Gated Graph Neural Networks (GG-NNs) for Abstractive Multi-Comment Summarization

Huixin Zhan[§], Kun Zhang[†], Chenyi Hu[‡] and Victor S. Sheng[§*]
[§]Department of Computer Science, Texas Tech University
[†]Department of Computer Science, Xavier University of Louisiana
[‡]Department of Computer Science, University of Central Arkansas
Email: [§]{huixin.zhan,victor.sheng}@ttu.edu, [†]kzhang@xula.edu, [‡]chu@uca.edu
[*]Corresponding author: Victor S. Sheng, victor.sheng@ttu.edu

2021 IEEE International Conference on Big Knowledge (ICBK) | 978-1-6654-3858-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICKG52313.2021.00050

***Abstract*—Summarization of long sequences into a concise statement is a core problem in natural language processing, which requires a non-trivial understanding of the weakly structured text. Therefore, integrating crowdsourced multiple users' comments into a concise summary is even harder because (1) it requires transferring the weakly structured comments to structured knowledge. Besides, (2) the users comments are informal and noisy. In order to capture the long-distance relationships in staggered long sentences, we propose a neural multi-comment summarization (MCS) system that incorporates the sentence relationships via graph heuristics that utilize relation knowledge graphs, i.e., sentence relation graphs (SRG) and approximate discourse graphs (ADG). Motivated by the promising results of gated graph neural networks (GG-NNs) on highly structured data, we develop a GG-NNs with sequence encoder that incorporates SRG or ADG in order to capture the sentence relationships. Specifically, we employ the GG-NNs on both relation knowledge graphs, with the sentence embeddings as the input node features and the graph heuristics as the edges' weights. Through multiple layer-wise propagations, the GG-NNs generate the salience for each sentence from high-level hidden sentence features. Consequently, we use a greedy heuristic to extract salient users' comments while avoiding the noise in comments. The experimental results show that the proposed MCS improves the summarization performance both quantitatively and qualitatively.**

***Index Terms*—graph neural network, multi-comment summarization, graph data structure**

I. Introduction

The advent of crowdsourcing systems, such as Amazon Mechanical Turk [17, 32], Figure Eight (CrowdFlower) [25], etc., have created a variety of new opportunities for machine learning research [30, 35]. In truth inference, a dataset with repeated crowdsourced labels is usually fed into these algorithms to obtain an integrated label for each instance before building a prediction model. This truth inference procedure increases the data quality and the quality of the subsequent models. Currently, with the prosperity of Web 2.0, users can freely provide their comments or reviews for any product and service. For example, there are 280 global reviews/comments[1] for the book titled "Data Science for Business" on Amazon[2]. It is difficult for the users to read all comments to make buying decisions. To reduce users' workload of reading these comments, we aim at summarizing these comments together to generate a summary. Summarizing multiple comments to a single summary is much more difficult than integrating multiple labels into one concise label in crowdsourcing. Comparing with the truth inference algorithms, it is closer to document summarization in the natural language processing domain.

Efforts on document summarization can be categorized into extractive and abstractive methods. Extractive methods produce the summary of a document by extracting sentences from the original document while abstractive summarization generates new summaries with the use of arbitrary words and expressions via the encoder-decoder framework [31]. Recently, in the abstractive document summarization domain, in order to capture the structural information in text, a line of algorithms enable graph-structured input by extending the existing sequence encoders with graph neural networks (GNNs) [3, 4, 14, 33]. Note that these methods only apply the graph component to the encoders and the decoders remain unchanged from the typical recurrent neural networks [34]. Among them, gated graph neural networks (GG-NNs) [14] achieve better results when comparing with the typical GNNs, where GG-NNs introduce gated recurrent units (GRUs) [11] into typical GNNs. The advantage of the GG-NNs is that it introduces favorable inductive biases for long sequence outputs, compared to purely sequence-based models when a problem is graph-structured. However, there are arguments about the promising results of GG-NNs are only on highly-structured data [14, 21]. These methods may not be useful and result in lost sentence relationships in staggered long sentences. The reasons are, (1) due to the nature of the weakly structured data, models heavily rely on well-designed features at the word level or take advantage of other large, manually annotated datasets [5], and (2) all sentences in the same collection of documents are processed independently most of the time. Therefore, in our work, in order to capture the long-distance relationships in staggered long sentences, GG-NNs with sequence encoders on two relation knowledge graphs, i.e., sentence relation graphs (SRG) and approximate discourse graphs (ADG), is developed for our neural multi-

[1]In this paper, "comment" and "review" are exchangeable.
[2] https://www.amazon.com/Data-Science-Business-Data-Analytic-Thinking/product-reviews/1449361323/ref=cm_cr_dp_d_show_all_btm?ie=UTF8&reviewerType=all_reviews

978-1-6654-3858-2/21/$31.00 ©2021 IEEE
DOI 10.1109/ICKG52313.2021.00050

comment summarization (MCS) system. Therefore, our work demonstrates the importance of considering either sentence relations or discourse relations among sentences in multi-comment summarization.

Briefly, we first explore GG-NNs with sequence encoders on two different relation knowledge graphs. We then apply the greedy heuristic to extract salient users comments while avoiding the noise in comments. The SRG is constructed by sentence relations and the ADG can construct edges between sentences by counting discourse relation indicators such as discourse markers and co-referent mentions. The sentence embeddings as the input features and the graph heuristics as the edges' weights are cooperated to produce the final sentence salience estimations. Consequently, we use a greedy heuristic to extract salient sentences while avoiding the noise in comments.

Besides, since the users' comments are informal and noisy, the summarization task is more challenging because more noises occur on real-world multi-comment summarization datasets. An example about the noise (inconsistency and redundancy) in a real-world dataset is shown in Figure 1, where all good comments are in blue and one bad comment is in red. When users' opinions differ, our MCS needs to generate the summary based on the salience of the sentences. Therefore, our MCS focuses on breaking down the real-world summarization task into salience estimation and salience selection using the greedy heuristic.

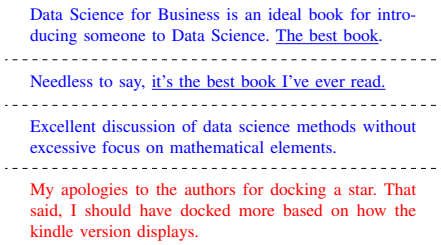


Fig. 1: An example about the noise in a real-world dataset, where all good comments are in blue and one bad comment is in red shows the inconsistency. The underlined text shows the redundancy.

The major contributions of this work are as follows:

- GG-NNs with sequence encoders are developed in order to capture the long-distance relationships in the relation knowledge graphs, such as SRG and ADG.
- The MCS breaks down the real-world summarization task into salience estimation and salience selection. A greedy heuristic is proposed to extract salient users' comments while avoiding the noise in comments.
- The multi-comment summarization is evaluated on a noisy real-world dataset and achieves the best performance among existing works both quantitatively and qualitatively.

II. Related Works

In this section, we summarize the related works from three aspects, including knowledge graph construction, graph-based document summarization, and graph-based multi-comment summarization (in noisy real-world datasets).

A. Knowledge Graph Construction

There are different types of graph construction methods for multiple documents. A typical one is the sentence relation graph adherer to the standard of cosine similarity. Besides, a G-Flow system was proposed to utilize discourse relationships between sentences to create its graph representations, known as Approximate Discourse Graph (ADG) [10]. Features in ADG reflect more diverse information about documents because ADG allows characterizing sentence relationships, rather than simply their similarity. In paper [10], the ADGs are constructed on Amazon Mechanical Turk via human annotators. However, in our work, the ADGs have constructed automatically, and we will show that the graph heuristics for the automatically constructed ADGs are effective enough for the summarization performance.

B. Graph-based Document Summarization

Graph-based document summarization models have traditionally employed surface level [12] or deep level [24] approaches based on topological features and the number of nodes [1]. Efforts have been made to improve the decision making of these systems by using discourse relationships between sentences [26]. For example, Radev et al. (2004) [26] proposed summarizing multiple documents based on Centroids. Erkan and Radev (2004) [12] introduces LexRank to compute sentence importance based on the eigenvector centrality in the connectivity graph of the inter-sentence cosine similarity. Christensen et al. (2013) [10] builds multi-document graphs to identify pairwise ordering constraints over the sentences by accounting for discourse relationships between sentences. In our work, we build MCS on top of both SRG and ADG. Recently, due to the large-scale datasets, a summarization system with attention-based encoder-decoder RNNs to sequentially label summary-worth sentences in single documents is trained in extracting sentences and words [8]. See et al. (2017) [29] then augments the standard attention-based encoder-decoder RNNs to keep track of what has been summarized with the ability to copy words from the source text via pointing. These models [8, 29] achieved a state-of-the-art performance on the DUC 2002 document summarization task [13]. However, scaling up these RNN sequence-to-sequence approaches to integrating multi-document with long distance relations is hard because (1) the necessity of training a computationally expensive sequence-to-sequence model on customized large multi-document summarization datasets, and 2) the inadequacy of RNNs to capture the long distance relationships across multiple documents. Recently, GRU+GCN [33] and SemSentSum [3] are proposed to learn the saliences for sentences for multi-document summarization. NLSUMMARIZATION [14] is developed to extend existing

sequence encoders with a graph component that can reason about long-distance relationships. However, these methods still cannot compensate for the noise in real-world datasets such as redundant or different opinions from users. Our proposed MCS system resolves these issues by breaking down the real-world summarization task into salience estimation and salience selection, which do not require training the computationally expensive sequence-to-sequence models extensively. We would also like the authors to notice that RASG [15] is an approach that deals with comments that are informal and noisy. However, our proposed MCS performs better by leveraging the graph heuristics for both SRG and ADG.

C. Graph-based Multi-comment Summarization

In a crowdsourcing scenario, individuals or organizations obtain goods and services from a large, relatively open and often rapidly evolving group of internet users [36]. In this paper, we aim at integrating multiple comments for any products or services, which are posted by participants (customers) with high inconsistency and redundancy. It is obvious that integrating such multiple comments together is a challenging problem. According to our knowledge, only one article focuses on this problem [27] and there is no existing graph-based works address this problem.

Inspired by the promising results of Graph Neural Networks [28] on highly structured data, our proposed MCS generates the salience for each sentence from high-level hidden sentence features from GG-NNs with sequence encoders. We then use a greedy heuristic to extract salient users' comments while avoiding the noise in comments.

III. Method

In order to summarize multi-comment, we propose an extension of GG-NNs that allows us to leverage known (or inferred) relationships among sentences for the input text. To achieve that, we combine sequence encoders with GG-NNs. We first construct the SRG or ADG using multi-comment (See subsection III-A for details). After this, we utilize a standard GG-NNs to obtain a per-sentence representation h_v^1 , and then we propogate it for t timesteps. The resulting per-node (i.e. per-sentence) representations h_v^t can be used for the GRU. The salience is then estimated from the output embedding for the user's comment. Figure 2 illustrates the

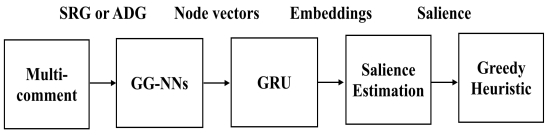


Fig. 2: Illustration of our MCS system for users comments' salience estimation.

MCS system. In this example, the GG-NNs take SRG or ADG, and outputs high-level hidden features for individual sentences. The GRU produces the cluster embedding for all comments by averaging all sentence embeddings for all comments within one product. Then the salience score is estimated from the sentence embeddings and the cluster embedding.

A. Knowledge graph construction

In our method, two kinds of relation knowledge graphs are constructed as following:

a) Sentence Relation Graphs (SRG): To best evaluate the architecture, we consider modeling sentence relationships within multi-comment. Since prior methods on representing document clusters often adhere to the standard of cosine similarity [33], our initial baseline approach naturally use this representation. Specifically, we add an edge between two sentences if the tf-idf cosine similarity based on the bag-of-words model [37] is above a threshold (0.2) [33]. Note that in our proposed SRG, all edges are considered as bi-directional.

b) Approximate Discourse Graph (ADG): The G-Flow system [10] utilizes discourse relationships between sentences to create its graph representations, known as Approximate Discourse Graph (ADG). The ADG constructs edges between sentences by counting discourse relation indicators such as discourse markers and co-referent mentions. These features provide different characteristics of sentence relationships, rather than simply compute their similarity. Because the weights are discretely incremented, they are multiples of 0.5 [10] in our later experiment. Note that ADG is a directed graph. We weight each edge in the ADG by adding the number of distinct indicators used to construct that edge. For example, if node annotations x_s and $x_{s'}$ have an edge because of a discourse marker and a co-referent mention, the edge weight 0.5 will be doubled.

Figure 3 shows an example of an ADG with each node presenting a sentence. First, a black edge from v_i to v_j indicates that v_j can be placed right after v_i in a coherent integration. In other words, these two nodes share a discourse relationship. For example, "Bombing in Jerusalem" and "Palestinians condemn attack" maintain this relationship. Our indicator is related to lexical chains [10]. We add a red dot edge in the ADG from a sentence v_i to v_j if they contain a co-referent mention (shown in yellow text with underline) and the timestamp of v_i is less than or equal to the timestamp of v_j (timestamps are generated via [6]). We construct the ADG based on the Stanfords coreference system [20]. For discourse markers, such as "therefore", a blue dash edge $(v_i; v_j)$ can be linked. Specifically, we are able to identify a syntactic edge between d_1s_3 and d_4s_1, where $d_{DocumentID}s_{SentenceID}$ denotes the $DocumentID$-th document and the $SentenceID$-th sentence. We weight each edge in the ADG by adding the number of textual cues used to construct that edge. For example, if sentences v_i and v_j have two overlapped edges (e.g., d_1s_3 and d_4s_1), the edge weight will be doubled.

B. GG-NNs

We now describe gated graph neural networks (GG-NNs) [21] used in our MCS system. This method unrolls the recurrence for a fixed number of steps T and uses back-propagation through for computing the gradients.

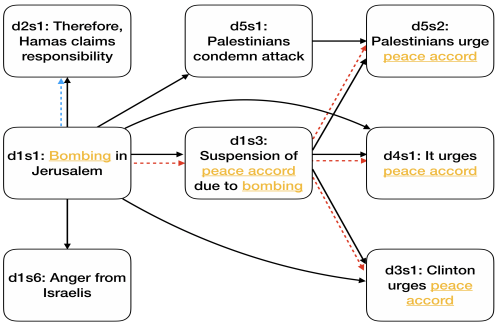


Fig. 3: An example of an ADG with the sentences for each node. Sentences are truncated.

1) Node Annotations: In GG-NNs, we will incorporate node labels as additional inputs. We denote these node labels as node annotations x. Because each sentence represents each node v, we can encode each node annotation x_v with the embedding of the sentence via Doc2Vec [19]. We then initialize the node state vectors h_v^1 using these label vectors by copying x_v into the first dimensions and padding with extra 0s to allow hidden states that are larger than the annotation size.

2) Propagation Model: The basic recurrence of the propagation model is

$$h_v^1 = [x_v^{\mathbb{T}}, 0]^{\mathbb{T}} \tag{1}$$

$$a_v^t = A_v^{\mathbb{T}}[h_1^{(t-1)\mathbb{T}}, \cdots, h_{|V|}^{(t-1)\mathbb{T}}]^{\mathbb{T}} + b \tag{2}$$

$$z_v^t = \sigma(W^z a_v^t + U^z h_v^{t-1}) \tag{3}$$

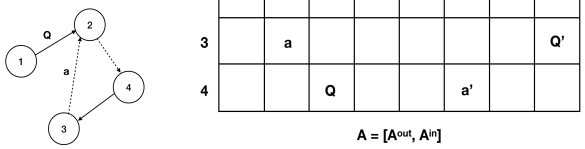
$$r_v^t = \sigma(W^r a_v^t + U^r h_v^{t-1}) \tag{4}$$

$$\widetilde{h_v^t} = \tanh(W a_v^t + U(r_v^t \circ h_v^{t-1})) \tag{5}$$

$$h_v^t = (1 - z_v^t) \circ h_v^{t-1} + z_v^t \circ \widetilde{h_v^{t-1}}, \tag{6}$$

where z_v^t is the latent embedding for the i-th sentence, h_v^t is the hidden state of the GG-NNs at t-th timestep, W, W^r, W^z, U, U^r, U^z are learnable parameters, $\mathbb{T}$ is the transpose of the matrix, and $\circ$ is the product operation. Eq. 1 is the initialization step, which copies node annotations into the first components of the hidden state and pads the rest with zeros. Eq. 2 is the step that passes information between different nodes of the graph via incoming and outgoing edges with parameters dependent on the edge type and direction. $A_v \in \mathbb{R}^{D|V| \times 2D}$ are the two columns of blocks in A^{out} and A^{in} corresponding to node $v \in V$, where V denoted all nodes and $|V|$ is the number of nodes (sentences) in the relation knowledge graph. In Eq.3, $a_v^t \in \mathbb{R}^{2D}$ contains activations from edges in both directions, where D is the dimension of h_v^t. The remaining equations are GRU-like updates that incorporate information from the other nodes and from the previous timestep to update each nodes hidden state. z and r are the update and reset gates respectively. $\sigma(\bullet) = \frac{1}{(1+e^{-\bullet})}$ is the logistic sigmoid function, and $\circ$ is element-wise multiplication.

The matrix $A \in \mathbb{R}^{D|V| \times 2D|V|}$ determines how nodes in the graph communicate with each other. For the reachability issue, it is easy for the propagation model to learn the propagation from node annotation x_s to all nodes reachable from x_s. For example, set the propagation matrix associated with forward edges to its weight for the corresponding entry for A^{out}. This will cause the node annotation x_s to be copied along forward edges. With this setting of parameters, the propagation step will cause all nodes reachable from x_s to gain the value from x_s. Figure 4a shows an example graph. The graph heuristics and the parameter tying in A is illustrated in Figure 4b. The graph heuristics are correspond to the edges of the graph, and the parameters are determined by the edge weights.



(a) An example graph, Q and a denote different edge weights.

(b) Parameter tying and sparsity in recurrent matrix A. Q and a denote weights caused by the outgoing edges, and Q' and a' denote weights caused by the incoming edges. In SRG, $Q' = Q$ and $a' = a$.

Fig. 4: An example relation knowledge graph (can be either SRG or ADG).

3) Cluster Embedding: In order to gain a global view of the entire comments for one product or service, we define a cluster as all users' comments within this product. We here introduce how do we compute the cluster embedding. We first extract the last hidden state h_v^T in sentence embedding as the initial i-th users' embedding for u^i for node v for timestep T. We then apply an RNN, i.e. GRU^{user}, to encode the entire cluster (all comments within one product). Given a cluster C with comments for all users. The i-th user's comment is denoted as $u^i = \{h_1^T, \cdots, h_M^T\}$ with M sentences. All sentence embeddings are grouped as the node feature matrix H:

$$H = [h_1^T, h_2^T, \cdots, h_M^T]^{\mathbb{T}}. \tag{7}$$

Here we abuse the notation a little bit. Each h_v^T is user-specific, and can be reflected as $h_v^{T,i}$ in practice. We denote it as h_j^T from now for simplification purposes, where j is the index of sentences for the i-th user's comment and $j \in \{1, \cdots, M\}$. The GRU^{user} first builds the clusters embedding $\mathbf{C}_e$ on top of sentence embeddings h_j^T. We compute every j-th sentence embedding from the $(j-1)$-th sentence embedding when $j > 1$:

$$u_j^i = GRU^{user}(u_{j-1}^i, h_j^T) \tag{8}$$

The cluster embedding $\mathbf{C}_e$ can be computed via average over all embeddings:

$$\mathbf{C}_e = \frac{1}{M}\frac{1}{I}\sum_{i}^{I}\sum_{j=1}^{M} u_j^i, \tag{9}$$

where I denotes the total number of users (comments).

All the GRUs we used are forward. Note that we also experimented with backward GRUs and bi-directional GRUs, but neither of them perform better than forward GRUs.

4) Salience Estimation: For the sentence h_j^T in the cluster C, we calculate the salience of h_j^T as the following, similarly to the attention mechanism in neural machine translation [9]:

$$f(h_j^T) = \sigma(nn(\mathbf{C}_e, h_j^T, x_j)) \tag{10}$$

$$salience(h_j^T) = \frac{f(h_j^T)}{\sum_{h_v^T \in C} f(h_v^T)}, \tag{11}$$

where $\sigma(nn(h_j^T, x_j))$ acts as a soft attention mechanism that decides which nodes are relevant to the current graph-level task. nn is a neural network that take the concatenation of h_j^T and x_j as input and outputs real-valued vectors. In Eq. 10, we first calculate the score $f(h_j^T)$ by considering the sentence embedding itself, h_j^T, and the cluster embedding $\mathbf{C}_e$ for the global context of the multiple users' documents. The score is then normalized as $salience(h_j^T)$ via softmax in Eq. 11.

5) Training: The model is then trained end-to-end to minimize the following cross-entropy loss between the salience estimation results and the normalized ROUGE score of each sentence:

$$\mathcal{L} = -\sum_{C}\sum_{h_j^T \in C} R(h_j^T)\log(salience(h_j^T)), \tag{12}$$

where $R(h_j^T)$ is calculated by $R(h_j^T) = softmax(\alpha r(h_j^T))$, and $r(h_j^T)$ denotes the average of ROUGE-1 and ROUCE-2. α is a constant rescaling factor to make the distribution sharper.

6) Salience Selection: Given the salience score estimation, we apply a simple greedy procedure to select sentences. Sentences with higher salience scores have higher priorities to be selected. First, we sort sentences in descending order of the salience scores. Then, we select one sentence from the top of the list and append to the summary if the sentence is of reasonable length (3- 55 words) and is not redundant. The sentence is redundant if the tf-idf cosine similarity between the sentence and the current summary is above 0.5. We select sentences this way until we reach the length limit (200 words).

IV. Experiments

A. Dataset and Evaluation

We investigate the performance of our approaches and baselines on a real-world dataset SSECIF 200. SSECIF 200 contains comments on 200 books from Amazon[3]. Specifically, for each book, comments from 10 participants, yet with different lengths, have been packaged as a "source document". Each source document has an expert-integrated comments. The human summaries have gone through three rounds of verification by professional researchers who have expertise in natural language processing. We utilize 80% data for training, 10% data for validation, and 10% data for testing.

[3]https://www.amazon.com/

B. Implementation Details

For the experiments with both relation knowledge graphs, we tokenize all the comments into sentences and construct the graphs by two methods: SRG and ADG. Stanford CoreNLP (version 3.9.1) [23] is used to tokenize the text and provide the resulting tokens to the encoder. Each sentence is mapped to a node, and each relation is mapped to an edge with an edge weight given by the relation. Each source document is consumed and mapped to a single graph. Processing large graphs of different shapes efficiently requires to overcome some engineering challenges. For example, the SSECIF 200 has (on average) about 100 nodes per graph. To allow efficient computation, we use the trick of [2], where all graphs in a minibatch are flattened into a single graph with multiple disconnected components. For evaluation, we use the ROUGE score metrics [22], with stemming and stop words not removed.

Besides, we used GG-NNs with the size of a node vector h_v^t set to $D = 128$ and two graph convolutional hidden layers ($L = 2$). The hidden states in GRU^{user} are all 64 dimensional vectors. The rescaling factor α is chosen as 50 from $\{10, 20, 30, 40, 50, 100\}$ based on the validation performance. We additionally perform an experiment with the model of See et al. (2017) [29] (as implemented in OpenNMT [18]) but using our parameters. The objective function is optimized using Adam [16] stochastic gradient descent with a learning rate of 0.001 and a batch size of 1. The model is validated every 10 iterations, and the training is stopped early if the validation performance does not improve for 10 consecutive steps.

C. Quantitive Evaluation

We show the quantitive evaluation results in Table I. Across all tasks, the results show the advantage of our MCS system.

TABLE I: Evaluation results for SRG and ADG, respectively. The results fo our proposed method are in bold.

SRG	ROUGE-1	ROUGE-2	ROUGE-L
(BiLSTM) + (LSTM)	52.3	31.4	47.9
(BiLSTM + GG-NNs) + (LSTM)	53.4	33.3	48.3
(BiLSTM + GG-NNs) + (LSTM + GRU)	**56.0**	**35.4**	**49.5**
See et al. (2017) + (Pointer)	54.5	35.7	43.4
See et al. (2017) + (Pointer + GRU)	**58.2**	**37.2**	**47.4**
(BiLSTM) + (LSTM + Pointer)	55.9	33.9	40.3
(BiLSTM + GG-NNs) + (LSTM + Pointer)	58.7	36.1	43.2
(BiLSTM + GG-NNs) + (LSTM + Pointer + GRU)	**62.8**	**38.2**	**47.3**
See et al. (2017) + (Pointer + Coverage)	59.5	36.7	46.4
See et al. (2017) + (Pointer + Coverage + GRU)	**63.6**	**39.9**	**49.2**
ADG			
(BiLSTM) + (LSTM)	54.9	31.5	48.2
(BiLSTM + GG-NNs) + (LSTM)	54.7	33.7	48.5
(BiLSTM + GG-NNs) + (LSTM + GRU)	**58.3**	**36.9**	**50.8**
See et al. (2017) + (Pointer) [29]	56.9	36.3	43.5
See et al. (2017) + (Pointer + GRU)	**57.6**	**38.0**	**49.6**
(BiLSTM) + (LSTM + Pointer)	57.4	35.8	41.5
(BiLSTM + GG-NNs) + (LSTM + Pointer)	59.1	36.9	44.0
(BiLSTM + GG-NNs) + (LSTM + Pointer + GRU)	**64.8**	**39.5**	**48.4**
See et al. (2017) + (Pointer + Coverage) [29]	59.8	37.5	46.8
See et al. (2017) + (Pointer + Coverage + GRU)	**66.0**	**40.0**	**50.9**

In both SRG and ADG, we can see that all GG-NNs augmented models are able to strongly outperform the typical encoder-decoder methods for abstractive summarization, such as (BiLSTM) + (LSTM) and (BiLSTM) + (LSTM+ Pointer). Our method also outperforms the state-of-the-art approach, i.e. See et al. (2017) + (Pointer + Coverage) [29] by leveraging the sentence relationships in SRG and ADG. The performance results in terms of different encoder and decoder configurations nicely show that their effects are mostly orthogonal. Furthermore, the addition of ADG gives a slightly better performance than the typical SRG.

TABLE II: Comparing our method with conventional multi-document summarizers. The result for our proposed method is in bold.

	ROUGE-1	ROUGE-2
Centroid [26]	34.9	25.0
LexRank [12]	38.1	24.8
G-Flow [10]	38.0	25.4
GRU+GCN [33]	40.2	27.4
SemSentSum[3]	42.4	27.2
NLSUMMARIZATION [14]	47.9	30.8
RASG [15]	60.4	37.0
(BiLSTM + GG-NNs) + (LSTM + Pointer + GRU)	**66.0**	**40.0**

To gain a global view of the performance of our approach, we also compare our approaches with other baseline multi-document summarizers. In ADG, as our model uses a standard sequence decoder, we do not expect it outperforms more recent models that introduce substantial novelty in the structure or training objective of the decoder [7]. However, as shown in Table II, with more fine-grained counting discourse relation indicators and the greedy heuristics for reducing the noise for multi-comment, we observe that our MCS system significantly outperforms the commonly used baselines and traditional graph approaches such as Centroid, LexRank, and G-Flow. This indicates the advantage of the combinatorial GG-NNs with sequence encoder and GRU used in our model. Our system also exceeds the state-of-the-art multi-document summarizers, i.e., NLSUMMARIZATION, and RASG, in terms of ROUGE-1 and ROUGE-2.

D. Salience Estimation

1) Node Degree and Salience: In order to analyze the salience score for multi-comment, We first compute the characteristics of both relation knowledge graphs by averaging the document clusters. Table III summarizes the following basic statistics: the number of nodes (i.e. the number of sentences), the number of edges, the average edge weight, the average node degree per graph, and the corresponding correlation coefficient as computed in paper [33].

As seen from Table III, the SRG has slightly fewer edges than ADG does. As shown in Figure 5a and Figure 5c, we choose the matrix A from the first cluster of the SSECIF 200 to illustrate this phenomenon. Moreover, the ADG has a significantly higher average edge weights and node degrees as compared to those of SRG. These values reflect the discrete nature of the ADGs edge assignment. Because the ADGs raw edge weight assignment is done by increments of 0.5, the average node degree tends to be significantly large. This

TABLE III: Characteristics of both graph heuristics, averaged over all clusters (i.e. graphs) in SSECIF 200. Note that max edge weight in the sentence relation graph is 1.0 and the weights in ADG are multiples of 0.5. The degree of each node is calculated as the sum of edge weights.

	SRG	ADG
Number of nodes	100	100
Number of edges	996	1204
Average edge weight	0.274	2.8
Average node degree	7.640	54.905
Correlation coefficient of degree and salience	0.45	0.69

phenomenon helps the GG-NNs in identifying the most important edge connections along with the affiliated sentences. As a case study to illustrate this observation, we again chose the cluster $\#1$ from the SSECIF 200. Figure 5b and Figure 5d show the scatter plots of the node degree and salience score of each sentence. We observe that all the relation knowledge graphs show positive correlation between the node degree and the salience score. Moreover, the correlation strength of ADG is higher than that of the SRG. Though node degree is a simple measure of these graphs, this observation supports our hypothesis on the efficacy of ADG and provides a guide to salience estimation.

E. Ablation Study

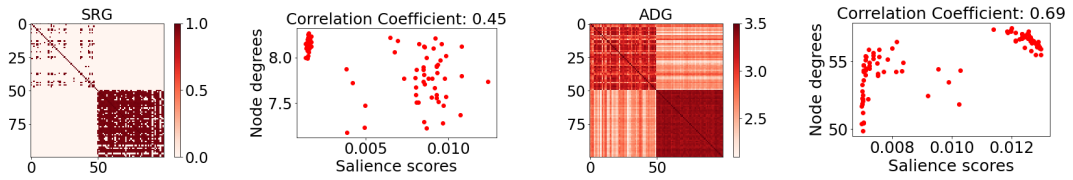
In this section, we will investigate how much additional textual cues provided by ADG help. Our experimental results are shown in Table IV. First, we add the discourse markers to the baseline (BiLSTM + GG-NNs) + (LSTM + Pointer) model (by extending the embedding of tokens with an embedding of the entity information) and observe only minimal improvements. This suggests that our GG-NNs structured encoder is better-suited to exploit additional structured information compared to a (BiLSTM+GG-NNs) encoder. We then add long-range dependency edges by inserting the discourse markers (dm) and the co-referent mention (cr). These additions improve the performance of the baseline (BiLSTM + GG-NNs) + (LSTM + Pointer) model (especially in terms of the ROUGE-2 score). This suggests that the GG-NNs can be further improved through combining the syntactical long-range dependency information.

TABLE IV: Evaluation results for ADG.

ADG	ROUGE-1	ROUGE-2	ROUGE-L
(BiLSTM + GG-NNs) + (LSTM + Pointer $w.o.$ dm, cr)	57.0	34.5	43.0
(BiLSTM + GG-NNs) + (LSTM + Pointer)(+ dm)	57.8	34.9	43.2
(BiLSTM + GG-NNs) + (LSTM + Pointer)(+ dm + cr)	59.1	36.9	44.0

F. Qualitative Evaluation

We look at a few sample suggestions in our dataset in ADG. Here we highlight some observations to point out interesting aspects of our model. The following text shows one sample summarization. In our proposed method, the final summary is rephrased from several segments in blue in two original documents, shown in Figure 6. We can notice that the model produces natural-looking summary with no noticeable negative impact on the fluency of the language over all comparing methods. Furthermore, the GG-NNs based model seems to



(a) A in SRG of cluster #1 (b) The node degree and salience score in SRG of cluster #1 (c) A in ADG of cluster #1 (d) The node degree and salience score in ADG of cluster #1

Fig. 5: The results for salience estimation

Original Comment #4 (truncated): I was impressed to see this is a debut novel. I read it over a twelve hour period and when I had to put the book down, I could not could not quit thinking about it. The ending was a little drawn out when the storylines are pulled altogether.

Original Comment #8 (truncated): At first, I was not sure if this would be a fantasy or literary fiction but in the end, the story was told well and made me think about it, which is a well-written novel worth five stars.

(BiLSTM) + (LSTM): I read it over a twelve hour period and I could not stop thinking about it. when the storylines are pulled altogether. The UNK UNK is told well and made me think about it.

(BiLSTM + GG-NNs) + (LSTM + Pointer): When I put the book down, iI could not stop thinking about it. The end is drawn out when the storylines are pulled altogether. The storyline is told well and imade me think about it.

(BiLSTM + GG-NNs) + (LSTM + Pointer + GRU) : In the end, the story pulled altogether and made me think about it, which is a well-written novel worth five stars.

Fig. 6: Comparisons of the outputs of three abstractive models for multi-comment summarization. The (BiLSTM) + (LSTM) model makes factual errors, such as a nonsensical sentence and struggles with OOV words. The (BiLSTM + GG-NNs) + (LSTM + Pointer) model is accurate but repeats itself. The final summary is composed from several sentences.

capture the central sentence in the article and creates a summary centered around that central sentence such as "it made me think about it". We hypothesize that the ADG weights that link long distance relationships help capture and maintain a better "global" view of the article, allowing for a better identification of central sentences. Besides, our model does not suffer from repetition of information in most of the cases. When comparing our proposed method with other baselines, we can see that the (BiLSTM) + (LSTM) model makes factual errors, a nonsensical sentence and struggles with out of vocabulary words (marked in red). The (BiLSTM + GG-NNs) + (LSTM + Pointer) model is accurate but repeats itself (marked in green). Our method eliminates repetition as shown in Figure 6.

V. Conclusion

In this paper, we presented a novel multi-comment summarization (MCS) system that exploits the representational graph structure of sentence relationships. We proposed a GG-NNs with sequence encoder for abstractive summarization and build an additional simple GRU on top of this structure to serve as a salience estimator. We applied this structure on both sentence relation graph (SRG) and approximate discourse graphs (ADG). Our model, unlike traditional sequential models or GNNs, can demonstrate improved summarization quality for both quantitive evaluation and qualitative evaluation. For quantitive evaluation, our method achieved a much better performance (e.g. 66.0 in terms of ROUGE-1 and 40.0 in terms of ROUGE-2) when comparing with the current state-of-the-art methods. For quantitive evaluation, we have validated that our method can leverage the relationships for long staggered sentences in weakly structured text. Besides, our final summary does not suffeer from redundancy.

Acknowledgement

Chenyi Hu is partially supported by US National Science Foundation through the grant award OIA 1946391.

References

[1] Réka Albert and Albert-László Barabási. Statistical mechanics of complex networks. *Reviews of modern physics*, 74(1):47, 2002.

[2] Miltiadis Allamanis, Earl T Barr, Premkumar Devanbu, and Charles Sutton. A survey of machine learning for big code and naturalness. *ACM Computing Surveys (CSUR)*, 51(4):1–37, 2018.

[3] Diego Antognini and Boi Faltings. Learning to create sentence semantic relation graphs for multi-document summarization. *arXiv preprint arXiv:1909.12231*, 2019.

[4] Mohit Bajaj, Lanjun Wang, and Leonid Sigal. G3raphground: Graph-based language grounding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4281–4290, 2019.

[5] Ziqiang Cao, Furu Wei, Sujian Li, Wenjie Li, Ming Zhou, and Houfeng Wang. Learning summary prior representation for extractive summarization. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 829–833, 2015.

[6] Angel X Chang and Christopher D Manning. Sutime: A library for recognizing and normalizing time expressions. In *Lrec*, volume 2012, pages 3735–3740, 2012.

[7] Yen-Chun Chen and Mohit Bansal. Fast abstractive summarization with reinforce-selected sentence rewriting. *arXiv preprint arXiv:1805.11080*, 2018.

[8] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252*, 2016.

[9] Jan K Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in Neural Information Processing Systems*, pages 577–585, 2015.

[10] Janara Christensen, Stephen Soderland, Oren Etzioni, et al. Towards coherent multi-document summarization. In *Proceedings of the 2013 Conference of the North American Chapter of The Association for Computational Linguistics: Human Language Technologies*, pages 1163–1173, 2013.

[11] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[12] Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479, 2004.

[13] Atefeh Farzindar, Guy Lapalme, and Horacio Saggion. Summaries with sumum and its expansion for document understanding conference (duc 2002). In *Workshop on Text Summarization*, 2002.

[14] Patrick Fernandes, Miltiadis Allamanis, and Marc Brockschmidt. Structured neural summarization. *arXiv preprint arXiv:1811.01824*, 2018.

[15] Shen Gao, Xiuying Chen, Piji Li, Zhaochun Ren, Lidong Bing, Dongyan Zhao, and Rui Yan. Abstractive text summarization by incorporating reader comments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6399–6406, 2019.

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

[17] Aniket Kittur, Ed H Chi, and Bongwon Suh. Crowdsourcing user studies with mechanical turk. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 453–456, 2008.

[18] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. Opennmt: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810*, 2017.

[19] Jey Han Lau and Timothy Baldwin. An empirical evaluation of doc2vec with practical insights into document embedding generation. *arXiv preprint arXiv:1607.05368*, 2016.

[20] Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Stanfords multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the 15th conference on computational natural language learning: Shared task*, pages 28–34. Association for Computational Linguistics, 2011.

[21] Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

[22] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004.

[23] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[24] Thiago Alexandre Salgueiro Pardo, Lucas Antiqueira, Maria das Graças Volpe Nunes, Osvaldo N Oliveira, and Luciano da Fontoura Costa. Modeling and evaluating summaries using complex networks. In *International Workshop on Computational Processing of the Portuguese Language*, pages 1–10. Springer, 2006.

[25] Lisa Posch, Arnim Bleier, Fabian Flöck, and Markus Strohmaier. Characterizing the global crowd workforce: A cross-country comparison of crowdworker demographics. *arXiv preprint arXiv:1812.05948*, 2018.

[26] Dragomir R Radev, Hongyan Jing, Małgorzata Styś, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing & Management*, 40(6):919–938, 2004.

[27] Huan Rong, Victor S Sheng, Tinghuai Ma, Yang Zhou, and Mznah A Al-Rodhaan. A self-play and sentiment-emphasized comment integration framework based on deep q-learning in a crowdsourcing scenario. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

[28] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.

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

[30] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622, 2008.

[31] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017.

[32] Amazon Mechanical Turk. Amazon mechanical turk. *Retrieved August*, 17:2012, 2012.

[33] Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. Graph-based neural multi-document summarization. *arXiv preprint arXiv:1706.06681*, 2017.

[34] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329*, 2014.

[35] Jing Zhang, Victor S Sheng, Jian Wu, and Xindong Wu. Multi-class ground truth inference in crowdsourcing with clustering. *IEEE Transactions on Knowledge and Data Engineering*, 28(4):1080–1085, 2015.

[36] Jing Zhang, Xindong Wu, and Victor S Sheng. Learning from crowdsourced labeled data: a survey. *Artificial Intelligence Review*, 46(4):543–576, 2016.

[37] Yin Zhang, Rong Jin, and Zhi-Hua Zhou. Understanding bag-of-words model: a statistical framework. *International Journal of Machine Learning and Cybernetics*, 1(1-4):43–52, 2010.