Event Graph based Sentence Fusion

Ruifeng Yuan *
The Hong Kong Polytechnic University
csryuan@comp.polyu.edu.hk

Zili Wang *
Xidian University
ziliwang.do@gmail.com

Wenjie Li
The Hong Kong Polytechnic University
cswjli@comp.polyu.edu.hk

Abstract

Sentence fusion is a conditional generation task that merges several related sentences into a coherent one, which can be deemed as a summary sentence. The importance of sentence fusion has long been recognized by communities in natural language generation, especially in text summarization. It remains challenging for a state-of-the-art neural abstractive summarization model to generate a well-integrated summary sentence. In this paper, we explore the effective sentence fusion method in the context of text summarization. We propose to build an event graph from the input sentences to effectively capture and organize related events in a structured way and use the constructed event graph to guide sentence fusion. In addition to make use of the attention over the content of sentences and graph nodes, we further develop a graph flow attention mechanism to control the fusion process via the graph structure. When evaluated on sentence fusion data built from two summarization datasets, CNN/DaliyMail and Multi-News, our model shows to achieve state-of-the-art performance in terms of Rouge and other metrics like fusion rate and faithfulness.

1 Introduction

Sentence fusion aims to combine several related sentences into a single coherent text. It is important in many NLP tasks such as text summarization, question answering and retrieval-based dialogue. In text summarization, it is a common practice for a proficient editor to fuse the information from several related sentences, however, it remains challenging for a state-of-the-art neural abstractive summarization model to achieve effective sentence fusion. As pointed out in (Lebanoff et al., 2019a), the human-written summaries contain 32% fusion sentences on the CNN/DailyMail dataset, while only 6% of the summary sentences generated by the Pointer-Generator model (See et al., 2017) are shown to fuse the information spread over sentences. Besides, without proper guidance, many sentences generated by fusion contain factual errors. Therefore, it is worthwhile to explore effective sentence fusion methods in the context of text summarization.

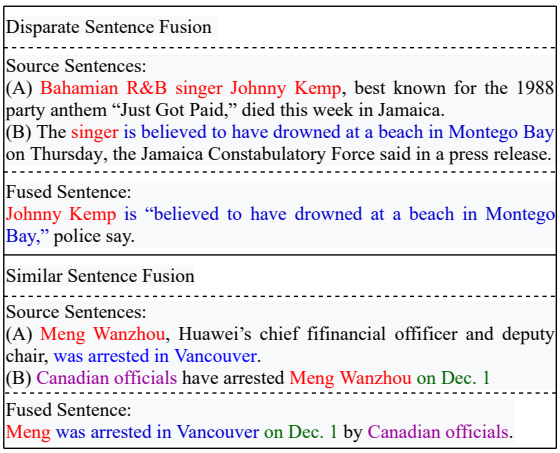

Figure 1: Examples of two types of sentence fusion in text summarization.

In fact, the importance of sentence fusion has long been recognized by researchers in the text summarization community. As shown in Figure 1, the researchers have been concerned with two types of sentence fusion task in the past. One is *similar* sentence fusion and the other one is *disparate* sentence fusion. For similar sentence fusion, a word graph or a dependency tree is often explored to find a coherent fusion path (Marsi and Krahmer, 2005; Filippova and Strube, 2008; Thadani and McKeown, 2013). For disparate sentence fusion, the coreference relations are typically considered as the key to tie the sentences together (Lebanoff et al., 2020b,a). Although both types of sentence fusion benefit text summarization, especially multi-document summarization, the solutions are rarely

[1]These authors contributed equally to this work.

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 4075–4084
November 7–11, 2021.

proposed to deal with the two types together. In this paper, we propose to apply the structured event information to guide the two types of sentence fusion in a unified framework.

We address the challenge of sentence fusion by building an *event graph* to capture the semantic relationships among the input sentences. The event graph is a directed graph composed of the nodes representing the predicate and event arguments and the edges that connect these event components together. Compared to the word graph or the dependency tree, the event graph provides more informative event-level (or to say entity-level) information. Meanwhile, it maintains the semantic integrity of each node, which allows us to add additional edges to represent some crucial relationships in disparate sentence fusion like co-reference. Such a structured representation is capable of preserving inherent event information and meanwhile formulating cross-sentence information such as entity interactions and proximity of relevant concepts.

With the target to guide sentence fusion, we develop a decoder that utilizes the information from both the sentence sequence and the event graph equipped with different attention mechanisms. We employ sequence attention and graph attention to determine what information is important to be select to generate the appropriate word token at each decoding step. Note that sentence fusion requires not only selecting the right salient information but also organizing the selected information logically and orderly. Otherwise, the models may tend to randomly combine the key event components or simply copy the most important text span. To this end, we develop a graph flow attention to explore potential fusion paths via the graph structure and control the fusion process. Moreover, how to avoid factual errors in a fused sentence is also a critical issue in sentence fusion. Inspired by (Scialom et al., 2020), we incorporate faithful beam search at the inference stage to reduce possible factual errors. This allows the model to remove the unfaithful candidate output sequence during the generation process by refining the generation probability with a faithful score.

Since there is no available dataset to evaluate the effectiveness of the sentence fusion models in the context of text summarization, following previous work (Lebanoff et al., 2020b), we automatically generate sentence fusion data from summarization datasets including CNN/DaliyMail (Hermann et al., 2015) and Multi-News (Fabbri et al., 2019). The experiments show that our proposed model indeed improves Rouges and the other metrics like faithfulness and the fusion rate. The contribution of our work can be summarized as follows:

(1) We propose a model to address both similar sentence fusion and disparate sentence fusion, which are critical for abstractive summarization.

(2) We build an event graph to guide sentence fusion, which allows our model to utilize the structural event information and various cross-sentence relations.

(3) We innovatively apply a graph flow attention to control the fusion process via the graph structure.

2 Related Work

2.1 Sentence Fusion in Text Summarization

Sentence fusion has been considered as an essential step for generating abstractive summaries. Its importance has long been recognized in the traditional text summarization research (Barzilay et al., 1999). The early attempts mainly focus on fusing a set of similar sentences (Marsi and Krahmer, 2005; Filippova and Strube, 2008; Elsner and Santhanam, 2011; Thadani and McKeown, 2013). They often build a dependency graph or a word graph from multiple similar sentences, and then adopt linear programming to generate the fused sentence from the graph. Recently, (Lebanoff et al., 2019a) conducts a comprehensive analysis of sentence fusion in neural abstractive summarization and finds that it remains a challenge for current state-of-the-art models. To address this problem, (Lebanoff et al., 2020a,b) propose to utilize points of correspondence between sentences to fuse disparate sentences, and develop a transformer enhanced with the links between the co-referred entities. Similar to above-mentioned works, our research also focuses on the research of sentence fusion in the context of text summarization.

Moving beyond sentence fusion alone, (Mehdad et al., 2013; Lebanoff et al., 2019b) discusses the potential application scenarios for enhancing text summarization with sentence fusion. Their models follow a similar framework that first extracts a few related sentences from the source document and then fuses them to obtain a summary sentence. Our model can be considered as a better replacement of the fusion model in such a framework.

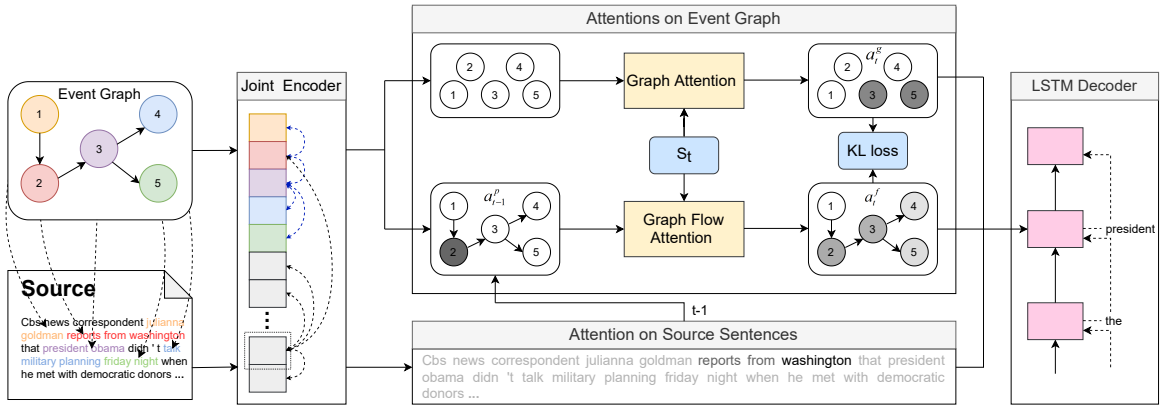

Figure 2: The framework of our proposed sentence fusion model. The various colors in the left refer to nodes and corresponding event components, while dotted lines represent how information disseminates in the BERT attention layer. In the middle part, different gray scales stand for different levels of attention on the tokens or nodes.

2.2 Event-aware Generation Model

Currently, in the conditional generation tasks like text summarization and question answering, most of the source documents are usually composed of a series of events. Understanding how to leverage event information in these generation models becomes crucial. (Moryossef et al., 2019) learns to generate a fluent sentence with an input subject-verb-object triple that describes an event. (Huang et al., 2020) transfers event triples extracted with OpenIE to an event graph to acquire semantic interpretation over input to assist text summarization. (Zheng and Kordjamshidi, 2020) adopts an event graph to understand the path of multi-hop reasoning in question answering. To control the generation process and avoid factual errors, (Cao et al., 2017) proposes an additional event relation encoder to produce representations of event triples. Considering the importance of the relations between events in sentence fusion and inspired by the above-mentioned works, we adopt the event graph to guide sentence fusion.

3 Method

Our sentence fusion model follows the typical encoder-decoder architecture, as shown in Figure 2. It is composed of a joint encoder that produces both source sentences and event graph representations, and a decoder that incorporates the information from the source sentences and the event graph to generate a fused sentence.

3.1 Event Graph Construction

The event graph is built to capture the semantic relationships in the source sentences. We utilize AllenNLP-OpenIE (Stanovsky et al., 2018) to extract a set of events, where each event is composed of a predicate and an arbitrary number of arguments. When there is an overlap between two events, only the longer one is retained. These predicates and arguments are represented as the nodes in the event graph. When two nodes share the same content, we merge them into one. The graph is a directed graph. Two types of edges are considered. (1) Directional edges connect a predicate and its corresponding arguments in an event and the direction follows the order of subject to predicate and predicate to other arguments. (2) Bi-directional edges connect two nodes if they share the same entity or there is a coreference relation between them.

3.2 Encoder

We apply a BERT-based encoder to jointly generate contextualized representations of the tokens in concatenated input sentences and the nodes in the event graph. Each node is represented by a special $[cls]$ token and the output representation of this token is considered as the representation of the node. The input of our encoder is the concatenation of sentence tokens and a set of graph node tokens. Since each node only corresponds to several words in the input sentences, one node token will only be attended by the sentence tokens that belong to this node in the attention layer of BERT. To distinguish the two kinds of tokens, we assign two different segment embeddings to sentence tokens and node tokens. Since there is no sequential relationship between nodes, we initialize the positional embedding for node tokens as a special pad embedding.

We use an additional mask matrix M similar to the one presented in (Yuan et al., 2020) to control the attention of the BERT-based encoder. $M_{ij} = 0$ means token i is allowed to attend to j, while $M_{ij} = -\infty$ prohibits i from attending to j. In our model, three possible situations can happen: (1) a sentence token attends to all other sentence tokens; (2) a sentence token attends to its corresponding graph node token; (3) a node token attends to other adjacent nodes on the event graph. After defining the mask matrix M, we calculate attention with Equation (1) below, where Q, K and V refer to the query matrix, the key matrix and the value matrix, respectively, d^k is a scaling factor.

$$Attention(Q, K, V) = softmax(\frac{QK^T + M}{\sqrt{d^k}})V \quad (1)$$

In our preliminary study, we have also considered using the graph neural network as the encoder for the event graph, but we find that the current approach achieves a better result.

3.3 Decoder

Overview of the Decoder. The decoder aims to generate the fused sentence utilizing both the (sentences) sequence information and the (event) graph information. We employ a one-layer LSTM as the decoder with the hidden state s_t at step t. The decoder generates tokens recurrently based on three types of attentions, i.e., the sequence attention, the graph attention and the graph flow attention.

Sequence Attention. At each decoding step t, we calculate the context vector c_t^s over a sequence of input sentences using the attention mechanism proposed in (Bahdanau et al., 2014). We also employ a coverage mechanism to avoid redundancy.

$$c_t^s = \sum_k a_{t,k}^s h_k \quad (2)$$

$$a_{t,k}^s = softmax(W_k tanh(W_1 s_t + W_2 h_k + W_3 Cov)) \quad (3)$$

where h_k represents the token representation obtained from the encoder, Cov refers to the coverage vector generated at the last step.

Graph Attention. The graph attention applies the mechanism analogous with the sequence attention but to the node embedding v_i and current hidden state s_t to compute the attention score. The graph vector c_t^g is computed over the node embeddings with attentions.

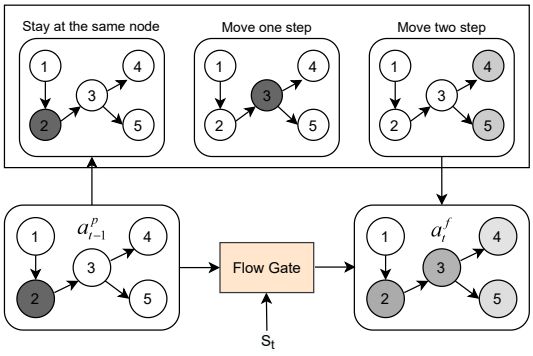

Figure 3: Calculation process of graph flow attention.

$$c_t^g = \sum_i a_{t,i}^g v_i \quad (4)$$

$$a_{t,i}^g = softmax(W_v tanh(W_4 s_t + W_5 v_i)) \quad (5)$$

Graph Flow Attention. When the graph structure is ignored during the decoding process, the graph attention tends to reflect the importance of individual nodes rather than the connections between nodes. We thereby propose a novel graph flow attention to explore potential fusion paths by capturing the content coherence embedded in the graph structure. The graph flow attention is designed to inherit the attention tendency of nodes from the previous decoding step and focuses on neighboring nodes at the current step.

The attention tendency of nodes is expected to be strongly correlated to the output of the decoder. In this way, the model can maintain the coherence between the generated tokens and the nodes focused by the graph flow attention. Considering the graph attention is not fully synchronized with the decoding process, the following situation may happen. It first focuses on one node, and then teleport to another one far from the current node across the two consecutive decoding steps. Therefore, we choose to compute the distribution of attention tendency of nodes in the last step a_{t-1}^p based on the sequence attention in the last decoding step. Suppose $Map \in i \times j$ is the mapping matrix between tokens and nodes, where $Map_{ij} = 1$ denotes that the i token in the source sequence is in the j node of the event graph. The a_{t-1}^p is then calculated based on the following equation.

$$a_{t-1}^p = softmax(Map^T a_{t-1}^s) \quad (6)$$

Given the adjacent matrix A of the event graph, the i row refers to the normalized in-degree of the node i. As shown in Figure 3, the graph flow attention transmits a^p_{t-1} in the following three ways:

(1) Remain in the previous node $f_{t,0} = a^p_{t-1}$. Since one node usually contains multiple tokens, the model may focus on the same node in several steps.

(2) Move one step $f_{t,1} = Aa^p_{t-1}$. For example, the attention moves from one node to its neighbor.

(3) Move two steps $f_{t,2} = A^2a^p_{t-1}$. The attention is allowed to skip a middle connection node.

The graph flow attention is then the weighted sum of the scores of the three flows controlled by a dynamic gate $Gate_t \in 1 \times 3$. And the graph flow vector c^f_t is computed by the following equation.

$$c^f_t = \sum_i a^f_{t,i} v_i \tag{7}$$

$$a^f_t = \sum_{h=0}^{2} f_{t,h} Gate_{t,h} \tag{8}$$

$$Gate_t = \text{softmax}(\text{W}_f tanh(W_6 s_t + W_7 \sum_i a^p_{t-1,i} v_i)) \tag{9}$$

Token Prediction. After obtaining the three vectors from the input sequence and the graph, we regard them as the representations of the information summarized from different points of view. Then they are concatenated with the decoder hidden state s_t to produce the vocabulary distribution D_{vocab} as follows.

$$D_{vocab} = softmax(W_{out}[s_t; c^s_t; c^g_t; c^f_t]) \tag{10}$$

We add a copy mechanism to directly copy words from source text based on the sequence attention. The copy probability is:

$$p_{copy} = sigmoid(W_{copy}[y_{t-1}; s_t; c^s_t; c^g_t; c^f_t]) \tag{11}$$

where y_{t-1} denotes the embedding of the token predicted at step $t-1$.

3.4 Training

Generation Loss. With the generation loss, the training goal is to maximize the estimated probability of the reference sequence. Following most current works, we adopt the maximum likelihood training objective function that minimizes the following loss.

$$L_{seq} = -\frac{1}{|D|} \sum_{(x,y,g) \in D} logp(y|x, g; \theta) \tag{12}$$

where θ represents model parameters and D stands for the training data including source sentences x, reference sequence y, and event graph g.

KL Loss. Our preliminary study reveals that simply concatenating the graph vector and graph flow vector in the decoding process fails to achieve a good performance. We figure out that it is difficult for a model to obtain effective information from two disparate vectors. Therefore, we introduce another training objective that computes the KL loss between the graph attention and the graph flow attention. In this way, the two attentions take advantage of each other. The KL loss is shown below and T is the total number of decoding steps.

$$L_{kl} = -\frac{1}{|D|T} \sum_D \sum_{t \in T} KL(a^g_t || a^f_t) \tag{13}$$

Node Salience Labeling. We further enhance the node representation via the third objective that models the salience of nodes. The goal of it is to identify whether the non-stop words in a node are mentioned in the reference fused sentence. We incorporate a classification layer over each node v_i above the joint encoder to predict a probability m_i ranged in [0,1]. During training, the gold label n_i is set to 1 if the node contains at least one non-stop word in the reference, and 0 otherwise. The loss function is shown below.

$$L_{node} = -\frac{1}{N_v} \sum_i (n_i log(m_i) + (1-n_i) log(1-m_i)) \tag{14}$$

where N_v is the number of the nodes in the graph. To summarize, the full training objective function consists of three terms: $L = L_{seq} + L_{kl} + L_{node}$.

3.5 Faithful Beam Search

Inspired by (Scialom et al., 2020), we propose faithful beam search to reduce possible factual errors at the inference stage. Given a factual consistency checking model F and a sentence fusion model G, the goal is to re-rank every generated token based on both the generation probability calculated by G and the faithful score derived from F. In our work, we adopt the FactCC model developed by (Kryscinski et al., 2020), a BERT-based faithfulness checking model, to evaluate faithfulness. The input to FactCC consists of a hypothesis sentence and several source sentences, while the output from FactCC is a probability that refers to whether the hypothesis sentence is faithful to the source sentences. Since what we need here is to verify the

faithfulness of an incomplete fused sentence during the decoding process, we made a corresponding change when training FactCC with sentence fusion data. We truncate all the fused sentences in positive samples to random length. For the negative samples, we remove the tokens after the position of the error in fused sentences. At the inference stage, the objective function aims to maximize the cumulative probability of the output tokens. At each decoding step, the top-b sequence with the highest probability is carried into the next step, where b stands for the beam size. We add an additional faithful score to refine the generation probability during beam search, such that:

$$S(y_t) = S(y_{t-1}) + \alpha log F(x, y) + log G(x, y_{1:t-1}) \quad (15)$$

where y refers to the generated sequence, x represents the source sentences and α is a weighting factor. F and G stand for the consistency checking model and the sentence fusion model respectively. In the experiments, the α is set to 0.05.

4 Experiments

4.1 Experimental Set-Up

Datasets: We follow the practice of (Lebanoff et al., 2019b) to sample the sentence fusion data from summarization datasets. We choose the well-known single-document summarization dataset CNN/DaliyMail and multi-document summarization dataset Multi-News for the purpose of evaluation. With the CNN/DaliyMail dataset, the fusion data is directly obtained according to the set of heuristics suggested in (Lebanoff et al., 2020a), which we call CNN/DaliyMail Fusion. With the Multi-News dataset, we use a strategy similar to the one proposed in (Lebanoff et al., 2020a) to generate the fusion data, which we call Multi-News Fusion. Note that there is a 60-70% compression rate on both sentence fusion datasets. Hence, they are different from the one proposed by (Geva et al., 2019) where the compression rate is lower than 5%. This explains why we create the sentence fusion data generated from summarization datasets rather than using the existing one.

Evaluation Metrics: Sentence fusion can be approximately regarded as multi-sentence summarization. Following the common practice, we adopt ROUGE F_1 as the basic evaluation metric. We also apply FactCC (Kryscinski et al., 2020) to evaluate faithfulness (Fai) automatically. FactCC is trained on the CNN/DaliyMail Fusion dataset and the Multi-News Fusion dataset following the method presented in the original paper. It achieves 90% of accuracy on the test set of two sentence fusion datasets and we believe that it is reasonably good for our evaluation. Note that it is distinct from the one used in our faithful beam search, where the fused sentences are not modified in the training. Besides, we also report the results of another two metrics, including (1) fusion rate (Fus), which is the percentage of the fused sentence that contain at least two unique non-stop words from multiple source sentences; and (2) length (Len), which is the average length of the fused sentences.

CNN/DaliyMail Fusion	**Train**	**Validate**	**Test**
Number	107347	5948	5100
Source length	53.8	53.5	53.2
Target length	16.3	16.3	16.4
Multi-News Fusion	**Train**	**Validate**	**Test**
Number	19984	2496	2512
Multi / Single	9402/10582	1184/1312	1124/1388
Source length	72.5	71.5	72.4
Target length	28.5	28.6	28.6

Table 1: Statistic of CNN/DaliyMail Fusion dataset and Multi-News Fusion dataset. Multi/Single indicates whether the source sentences are from multiple documents or a single document.

Implementation Details: We build the encoder using the BERT-base-uncased version of BERT. We employ the LSTM models with 768-dimensional hidden states as the decoder. We truncate the input sentences to 150 tokens and limit the decoder to a maximum of 60 steps. The batch size is set to 32 and we train the model for 20 epochs. After training, we select top-3 checkpoints on the validation dataset, and report the one with the best record on the test set among the three. For inference, the beam size is set to 5 in CNN/DaliyMail Fusion and 2 in Multi-News Fusion.

4.2 Automatic Evaluation

To examine the effectiveness of our model, we compare our model with two widely adopted seq2seq baseline models. They are **Pointer-Generator** (See et al., 2017) and **BERT+LSTM**, which is our basic encoder-decoder architecture before integrating the graph information. We also implement the state-of-the-art sentence fusion model for comparisons. **Tranformer-Linking** (Lebanoff et al., 2020a) is a BERT based model proposed for disparate sentence fusion. It utilizes coref-

CNN/DaliyMail Fusion	Rouge-1	Rouge-2	Rouge-L	%Fai	%Fus	#Len
Concat-Baseline	37.29	20.06	28.77	100	100	53.07
Random-Baseline	36.25	17.64	30.72	100	-	26.10
Pointer-Generator	33.37	16.29	29.51	80.13	31.37	13.79
BERT+LSTM	37.56	19.50	33.59	88.77	45.66	16.65
BERTSUMABS	37.96	19.32	33.36	86.17	60.24	16.34
Transformer-linking	**39.79**	**21.08**	**35.35**	90.68	59.42	15.78
Our Model	39.30	21.03	35.12	**91.56**	**61.30**	15.12
Multi-News Fusion	Rouge-1	Rouge-2	Rouge-L	%Fai	%Fus	#Len
Concat-Baseline	48.63	32.95	36.56	100	100	71.28
Random-Baseline	44.60	27.04	37.16	100	-	31.48
Pointer-Generator	49.01	31.57	40.65	81.45	44.39	29.28
BERT+LSTM	50.93	33.99	43.00	85.84	48.30	26.16
BERTSUMABS	51.85	31.60	44.62	78.32	56.48	26.32
Our Model	**53.06**	**36.02**	**45.40**	**89.31**	**59.82**	25.36

Table 2: Automatic evaluation on Rouge, faithfulness(Fai), fusion rate(Fus), and generated sentence length (Len).

erence relationships between entities to enhance sentence fusion. Since our data can be approximately regarded as multi-sentence summarization, we also adopt BERT based document summarization model, **BERTSUMABS** (Liu and Lapata, 2019), for comparisons. Most of these models are trained on the two sentence fusion datasets by ourselves except that the output result of Transformer-Linking is directly obtained from its author.

As shown in Table 2, our proposed model obtains the highest Rouge scores on the Multi-News Fusion dataset and the competitive Rouge scores on the CNN/DaliyMail Fusion dataset. Meanwhile, our model achieves the best performance in fusion rate and faithfulness on both datasets. These suggest the effectiveness of our model in fusing sentences and its ability to reduce factual errors. We also notice that the transformer decoder has a clear advantage over the LSTM decoder in fusion rate. One possible reason is that the transformer decoder can generate a more abstractive sentence, which makes fusion a lot easier. Considering our model adopt a LSTM based decoder, we believe the event graph effectively assists the fusion process by providing cross-event connections and reduce the shifting distance between event components.

4.3 Ablation Study

To look into more detail, we design an experiment to understand how different components contribute to our model. We remove the KL loss, the graph attention and the graph flow attention independently from the full model and report the results in Table 3. On the one hand, we find that the graph flow attention boosts the fusion rate. We believe that the flow attention indeed benefits the fusion process when utilizing the graph structure to find possible fusion paths. On the other hand, the graph attention leads to relatively high Rouge scores but a lower fusion rate. This suggests that although the graph attention does not contribute to sentence fusion, it assists to select important information from source sentences. More importantly, when the KL loss is taken out, the model performance drops more compared to the other two reductions. It indicates that the KL loss is essential for our model to take advantage of both attentions.

Model	R-1	R-2	R-L	%Fai	%Fus
Our Model	**53.06**	**36.02**	**45.40**	**89.31**	**59.82**
- KL loss	52.63	35.63	45.42	87.10	55.52
- Flow Attention	52.71	35.97	45.37	88.79	51.42
- Graph Attention	52.81	35.94	45.22	86.62	56.13

Table 3: The results of ablation study on Multi-News Fusion test set.

4.4 Human Evaluation

Automatic evaluation results are often not enough to fully reflect the quality of the generated fused sentence. We further conduct human evaluation to analyze unfaithful errors and fusion quality. We randomly extract 50 samples from the Multi-News Fusion test set and invite three fluent English speakers as human judges. Given a sentence fusion instance, the judges are asked to answer yes or no to

Source: (1) Police identified the rite aid shooter as Snochia Moseley, 26, who lived in the marsh neighborhood of Baltimore. (2) The shooter was found with a self-inflicted gunshot wound and died at an area hospital. (3) The woman died at a nearby hospital after shooting herself in the head.
BERT+LSTM: Police say the shooter as Snochia Moseley, 26, was found with a self-inflicted gunshot wound and died at an area hospital.
BERTSUMABS: The woman, who died at a hospital, was found with a self-inflicted gunshot wound and died at an area hospital.
Our: Snochia Moseley was found with a self-inflicted gunshot wound and died at a nearby hospital after shooting herself in the head.
Reference: Police say the 26-year-old woman, who has not been identified, died of a self-inflicted gunshot wound to the head.

Table 4: Examples from the Multi-News Fusion test dataset.

Model	%Fluency	%Fusion	%Faithful
BERT+LSTM	85.3	51.3	56.7
BERTSUMABS	81.3	56.7	40.0
Our Model	**88.7**	**58**	**58.6**

Table 5: The results of the human evaluation on Multi-News Fusion test set.

the following three questions. (1) Fluency: whether the generated sentence is grammatically correct and readable. (2) Fusion: whether the generated sentence is generated through sentence fusion. (3) Faithful: whether the generated sentence is faithful to the source sentences. Table 5 shows the percentage of yes on the three questions. We adopt Fleiss' kappa (Fleiss, 1971) to conduct the inter-annotator agreement test and the result is 0.53. The result shows a similar trend to the automatic evaluation, where our model achieves the best result in both fusion rate and faithfulness. The performance of BERTSUMABS further indicates that sentence fusion will lead to the decline of fluency and more faithful errors if there is no proper guidance.

We illustrate a sentence fusion example that contains both similar and disparate sentence fusion in Table 4. As shown, BERT+LSTM tends to fuse sentences by directly copying the text spans from the source text. BERTSUMABS attempts to utilize the coreference relations between "the shooter" and "the woman" to fuse the last two source sentences, but generates redundancy when merging similar

	Rouge-1	Rouge-2	Rouge-L
Oracle	51.67	29.12	48.06
Oracle_all	48.92	26.43	45.42
Fusion	**52.14**	**29.19**	**48.62**

Table 6: The result of sentence fusion application on CNN/DaliyMail test set.

content. On the contrary, our model successfully fuses the information from all source sentences. It shows that our model can effectively handle both types of sentence fusion at the same time.

4.5 Application in Text Summarization

We further design an experiment to investigate the effectiveness of the sentence fusion model in text summarization using a framework from (Lebanoff et al., 2019b). It aims to extract a single sentence (no need for fusion) or a pair of sentences (need fusion), then rewriting them to produce a summary sentence. Each sentence pair consists of a primary sentence and a secondary sentence provides complementary information. We use the oracle extractive results as input to conduct the generation experiment. Table 6 shows the summarization results with three different strategies: (1) Oracle: concatenating oracle single sentences and primary sentences in oracle pairs as the summary; (2) Oracle_all: concatenating oracle single sentences and both sentences in oracle pairs as the summary; (3) Fusion: concatenating oracle single sentences and fused sentences as the summary, where the fused sentences are generated by our model using oracle pairs as input. All the summaries are truncated to 100 words. The result shows that the sentence fusion model has the potential to improve the performance of summarization models by fusing information from multiple sentences.

5 Conclusion

In this paper, we investigate the sentence fusion problem in the context of text summarization by exploring the event graph. Our model captures both node representations and the structural information embodied in the event graph to guide the fusion. We further propose a faithful beam search to reduce the possible faithful errors. The experiment results suggest that event graph is crucial for effective sentence fusion and both node representations and graph structure play important roles in sentence fusion. In the future, we would like to

further explore the direct incorporation of event information and the sentence fusion model to text summarization.

Acknowledgements

The work described in this paper was supported by and Research Grants Council of Hong Kong(PolyU/15203617 and PolyU/5210919) and National Natural Science Foundation of China (61672445).

References

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

Regina Barzilay, Kathleen McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics*, pages 550–557.

Ziqiang Cao, Furu Wei, Wenjie Li, and Sujian Li. 2017. Faithful to the original: Fact aware neural abstractive summarization. *arXiv preprint arXiv:1711.04434*.

Micha Elsner and Deepak Santhanam. 2011. Learning to fuse disparate sentences. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 54–63.

Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *arXiv preprint arXiv:1906.01749*.

Katja Filippova and Michael Strube. 2008. Sentence fusion via dependency graph compression. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 177–185.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Mor Geva, Eric Malmi, Idan Szpektor, and Jonathan Berant. 2019. Discofuse: A large-scale dataset for discourse-based sentence fusion. *arXiv preprint arXiv:1902.10526*.

Karl Moritz Hermann, Tomáš Kočiskỳ, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *arXiv preprint arXiv:1506.03340*.

Luyang Huang, Lingfei Wu, and Lu Wang. 2020. Knowledge graph-augmented abstractive summarization with semantic-driven cloze reward. *arXiv preprint arXiv:2005.01159*.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346.

Logan Lebanoff, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020a. Learning to fuse sentences with transformers for summarization. *arXiv preprint arXiv:2010.03726*.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019a. Analyzing sentence fusion in abstractive summarization. *arXiv preprint arXiv:1910.00203*.

Logan Lebanoff, John Muchovej, Franck Dernoncourt, Doo Soon Kim, Lidan Wang, Walter Chang, and Fei Liu. 2020b. Understanding points of correspondence between sentences for abstractive summarization. *arXiv preprint arXiv:2006.05621*.

Logan Lebanoff, Kaiqiang Song, Franck Dernoncourt, Doo Soon Kim, Seokhwan Kim, Walter Chang, and Fei Liu. 2019b. Scoring sentence singletons and pairs for abstractive summarization. *arXiv preprint arXiv:1906.00077*.

Yang Liu and Mirella Lapata. 2019. Text summarization with pretrained encoders. *arXiv preprint arXiv:1908.08345*.

Erwin Marsi and Emiel Krahmer. 2005. Explorations in sentence fusion. In *Proceedings of the Tenth European Workshop on Natural Language Generation (ENLG-05)*.

Yashar Mehdad, Giuseppe Carenini, Frank Tompa, and Raymond Ng. 2013. Abstractive meeting summarization with entailment and fusion. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 136–146.

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. *arXiv preprint arXiv:1904.03396*.

Thomas Scialom, Paul-Alexis Dray, Sylvain Lamprier, Benjamin Piwowarski, and Jacopo Staiano. 2020. Discriminative adversarial search for abstractive summarization. *arXiv preprint arXiv:2002.10375*.

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

Gabriel Stanovsky, Julian Michael, Luke Zettlemoyer, and Ido Dagan. 2018. Supervised open information extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 885–895.

Kapil Thadani and Kathleen McKeown. 2013. Supervised sentence fusion with single-stage inference. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 1410–1418.

Ruifeng Yuan, Zili Wang, and Wenjie Li. 2020. Fact-level extractive summarization with hierarchical graph mask on bert. *arXiv preprint arXiv:2011.09739*.

Chen Zheng and Parisa Kordjamshidi. 2020. Srlgrn: Semantic role labeling graph reasoning network. *arXiv preprint arXiv:2010.03604*.