Heterogeneous Graph Based Extractive Summarization Considering Discourse and Coreference Relations

Yin Jou Huang† and Sadao Kurohashi†

Modeling the relations between text spans in a document is a crucial yet challenging problem for extractive summarization. Various kinds of relations exist among text spans of different granularity, such as discourse relations between elementary discourse units and coreference relations between phrase mentions. In this paper, we utilize heterogeneous graphs that contain multiple edge/node types to model the input document as well as the various relations among text spans in it. Also, we propose a heterogeneous graph based model for extractive summarization that considers the heterogeneity of the document graph. Experimental results on a benchmark summarization dataset verify the effectiveness of our proposed method.

Key Words: Extractive Summarization, Graph Neural Network, Heterogeneous Graph, Discourse Relation, Coreference Relation

1 Introduction

Automatic summarization aims to condense the information of the input document into a shorter summary. Previous works on the task of automatic summarization can be categorized into two main paradigms: abstractive summarization and extractive summarization. Abstractive summarization methods encode the input document and generate output summaries from scratch. Abstractive summarizers can generate concise and flexible summaries, but they also suffer from the problem of not being able to reproduce factual details correctly (See et al. 2017). On the other hand, extractive summarization methods aim to select salient text spans (mostly sentences) from the input document as summaries. Conceptually, extractive summarization resembles the process of highlighting the important parts of the input document. Due to the above nature, extractive summarizers have the advantage of being computationally efficient and factually reliable. In this paper, we will focus on extractive summarization.

Most existing extractive summarization models operate on sentence level. However, the extracted sentences may contain redundant details that harm the conciseness of the summary. This kind of sentence-level redundancies exist pervasively even in the salient sentences. Consider the
following examples:

(a) \[
\text{[Boston native Mark Wahlberg will star in a film about the Boston Marathon bombing and the manhunt,]}_{EDU^1} \quad \text{[Deadline reported Wednesday.]}_{EDU^2}
\]

(b) \[
\text{[Yahya Rashid,]}_{EDU^a} \quad \text{[a UK national from northwest London,]}_{EDU^b} \quad \text{[was detained at Luton Airport on Tuesday]}_{EDU^3} \quad \text{[after he arrived on a flight from Istanbul,]}_{EDU^4} \quad \text{[police said.]}_{EDU^5}
\]

Both sentence (a) and sentence (b) contain important information about the salient events (highlighted in bold font in the above example). However, there are also plenty of peripheral details that could safely be excluded from the summary without harming the overall understanding of the events (colored in gray in the above example).

To avoid the sentence-level redundancy as shown in the examples above, we perform extraction on a smaller granularity—Elementary Discourse Unit (EDU)—in this work. EDU is a sub-sentence unit that originated from discourse analysis (Mann and Thompson 1988). In the examples above, sentence (a) can be further segmented into two EDUs, and sentence (b) can be segmented into five EDUs. Among them, only \( EDU^a \), \( EDU^b \), \( EDU^3 \) and \( EDU^4 \) present central concept of the sentences. By performing EDU-level extractive summarization, we can eliminate unwanted trivial details like \( EDU^2 \), \( EDU^b \), and \( EDU^5 \).

The goal of extractive summarization is to identify salient text spans that represent the central ideas of the input document. Thus, it is crucial to model the overall document structure and the various relations between the text spans across the document. Natural language documents have a hierarchical nature, with each level corresponding to a different level of granularity: document, sentences, EDUs, words and phrases (Figure 1(a)). Between text spans of different granularity,
there exist many different kinds of relations. For example, discourse relations exist between EDUs within a document and coreference relations exist between mention phrases that correspond to the same real-world entity (Figure 1(b)). The discourse relations between EDUs provide important clues for extractive summarization. The discourse structure describes the high-level linguistic structure of the document, and the concept of nucleus and satellite discourse units in the Rhetorical Structure Theory (Mann and Thompson 1988) defines the relative saliency of discourse units. Thus, the discourse relations are helpful in identifying the salient EDUs of an input document. Also, since salient entities are often mentioned multiple times in a given document, the coreference relations between mention phrases can implicitly capture the narrative structure of the document. Figure 2 shows a document where the protagonist ‘Yahya Rashid’ is mentioned multiple times. By observing each mention of the entity ‘Yahya Rashid’ as well as its context, we can observe the narrative revolving around the given protagonist.

Due to its complex nature, modeling the various relations among text spans of a document remains an open challenge. Some recent works capture inter-sentential relations by utilizing recurrent neural networks (RNNs) or Transformer (Vaswani et al. 2017) based encoders on top of the acquired sentence representations (Cheng and Lapata 2016; Nallapati et al. 2017; Liu and Lapata 2019). However, empirical observations show that these sentence-level encoders do not bring much performance gain (Liu and Lapata 2019). Graph structure is an intuitive way to model long-range relations among text spans throughout a document. Early works build connectivity graphs based on content similarity between sentences (Erkan and Radev 2004; Mihalcea and Tarau 2004). Some recent works incorporate discourse or coreference relations into the graph structure and utilize graph neural networks (GNNs) to obtain a high-level representation of text

[Yahya Rashid was detained at Luton Airport on Tuesday] [after he arrived on a flight from Istanbul] ...... [He has been charged with conduct of terrorism]...
...[Rashid is due to appear in Westminster court on Wednesday]

Narrative structure centering around protagonist Yahya Rashid (X):

- X arrives at Luton Airport from Istanbul → X be detained
- X be charged with conduct of terrorism → X appears in Westminster court (future)

Fig. 2 Coreference relations around a given entity reflects the narrative structure in which the entity being the protagonist.
spans (Yasunaga et al. 2017; Xu and Durrett 2019; Xu et al. 2020). Most of these works operate on homogeneous graphs such as Approximate Discourse Graph (ADG) (Christensen et al. 2013) or Rhetorical Structure Theory (RST) (Mann and Thompson 1988) dependency graph. By definition, a homogeneous graph contains only one type of node as well as only one type of edge. On the other hand, a heterogeneous graph consists of more than one type of nodes or more than one type of edge. As illustrated in Figure 1, the various types of relations exist between text spans of different granularity. To model the various text spans of different granularity (nodes) as well as the various types of relations (edges) among them, a heterogeneous graph is a more natural choice than homogeneous graphs.

In this paper, we propose a novel heterogeneous graph based model for extractive summarization. First, we model each input document as a heterogeneous document graph. The heterogeneous graph contains three types of nodes of different granularity: sentence nodes, EDU nodes, and entity nodes. We encode both discourse and coreference relations into the heterogeneous document graph. The discourse relations are encoded with the edges between EDU nodes. As for the coreference relations, edges between EDU nodes and entity nodes are introduced. In addition, we model the hierarchical structure of the document by adding edges between sentence nodes and their constituent EDU nodes. Further, we adopt a Graph Attention network (GAT) (Veličković et al. 2018) based graph encoder to capture the various relations in the heterogeneous document graph. Different from the original GAT network, which is proposed for handling homogeneous graphs, our proposed graph encoder considers the heterogeneous nature of the document graph and applies separate processing for each type of node. To the best of our knowledge, we are the first to utilize heterogeneous graphs to incorporate multiple types of relations (discourse relations and coreference relations) simultaneously for extractive summarization.

Our main contribution is threefold: (1) We propose a heterogeneous document graph that incorporates multiple types of relations simultaneously for extractive summarization. (2) We propose a GAT-based graph encoder that considers the heterogeneity of the document graph. (3) We conducted experiments on summarization benchmark corpora CNN/DailyMail (CNNDM) (Hermann et al. 2015) and New York Times (Sandhaus 2008) and verified the effectiveness of our proposed method.

2 Document Graph Construction

In this section, we introduce the construction of document graphs. Section 2.1 gives details of the pre-processing steps to obtain the discourse and coreference information required for the
construction of document graphs. Section 2.2 gives a brief introduction of the homogeneous document graphs used in previous works as well as their limitations. At last, Section 2.3 and 2.4 introduce two heterogeneous document graphs which tackle the limitations of the homogeneous document graphs.

2.1 Pre-processing

Both discourse information and coreference information are helpful in deciding the content saliency, and have been incorporated as external knowledge into extractive summarization systems.

Given an input document $D$, we first perform the following pre-processing steps and construct the document graphs based on the discourse information and coreference information acquired.

**Discourse Parsing**

Given document $D$ with $m$ sentences $\{s_1, ..., s_m\}$, we first segment the sentences into $n$ contiguous, adjacent and non-overlapping EDUs $\{d_1, ..., d_n\}$.

Further, we perform RST discourse parsing to identify the rhetorical relations between the EDUs. In the RST framework, the discourse structure of document $D$ is represented by a continuous constituency tree. For example, Figure 3 illustrates the RST parse tree consisting of five EDUs. The RST parse tree is constructed by continuously merging adjacent discourse units into a larger discourse unit. In addition, the merged discourse units are tagged as either nucleus(N) or satellite(S), which indicates their relative nuclearity/saliency. Nucleus units are

![Fig. 3](image_url) An example of RST discourse parse tree. Based on the rhetorical relations among the discourse units, we can derive the RST dependency graph.
considered more salient, while satellite units are less important in content. RST defines two types of rhetorical relations between discourse units:

- **Mononuclear relation** that links a satellite unit and a nucleus unit. Such as relation between $EDU_1$ and $EDU_2$.
- **Multinuclear relation** that links two nucleus units. Such as the relations between $EDU_{1-2}$ and $EDU_{3-5}$.

Similar to (Xu et al. 2020), we convert the RST tree to the dependency form based on the rhetorical relations in the RST tree. The RST dependency graph consists of EDU nodes. In the case of mononuclear relation, the dependency graph contains directed edges from satellite nodes to the corresponding nucleus nodes. In the case of multinuclear relation, we link the nucleus nodes in both directions.

### Coreference Resolution

In addition, we perform coreference resolution to identify coreferent relations among mention phrases in the input document $D$. The goal is to identify mention phrases that refer to the same real-world entity, such as the mentions 'Boston native Mark Wahlberg' and 'Wahlberg' (highlighted in red) in Figure 1. The mentions in document $D$ are clustered into $k$ entities $\{e_1, \ldots, e_k\}$, with each entity $e_i$ representing a cluster of mentions among which coreference relations holds.

#### 2.2 Homogeneous Document Graphs

In many works on graph-based extractive summarization, discourse information and coreference information are used to build homogeneous document graphs with sentence/EDU nodes. By definition, homogeneous graphs contain only one type of node and one type of edge.

For example, we can embed discourse relations with the RST dependency graph (Figure 3). Also, we can build a homogeneous coreference graph (Figure 4(a)) with EDU nodes in which EDU nodes containing a common entity are linked. However, it is not straightforward to incorporate discourse relations and coreference relations together in a single homogeneous graph.

Some previous works like DiscoBERT (Xu et al. 2020) utilize two different graphs to embed discourse and coreference relations separately. However, this method neglects the interaction between different relation types. Some other works use weighted graphs (Figure 4(b)) like ADG to embed multiple types of textual relations, with the edge weights representing the overall 'strength' of relations between text nodes (Christensen et al. 2013). Although it is possible to combine multiple different relation types within the same graph in this manner, it is difficult to design the proper edge weights.
(a) Homogeneous coreference graph.

(b) Weighted homogeneous document graph.

(c) Heterogeneous document graph with multiple edge types ($G_1$).

(d) Heterogeneous document graph with multiple node types ($G_2$).

Fig. 4 Homogeneous and heterogeneous document graphs.
2.3 Heterogeneous Document Graph with Multiple Edge Types

To tackle the limitations of homogeneous document graphs, we consider a heterogeneous document graph with multiple edge types. As shown in Figure 4(c), we represent each input document $D$ with a heterogeneous graph $G_1 = \{V, E\}$, with $V$ and $E$ being the set of nodes and edges, respectively.

In the pre-processing step, we perform discourse parsing and coreference resolution on $D$ (Section 2.1). Based on the pre-processing results, we construct the heterogeneous document graph $G_1$.

$V = \{d_1, ..., d_n\}$ consists of the EDU nodes, with each node $d_i$ representing the $i^{th}$ EDU in document $D$.

$E$ consists of three types of edges:
- **Discourse edge**: Based on the discourse parsing results, we add the RST dependency edges between EDU nodes.
- **Coreference edge**: Based on the coreference resolution results, the two EDUs which contain mentions of the same entity are connected.
- **Same-sentence edge**: EDU nodes that belong to the same sentence are connected.

In this work, we use the above-mentioned document graph $G_1$ as a baseline to compare with our proposed heterogeneous document graph with multiple node types (introduced in Section 2.4).

2.4 Heterogeneous Document Graph with Multiple Node Types

In Figure 4(a)–(c), the coreference relations are represented by the edges between EDU nodes. However, the important information about the coreference entities is overlooked. For example, in Figure 4(c), a coreference edge only indicates that the two EDUs share one or more common entities, but does not give information regarding what entities actually are shared by the EDUs.

To tackle the above limitations, we propose a heterogeneous document graph with multiple node types. As shown in Figure 4(d), input document $D$ is represented with a heterogeneous document graph $G_2 = \{V, E\}$, where $V$ and $E$ are the set of nodes and edges.

$V$ contains three types of nodes:
- **Sentence nodes** $V_s = \{s_1, ..., s_m\}$, with $s_i$ representing the $i^{th}$ sentence in $D$.
- **EDU nodes** $V_d = \{d_1, ..., d_n\}$, with $d_i$ representing the $i^{th}$ EDU in $D$.
- **Entity nodes** $V_e = \{e_1, ..., e_k\}$, with each $e_i$ representing a phrase entity in $D$.

We utilize the pre-processing results acquired in Section 2.1 to construct the heterogeneous document graph $G_2$ as follows:
• **Discourse edge**: We add these RST dependency edges between EDU units to our document graph $G_2$ to model the discourse structure of the document.

• **Coreference edge**: We use edges between EDU nodes and entity nodes to embed the coreference relations. If EDU $d_i$ contains a mention of entity $e_j$, then we add an undirected edge $(d_i, e_j)$ to $E$. That is, each entity node indirectly connects all EDUs that contain mentions of the entity. In this way, the subgraph around a specific entity node implicitly models the narrative structure related to the entity.

• **Same-sentence edge**: Based on the discourse segmentation result, we connect each sentence node to its constituent EDU nodes.

3 Proposed Method

3.1 Overview

**Problem Formulation**

Given an input document $D$ with $n$ EDUs $\{d_1, d_2, ..., d_n\}$, we formulate EDU-based extractive summarization task as a sequence labeling problem. The model predicts a sequence of binary labels $Y = \{y_1, y_2, ..., y_n\}$, where $y_i = 1$ indicates that the $i^{th}$ EDU, $d_i$, should be included in the summary.

From the human-written summary, we heuristically obtain the oracle labels $\{y^*_1, y^*_2, ..., y^*_n\}$, which can be used to train our extractive summarization model. Further details will be given in Section 4.1. In the inference stage, the model predicts the binary labels for each EDU in the input document. The EDUs with label $y_i = 1$ will be concatenated to form the summary.

**Model Overview**

Figure 5 provides an overview of our proposed model. First, the input document $D$ is feed into a Longformer (Beltagy et al. 2020) document encoder. With the self-attention based EDU and entity encoders, we acquire the initial node representation of the heterogeneous document graph (Section 3.2). We then apply a heterogeneous graph encoder to obtain high-level node representations considering the document graph structure (Section 3.3). Finally, we make predictions based on the learned EDU node representations (Section 3.4).

3.2 Graph Node Initialization

Following the settings in (Liu and Lapata 2019), we utilize pretrained Longformer (Beltagy et al. 2020) to encode the input document $D$. We insert the $\langle CLS \rangle$ and $\langle SEP \rangle$ special tokens to
the beginning and the end of each sentence $s_i$, respectively. $\langle CLS \rangle$ special token was originally used to aggregate features from one or a pair of sentences.

With the output vectors of Longformer, we acquire the initial node representations. In the following, we describe how we obtain the initial representations of sentence nodes, EDU nodes, and entity nodes, respectively.

**Sentence Node Representations**

For each sentence node $s_i$ in $V_s$, we take the Longformer output vector of the $\langle CLS \rangle$ token before $s_i$ as the sentence node representation $h_s^i$. We expect the $\langle CLS \rangle$ symbol to aggregate the information of the tokens of the sentences that follows it.

**EDU Node Representations**

We use a self-attention based EDU encoder to encode each EDU node in $V_d$. Given an EDU $d_i$ consisting of tokens $\{w_1^i, w_2^i, ..., w_N^i\}$, we obtain the node representation $h_d^i$ by taking self-attention on the Longformer output vectors $\{v_1^i, v_2^i, ..., v_N^i\}$ of its consisting tokens:

$$\alpha_{ij} = v_2ReLU(W_1v_i^j + b_1)$$

(1)
\[ a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_{k=1}^{N} \exp(\alpha_{ik})} \]  \hspace{1cm} (2)

\[ h_i^d = \sum_j a_{ij} v_i^j \]  \hspace{1cm} (3)

where \(W_1, b_1, v_2\) are trainable weights.

**Entity Node Representations**

The structure of the entity encoder is identical to the EDU encoder. For each entity \(e_i\) in \(V_e\), we consider all mentions of it. By taking self-attention among the Longformer output vectors which correspond to tokens of these mentions, we can acquire the entity representation \(h_i^e\).

### 3.3 Heterogeneous Graph Encoder

We initialize the representation of each node in \(G\) with the sentence representations \(\{h_i^s\}\), EDU representations \(\{h_i^d\}\), and entity representations \(\{h_i^e\}\) acquired in Section 3.2. We then use the heterogeneous graph encoder to learn high-level node representations considering the structure of the document graph.

Figure 6 illustrates the structure of our heterogeneous graph encoder. Each graph encoding layer \(L_i\) consists of two Transformer sub-layers and a graph attention sub-layer. The Transformer sub-layers aim to model the interactions among nodes with the same granularity. In the graph attention sub-layer, we apply graph attention network (GAT) (Veličković et al. 2018) to model the interactions among all types of nodes in \(G\) and update the node representations. The complete heterogeneous graph encoder consists of \(N\) stacked graph encoding layer \(\{L_1, ..., L_N\}\).

#### 3.3.1 Transformer Sub-layers

We first feed the sentence vectors and EDU vectors to the sentence-level and the EDU-level Transformer (Vaswani et al. 2017) sub-layers.

At the center of the Transformer sub-layer is a multi-head self-attention layer follow by a feed-forward layer. The self-attention mechanism can be seen as a fully-connected version of a graph attention network. To model the interactions among nodes of the same granularity, we utilized two types of Transformer layers, which operate on the sentence-level and the EDU-level, respectively.

#### 3.3.2 Graph Attention Sub-layer

The graph attention sub-layer consists of a multi-head GAT network followed by a feed-
forward layer. Taking the document graph and the node representations as input, the purpose of the graph attention sub-layer is to learn a higher-level representation of each node by aggregating information from its neighboring nodes. Here, we introduce two types of GAT networks, the vanilla GAT network and the heterogeneous GAT network. Similar to the one proposed in (Veličković et al. 2018), the vanilla GAT network handles the document graph $G$ as a homogeneous graph and treats all types of nodes in the same way. On the other hand, our proposed heterogeneous GAT network considers the heterogeneity of $G$ and applies different processing to different node types.

**Vanilla GAT network**

We apply graph attention networks (GAT) (Veličković et al. 2018) to update the node representations in $G$. For the $i^{th}$ node, we update the representation $h_i$ of node $i$ with the representations of its neighbors $\{h_j\}$:

$$\alpha_{ij} = \text{LeakyReLU}(W_a[W_q h_i; W_k h_j])$$

(4)
Huang and Kurohashi  Heterogeneous Graph Based Extractive Summarization

Fig. 7 Graph attention sub-layer.

\[ a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_k \exp(\alpha_{ik})} \]  (5)

\[ h_i \leftarrow W_t(\sigma(\sum_j a_{ij}W_v h_j) + h_i) \]  (6)

where \( W_a, W_q, W_k, W_v, W_k \) are trainable weights.

Figure 7 illustrates an example of graph attention mechanism. The subgraph centering around node \( EDU_1 \) is highlighted in Figure 7(a). \( EDU_1 \) is connected to a sentence node \( sent_1 \), two EDU nodes \( EDU_2 \) and \( EDU_3 \), and two entity nodes \( entity_1 \) and \( entity_2 \). With vanilla GAT network, we calculate the attention weight across the five neighbors of \( EDU_1 \) and updated the node representation of \( EDU_1 \) (\( h_i^d \)) accordingly (Figure 7(b)).

Although a single GAT network only considers the first-degree neighbors, by stacking several layers of GAT network, we can obtain a higher-level representation for each node in \( G \).

**Heterogeneous GAT Network**

The vanilla GAT network disregards the heterogeneity of the document graph and treats different types of nodes identically. Take Figure 7 for example, all three types of neighboring nodes of \( EDU_1 \) (sentence, EDU, and entity nodes) use the same equation for calculating attention scores \( \alpha \) (Figure 7(b)).

Considering the heterogeneous nature of the document graph \( G_2 \), we introduce a heterogeneous version of the GAT network (Figure 7(c)). To incorporate the multiple node types of heterogeneous document graph \( G_2 \), we introduce different query and key matrices for each type.
of node. For example, we use query matrix $W^d_q$ and key matrix $W^d_k$ for EDU nodes, query matrix $W^e_q$ and key matrix $W^e_k$ for entity nodes, and query matrix $W^s_q$ and key matrix $W^s_k$ for sentence nodes.

Different query ($\{W^d_q, W^e_q, W^e_q\}$) and key matrices ($\{W^d_k, W^e_k, W^e_k\}$) are used during the attention calculation process for different types of nodes:

$$
\alpha_{ij} = \text{LeakyReLU}(W_a[W_q^t h_i; W_k^t h_j])
$$

$$
a_{ij} = \frac{\exp(\alpha_{ij})}{\sum_k \exp(\alpha_{ik})}
$$

$$
h_i \leftarrow W_t(\sigma(\sum_j a_{ij} W_v h_j) + h_i)
$$

where $t_i$ indicates the type of node $i$, and $W_a$, $W^d_q$, $W^e_q$, $W^e_k$, $W^s_q$, $W^s_k$ are trainable weights.

For example, the attention score $\alpha$ between $\text{sent}_1$ and $\text{EDU}_1$ is calculated with the key matrix for sentence nodes $W^s_k$, while the attention score $\alpha$ between $\text{entity}_1$ and $\text{EDU}_1$ is calculated with the key matrix for entity nodes $W^e_k$.

As for the baseline document graph $G_1$, we consider a heterogeneous GAT network that considers the multiple edge types. We introduce binary variables $d_{ij}$, $c_{ij}$ and $s_{ij}$ to indicate the existence of discourse edge, coreference edge and same-sentence edge between node $i$ and node $j$. For example, if there is a discourse edge between node $i$ and node $j$, then $d_{ij} = 1$, otherwise, $d_{ij} = 0$. The attention weight between node $i$ and node $j$ is calculated as follows:

$$
\alpha_{ij} = \text{LeakyReLU}(W_a[W_q^t h_i; d_{ij} W_d^t h_j; c_{ij} W_c^t h_j; s_{ij} W_s^t h_j])
$$

### 3.4 Prediction Layer

We feed the final representation of the EDU nodes ($h^d_i$) to the prediction layer with sigmoid activation to predict binary labels:

$$
\hat{y}_i = \sigma(W_p h^d_i + b_p)
$$

The training loss of the model is the binary cross-entropy loss $L$ against the oracle extraction labels:

$$
L = -\sum_i y^{*}_i \log(\hat{y}_i) + (1 - y^{*}_i) \log(1 - \hat{y}_i)
$$

in which $\{y^{*}_i\}$ are the oracle labels and $\{\hat{y}_i\}$ are the binary labels predicted by the model.
4 Experiment

4.1 Experimental Settings

Dataset

We evaluated our proposed model on the benchmark CNN/DailyMail dataset (Hermann et al. 2015) and New York Times dataset (Sandhaus 2008).

For the CNN/DailyMail dataset, we use the non-anonymized version of the standard split, which contains 287,227/13,368/11,490 documents for training, validation, and test split, respectively. As for the New York Times (NYT) dataset, we follow previous works and split the dataset into 137,7787/17,222/17,223 documents for training, validation, and test split, respectively.

Since both datasets only contain abstractive gold summaries, we have to construct oracle labels heuristically. Similar to (Liu and Lapata 2019), we created the oracle labels on EDU-level by greedily selecting EDUs which maximize the ROUGE scores (Lin 2004). We used the average of ROUGE$_1$-$F_1$ and ROUGE$_2$-$F_1$ as selection criteria. For each document, we selected up to 5 EDUs.

Pre-processing

We used the Stanford CoreNLP (Manning et al. 2014) to split sentences. Further, we used the RST discourse parser proposed by (Ji and Eisentstein 2014) for both discourse segmentation and discourse parsing. For coreference resolution, we used the spanBERT-based (Joshi et al. 2020) version of the end-to-end coreference resolver proposed by (Lee et al. 2017).

Hyper-parameter Settings

We used the ‘longformer-base-4096’ version of Longformer to encode the input document. The length of each document is truncated to 1024 BPEs. The hidden size of the EDU encoder and the entity encoder is 128. Based on the evaluation losses on the validation set, we set the number of stacked graph encoding layers to $N = 2$. For both Transformer sub-layers and the graph attention sub-layers, the number of attention heads is set to 8, with each head having a hidden size of 96.

Training and Evaluation

During training, we used a batch size of 20. We used Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ and followed the learning rate (lr) scheduling in (Vaswani et al. 2017) with warm-up of 4000 steps ($n_{\text{warmup}}$):
\[ lr = 2e^{-3} \min(n_{\text{step}}^{-0.5}, n_{\text{step}}n_{\text{warmup}}^{-1.5}) \]

All models are trained for 60000 steps. We selected the top-3 checkpoints based on the evaluation losses on the validation set and report the average scores of them on the test set. During the inference phase, the trained model is used to obtain likelihood score for each EDU. The top-5 EDUs with the highest likelihood scores are concatenated to generate the final summary. We also perform trigram blocking in the inference phase, which is a simple yet effective way to reduce redundancy in extractive summarization.

We adopt ROUGE (Lin 2004) as the evaluation metrics. We report the F1 scores of the ROUGE1, ROUGE2, and ROUGE\textsubscript{L} metrics of our proposed models.

### 4.2 Results and Analysis

#### Results on CNN/DailyMail Dataset

Table 1 shows the results on CNN/DailyMail dataset. The first part contains the LEAD-3 baseline and Oracle upper bounds. The second part of the table includes the sentence-based extractive models, and the third part includes the EDU-based extractive models. In the third part of the table, we present the evaluation scores of our proposed models with vanillaGAT

| Model                        | CNN/DailyMail | NYT     |
|------------------------------|---------------|---------|
|                              | R-1  | R-2  | R-L  | R-1  | R-2  | R-L  |
| LEAD-3                       | 40.34| 17.70| 36.57| 41.80| 22.60| 35.00|
| Oracle (Sentence-based)      | 52.59| 31.24| 48.87| 60.49| 42.09| 57.02|
| Oracle (EDU-based)           | 55.96| 34.64| 53.26| 62.21| 43.97| 59.54|
| (Sentence-based extraction)  |      |      |      |      |      |      |
| BanditSum (Dong et al. 2018) | 41.50| 18.70| 37.60| —    | —    | —    |
| NEUSUM (Zhou et al. 2018)    | 41.59| 19.01| 37.98| —    | —    | —    |
| HIBERT (Zhang et al. 2019)   | 42.37| 19.95| 38.83| 49.06| 29.07| 41.23|
| HSG (Wang et al. 2020)       | 42.95| 19.76| 39.23| —    | —    | —    |
| BertSum (sent) (Liu and Lapata 2019) | **43.25** | **20.24** | **39.63** | 48.11 | 28.19 | 40.91 |
| (EDU-based extraction)       |      |      |      |      |      |      |
| BertSum (EDU)                | 42.73| 20.03| 40.16| 47.27| 27.76| 40.35|
| DiscoBERT (Xu et al. 2020)   | 43.77| **20.85**| 40.67| **50.00**| **30.38**| 42.70|
| Proposed (vanillaGAT, 768)   | 43.33| 20.55| 40.74| 47.92| 28.46| 42.99|
| Proposed (hetGAT, 768)       | 43.54| 20.63| 40.91| 48.14| 28.68| 43.20|
| Proposed (vanillaGAT, 1024)  | 43.61| 20.67| 40.95| 48.07| 28.57| 43.16|
| Proposed (hetGAT, 1024)      | **43.78**| 20.81| **41.12**| 48.28| 28.82| **43.35**|

*Table 1*  Results on CNN/DailyMail dataset and NYT dataset.
and the *hetGAT* version. heterogeneous document graph $G_2$ (introduced in Section 2.4) is used here. Also, we included results of maximum input size 1024 and 768. For the input size 768, we use BERT (Devlin et al. 2019) as encoder instead of Longformer, which is the same setting as DiscoBERT (Xu et al. 2020).

As Table 1 shows, all our proposed models outperform the LEAD-3 and all sentence based extractive baseline models. Compared to the BertSum(*sent*) baseline, our proposed model (*hetGAT, 1024*) achieved a higher ROUGE on all three metrics (R-1/R-2/R-L). We conclude that EDU-based extraction is a promising direction in extractive summarization.

Our proposed model (*hetGAT*) also outperform the BertSum(*EDU*) baseline by a significant margin in all three metrics (R-1/R-2/R-L). This result shows the effectiveness of our graph encoder module to capture the complex relations among the text spans of the input documents.

Compared to the state-of-the-art EDU-extraction model DiscoBERT, our proposed model (*hetGAT, 1024*) achieved comparable performance on R-1/R-2 metrics and outperformed it on the R-L metrics by 0.45 of F$_1$ score. However, the performance of the proposed model with 768 maximum input size (*hetGAT, 768*) has worse performance compare to DiscoBERT on R-1/R-2 metrics. DiscoBERT incorporates a strict RST-based rule during oracle construction and post-processing to ensure discourse consistency. Since the purpose of this paper is to propose a heterogeneous graph based method for modeling text span relations, we will leave the question of discourse consistency to future work.\(^1\)

Finally, for both types of input length (768 and 1024), we can observe that the *hetGAT* model outperforms the vanillaGAT model. This shows the effectiveness of our proposed heterogeneous GAT networks in capturing and aggregating the various text relations in the heterogeneous document graphs.

**Results on New York Times Dataset**

The results on New York Times dataset is also included in Table 1. Our proposed method (*hetGAT*) outperforms the BertSum(*EDU*) baseline in all R-1/R-2/R-L metrics with a significant margin. Compared to the state-of-the-art model DiscoBERT, our proposed model is

---

\(^1\) The SpanBERT based end-to-end coreference resolver (F1 = 0.77, OntoNote corpus) has a better performance than the Stanford CoreNLP coreference resolver (F1 = 0.69, OntoNote corpus) used in DiscoBERT.

Compared to DiscoBERT, our proposed model can better adapt to long input documents. First, the homogeneous structure of their work is not efficient in embedding coreference relations. Take the example of an entity with $k$ mentions, the homogeneous coreference graph (like Figure 4(a)) needs to use $\frac{k(k-1)}{2}$ edges, while our proposed heterogeneous graph only needs $k$ edges to represent the coreference relation. Since both GCN (used in DiscoBERT) and GAT have time complexity $O(|E|)$, the larger number of edges will make it difficult to adapt to longer documents. Also, unlike their GCN-based method, the GAT-based method we adopted is exempted from eigen-compositions and costly matrix operations.
inferior in R-1/R-2 metrics but outperforms DiscoBERT in R-L metrics.

Similar to the results on CNN/Daily Mail dataset, the performance of the hetGAT models are better than the vanillaGAT models, for both maximum input size settings (768 and 1024).

Ablation Study

We conduct ablation studies on the CNN/Daily Mail dataset by removing components from our proposed document graph (heterogeneous document graph $G_2$ introduced in Section 2.4). The results of the ablation studies are shown in Table 2.

The first part shows the ablation study of the proposed (hetGAT) model. First, we remove the RST dependency edges between EDU nodes (-discourse). Next, we remove the coreferential edges between EDU nodes and entity nodes (-coref). We can see that both discourse and coreference information contributes significantly to the model performance, with discourse information being slightly more important than the coreference information. We also try to remove the edges between sentence nodes and their constituent EDU nodes (-sent). However, linking the sentence and EDU nodes does not seem to have a significant impact on model performance.

The second part of Table 2 shows the ablation study of the proposed (vanillaGAT) model. The results of the ablation study show a similar tendency compared to the hetGAT model. We can observe from the results that both discourse and coreference information contributes to the model performance separately, but these two different types of information do not aggregate well compared to the case in the hetGAT model. This result illustrates the effectiveness of our proposed heterogeneous GAT network in handling various types of text relations simultaneously, compared to the vanilla GAT networks proposed for homogeneous graphs.

| Model            | R-1 (ΔR-1) | R-2 (ΔR-2) | R-L (ΔR-L) |
|------------------|------------|------------|------------|
| Proposed (hetGAT)| 43.78      | 20.81      | 41.12      |
| - discourse      | 43.45(-0.33)| 20.55(-0.26)| 40.83(-0.29)|
| - coref          | 43.57(-0.21)| 20.68(-0.13)| 40.93(-0.19)|
| - sent           | 43.74(-0.04)| 20.79(-0.02)| 41.06(-0.06)|
| Proposed (vanillaGAT)| 43.61      | 20.67      | 40.95      |
| - discourse      | 43.42(-0.19)| 20.54(-0.13)| 40.77(-0.18)|
| - coref          | 43.51(-0.10)| 20.61(-0.06)| 40.85(-0.10)|
| - sent           | 43.57(-0.04)| 20.66(-0.01)| 40.90(-0.05)|

Table 2 Ablation studies on CNN/DailyMail dataset.
Results of Different Graph Structure

We perform experiments on different graph structures with the CNN/Daily Mail dataset. We compare the system performance on the baseline document graph $G_1$ (Section 2.3) and our proposed heterogeneous document graph $G_2$ (Section 2.4).

As shown in Table 3, for both vanillaGAT and hetGAT, using document graph $G_2$ gives a better performance than using $G_1$. Although both $G_1$ and $G_2$ embeds the discourse, coreference and same-sentence information within the document graph, $G_2$ includes richer information of the coreferent entities and sentences. The experimental results show the effectiveness of introducing extra entity and sentence nodes in the document graph.

Results of Different Hyper-parameter Settings

We perform experiments on different hyper-parameter settings to observe how the model performance changes under the changes of hyper-parameters. Same as ablation study, we use heterogeneous graph $G_2$ in the following experiments.

Number of stacked graph encoding layers

We modify the number of stacked graph encoding layers ($N$) and observe how it affects the performance. We report the performance on the validation set from $N = 1$ to $N = 3$ in terms of validation loss, ROUGE-1, ROUGE-2 and ROUGE-L in Figure 8.

As can be observed in Figure 8, we can see that there is a significant performance gap between $N = 1$ and $N = 2$. On the other hand, there is no significant difference in model performance from $N = 2$ to $N = 3$.

The phenomenon can be explained by the theory of meta-path of the document graph. Meta path is a widely used concept proposed to model the various types of relations between nodes in a heterogeneous graph. A meta path captures a specific type of relations within the given graph. Figure 9 illustrates the various meta paths in our proposed heterogeneous document graph:

(a) **EDU-EDU meta path** represents the discourse relation between discourse units.
(b) **EDU-entity-EDU meta path** connects two EDU nodes sharing a coreferent entity.

| Model                   | R-1  | R-2  | R-L  |
|-------------------------|------|------|------|
| Baseline (G1, vanillaGAT) | 43.43| 20.58| 40.77|
| Baseline (G1, hetGAT)    | 43.53| 20.64| 40.85|
| Proposed (G2, vanillaGAT)| 43.61| 20.67| 40.95|
| Proposed (G2, hetGAT)    | **43.78**| **20.81**| **41.12**|

Table 3 Results of different graph structure on CNN/Daily Mail dataset.
Fig. 8 Result of different number of stacked graph encoding layers (N) on validation set of CNN/Daily Mail dataset.

Fig. 9 Meta paths in the heterogeneous document graph.

2-hop meta path describes the coreference relation of the entity in the center.

(c) entity-EDU-entity meta path describes the collocation relation between two entities that appear in the same EDU unit.

(d) EDU-sent-EDU meta path represents the hierarchical structure of a sentence and its consisting EDUs.

The \(N = 1\) model cannot capture the 2-hop meta-paths like EDU-entity-EDU, entity-EDU-entity, and EDU-sent-EDU. We speculate that this could account for the performance loss of the \(N = 1\) model.

**Maximum Input Length** We modify the maximum input length (in BPEs) and observe how it affects the performance. We report the ROUGE-1, ROUGE-2 and ROUGE-L scores on the
validation set with the maximum input size set to 512, 768, 1024, 2048, 4096 BPEs in Figure 10. Generally, we can observe a performance gain by increasing the maximum input length. The performance gain is significant for the maximum input size under 1024. However, the gain is less significant if we increase the input size further.

The average document length of the CNN/DailyMail dataset is around 864 BPEs, with a standard deviation of 443 BPEs. Also, consider the fact that summary-worthy, salient sentences tend to appear at the beginning of the document. The above two facts altogether support that setting the maximum input length to 1024 BPEs should give satisfactory results.

**Qualitative Analysis**

We also conduct a qualitative analysis of the proposed model. The effectiveness of discourse relations is more straightforward and widely studied in previous research. Thus, we focus on the analysis of the role of coreference information in our proposed summarization model.
In the heterogeneous document graph, EDUs containing the same entity phrase are indirectly connected through the node of the given entity. By analyzing the output of the full proposed model and the model without coreference information (-coref), we found that the models rank the importance of coreferent EDUs differently. Table 4 indicates a common pattern of the improved cases by incorporating coreference information. The table shows examples of coreferent EDUs and the ranking of their likelihood scores to be included in the summary. Comparing the EDU ranking of the full model (\text{Rank}_{\text{coref}}) and the model without coreference information (\text{Rank}_{w/o\text{ coref}}), we argue that the model with coreference information is better in discriminating the important EDUs among all EDUs sharing the same entity.

| EDUs with coreferent entity | Rank\text{_{coref}} | Rank\text{_{w/o coref}} | Oracle Label |
|-----------------------------|---------------------|------------------------|--------------|
| Mexican state oil company Pemex said 45 workers were injured ... | 1                   | 2                      | 1            |
| ... two of them are in serious condition | 2                   | 1                      | 0            |
| Kim was accused of stabbing U.S. ambassador Mark Lippert ... | 1                   | 3                      | 1            |
| Before Lippert was supposed to give a speech, the attacker slashed him in the face a jaw | 3                   | 1                      | 0            |
| Kim stabbed Lippert with a 10-inch knife | 2                   | 2                      | 0            |

Table 4 Qualitative studies on CNN/DailyMail dataset.

5 Related Work

Graph-based Summarization

Graph-based summarization models have been broadly explored. Early works build connectivity document graphs based on inter-sentential similarity (Erkan and Radev 2004; Mihalcea and Tarau 2004). With the promising results of graph neural networks (GNNs) (Kipf and Welling 2017; Veličković et al. 2018), some recent works utilize GNN to incorporate external knowledge into the model. For instance, (Yasunaga et al. 2017) utilizes a sentence-level ADG graph to model discourse and coreference relations. Some works convert the RST tree of the input document into dependency form in either sentence or EDU level (Xu and Durrett 2019; Xu et al. 2020). Most of these models operate on homogeneous graphs with only one type of node. Fewer summarization models operate on heterogeneous graphs with different types of nodes. (Wei 2012) introduces a heterogeneous graph of sentence, word, and topic nodes, and (Wang et al. 2020) also utilizes a heterogeneous graph of sentence and word nodes. However, neither of the above works incorporates external knowledge into the graph. (Cui et al. 2020) performs sentence-based extractive summarization based on the heterogeneous sentence node and nodes representing latent topics.
EDU-based Extractive Summarization

(Li et al. 2016) illustrates the potential of using EDU as the extraction unit for summarization. (Xu et al. 2020) also introduces an end-to-end EDU-based extractive summarization model. By using a heuristic based on RST dependency structure, they enhanced the grammaticality and discourse consistency of the extracted summary.

6 Conclusion

In this paper, we proposed a novel heterogeneous graph based model for extractive summarization. By introducing nodes of different granularity, the heterogeneous document graph has the capacity to embed various types of relations between text spans. In addition, we proposed a heterogeneous GAT network that considers the heterogeneous nature of the document graph. Experiments on the benchmark datasets illustrated the effectiveness of our proposed method.

References

Beltagy, I., Peters, M. E., and Cohan, A. (2020). “Longformer: The Long-Document Transformer.” arXiv:2004.05150.

Cheng, J. and Lapata, M. (2016). “Neural Summarization by Extracting Sentences and Words.” In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 484–494, Berlin, Germany. Association for Computational Linguistics.

Christensen, J., Mausam, Soderland, S., and Etzioni, O. (2013). “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, pp. 1163–1173, Atlanta, Georgia. Association for Computational Linguistics.

Cui, P., Hu, L., and Liu, Y. (2020). “Enhancing Extractive Text Summarization with Topic-Aware Graph Neural Networks.” In Proceedings of the 28th International Conference on Computational Linguistics, pp. 5360–5371, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pp. 4171–4186, Minneapolis,
Minnesota. Association for Computational Linguistics.

Dong, Y., Shen, Y., Crawford, E., van Hoof, H., and Cheung, J. C. K. (2018). “BanditSum: Extractive Summarization as a Contextual Bandit.” In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 3739–3748, Brussels, Belgium. Association for Computational Linguistics.

Erkan, G. and Radev, D. R. (2004). “LexRank: Graph-based Lexical Centrality as Salience in Text Summarization.” Journal of Artificial Intelligence Research (JAIR), 22, pp. 457–479.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). “Teaching Machines to Read and Comprehend.” In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (Eds.), Advances in Neural Information Processing Systems 28, pp. 1693–1701. Curran Associates, Inc.

Ji, Y. and Eisenstein, J. (2014). “Representation Learning for Text-level Discourse Parsing.” In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 13–24, Baltimore, Maryland. Association for Computational Linguistics.

Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. (2020). “SpanBERT: Improving Pre-training by Representing and Predicting Spans.” Transactions of the Association for Computational Linguistics, 8, pp. 64–77.

Kipf, T. N. and Welling, M. (2017). “Semi-Supervised Classification with Graph Convolutional Networks.” In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings. OpenReview.net.

Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). “End-to-end Neural Coreference Resolution.” In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 188–197, Copenhagen, Denmark. Association for Computational Linguistics.

Li, J. J., Thadani, K., and Stent, A. (2016). “The Role of Discourse Units in Near-Extractive Summarization.” In Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 137–147, Los Angeles. Association for Computational Linguistics.

Lin, C.-Y. (2004). “ROUGE: A Package for Automatic Evaluation of Summaries.” In Text Summarization Branches Out, pp. 74–81, Barcelona, Spain. Association for Computational Linguistics.

Liu, Y. and Lapata, M. (2019). “Text Summarization with Pretrained Encoders.” In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP),
pp. 3730–3740, Hong Kong, China. Association for Computational Linguistics.
Mann, W. C. and Thompson, S. A. (1988). “Rhetorical Structure Theory: Toward a Functional
Theory of Text Organization.” Text, 8 (3), pp. 243–281.
Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). “The
Stanford CoreNLP Natural Language Processing Toolkit.” In Proceedings of 52nd An-
nual Meeting of the Association for Computational Linguistics: System Demonstrations,
pp. 55–60, Baltimore, Maryland. Association for Computational Linguistics.
Mihalcea, R. and Tarau, P. (2004). “TextRank: Bringing Order into Texts.” In Proceedings of
EMNLP-04and the 2004 Conference on Empirical Methods in Natural Language Processing.
Nallapati, R., Zhai, F., and Zhou, B. (2017). “SummaRuNNer: A Recurrent Neural Network
based Sequence Model for Extractive Summarization of Documents.” In Proceedings of the
31st AAAI Conference on Artificial Intelligence (AAAI’17), pp. 3075–3081. AAAI Press.
Sandhaus, E. (2008). “The New York Times Annotated Corpus.” Abacus Data Network. https:
//hdl.handle.net/11272.1/AB2/GZC6PL.
See, A., Liu, P. J., and Manning, C. D. (2017). “Get To The Point: Summarization with Pointer-
Generator Networks.” In Proceedings of the 55th Annual Meeting of the Association for
Computational Linguistics (Volume 1: Long Papers), pp. 1073–1083, Vancouver, Canada.
Association for Computational Linguistics.
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and
Polosukhin, I. (2017). “Attention is All you Need.” In Guyon, I., Luxburg, U. V., Bengio,
S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R. (Eds.), Advances in Neural
Information Processing Systems 30, pp. 5998–6008. Curran Associates, Inc.
Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2018). “Graph
Attention Networks.” In International Conference on Learning Representations.
Wang, D., Liu, P., Zheng, Y., Qiu, X., and Huang, X. (2020). “Heterogeneous Graph Neural Net-
works for Extractive Document Summarization.” In Proceedings of the 58th Annual Meeting
of the Association for Computational Linguistics, pp. 6209–6219, Online. Association for
Computational Linguistics.
Wei, Y. (2012). “Document Summarization Method Based on Heterogeneous Graph.” In 2012
9th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 1285–1289.
Xu, J. and Durrett, G. (2019). “Neural Extractive Text Summarization with Syntactic Compress-
ion.” arXiv:1902.00863.
Xu, J., Gan, Z., Cheng, Y., and Liu, J. (2020). “Discourse-Aware Neural Extractive Text Summa-
ration.” In Proceedings of the 58th Annual Meeting of the Association for Computational
Yasunaga, M., Zhang, R., Meelu, K., Pareek, A., Srinivasan, K., and Radev, D. (2017). “Graph-based Neural Multi-Document Summarization.” In Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), pp. 452–462, Vancouver, Canada. Association for Computational Linguistics.

Zhang, X., Wei, F., and Zhou, M. (2019). “HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization.” In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 5059–5069, Florence, Italy. Association for Computational Linguistics.

Zhou, Q., Yang, N., Wei, F., Huang, S., Zhou, M., and Zhao, T. (2018). “Neural Document Summarization by Jointly Learning to Score and Select Sentences.” In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 654–663, Melbourne, Australia. Association for Computational Linguistics.

Yin Jou Huang: Yin Jou Huang received his B.S. in Taiwan University, Taipei, Taiwan in 2015. From 2015 she became a student of Graduate School of Informatics, Kyoto University and received her M.S. here in 2017. She is currently a program-specific researcher in Graduate School of Informatics, Kyoto University. Her research interest center on natural language processing, particularly graph-based neural networks.

Sadao Kurohashi: Sadao Kurohashi received the B.S., M.S., and Ph.D. in Electrical Engineering from Kyoto University in 1989, 1991, and 1994, respectively. He has been a visiting researcher of IRCS, the University of Pennsylvania in 1994. He is currently a professor of the Graduate School of Informatics in Kyoto University. His research interest include natural language processing and knowledge acquisition/representation.

(Received November 1, 2020)
(Revised February 9, 2021)
(Accepted March 25, 2021)