Outline Generation: Understanding the Inherent Content Structure of Documents

Abstract

In this paper, we introduce and tackle the Outline Generation (OG) task, which aims to unveil the inherent content structure of a multi-paragraph document by identifying its potential sections and generating the corresponding section headings. Without loss of generality, the OG task can be viewed as a novel structured summarization task. To generate a sound outline, an ideal OG model should be able to capture three levels of coherence, namely the coherence between context paragraphs, that between a section and its heading, and that between context headings. The first one is the foundation for section identification, while the latter two are critical for consistent heading generation. In this work, we formulate the OG task as a hierarchical structured prediction problem, i.e., to first predict a sequence of section boundaries and then a sequence of section headings accordingly. We propose a novel hierarchical structured neural generation model, named HiStGen, for the task. Our model attempts to capture the three-level coherence via the following ways. First, we introduce a Markov paragraph dependency mechanism between context paragraphs for section identification. Second, we employ a section-aware attention mechanism to ensure the semantic coherence between a section and its heading. Finally, we leverage a Markov heading dependency mechanism and a review mechanism between context headings to improve the consistency and eliminate duplication between section headings. Besides, we build a novel WIKIOG dataset, a public collection which consists of over 1.75 million document-outline pairs for research on the OG task. Experimental results on our benchmark dataset demonstrate that our model can significantly outperform several state-of-the-art sequential generation models for the OG task.

Introduction

Document understanding is one of the critical and challenging tasks in information processing. There

have been many related research topics in this direction, such as keyword detection BIBREF0 , BIBREF1 , topic modeling BIBREF2 , BIBREF3 , headline generation BIBREF4 , BIBREF5 and text summarization BIBREF6 , BIBREF7 . Keyword detection and topic modeling aim to describe a document by a few important words or topics (i.e., distributions of words) for concise representation; While headline generation and text summarization attempt to compress the document into one or a few sentences to capture the key information. As we can see, most existing research on document understanding has focused on the coarse-grained understanding of documents by capturing its global semantics. In this paper, we attempt to provide fine-grained understanding of documents by unveiling its inhere content structure BIBREF8 , BIBREF9 , i.e., to understand how the document is organized and what it talks about in each part .

We thus introduce the Outline Generation (OG) task in this work. Given a multi-paragraph document, the OG task aims to identify its potential sections and generate the corresponding section headings. Figure FIGREF3 shows some typical outline of articles, where Figure FIGREF3 (a) depicts the outline of a Wikipedia article with a two-level hierarchy, and Figure FIGREF3 (b) depicts a typical outline of a research paper. As we can see, the outline can clearly capture the content structure of a document with concise text descriptions (i.e., section headings), which can not only help navigate the reading but also significantly reduce the cognitive burden over the document. Moreover, outlines can also facilitate a variety of text analysis applications such as text clustering and topic survey.

In a conceptual level, the OG task could be viewed as a kind of summarization task. However, from the examples shown in Figure FIGREF3 , we can find clear differences between the OG task and traditional summarization tasks. Firstly, the OG task produces a structured output with short descriptions (i.e., keywords or key phrases), while the output of traditional summarization is usually a set of unstructured sentences. Secondly, the OG task needs to summarize the paragraphs (into sections) in a strict sequential order, while the sentences in traditional summarization usually do not map to the paragraphs

linearly. Thirdly, the section headings in one outline usually follow a similar style (e.g., topical headings as in Figure FIGREF3 (a) and functional headings as in Figure FIGREF3 (b)), while there is no such requirements in traditional summarization. Therefore, the OG task is actually a novel structured summarization task with its own special challenges.

If we take a further look at the OG task, we can find there are actually two structured prediction problem within it, i.e., to identify a sequence of sections (i.e., paragraphs with coherent information/topics), and to generate a sequence of section headings (i.e., short descriptions that summarize the sections) accordingly. Both problems are non-trivial. For section identification, it is unknown how many sections there are in a document. For section heading generation, headings should be able to reflect the section content in a consistent style. To achieve these two goals, an ideal OG model should be able to capture three levels of coherence, namely the coherence between context paragraphs, that between a section and its heading, and that between context headings. The first one is the foundation for section identification, while the latter two are critical for consistent heading generation.

In this work, we formulate the OG task as a hierarchical structured prediction problem and introduce a novel hierarchical structured neural generation model, named HiStGen, to solve it. In this model, we view the section boundary prediction problem as a first-level sequential labeling process, and the section heading generation as a second-level structured prediction which depends on the predicted boundary labels from the lower level. For section identification, we employ a Markov paragraph dependency mechanism to model the coherence in adjacent paragraphs to help decide the section boundaries. For section heading generation, we leverage a section-aware attention mechanism BIBREF10 to allow the decoder to focus on the most informative content within a section for heading generation. Furthermore, we introduce a Markov heading dependency mechanism and a review mechanism BIBREF11 between context headings. The Markov heading dependency mechanism is used for modeling the consistency between adjacent headings, while the review mechanism is employed to avoid the repetition in the

generated headings.

To facilitate the study and evaluation of the OG task, we build a new benchmark dataset based on Wikipedia articles. As we can see, in most multi-paragraph Wikipedia articles, human editors would segment the article into several sections and provide the outline as an overview of the content structure. Therefore, we can directly leverage these articles to build the benchmark. Specifically, we collect Wikipedia articles with outlines under “celebrity", “cities” and “music” category, and obtain hundreds of thousands of articles respectively. We remove the outlines from Wikipedia articles to form the raw text input. The task is to recover the sections and section headings simultaneously. We call this benchmark dataset as WIKIOG.

For evaluation, we compare with several state-of-the-art methods to verify the effectiveness of our model. Empirical results demonstrate that outline generation for capturing the inherent content structure is feasible and our proposed method can outperform all the baselines significantly. We also provide detailed analysis on the proposed model, and conduct case studies to provide better understanding on the learned content structure.

The main contributions of this paper include:

Related Work

To the best of our knowledge, outline generation over a multi-paragraph document is a new task in the natural language processing community. The most closely related tasks to the OG task are keyword extraction, headline generation, text summarization and storyline generation tasks, which have been studied extensively in the past decades.

Keyword extraction aims to automatically extract some keywords from a document. Most of the existing keyword extraction methods have addressed this problem through two steps. The first step is to acquire a list of keyword candidates (e.g., n-grams or chunks) with heuristic methods BIBREF12 , BIBREF13 . The second step is to rank candidates on their importance to the document, either with supervised machine learning methods BIBREF14 , BIBREF15 , BIBREF16 , BIBREF17 or unsupervised machine learning methods BIBREF18 , BIBREF19 , BIBREF20 , BIBREF0 . However, these approaches could neither identify keywords that do not appear in the text, nor capture the real semantic meaning behind the text. Recently, natural language generation models are used to automatically generate keywords. BIBREF21 BIBREF21 applied an encoder-decoder framework BIBREF22 with a copy mechanism BIBREF23 to this task, achieving state-of-the-art performance. BIBREF11 BIBREF11 modeled correlation among multiple keywords in an end-to-end fashion to eliminate duplicate keywords and improve result coherence.

Headline generation aims to describe a document by a compact and informative headline, with the constraint that only a short sequence of words is allowed to generate BIBREF4 . Early work has pointed out that a purely extractive approach is not appropriate to generate headlines from the document text BIBREF24 . This is due to two major reasons: (1) The single sentence extracted from the document is often longer than the desired headline size; (2) Sometimes the most important information is distributed across several sentences in the document. Hence, many studies have focused on either extracting and reordering n-grams from the document BIBREF24 , or selecting one or two informative sentences from the document, and then reducing them to the target headline size BIBREF4 . Recently, the task is formulated as a Seq2Seq learning problem and neural encoder-decoder architectures have been widely adopted to solve it. BIBREF25 BIBREF25 trained an encoder-decoder recurrent neural network with attention for generating news headlines using the news articles from the English Gigaword corpus. BIBREF26 BIBREF26 proposed to generate the headline from multiple summaries using a hierarchical attention model for the New York Times corpus.

Text summarization is the process of automatically generating one or more natural summaries from an input document that retain the most important information. Most summarization models studied in the past are extractive in nature BIBREF27 , BIBREF28 , BIBREF29 , which try to extract the most important sentences in the document and rearranging them into a new summary. Recent abstractive summarization models have shown better flexibility and can generate more novel summaries. Many abstractive models BIBREF30 , BIBREF5 , BIBREF31 are based on the neural encoder-decoder architecture. To facilitate the research, a set of summarization tasks have been proposed in the Document Understanding Conference (DUC). These tasks often provide multiple human-generated reference summaries of the document for evaluation.

Storyline generation aims to summarize the development of certain events and understand how events evolve over time. BIBREF32 BIBREF32 formalized different types of sub-events into local and global aspects. Some studies have been conducted in storyline generation with Bayesian networks to detect storylines BIBREF33 , BIBREF34 . BIBREF35 BIBREF35 firstly obtained relevant tweets and then generate storylines via graph optimization for the Tweets2011 corpus.

The OG task introduced in our work is related to the keyword extraction, headline generation, text summarization and storyline generation tasks but with some clear differences. Firstly, the output of keyword extraction is usually a set of unstructured keywords, while the OG task produces a structured output with short descriptions. Secondly, the output of the headline generation task is a single heading at the document-level with coarse-grained semantics, while the output of our OG task is a sequence of headings at the section-level with fine-grained semantics. Thirdly, text summarization aims to capture the major content of a document by producing a few unstructured sentences, while our OG task attempts to unveil the inherent content structure of a document by identifying its potential sections and generating the corresponding section headings. Finally, storyline generation is based on the multiple sub-events along the timeline, while the OG task focuses on the multiple sections. Therefore, most existing methods

applied for these related tasks may not fit the OG task directly.

Problem Statement

In this section, we introduce the OG task, and describe the benchmark dataset WIKIOG in detail. A summary of key notations in this work is presented in Table TABREF7 .

Task Description

Given a multi-paragraph document, the OG task aims to unveil its inherent content structure, i.e., to identify the potential sections (i.e., sequential paragraphs with coherent information/topics) of the document, as well as to generate the section headings (i.e., a short description that summarizes the section) correctly. Specifically, headings over different sections should be consistent in style and exclusive on topics, i.e., they should cover different aspects in a similar style. For example, as shown in Figure FIGREF3 (b), headings in a research paper might include introduction, related work, method and so on. These headings are exclusive to each other and mainly describe the function of each section in the paper.

Formally, given a document INLINEFORM0 composed of a sequence of paragraphs INLINEFORM1 , the OG task is to learn a structured prediction model INLINEFORM2 for INLINEFORM3 to identify a sequence of sections INLINEFORM4 and produce the corresponding section headings INLINEFORM5 simultaneously, DISPLAYFORM0

where INLINEFORM0 .

Benchmark Construction

In order to study and evaluate the OG task, we build a new benchmark dataset WIKIOG. We take Wikipedia articles as our source articles since (1) Wikipedia is publicly available and easy to collect; (2) Most multi-paragraph Wikipedia articles contain outlines as an overview of the article, which are constructed by professional human editors. Specifically, we collect English Wikipedia articles under three categories, i.e., “celebrity”, “cities” and “music”. We only make use of the first-level headings as our ground-truth, and leave the deeper-level headings (e.g., second-level headings) generation for the future study. Articles with no headings or more than ten first-level headings are removed, leaving us roughly INLINEFORM0 million articles in total. Table TABREF9 shows the overall statistics of our WIKIOG benchmark dataset.

For the OG task, we remove the outlines from Wikipedia articles, and concatenate all the paragraphs together to form the raw text input INLINEFORM0 . We record all the sections by their boundaries INLINEFORM1 as well as all the corresponding section headings INLINEFORM2 . In this way, we obtain the INLINEFORM3 paragraph, section boundary label, section heading INLINEFORM4 triples, i.e., INLINEFORM5 , as ground-truth data for training/validation/testing.

Our Approach

In this section, we introduce our proposed approach for the OG task in detail. We first give an overview of the problem formulation and the model architecture. We then describe each component of our model as well as the learning procedure specifically.

Overview

Without loss of generality, the OG task can be decomposed into two structured prediction problems: 1) Section Identification: a sequential labeling process to identify the section boundaries; and 2) Section

Heading Generation: a sequential generation process to produce short text descriptions for each identified section. These two structured prediction problems are coupled in the sense that the section heading prediction is dependent on the section prediction results. Therefore, in this work, we formulate the OG task as a hierarchical structured prediction problem and introduce a novel hierarchical structured neural generation model, named HiStGen for short, to solve it. The overall architecture of HiStGen is illustrated in Figure FIGREF8 .

Basically, the HiStGen employs the encoder-decoder framework. In the encoding phase, to obtain the representation of a multi-paragraph document, HiStGen utilizes the hierarchical encoder framework BIBREF36 to obtain the document representation. The decoding phase is hierarchical, where we exploit three-level coherence for better OG prediction. Specifically, we employ a Markov paragraph dependency mechanism between context paragraphs for the section boundary prediction problem. Moreover, HiStGen employs a section-aware attention mechanism between a section and its heading, and a Markov heading dependency mechanism and a review mechanism between context headings for the heading generation problem whenever a new section is identified. We will discuss the details of these model designs in the following sections.

Encoder

The goal of the encoder is to map the input document to a vector representation. In HiStGen, we adopt a hierarchical encoder framework, where we use a word encoder to encode the words of a paragraph INLINEFORM0 , and use a paragraph encoder to encode the paragraphs of a document INLINEFORM1 .

As depicted in Figure FIGREF8 , each word INLINEFORM0 in each paragraph INLINEFORM1 is represented by its distributed representation INLINEFORM2 . We use a bi-directional GRU as both the word and paragraph encoder, which summarizes not only the preceding words/paragraphs, but also the

following words/paragraphs. The forward GRU in word encoder reads the words in the INLINEFORM3 -th paragraph INLINEFORM4 in the left-to-right direction, resulting in a sequence of hidden states INLINEFORM5 . The backward GRU reads INLINEFORM6 in the reversed direction and outputs INLINEFORM7 . We obtain the hidden state for a given word INLINEFORM8 by concatenating the forward and backward hidden states, i.e., INLINEFORM9 . Then, we concatenate the last hidden states of the forward and backward passes as the embedding representation of the paragraph INLINEFORM10 , denoted as INLINEFORM11 . A paragraph encoder is used to sequentially receive the embeddings of paragraphs INLINEFORM12 in a similar way. The hidden representation of each paragraph is given by INLINEFORM13 , where INLINEFORM14 and INLINEFORM15 are the forward and backward hidden states of the paragraph encoder respectively.

Hierarchical Decoder

The goal of the hierarchical decoder is to produce an outline for an input article, which could be decomposed into two dependent steps: (1) Section Boundary Prediction: to predict a sequence of section boundary labels over the paragraphs; and (2) Section Heading Generation: to generate the section heading for a newly detected section.

This step is to break up a multi-paragraph document INLINEFORM0 into multiple successive sections INLINEFORM1 by predicting the section boundary labels INLINEFORM2 , where INLINEFORM3 . If INLINEFORM4 , INLINEFORM5 is the inner paragraph of a section and the section prediction continues. If INLINEFORM6 , INLINEFORM7 is the last paragraph of one section and the corresponding heading should be generated. Note that a section is a sequence of information coherent paragraphs, while the coherence modeling is non-trivial in nature. In this paper, we introduce a Markov paragraph dependency mechanism for modeling the coherence between context paragraphs and identifying section boundaries.

[leftmargin=*]

Markov Paragraph Dependency Mechanism. The key assumption of the Markov paragraph dependency mechanism is that the coherence between paragraphs has a Markov property. Therefore, we can identify a section, i.e., to decide whether a target paragraph is a last paragraph of a section, by looking at its previous and successive paragraph. As shown in Figure FIGREF8 , we utilize the hidden representation of the current paragraph INLINEFORM0 , the previous paragraph INLINEFORM1 , and the next paragraph INLINEFORM2 to predict the section boundary label INLINEFORM3 . Specifically, the section boundary label INLINEFORM4 is modeled with binary output: DISPLAYFORM0 DISPLAYFORM1

where INLINEFORM0 stands for the sigmoid function, INLINEFORM1 , and INLINEFORM2 are learned parameters.

This step executes when a new section is detected, i.e., INLINEFORM0 . Based on the detected section INLINEFORM1 , to generate the heading INLINEFORM2 , we employ 1) a section-aware attention mechanism: maintaining a section-aware context vector to make sure more important content in the target section is attended; 2) a Markov heading dependency mechanism: maintaining the representation of the previously generated heading for new heading generation to improve the consistency between headings; and 3) a review mechanism: maintaining a heading-aware context vector to utilize contextual information of generated headings to eliminate duplication between headings. The first one is used to capture the coherence between a section and its heading, and the latter two are used to capture the coherence between context headings.

Afterwards, the section-aware context vector INLINEFORM0 and the heading-aware context vector INLINEFORM1 are provided as extra inputs to derive the hidden state INLINEFORM2 of the INLINEFORM3 -th word INLINEFORM4 in INLINEFORM5 , and later the probability distribution for

choosing the word INLINEFORM6 .

Concretely, INLINEFORM0 is defined as DISPLAYFORM0

where INLINEFORM0 is a GRU unit, INLINEFORM1 is the predicted word from vocabulary at INLINEFORM2 -th step when decoding the heading INLINEFORM3 . The probability distribution for choosing the word INLINEFORM4 is defined as DISPLAYFORM0

where INLINEFORM0 is a nonlinear function that computes the probability vector for all legal output words at each output time. We now describe the specific mechanism in the follows.

[leftmargin=*]

Section-Aware Attention Mechanism. The key idea of the section-aware attention mechanism is to make the generation of a section heading focusing on the target section. Concretely, as shown in Figure FIGREF21 , we maintain a section-aware context vector INLINEFORM0 for generating the INLINEFORM1 -th word INLINEFORM2 in the INLINEFORM3 -th heading INLINEFORM4 . Based on the INLINEFORM5 -th section INLINEFORM6 , INLINEFORM7 is a weighted sum of the hidden representations of all the paragraphs in INLINEFORM8 : DISPLAYFORM0

where INLINEFORM0 indicates how much the INLINEFORM1 -th paragraph INLINEFORM2 from the source section INLINEFORM3 contributes to generating the INLINEFORM4 -th word in target heading INLINEFORM5 , and is usually computed as: DISPLAYFORM0

where INLINEFORM0 represents the hidden state (just before emitting the INLINEFORM1 -th word INLINEFORM2 in INLINEFORM3 -th heading INLINEFORM4) of the decoder.

Markov Heading Dependency Mechanism. The headings in an outline should be consistent in style and it is necessary to capture the dependence between context headings. To achieve this purpose, we introduce a Markov heading dependency mechanism, for the section heading generation process. Note that different from the Markov paragraph dependency mechanism, the Markov heading dependency mechanism only looks at the previous generated heading since there is no successive heading generated yet.

Concretely, as shown in Figure FIGREF21 , the Markov heading dependency mechanism uses the accumulation of all the hidden states of the previous decoder and pass it to the next decoder. In this way, the generation of a new heading is decided by both the section content and the previous generated heading.

As we can see, the Markov heading dependency mechanism conveys strong dependency requirement between headings by involving all the previous states. The initial hidden state of the decoder INLINEFORM0 of heading INLINEFORM1 is the “mixture” of probabilities: DISPLAYFORM0

where INLINEFORM0 are learned parameters. INLINEFORM1 is the representation of paragraph INLINEFORM2 , where INLINEFORM3 is the last paragraph of the section INLINEFORM4 . The passed information INLINEFORM5 is the average of all the output states of the decoder for the heading INLINEFORM6 and defined as: DISPLAYFORM0

where INLINEFORM0 is the output state of the decoder for the heading INLINEFORM1 at the INLINEFORM2 -th step.

Review Mechanism. Headings should cover all topics in the source document and be exclusive to each other. To avoid duplicate generation, we incorporate a review mechanism BIBREF11 between context

headings as shown in Figure FIGREF21 . It models the correlation between the headings that have been generated and the heading that is going to be generated to generate a heading to cover topics that have not been summarized by previous headings.

Specifically, we construct a heading-aware review set which contains contextual information of generated headings. The heading-aware review set is defined as INLINEFORM0 , which is the collection of all the decoder hidden states before generating the INLINEFORM1 -th word INLINEFORM2 in the INLINEFORM3 -th heading INLINEFORM4 . When decoding the word INLINEFORM5 , the heading-aware review set INLINEFORM6 is integrated into the heading-aware context vector INLINEFORM7 : DISPLAYFORM0

where INLINEFORM0 indicated how much the INLINEFORM1 -word in the INLINEFORM2 -th heading contributed to generating the INLINEFORM3 -th word in target heading INLINEFORM4 , and is computed as: DISPLAYFORM0

where INLINEFORM0 is defined as DISPLAYFORM0

where INLINEFORM0 are learned parameters. The heading-aware review set gets updated consequently as INLINEFORM1 in the decoding process.

Model Training and Testing

In the training phase, we employ maximum likelihood estimation (MLE) to learn our HiStGen model in an end-to-end way. Specifically, the training objective is a probability over the training corpus INLINEFORM0 with decomposition into the ordered conditionals: DISPLAYFORM0

We apply stochastic gradient decent method Adam BIBREF37 to learn the model parameters INLINEFORM0 and INLINEFORM1 . Note that, during the training, we provide the model with the specific section boundary label INLINEFORM2 , and thus we do not have to sample.

In the testing phase, given a new multi-paragraph document, we compute Eqn. (EQREF19) and (EQREF20) to predict the section boundary label for each paragraph, and then pick the word with the highest probability using Eqn. (EQREF24) to generate the heading for each identified section.

Experiments

In this section, we conduct experiments to verify the effectiveness of our proposed model.

Experimental Settings

To evaluate the performance of our model, we conducted experiments on our WIKIOG benchmark dataset. In preprocessing, all the words in documents and headings are white-space tokenized and lower-cased, and pure digit words and non-English characters are removed. Beyond the three separate datasets (i.e., “celebrity”, “cities” and “music”), we also mix them together to form a “mixture” dataset. For each dataset in WIKIOG, we randomly divide it into a training set (80%), a development set (10%), and a test set (10%).

We construct two separate vocabularies for input documents and target headings by using 130000 and 16000 most frequent words on each side in the training data. All the other words outside the vocabularies are replaced by a special token INLINEFORM0 UNK INLINEFORM1 symbol. We implement our models in Tensorflow. Specifically, we use a bi-directional GRU for the word/paragraph encoder respectively and another forward GRU for the heading decoder, with the GRU hidden unit size set as 300 in both the

encoder and decoder. The dimension of word embeddings in documents and headings is 300. The learning rate of Adam algorithm is set as INLINEFORM2 . The learnable parameters (e.g., the parameters INLINEFORM3 , INLINEFORM4 and INLINEFORM5) are uniformly initialized in the range of INLINEFORM6 . The mini-batch size for the update is set as 64. We clip the gradient when its norm exceeds 5.

We run our model on a Tesla K80 GPU card, and we run the training for up to 12 epochs, which takes approximately two days. We select the model that achieves the lowest perplexity on the development set, and report results on the test set.

Baselines

Here, we first employ some degraded HiStGen models to investigate the effect of our proposed mechanisms, namely

[leftmargin=*]

HiStGen INLINEFORM0 removes the Markov paragraph dependency mechanism between context paragraphs, and the section boundary label is only decided by the representation of current paragraph.

HiStGen INLINEFORM0 removes the section-aware attention mechanism between a section and its heading.

HiStGen INLINEFORM0 removes the Markov heading dependency mechanism between context headings, and the initial hidden state of the decoder is only decided by the representation of last paragraph in the section.

HiStGen INLINEFORM0 removes the review mechanism between context headings.

HiStGen INLINEFORM0 removes all the mechanisms and reduces to a vanilla hierarchical sequence-to-sequence generation model.

We also apply two types of step-wise process for the OG task.

[leftmargin=*]

First-Identify-then-Generate (IG). The first step is to identify the potential sections, and the second step is to generate the heading for each section. For the section identification step, based on the hidden representations of the input paragraphs (described in Section SECREF15), we employ two methods:

[leftmargin=*]

Conditional random field (CRF) is a well-known sequential labeling model. Here we follow the work BIBREF38 where the CRF model is built upon the hierarchical encoder, and use the representation of the target paragraph and meanwhile take a chain dependence assumption between the labels, for section boundary prediction.

Global paragraph dependency mechanism (GPD) considers all the context paragraphs in a document, not just the previous and successive paragraph as in our Markov paragraph dependency mechanism, to predict the section boundary label for a target paragraph.

For the heading generation step, we employ both extractive (TextRank and TopicRank) and generative (Hier and GHD) methods over the detected sections:

[leftmargin=*]

TextRank BIBREF18 is a graph-based method inspired by the PageRank algorithm.

TopicRank BIBREF20 represents a document as a complete graph depending on a topical representation of the document.

Hier BIBREF36 takes the section as input using a hierarchical encoder structure (words form paragraph, paragraphs form section) and employs the section-aware attention (described in Section UID22) in the decoding phase.

GHD further employs a global heading dependency mech- anism based on the Hier, where all the previous generated headings are taken into account to initialize the hidden state of the current decoder, not just the previous one as in our Markov heading dependency mechanism.

By combining these two-step methods, we obtain eight types of IG methods denoted as IG INLINEFORM0 , IG INLINEFORM1 , IG INLINEFORM2 , IG INLINEFORM3 , IG INLINEFORM4 , IG INLINEFORM5 , IG INLINEFORM6 and IG INLINEFORM7 .

First-Generate-then-Aggregate (GA). The first step is to generate the heading for each paragraph, and the second step is to aggregate the paragraph with respect to their headings. For the heading generation step, we also employ the TextRank, TopicRank, Hier and GHD method over the paragraphs. For the heading aggregation step, we combine successive paragraphs with the same heading into one section. Similarly, we refer to these four types of GA process as GA INLINEFORM0 , GA INLINEFORM1 , GA INLINEFORM2 and GA INLINEFORM3 .

Evaluation Metrics

To measure the quality of outline generated by our model and the baselines, we employ three automatic metrics, namely

[leftmargin=*]

EM INLINEFORM0 : evaluates the overall accuracy of the generated outline based on exact matching. That is, if both the predicted section boundaries and the generated section headings in a document exactly match with the ground-truth, we treat the document as a positive sample. Otherwise the document is a negative sample.

EM INLINEFORM0 : evaluates the accuracy of the section boundary prediction based on exact matching. Namely, if the predicted section boundaries in a document exactly match with the ground-truth, we treat the document as a positive sample. Otherwise the document is a negative sample.

Rouge INLINEFORM0 evaluates the similarities between generated headings and referenced headings only for the correctly predicted sections. Specifically, we employ Rouge-1 BIBREF39 to measure the uni-gram recall on the reference headings.

Model Ablation

We conduct ablation analysis to investigate the effect of proposed mechanisms in our HiStGen model. As shown in table TABREF55 , we can observe that: (1) By removing the Markov paragraph dependence mechanism, the performance of INLINEFORM0 in terms of EM INLINEFORM1 has a significant drop as compared with INLINEFORM2 . The results indicate that modeling the dependency between adjacent

paragraphs does help decide the section boundaries. (2) INLINEFORM3 performs worse than INLINEFORM4 and INLINEFORM5 in terms of Rouge INLINEFORM6 , showing that the coherence between a section and its heading (captured by the section-aware attention mechanism) has much bigger impact than that between context headings (captured by the Markov heading dependency mechanism and review mechanism) for heading generation. (3) HiStGen INLINEFORM7 gives the worst performance, indicating that traditional seq2seq model without considering three-level coherence is not suitable for the OG task. (4) By including all the mechanisms, INLINEFORM8 achieves the best performance in terms of all the evaluation metrics.

Baseline Comparison

The overall performance comparisons between our HiStGen and the step-wise baselines are shown in Table TABREF61 . We have the following observations: (1) The INLINEFORM0 process (i.e., INLINEFORM1 , INLINEFORM2 , INLINEFORM3 and INLINEFORM4) performs very poorly. By looking at the results of the INLINEFORM5 methods, we find that INLINEFORM6 tends to segment the document into too much sections since it usually generates different headings even for paragraphs that should belong to a same section. (2) For the INLINEFORM7 process, the methods based on INLINEFORM8 perform better than that based on INLINEFORM9 . For example, the relative improvement of INLINEFORM10 over INLINEFORM11 is about INLINEFORM12 in terms of EM INLINEFORM13 on the mixture set. We analyze the results and find that using INLINEFORM14 can obtain better section prediction results, showing that the dependency on the context labels is more important than that on all the paragraphs for section identification. Moreover, for the INLINEFORM15 process, the generative methods can achieve significantly better results than the extractive methods, since those extractive methods are unsupervised in nature. (3) Our INLINEFORM16 model can outperform all the step-wise baselines significantly (p-value INLINEFORM17 0.01). As compared with the best-performing baseline INLINEFORM18 , the relative improvement of INLINEFORM19 over INLINEFORM20 is about

INLINEFORM21 in terms of EM INLINEFORM22 on the mixture set. The results demonstrate the effectiveness of our end-to-end learning model.

We further compare the section boundary prediction performance between our Markov paragraph dependency mechanism (MPD for short) and the two baseline methods, i.e., INLINEFORM0 and INLINEFORM1 , by keeping the rest components the same. The results are shown in Figure FIGREF65 . We can find that: (1) The improvements of INLINEFORM2 over INLINEFORM3 , showing that the consideration of the previous and successive paragraph is better than the consideration of all the paragraphs in a document for section boundary prediction. The reason might be by considering all the paragraphs, INLINEFORM4 tends to bring noisy information that may hurt the prediction on section boundaries. Moreover, INLINEFORM5 leads to much higher computing complexity than INLINEFORM6 (i.e., INLINEFORM7). (2) INLINEFORM8 performs better than INLINEFORM9 , demonstrating that depending on the semantic representations of the previous and successive paragraph is more beneficial than only depending on the labels of the previous and successive paragraph in section boundary prediction. All the improvements over the baselines are statistically significant (p-value $<$ 0.01).

We evaluate the section heading generation ability to demonstrate the effectiveness of our Markov heading dependency mechanism and review mechanism. Here we suppose that sections in an article are already given, and only need to predict the corresponding headings for each section. We consider two generative baselines INLINEFORM0 and INLINEFORM1 , where INLINEFORM2 is an extension of INLINEFORM3 by employing a global heading dependency mechanism. We then introduce our Markov heading dependency mechanism based on the INLINEFORM4 , named Hier INLINEFORM5 , and further employ the review mechanism, named Hier INLINEFORM6 . All these methods employ the section-aware attention in generation. The performance under Rouge INLINEFORM7 is shown in Table TABREF68 . We can find that: (1) Hier performs worst among all the methods, showing that the independence between context headings is not good for section heading generation. (2) By incorporating all the

previous generated headings to model the dependence between context headings, INLINEFORM8 shows slight improvements on the heading generation performance. It indicates that the global dependency may not be effective in heading generation by involving too much context information, and also leads to high computing complexity. (3) The improvements of INLINEFORM9 over INLINEFORM10 indicate that the dependency between adjacent headings is sufficient for generating good and consistent section headings. (4) The improvements of INLINEFORM11 over INLINEFORM12 demonstrate that the review mechanism is also helpful in improving the quality of section heading generation. All the improvements over the baselines are statistically significant (p-value INLINEFORM13 0.01).

Case Study

To better understand how different models perform, we conduct some case studies. We take one Wikipedia article from the “celebrity” test data as an example. As shown in Figure FIGREF62 , there are 15 paragraphs in this article, which are segmented into 7 sections. We show the identified sections and generated headings from our model as well as that from the baseline model INLINEFORM0 . We can find that: (1) The number of sections predicted by INLINEFORM1 is larger than the ground-truth (i.e., INLINEFORM2) and the segmentation is totally wrong. The results show that using current paragraph representation and context label dependency, CRF may not be able to make correct section boundary prediction. (2) Without considering the coherence between context headings, INLINEFORM3 generates repetitive headings (e.g., “career” repeats twice) and the heading with inconsistent style (e.g., “citizen political” is not suitable for the description of a celebrity). (3) Our INLINEFORM4 can generate right section boundaries and consistent headings. Note that INLINEFORM5 generates “family” for the third section whose true heading is “personal life”. As we look at that section, we found that “family” is actually a very proper heading and INLINEFORM6 did not generate the “personal life” as the heading possibly due to the review mechanism by avoiding partial duplication with the “early life” heading.

Conclusion and future work

In this paper we introduced a challenging OG task to unveil the inherent content structure of a multi-paragraph document by identifying its potential sections and generating the corresponding section headings. To tackle the problem, we formulated the OG task as a hierarchical structured prediction problem and developed a novel hierarchical structured neural generation model to capture the three levels of coherence. Furthermore, we built a new benchmark dataset WIKIOG to study and evaluate the OG task. The experimental results demonstrated that our model can well capture the inherent content structure of documents. In the future work, we would like to extend our model to produce hierarchical outlines for documents.

Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 61425016, 61722211, 61773362, and 61872338, the Youth Innovation Promotion Association CAS under Grants No. 20144310, and 2016102, the National Key R&D Program of China under Grants No. 2016QY02D0405, and the Foundation and Frontier Research Key Program of Chongqing Science and Technology Commission (No. cstc2017jcyjBX0059).