$$p = (1 - p_{copy}) \times y + p_{copy} \times \alpha \tag{13}$$

where W_o, W, W_{copy}, b are all learnable parameters.

Topic	document #	comment #
Entertainment	116,138	287,889
Sport	90,979	378,677

Table 2: Document and comment number of Entertainment and Sport.

	ave word #		ave character #	
	Ent	Sport	Ent	Sport
content	456.1	506.6	754.0	858.7
title	16.4	15.7	28.1	27.4
comment	16.3	19.4	26.2	31.2
keyword	8.4	9.0	-	-

Table 3: Length of content, title, comment and keyword of the news for the topic of **Ent** (entertainment) and **Sport**.

4 Experiment

4.1 Corpus

We collect news and comments from Tencent Kuaibao,[4] which is a popular online news platform in Chinese. Because the number of news is very large and the comments vary a lot between different topics of news, we select the news from two most popular topics (topics that have the most news and comments) *Entertainment* and *Sport*. The data is available at https://pan.baidu.com/s/1b5zAe7qqUBmuHz6nTU95UA[5]. The document number and comment number of the two topics are listed in Table 2.

The average length with respect to words and characters of content, title, comment and keyword for the two topics are listed in Table 3. From the number we can see that the length of news content is too large for traditional sequence-to-sequence model.

4.2 Experiment Settings

We use a batch size of 32. The embedding size is set to 128. The word embeddings are shared between encoder and decoder. Because the vertex number (keyword number in Table 3) is relatively small, to ease the over-smoothing problem we use 1-layer convolution in GCN. For all the RNN based encoders, we use bidirectional LSTM and set the hidden size to 128. For the baseline hierarchical attention model, the hidden size of the second LSTM layer is 256. We use a vocabulary

[4] https://kuaibao.qq.com/

[5] The extraction code is 6xdw

size of 60,000. The sentences are truncated to 100 words. The maximum length for generating is set to 32. For multi-head attention, we use 4 heads. For RNN encoder, RNN decoder and multi-layer self-attention, we use a layer number of 2. We use a dropout rate of 0.1. We use Adam optimizer (Kingma and Ba, 2014) to train the parameters. The initial learning rate is set to 0.0005. For all the models, we train for 5 epochs, the learning rate is decayed to half after each epoch.

4.3 Evaluation Metrics

We choose three metrics to evaluate the quality of generated comments. For all the metrics, we ask the raters to score the comments with three gears, the scores are then projected to $0 \sim 10$.

- **Coherence**: This metric evaluates how Coherent (consistent) is the comment to the news document. It measures whether the comment is about the main story of the news, one side part of the news, or irrelevant to the news.
- **Informativeness**: This metric evaluates how much concrete information the comment contains. It measures whether the comment involves a specific aspect of some character or event, or is a general description of some character or event, or is a general comment that can be the answer to many news.
- **Fluency**: This metric evaluates whether the sentence is fluent. It mainly measures whether the sentence follows the grammar and whether the sentence accords with the logic including world knowledge.

We ask three raters to evaluate the generated comments of different models. Owing to the laborious evaluation process (reading the long news document is time consuming), we ask the raters to evaluate the generated comments from one hundred news documents of both topics. The raters are given both the title and the document content of the news which is the same as how a user would read the news online.

We use spearman's rank score to measure the correlation between raters.The p-values are all below $1e-50$. The ratings between raters have relatively good correlation with spearman's rank of around 0.6. Among the metrics, fluency is more divergent. This is expected as this metric is more flexible, different people may have more divided opinion.

4.4 Baseline Models

In this section, we describe the baseline models we use. The settings of these models are described in Section 4.2. Note that for fair comparison, all the baselines use RNN with attention as the decoder, the choice of the encoder is dependent on the input of the model (whether the input is in order or not).

- **Seq2seq** (Qin et al., 2018): this model follows the framework of sequence-to-sequence model with attention. We use three kinds of input, the title (**T**), the content (**C**) and the title together with the content (**TC**). The length of the input sequence is truncated to 100. For the input of title together with content, we append the content to the back of the title.
- **Self-attention** (Chen et al., 2018): this model follows the encoder-decoder framework. We use multi-layer self-attention with multi-head as the encoder, and a RNN decoder with attention is applied. We use two kinds of input, the bag of words (**B**) and the keywords (**K**). Since the input is not sequential, positional encoding is not applied. A special '*CLS*' label is inserted, the hidden vector of which serves as the initial state of decoder. For the bag of words input we use the words with top 100 term frequency (TF) in the news document. For the keywords input, we use the same extracted keywords (topic words) with the ones used in our topic interaction graph.
- **Hierarchical-Attention** (Yang et al., 2016): this model takes all the content sentences as input and applies hierarchical attention as the encoder to get the sentence vectors and document vector. A RNN decoder with attention is applied. The document vector is used as the initial state for RNN decoder.

4.5 Results

In Table 4 and Table 5, we show the results of different baseline models and our graph2seq model for the topic of entertainment and sport separately. From the results we can see that our proposed graph2seq model beats all the baselines in both coherence and informativeness.

Coherence: Our model receives much higher scores in coherence compared with all other baseline models. This indicates that our graph based

Models	Coherence	Informativeness	Fluency	Total
seq2seq-T (Qin et al., 2018)	5.38	3.70	8.22	5.77
seq2seq-C (Qin et al., 2018)	4.87	3.72	8.53	5.71
seq2seq-TC (Qin et al., 2018)	3.28	4.02	8.68	5.33
self-attention-B (Chen et al., 2018)	6.72	5.05	8.27	6.68
self-attention-K (Chen et al., 2018)	6.62	4.73	8.28	6.54
hierarchical-attention (Yang et al., 2016)	1.38	2.97	**8.65**	4.33
graph2seq (proposed)	**8.23**	**5.27**	8.08	**7.19**

Table 4: Comparison between our graph2seq model and baseline models for the topic of **entertainment**. T, C, B, K represents title, content, bag of words, keywords separately. *Total* is the average of other three metrics

Models	Coherence	Informativeness	Fluency	Total
seq2seq-T (Qin et al., 2018)	4.30	4.38	6.27	4.98
seq2seq-C (Qin et al., 2018)	3.88	3.85	6.02	4.58
seq2seq-TC (Qin et al., 2018)	4.70	5.08	6.37	5.38
self-attention-B (Chen et al., 2018)	5.15	5.62	6.28	5.68
self-attention-K (Chen et al., 2018)	6.68	5.83	**7.00**	6.50
hierarchical-attention (Yang et al., 2016)	4.43	5.05	6.02	5.17
graph2seq (proposed)	**7.97**	**6.18**	6.37	**6.84**

Table 5: Comparison between our graph2seq model and baseline models for the topic of **sport**. T, C, B, K represents title, content, bag of words, keywords separately. *Total* is the average of other three metrics

model can better get the main point of the article instead of referring to the high frequency terms that are only slightly related or even irrelevant to the article, which is often carried out by baseline models (especially seq2seq based models). Besides, other baseline models tend to generate general comments such as "*I still think I like him*" when encountering low frequency topics (similar to the dull response problem in dialogue). These two phenomena hurt the coherence performance severely. Compared with other baselines, self-attention based models receive higher coherence score, we assume that this is because the most relevant words are maintained by the bag of words and keywords input. However, it is hard to distinguish the main point of the article from all other input words with self-attention model. Therefore, they do not perform as well as our graph based model, which can make use of the structure of the article. For the hierarchical attention model, although it uses a hierarchical structure to organize the article, it is still very difficult for the model to understand the story. In fact, we observe in the experiment that the hierarchical structure even makes it harder to extract useful information because of the over-simplified attention performed in the word level.

Informativeness: For the metric of informativeness, our graph2seq model can generate comments with the most information because it can capture the plot of the article. We observe that this metric is related to the metric of coherence. Models with higher coherence score tend to be more informative. This phenomenon is related to the fact that many of the comments with low informative scores are general comments which are naturally not coherent to the news. In Figure 2 we show the number of generated *general* comments and number of generated *unique* words for both topics. By "*general comment*", we mean those comments that have no specific information, irrelevant to the news and can be the comment to many other news of different stories, e.g., "*I still think I like him*". Note that the notion of *general comment* is not strictly defined, but an information that is meant to help analyze *informativeness* score. The unique words are those not in a pre-defined stop word list. From the figure we can see that the number of general comments is loosely negatively correlated to the informative score, especially in entertainment topic. The number of generated unique words can also be an indicator for the informativeness of the comments, because the more words are involved in the comment, the more information the comment is able to provide.

Fluency: Our model receives comparable fluency score in the experiments, we assume that this is be-

Title	被王丽坤美到了，《上新了·故宫》里穿古装温婉又娴静，气质惊艳 *In "updates of the Palace Museum" Likun Wang appears so gentle, refined and astonishingly elegant wearing ancient costume that audiences are touched by her beauty.*
S2S-T	我觉得还是喜欢看的古装，古装扮相，古装扮相很好看 *I still think I like ancient costume, appearance in ancient costume, appearance in ancient costume is pretty.*
S2S-C	我觉得还是喜欢看的 *I still think I like to watch*
S2S-TC	我觉得还是喜欢看的 *I still think I like to watch*
SA-B	我觉得赵丽颖的演技真的很好 *I think the acting skill of Liying Zhao is very good*
SA-K	我觉得还是喜欢李沁 *I still think I like Qin Li*
HA	我觉得还是喜欢看她的剧 *I still think I like her plays*
graph2seq	王丽坤的演技真的好 *The acting skill of Likun Wang is really good*

Table 6: An example of comments generated by different models. **Title** is the original title of the article. *S2S*, *SA*, *HA* indicate seq2seq, self-attention and hierarchical attention respectively. T, C, B, K represents title, content, bag of words, keywords separately.

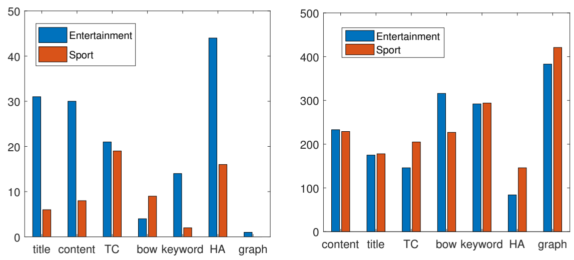

Figure 2: Number of generated **general comments** (Left, the lower the better) and number of **unique words** (Right, the higher the better) in the generated comments by different models. The comments from a total number of 100 news articles are inspected.

cause of the similar structure of decoder between different models. After inspecting a part of the generated comments, we observe that the following reasons may lead to low fluency cases.

(1) The generated comment is against the world knowledge, for instance, "The big feast is a good actor (大餐是个好演员)".

(2) The model can not distinguish between similar characters, for instance, "*Who is Han Lu? I only know Han Lu* (鹿晗是谁？我只认识鹿晗) ".

(3) The model sometimes repeatedly generates the same names. We assume that this is because repeated pattern appears in some of the real comments and the copy mechanism sometimes makes the problem more severe.

These phenomena are actually observed in comments generated by various models, problems such as the deficiency of understanding world knowledge are actually very hard to solve, which are beyond the discussion of this paper.

4.6 Case Study

In Table 6 we show an example of comments generated by different models.

For the seq2seq-**T** (**S2S-T**) model (Qin et al., 2018), the comment is generated mainly based on the clue "ancient costume" in the title. However, because "ancient costume" is not frequently seen in the comments (in the training set). The pattern of generating comments about "ancient costume" is not well learned by the model, which makes the language of the comment not fluent. The comment generated by the seq2seq-**C** (**S2S-C**) model is a typical *general comment*, which includes no specific information. This happens when the input to the model does not contain obvious signals that indicates what topic the comment should be about. Despite the fact that these comments are not what we desire, these comments get good fluency scores, which explains why the fluency scores of some of the baselines exceed our model's. The comment made by hierarchical attention model (**HA**) suffers from the same problem with seq2seq model. We assume that this is because even with the hierarchical structure, this model can not understand the long input well. Therefore, it can not extract the main point of the story and generate general comments.

The comments made by self-attention based models (**SA**) are generally more informative, which contain more specific plots or characters. Even though the input to these models are not in order, the combination of the keywords makes the model easier to associate the input with some learned pattern. However, this way of representing the article is incapable of getting the main point of the article. The main characters in the generated comments " 赵丽颖 " and " 李沁 " (names of Chinese actresses) are not much related to the news.

The comment generated by our proposed graph2seq model is the only model that mentions the main character of the news "王丽坤" (name of the Chinese actress), which accords with the expectation of the design of our graph based model.

5 Conclusion

In this paper, we propose to automatically generate comment of articles with a graph-to-sequence model that organizes the article into a topic interaction graph. Our model can better understand the structure of the article, thus capturing the main point of the article. Experiment results show that our model can generate more coherent and informative comments. We observe that there are still some comments conflicting with the world knowledge. In the future, we would like to explore how to introduce external knowledge into the graph to make the generated comments more logical.

Acknowledgement

We thank the anonymous reviewers for their thoughtful comments. This work was supported in part by National Natural Science Foundation of China (No. 61673028). Xu Sun is the corresponding author of this paper.

References

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

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. *arXiv preprint arXiv:1806.09835*.

Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *stat*, 1050:7.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Niki Parmar, Mike Schuster, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. *CoRR*, abs/1804.09849.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *CoRR*, abs/1603.06393.

Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. 2017. Knowledge transfer for out-of-knowledge-base entities: a graph neural network approach. *arXiv preprint arXiv:1706.05674*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 1024–1034. Curran Associates, Inc.

Michael Kampffmeyer, Yinbo Chen, Xiaodan Liang, Hao Wang, Yujia Zhang, and Eric P Xing. 2018. Rethinking knowledge graph propagation for zero-shot learning. *arXiv preprint arXiv:1805.11724*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2018. Learning comment generation by leveraging user-generated data. *arXiv preprint arXiv:1810.12264*.

Bang Liu, Ting Zhang, Di Niu, Jinghong Lin, Kunfeng Lai, and Yu Xu. 2018. Matching long text documents via graph convolutional networks. *CoRR*, abs/1802.07459.

Shuming Ma, Lei Cui, Furu Wei, and Xu Sun. 2018. Unsupervised machine commenting with neural variational topic model. *arXiv preprint arXiv:1809.04960*.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

Deokgun Park, Simranjit Sachar, Nicholas Diakopoulos, and Niklas Elmqvist. 2016. Supporting comment moderators in identifying high quality online news comments. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1114–1125, New York, NY, USA. ACM.

Hao Peng, Jianxin Li, Yu He, Yaopeng Liu, Mengjiao Bao, Lihong Wang, Yangqiu Song, and Qiang Yang. 2018. Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1063–1072. International World Wide Web Conferences Steering Committee.

Lianhui Qin, Lemao Liu, Wei Bi, Yan Wang, Xiaojiang Liu, Zhiting Hu, Hai Zhao, and Shuming Shi. 2018. Automatic article commenting: the task and dataset. *arXiv preprint arXiv:1805.03668*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Heung-Yeung Shum, Xiao-dong He, and Di Li. 2018. From eliza to xiaoice: challenges and opportunities with social chatbots. *Frontiers of Information Technology & Electronic Engineering*, 19(1):10–26.

Linfeng Song, Yue Zhang, Zhiguo Wang, and Daniel Gildea. 2018. A graph-to-sequence model for amr-to-text generation. *arXiv preprint arXiv:1805.02473*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Xiaolong Wang, Yufei Ye, and Abhinav Gupta. 2018. Zero-shot recognition via semantic embeddings and knowledge graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6857–6866.

Kun Xu, Lingfei Wu, Zhiguo Wang, and Vadim Sheinin. 2018a. Graph2seq: Graph to sequence learning with attention-based neural networks. *arXiv preprint arXiv:1804.00823*.

Kun Xu, Lingfei Wu, Zhiguo Wang, Mo Yu, Liwei Chen, and Vadim Sheinin. 2018b. Sql-to-text generation with graph-to-sequence model. *arXiv preprint arXiv:1809.05255*.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2018. Graph convolutional networks for text classification. *arXiv preprint arXiv:1809.05679*.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. *arXiv preprint arXiv:1806.01973*.

Ting Zhang, Bang Liu, Di Niu, Kunfeng Lai, and Yu Xu. 2018. Multiresolution graph attention networks for relevance matching. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 933–942.

Guoshuai Zhao, Jun Li, Lu Wang, Xueming Qian, and Yun Fu. 2018. Graphseq2seq: Graph-sequence-to-sequence for neural machine translation.

Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. 2018. Graph neural networks: A review of methods and applications. *CoRR*, abs/1812.08434.