

Unsupervised Text Summarization of Long Documents using Dependency-based Noun Phrases and Contextual Order Arrangement

Yen-Hao Huang¹, Hsiao-Yen Lan², Yi-Shin Chen^{*}

Institute of Information Systems and Applications¹

Department of Computer Science^{2*}

National Tsing Hua University

Hsinchu, Taiwan

{yenhao0218¹, penny16335², yishin*}@gmail.com

Abstract

Unsupervised extractive summarization has recently gained importance since it does not require labeled data. Among unsupervised methods, graph-based approaches have achieved outstanding results. These methods represent each document by a graph, with sentences as nodes and word-level similarity among sentences as edges. Common words can easily lead to a strong connection between sentence nodes. Thus, sentences with many common words can be misinterpreted as salient sentences for a summary. This work addresses the common word issue with a phrase-level graph that (1) focuses on the *noun phrases* of a document based on grammar dependencies and (2) initializes edge weights by term-frequency within the target document and inverse document frequency over the *entire corpus*. The importance scores of noun phrases extracted from the graph are then used to select the most salient sentences. To preserve summary coherence, the order of the selected sentences is re-arranged by a flow-aware *orderBERT*. The results reveal that our unsupervised framework outperformed other extractive methods on ROUGE as well as two human evaluations for semantic similarity and summary coherence.

Keywords: Extractive Summarization, Graph, Dependency, Summary Coherence

1 Introduction

Text summarization helps in preserving and compressing representative information from long documents. This work aims at the extractive summarization, which condenses a document by extracting a few salient sentences.

It produces fluent sentences with less training data than abstractive methods.

Most extractive summarization research focuses on supervised learning methods (Narayan et al., 2018; Dong et al., 2018; Yao et al., 2018; Wu and Hu, 2018; Liu and Lapata, 2019) to derive models for automatically observing salient sentences based on specified golden labels. Nonetheless, it is impractical to expect the availability of such high-quality training datasets. on the rich unpaired data. In this method, researchers model textual content into sentence-level (Erkan and Radev, 2004; Mihalcea and Tarau, 2004; Mallick et al., 2019; Zheng and Lapata, 2019) or hybrid (Tarau and Blanco, 2019) graphs and adopt PageRank-based algorithms (Page et al., 1999) to retrieve the salient sentences in a document. Due to the characteristics of the graph, the extracted salient sentences are easily affected by common or function words that have high connectivities and are overestimated as key nodes. In addition, coherence is considered as an important attribute in summarization as it keeps the flow of concepts smooth and logical. However, few studies take coherence into consideration.

To address the above issues, this work assumes the major concepts in a document are expressed by key noun phrases and constructs a phrase-level graph specific for noun phrases that leverage grammar dependencies. With salient sentences extracted by key noun phrases, a sentence re-ordering step is applied to ensure the flow of concepts is contextually correct for the reader’s understanding. There are two major steps in the proposed framework: *key noun phrase extraction* and *salient sentence extraction*, as shown in Figure 1. Our

^{*}The corresponding author.

contributions are summarized as:

- An unsupervised extractive framework is proposed by constructing a novel phrase-level graph to obtain key noun phrases for salient sentence extraction. The proposed framework not only outperformed all extractive baselines on ROUGE, but also achieved results closer to the SOTA supervised transformer-based methods.
- The proposed orderBERT reorders the summary sentences with respect to sentence-level context, which improves 9% over using sentence’s original position in a human-reader evaluation.
- The proposed noun phrase hyper relation extraction method can obtain more relations and less duplicates. These rich relations then provide more nodes and edges information to the phrase-level graph.

2 Key Noun Phrase Extraction

Keyphrases represent important information in sentences and documents but not all of them contribute the same amount of information. With noun phrases indicated as the most commonly occurring structures among different types of corpora (Le et al., 2016), the proposed method assumes that they potentially provide coverage of the major conceptual points of the document. By focusing on extracting key noun phrases from documents, this work proposes a graph-based keyphrase extraction for noun phrases in an unsupervised manner.

The key noun phrase extraction is separated into two steps: noun phrase hyper relation extraction and graph-based keyphrase scoring. The former is designed to extract the noun phrases along with the relations between them in a complete as possible manner according to the grammar dependency. To extract important noun phrases and avoid an undue influence of common words, further relations are adopted as a guide in constructing the dependency graph for specific noun phrases.

2.1 Noun Phrase Hyper Relation Extraction

In traditional relation extraction methods, noun phrases are extracted based on co-occurrence within a predefined window size,

which might encounter a window size limitation; in other words, noun phrases whose relations are outside of the window size will be ignored. To overcome this limitation, this paper adapted the concept of open information extraction (OpenIE) to enable the extraction of both short and long relations between noun phrases based on grammar relations, which is denoted by relation triples as in Definition 1.

Definition 1. (*Relation Triple*). Let s , o , r , \mathbf{N} , and \mathbf{NP} denote a subject, object, their relation, the set of nouns, and the set of noun phrases, respectively. The relation triple set \mathbf{RT} is presented as follows:

$$\mathbf{RT} = \{(s, r, o) | s, o \in \mathbf{N} \cup \mathbf{NP}\} \quad (1)$$

The goal of phrase relation extraction is to retrieve a set of triples \mathbf{RT} from each sentence. However, existing OpenIE tools mainly focus on the direct relations between subjects, verbs, and objects. Consequently, complex relations cannot be captured, e.g. the intra-clause relation of two nouns and nested clauses. Except for adopting existing OpenIE tools, our approach proposes rules to capture complex relations based on the grammar dependencies as defined in Definition 2.

Definition 2. (*Grammar Dependency*). Let ζ be the grammar dependency type between a source word γ and a target word τ in a given sentence st . A set of grammar dependencies \mathbf{GD}_ζ with type ζ in a given sentence st is presented as $\mathbf{GD}_\zeta^{st} = \{(\gamma, \tau)\}$.

With the dependency parsing tool from Stanford CoreNLP (Manning et al., 2014), a set of dependencies \mathbf{GD}^{st} and the corresponding word pairs are first extracted. In Figure 2, the extracted dependencies can be presented as $\mathbf{GD}^{st} = \{\mathbf{GD}_{\text{NSUBJ}}^{st}, \mathbf{GD}_{\text{CASE}}^{st}, \dots, \mathbf{GD}_{\text{AMOD}}^{st}\}$, $\mathbf{GD}_{\text{CASE}}^{st} = \{(\text{success, for}), (\text{models, of})\}$. These dependencies \mathbf{GD}^{st} are then used to construct the triples \mathbf{RT} for each sentence by algorithms proposed in the following sections.

2.1.1 Inter-clause Relation

To extract relations in the same clause, the proposed procedures are shown below.

Definition 3. (*RT from Nominal Subjects*). Let NSUBJ , OBJ , OBL , XCOMP , COP , and AUX

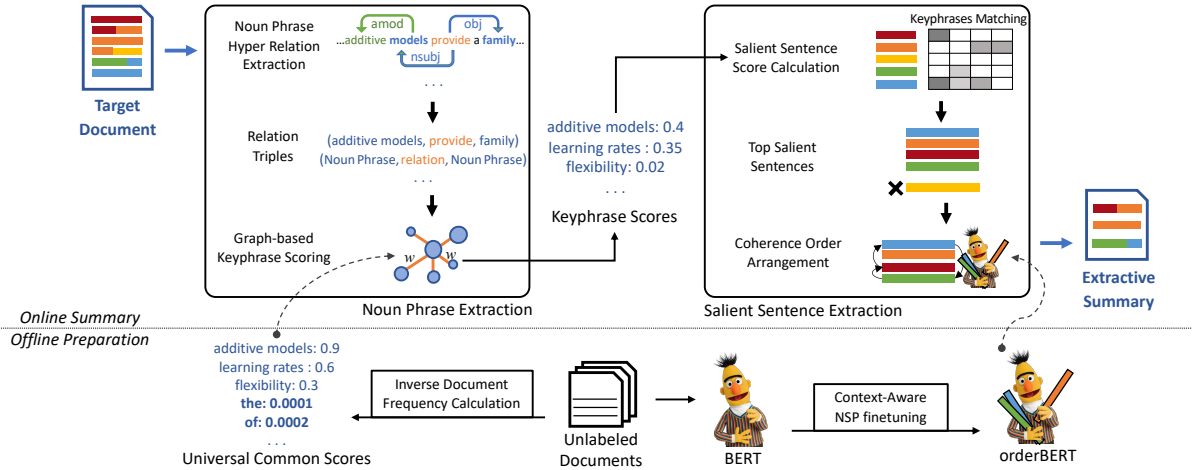


Figure 1: Overall framework flowchart

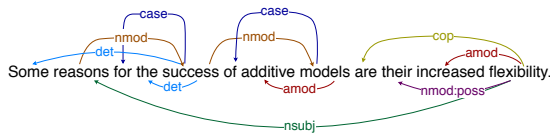


Figure 2: Example of dependency parsing

be the dependency types nominal subject, object, oblique nominal, open clausal complement, copula, and auxiliary. Let $\mathbf{GD}_{\text{NSUBJ}}$ denote the set of grammar dependencies with type NSUBJ, where the set consists of two types of word pairs: (γ_v, τ) , $(\gamma_v \text{ is verb})$; and (γ_n, τ) , $(\gamma_n \text{ is noun})$. Let $(\hat{\gamma}_v, \hat{\tau}) \in \mathbf{GD}_{\text{OBJ}} \cup \mathbf{GD}_{\text{OBL}}$, $(\tilde{\gamma}_v, \tilde{\tau}) \in \mathbf{GD}_{\text{XOMP}}$, and $(\check{\gamma}, \check{\tau}) \in \mathbf{GD}_{\text{COP}} \cup \mathbf{GD}_{\text{AUX}}$. Triples can be extracted as follows:

$$\mathbf{RT}_{\text{OBJ, OBL}}^{\text{NSUBJ}} = \{(\tau, \gamma_v, \hat{\tau}) | \gamma_v = \hat{\gamma}_v\} \quad (2)$$

$$\mathbf{RT}_{\text{XOMP}}^{\text{NSUBJ}} = \{(\tau, \tilde{\tau}, \hat{\tau}) | \gamma_v = \tilde{\gamma}_v, \tilde{\tau} = \hat{\gamma}_v\} \quad (3)$$

$$\mathbf{RT}_{\text{AUX, COP}}^{\text{NSUBJ}} = \{(\tau, \check{\gamma}, \gamma_n) | \gamma_n = \check{\tau}\} \quad (4)$$

Definition 4. (RT from Passive Nominal Subjects). Let NSUBJ:P denote the passive-nominal-subject type. Let $(\gamma, \tau) \in \mathbf{GD}_{\text{NSUBJ:P}}$, $(\hat{\gamma}, \hat{\tau}) \in \mathbf{GD}_{\text{OBL}}$, and $(\check{\gamma}, \check{\tau}) \in \mathbf{GD}_{\text{XOMP}}$. The triples can be extracted by

$$\mathbf{RT}_{\text{OBL}}^{\text{NSUBJ:P}} = \{(\tau, \gamma, \hat{\tau}) | \gamma = \hat{\gamma}\} \quad (5)$$

$$\mathbf{RT}_{\text{XOMP}}^{\text{NSUBJ:P}} = \{(\tau, \gamma, \check{\tau}) | \gamma = \check{\gamma}\} \quad (6)$$

Definition 5. (RT from Nominal Modifier). Let NMOD, CASE, PUNCT be dependencies types nominal modifier, case marking, and punctuation, respectively. Let $(\gamma, \tau) \in \mathbf{GD}_{\text{NMOD}}$ and $(\hat{\gamma}, \hat{\tau}) \in \mathbf{GD}_{\text{CASE}} \cup \mathbf{GD}_{\text{PUNCT}}$. Triples are built as $\mathbf{RT}^{\text{NMOD}} = \{(\gamma, \hat{\tau}, \tau) | \tau = \hat{\gamma}\}$.

Based on Definitions 3 to 5, these triples are first extracted and denoted as \mathbf{RT} for brevity.

However, some relations are still missing after these extraction processes, such as relations between nouns and the corresponding appositions, which are ignored. Two algorithms in Definitions 6 and 7 are then proposed.

Definition 6. (RT from Conjunction). Let CONJ denote the conjunction dependency type and $\tilde{\mathbf{RT}}$ denote the set of all extracted triples. The triple set $\mathbf{RT}^{\text{CONJ}}$ is extracted based on Algorithm 1 for each $(s, r, o) \in \mathbf{RT}$.

Algorithm 1 RT Construction from CONJ

```

for  $(\gamma, \tau) \in \mathbf{GD}_{\text{CONJ}}$  do
  if  $\gamma == r$  then
    if  $\tau.\text{isVerb}() \wedge \tau.\text{hasNoSubject}()$  then
      object  $\leftarrow$  getObj( $\mathbf{GD}_{\text{OBJ}}, \tau$ )
      return  $(s, \tau, \text{object})$ 
    else if  $\gamma == o$  then
      return  $(s, r, \tau)$ 

```

Definition 7. (RT from Appositional Modifier). Let APPOS denote the appositional-modifier dependency type. Let $(\gamma, \tau) \in \mathbf{GD}_{\text{APPOS}}$, $(s, r, o) \in \tilde{\mathbf{RT}}$, and ϕ denote an empty relation word. The triples are built as $\mathbf{RT}^{\text{APPOS}} = \{(\gamma, \phi, \tau) | \gamma = s \vee \gamma = o\}$.

2.1.2 Intra-clause Relation

For dependencies in the independence and subordinate clauses, the dependencies of which the object or subject in a subordinate clause provides complements for an independent clause are leveraged. Two subordinate clauses are considered and defined below.

Definition 8. (RT from Adjective Clausal Modifier of Noun). Let ACL be the adjective-clausal-modifier dependency type. Let $(\gamma, \tau) \in \mathbf{GR}_{\text{ACL}}$, $(\hat{\gamma}, \hat{\tau}) \in \mathbf{GR}_{\text{OBJ}}$. Triples are extracted as $\mathbf{RT}^{\text{ACL}} = \{(\tau, \gamma, \hat{\tau}) | \gamma = \hat{\gamma}\}$.

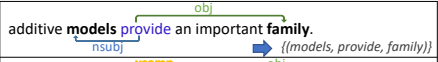
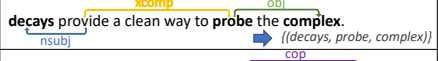
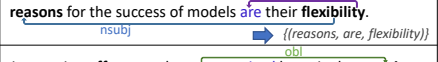


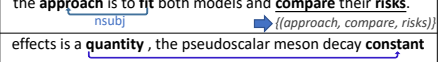
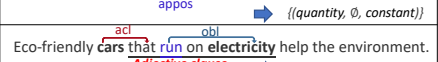
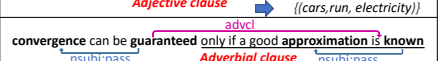

$RT^{NSUBJ}_{OBJ,OBL}$	additive models provide an important family . 
RT^{NSUBJ}_{XCOMP}	decays provide a clean way to probe the complex . 
RT^{NSUBJ}_{COP}	reasons for the success of models are their flexibility . 
$RT^{NSUBJ}_{P}_{OBL}$	interaction effects can be parametrized by a single quantity . 
RT^{NMOD}	reasons for the success of models are their flexibility . 
RT^{CONJ}	the approach is to fit both models and compare their risks . 
RT^{APPOS}	effects is a quantity , the pseudoscalar meson decay constant . 
RT^{ACL}	Eco-friendly cars that run on electricity help the environment. 
RT^{ADVCL}	convergence can be guaranteed only if a good approximation is known . 

Figure 3: Examples of RT construction

Definition 9. (*RT from Adverbial Clause Modifier*). Let $ADVCL$ and $ADVMOD$ be dependency types adverbial clause modifier and adverbial modifier, respectively. Let \mathbf{RT} be all the extracted triples. The subsets of the triples are built by Algorithm 2 for each $(\gamma, \tau) \in \mathbf{GR}_{ADVCL}$ and all the triples are merged as \mathbf{RT}^{ADVCL} .

Algorithm 2 RT Construction from $ADVCL$

```

tmpRT = ∅
for (s, r, o) ∈ RT do
  if γ == r then
    adverb ← getAdv(GRADVMOD, τ)
    object ← getObj(GROBJ ∪ GROBL, τ)
    tmpRT.add((s, adverb, object))
    continue
  for (γ̂, τ̂) ∈ GRNSUBJ:P do
    if γ == γ̂ then
      object ← getObj(GROBJ ∪ GROBL, τ)
      adverb ← getAdv(GRADVMOD, τ)
      tmpRT.add((τ̂, adverb, object))
return tmpRT

```

As the adjective clauses provide extra information for the noun, there exists relations between the corresponding object of the verb in an adjective clause and the noun. Such relations could be extracted by Definition 8. For an adverbial clause, there are two major cases to be captured. First, all the extracted triples are considered, as an adverbial clause describes the conditions or reasons (such as *if* or *since*) for the action of a subject with regard to the corresponding object in the independent clause. Second, for a subject in the passive voice, it may not have a corresponding object and thus cannot be extracted by the previous methods. The details and an example are shown in Definition 9 and Figure 3, respec-

tively. For brevity, all the extracted triples to this end are denoted as \mathbf{RT} .

Lastly, due to the design of CoreNLP, the extracted (noun) words in \mathbf{RT} are still uni-grams. To present the noun phrase, a uni-gram is combined with the previous word if there exists a grammar type for a compound or adjectival modifier. Let a target word be the subject s or object o from an extracted triple $(s, r, o) \in \mathbf{RT}$ of the sentence st . If the relation r between the target word and of any its previous word is compound, this work considers that this previous word is able to modify the meaning of the target word and, thus, combine these words.

2.2 Graph-based Keyphrase Extraction

Let \mathbf{S} , \mathbf{R} , and \mathbf{O} denote the sets of subjects, relations, and objects in the extracted relation triples \mathbf{RT} from all the sentences in a document, respectively. A bi-directional graph G is built as $G = (\mathbf{V}, \mathbf{E})$, where \mathbf{V} represents noun phrase nodes and \mathbf{E} denotes the edges, such that $\mathbf{V} = \mathbf{S} \cup \mathbf{O}$ and $\mathbf{E} = \mathbf{R}$.

With all the triples \mathbf{RT} transformed into a graph G , a keyphrase extraction method is proposed to retrieve important noun phrases through this graph. The PageRank algorithm (Page et al., 1999) was adapted to score all the nodes in the graph. However, due to the nature of PageRank, the score of common phrases could be too high as they have more edges than other nodes. Thus, to avoid this common word issue, the term-frequency inverse document frequency (TFIDF) ratio was adapted for the edge weight in advance from all the training documents. It is computed by:

$$TFIDF(w) = \log_2(freq(w)) * \log_2 \frac{|D|}{df(w)} \quad (7)$$

where $freq(w)$ denotes the frequency of word w in the source document, $|D|$ represents the number of total source documents in the collection, and $df(w)$ is the number of documents that contain the word w . It is worth noting that the although common words have lower IDF values, they still achieve high scores due to their overly high frequencies. The \log function is thus adopted for $freq(w)$. The TFIDF scores are added into

graph $G = (\mathbf{V}, \mathbf{E}, \Theta)$ as edge weights $\Theta = \{\theta\}$ by the following equation:

$$\theta_{i,j} = |Relation_{i,j}| * \frac{TFIDF_j}{TFIDF_i + TFIDF_j} \quad (8)$$

where $\theta_{i,j}$ is the edge weight, and $|Relation_{i,j}|$ is the number of relations among nodes v_i, v_j . PageRank is adopted in the end to obtain the importance scores for noun phrase nodes.

3 Salient Sentence Extraction

Consistent with our assumption that noun phrases potentially provide coverage of the major conceptual aspects of a document, the salient sentences are also selected based on these noun phrases with the corresponding importance scores derived by the weighted graph. In the following section, a context-aware BERT is derived to perform sentence reordering for the top salient scores in order to maintain summary coherence.

3.1 Salient Sentence Score Calculation

With important scores of noun phrases, they are utilized to calculate the salient score for each sentence in the document. Let $d = \{st_i\}$, $st_i = \{p_k\}$, $G^{(d)}$, and $\mathbf{V}^{(d)}$ be a document, sentence, graph, and set of nodes, respectively, where st_i represents the i^{th} sentence in document d , p_k denotes the k^{th} noun phrase in st_i , $G^{(d)}$ denotes the graph constructed by d , and $\mathbf{V}^{(d)} \in G^{(d)}$. Sum aggregation was applied to score the salience value for a given sentence:

$$Score_{sent_n} = \sum_{p_k \in (sent_n \cap \mathbf{V})} Score_{p_k} \quad (9)$$

where $Score_{p_k}$ is the importance score of noun phrase p_k calculated by $G^{(d)}$. Note that different sentence scoring methods were also proposed, such as the average aggregation, yet the sum aggregation performs the best.

3.2 Coherence Order Arrangement

With the salient score of each sentence in a document, sentences are ranked according to their scores. An intuitive solution to maintain the summary coherence in the extractive summarization is to reorder these top- n sentences according to their original position order in the source document (Zhong et al., 2020). However, the flow of selected sentences might be

disrupted and, hence, damage the readability of the generated summary. With regard to the challenge, a flow-aware **orderBERT** is proposed for sentence order arrangement.

3.2.1 orderBERT

The orderBERT is a fine-tuned BERT by a modified objective of the next sentence prediction (NSP) (Devlin et al., 2019). Given a sentence pair (st_α, st_β) , the goal of NSP is to predict whether the second sentence st_β is the sentence after the first sentence st_α .

In the original NSP, its negative samples are sentence pairs sampled from different documents. However, these training data may result in two objectives while pretraining: (1) BERT can successfully classify the ‘‘order’’ and ‘‘context’’ in which the given sentences are in the incorrect order or from different documents are classified as negative; instead, (2) it only predicts whether two input sentences originate from the same document or from different ones. It is difficult to ensure that BERT is aware of the order of the given sentences. Thus, this study finetunes BERT by the sentence order based on a **context-aware NSP fine-tuning** strategy. Specifically, as additional negative samples, a set of sentence pairs are constructed by inverse order of two consequent sentences within a single source article. The number of inverse-order false samples are set to be the same as the number of original consequent sentence pairs, as done in the original work (Devlin et al., 2019).

To this end, the training dataset contains (1) correct order consequent sentence pairs within a document (positive sample), (2) sentence pairs from different documents (negative sample), and (3) *inverse-order* consequent sentence pairs within a document (negative sample). With this training set, *orderBERT* was trained to predict whether the second sentence is next in order after the first sentence (Huang et al., 2021), thereby going beyond merely recognizing whether or not the sentences are from the same document. Note that this process remains unsupervised since its labels are obtained naturally from documents.

3.2.2 Summary Sentence Reordering

With the trained *orderBERT*, given a pivot sentence and a set of candidate sentences, we

let orderBERT go through all the pairs of pivot sentence and candidates to obtain the most suitable candidate connected after the pivot sentence. Finally, given a set of salient sentences, *orderBERT* then maintains the coherence of summary by re-ordering the extracted salient sentences, as defined in Algorithm 3.

Overall, after the sentences reordering, a machine-generated extractive summary is obtained in an unsupervised manner, in which the summary comprises of n number of salient sentences from the original document. It is important to note that the salient sentences are selected based on the key noun-phrases and, thus, several important terminologies are present in each sentence of the summary. For the summary coherence, the improvement is not only contributed by sentence rearrangement step, the important terms (noun phrases) also play important role in connecting the concept through different sentences while the readers go through the summary.

Algorithm 3 Sentence Reordering

```
ST = Salient sentence list ordered by each one's original positions
stpivot = ST.deque()
ordered_ST = [stpivot]
while len(ST) ≠ 0 do
    stpivot = GETNEXTBYORDERBERT(stpivot, ST)
    ST.remove(stpivot)
    ordered_ST.append(stpivot)
return ordered_ST
```

4 Experimental Setup

4.1 Dataset and Preprocessing

This work focuses on long document summarization as the key concepts in long documents are more dispersed than in short ones. Two different long-document datasets, PubMed and arXiv from Cohan et al. (2018), were considered with the introductions as the source documents and their abstracts as the summaries. For pre-processing, documents with its introduction less than 10 sentences were removed, as there were an insufficient number of sentences from which to select. The statistics of the datasets are summarized in Table 1.

4.2 Baselines

To evaluate performance, we compared our framework with different baselines as follows:

(a) **LEAD-5** and (b) **ORACLE** generally represent the lower-bound and upper-bound of extractive summarization tasks; for unsupervised methods, we adopted (c) **TextRank** (Mihalcea and Tarau, 2004) with co-occurrence relations with window size set at 2 for graph-based keyphrases and Equation 9 for sentences scores; (d) **DeepRank** (Tarau and Blanco, 2019) contains a word-sentence heterogeneous graph with PageRank for sentence scores; (e) **LexRank** (Erkan and Radev, 2004) is a sentence-level undirected graph with edge weight threshold set to 0.1 according to its paper and calculates a cosine similarity between sentences; and (f) **PacSum** (Zheng and Lapata, 2019) builds a sentence-level directed graph with TFIDF or BERT for the edge weights. For supervised methods, the following were adopted: (g) **Pointer Generator** (See et al., 2017) with attention and beam search algorithm; (h) **BertSum** (Liu and Lapata, 2019), three SOTA BERT-based models for both extractive and abstractive summarization that included BertExt, BertAbs, and BertExtAbs. The summary of each extractive method can contain at most 5 sentences.

For evaluation, ROUGE 1, 2, and L (Lin and Och, 2004) were first applied to examine the information-preserving capabilities. Secondly, a human evaluation of the coherence of the summaries was conducted.

5 Results and Discussion

5.1 Model Performance on ROUGE

The performance comparison for different methods on two datasets is demonstrated in Table 2. Overall, the proposed unsupervised keyword-based method outperformed all the extractive summarization baselines, including the SOTA transformer method, namely BertExt. For the BertAbs and BertExtAbs models that were trained under supervision, it is worth mentioning that our method still outperformed both of them with the PubMed dataset. As the size of the data in arXiv was ten times more than the PubMed dataset, BertAbs and BertExtAbs largely benefited from the supervised learning process; our methods then performed slightly worse than them. However, this still indicates that by leveraging key noun-phases using grammar de-

Dataset	# of Doc. train/valid/test	Avg. Abstract		Avg. Introduction		Avg. Doc.
		# word	# sentence	# word	# sentence	# word
PubMed	10k / 2k / 1.25k	201.9	6.8	1013.1	37.3	3224.4
arXiv	83k / 19.8k / 20k	177.7	6.6	1077.0	42.8	6913.8

Table 1: Dataset statistics

Method	Type	PubMed			arXiv		
		ROUGE-1	ROUGE-2	ROUGE-L	ROUGE-1	ROUGE-2	ROUGE-L
LEAD-5	*	0.2999	0.0865	0.2695	0.0137	0.0003	0.0137
ORACLE	*	0.4490	0.1817	0.3604	0.4610	0.1994	0.2784
TextRank	Unsup.	0.3514	0.0944	0.3115	0.3424	0.0972	0.3035
LexRank		0.3936	0.1169	0.3469	0.3592	0.1000	0.3151
DeepRank		0.3029	0.0651	0.2652	0.3257	0.0841	0.2861
PacSum (TFIDF)		0.3650	0.0904	0.3237	0.3835	0.1131	0.3341
PacSum (BERT)		0.3093	0.0677	0.2777	0.3595	0.1005	0.3146
Pointer Generator	Sup.	0.2999	0.0865	0.2695	0.3554	0.1255	0.3192
BertExt		0.3249	0.1012	0.2863	0.3829	0.1324	0.3311
BertAbs		0.3199	0.0730	0.2909	0.4105	0.1512	0.3667
BertExtAbs		0.3485	0.0802	0.3136	0.4269	0.1598	0.3802
Ours	Unsup.	0.3999	0.1174	0.3504	0.4075	0.1347	0.3569

Table 2: Overall performance on ROUGE

Method	Avg. # of Noun	
	arXiv	PubMed
DeepRank	29.3	35.6
LexRank	45.3	53.3
PacSum (TFIDF)	57.3	65.2
PacSum (BERT)	40.8	42.4
TextRank	38.8	49.0
Ours	64.5	71.0

Table 3: Noun usages of summaries

dependencies, there is a chance for unsupervised method to perform similarly to a supervised and pretrained method. In addition, one possible reason for the good performance with the more limited dataset (PubMed) was the usage of the grammar dependencies in constructing the graph for the keyphrases. Specifically, the rich grammar relations lay in the language usage implicitly, which allows our models to capture the key concepts of a document. The remaining evaluations focus on the comparisons among unsupervised baselines.

5.2 Noun for Information Preserving

As this work focuses on the representative noun phrases for concept preserving. Statistics were conveyed on the average frequencies of nouns from all graph-based baselines as shown in Table 3. This showed that the summary generated by our method contained mostly words that were nouns, which probably helped our method to preserve most of the concepts and obtain the best ROUGE performance.

5.3 Graph Construction Comparison

The other phrase-level method, TextRank, did not have as good a performance as ours. The only difference between our framework and TextRank is the way the phrase-level graph is constructed. Our framework utilizes rule-based relation extraction from grammar relations, while TextRank applies co-occurrence relations. To compare the differences, graphs were visualized with the same sentence as shown in Figure 4. In Figure 4a, there are only four adjective-noun combinations. The co-occurrence relations ignore many important relation between phrases due to the limited window size. In contrast, the graph by our method (Figure 4b) contains more relations between phrases that contributes to a dense graph and benefits for the keyword extraction.

As compared to a heterogeneous graph, DeepRank built a graph from both words and sentences. As there are many edges that connect from keyphrases to sentences, it results in the scores of important keyphrases being distributed uniformly to these sentences. The top-five salient sentences were examined as to whether they contained the top keyphrase. The average keyphrase counts in top-5 sentences from DeepRank were 0.748 and 0.864 while our method obtained 3.003 and 3.321 on arXiv and PubMed, respectively. This also indicates that it is better to separate phrases and sentences for summarization.

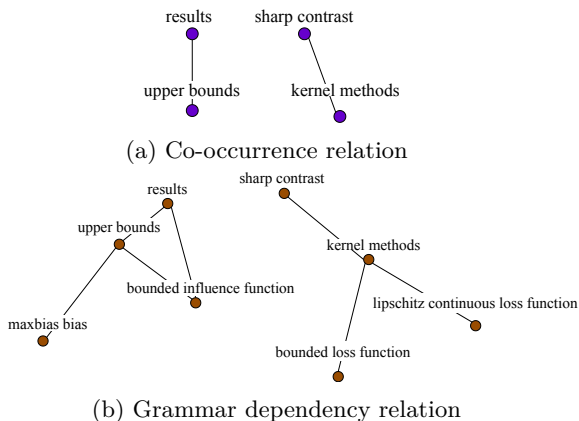


Figure 4: Graphs built from an example sentence

5.4 Human Questionnaire Evaluation

Human evaluation was conducted on Amazon Mechanical Turk (AMT) to compare the semantic similarity to gold summary and coherence performance among baselines that had the best ROUGE score or was the most coherent. An abstract (golden summary) and multiple summaries from the baselines were provided for each question. There were a total of 10 documents that were randomly sampled from arXiv and PubMed in the same proportion and assigned to 100 AMT workers for evaluations. Note that 21 workers were discarded as they submit inattentive answers to the questionnaire including the behaviors of quick answering, the same answer for all questions, and wrong answer for the trap question. The results from both datasets are together in Table 4.

Method	Chose Ratio	
	Similarity	Coherence
LexRank	25.96%	28.79%
PacSum (TFIDF)	19.61%	19.33%
Ours + Orig. Pos.	27.22%	21.17%
Ours + orderBERT	27.21%	30.71%

Table 4: Percentage of human-preferred methods

From the semantic similarity question, our proposed methods outperform other unsupervised baselines. Noting that the sentences of two summaries generated by our methods are identical, only the orders are different. Therefore, their percentages are, therefore, almost the same—27.22% and 27.21%. It indicates that labelers struggled to select one of ours as the best semantic-similar summary from all options; with two of our methods together, most

labelers selected our summaries as the most similar. For LexRank and PacSum (TFIDF), although they are comparable in ROUGE evaluation, the summaries by LexRank were more preferred by human readers.

With regard to summary coherence, with sentence re-ordering and key noun phrases, our method with *orderBERT* had better performance than others in terms of coherence evaluation. The results of two our methods also indicate a 9.5% improvement in coherence on the chosen ratio with the BERT reordering mechanism as compared to the summary that only reordered based on its original position (Ours + Orig. Pos.). Although adopting the original position for reordering works well in short document summarization, it may not be suitable to directly adopt for a long document. Interestingly, LexRank also obtained good results in the coherence questions. It is found that LexRank tends to select a few sentences with connecting/turning words that are helpful for coherence between sentences.

Example summaries are shown in Table 5. It is observed that the summary reordered by the original positions has multiple topic shifts and repetitions. The topic shifts from *model-checking problem* to *timed automata*, then to *model-checking problem*, and then to *timed automata* again. As for the summary by *orderBERT*, the topic first focuses on the *model-checking problem* and then provides the link between *timed automata* and *model-checking problem* instead of switching the topics between them. This shows that the original position method may produce topic gaps between salient sentences as there is more content in a long document. By reordering sentences at the sentence level, the mechanism with *orderBERT* could alleviate such an issue. Overall, the combined use of noun phrases and sentence reordering with *orderBERT* could provide better readability with respect to summary coherence than the other baselines.

5.5 Relation Extraction Comparison

To evaluate the proposed phrase extraction method, the latest OpenIE tool given by Stanford CoreNLP was compared as in Table 6.

For the Stanford CoreNLP tool, there was no triple extracted in Case 1 and all the results in Case 2 were almost the same. Al-

LexRank	For this class of parametric timed automata , they focus on the emptiness problem: are there concrete values for the parameters so that the automaton has an accepting run? they show that when only one clock is compared to parameters, the emptiness problem is decidable. The model-checking problem for tctl extended with parameters over discrete- and dense- timed automata (without parameters) is decidable. Unfortunately, in all those previous works, the parameters are only in the model (expressed as a timed automaton) or only in the property (expressed as a temporal logic formula). Nevertheless, when expressing a temporal property of a parametric system, it is natural to refer in the temporal formula to the parameters used in the system.
PacSum	In fact, the control has to leave <i>equation</i> at most <i>equation</i> time units after entering it and the control has to stay exactly <i>equation</i> time units in state <i>equation</i> . Let us consider the next three formulae for configuration <i>equation</i> , i.e. the control is in state <i>equation</i> and clock <i>equation</i> has value <i>equation</i> : a. <i>Equation</i> the parameter synthesis problem associated to formula <i>equation</i> , asks for which values of <i>equation</i> and <i>equation</i> , the formula is true at configuration <i>equation</i> . Formula <i>equation</i> formalizes the next question “in all the cases where the value assigned to parameter <i>equation</i> is greater than the value assigned to parameter <i>equation</i> , is it true that any cycle has a duration bounded by <i>equation</i> ”. On the positive side, we show that the model-checking problem becomes decidable and parameter synthesis problem is solvable for a fragment of logic where the equality is not allowed.
Ours +Orig.Pos.	In this paper, we further investigate the model-checking problem of real-time formalisms with parameters. For this class of parametric timed automata , they focus on the emptiness problem: Are there concrete values for the parameters so that the automaton has an accepting run? They show that when only one clock is compared to parameters, the emptiness problem is decidable. The model-checking problem for tctl extended with parameters over discrete- and dense- timed automata (without parameters) is decidable. In this paper, we study the model-checking problem of the logic tctl extended with parameters over the runs of a discrete-timed automaton with one parametric clock. On the negative side, we show that the model-checking problem of tctl extended with parameters is undecidable over timed automata with only one parametric clock.
Ours +order-BERT	In this paper, we further investigate the model-checking problem of real-time formalisms with parameters. On the negative side, we show that the model-checking problem of tctl extended with parameters is undecidable over timed automata with only one parametric clock. The model-checking problem for tctl extended with parameters over discrete- and dense- timed automata (without parameters) is decidable. In this paper, we study the model-checking problem of the logic tctl extended with parameters over the runs of a discrete-timed automaton with one parametric clock. For this class of parametric timed automata , they focus on the emptiness problem: Are there concrete values for the parameters so that the automaton has an accepting run? They show that when only one clock is compared to parameters, the emptiness problem is decidable.

Table 5: Example Summaries from Unsupervised Methods (Key noun-phrases are highlighted in bold).

Case 1	of such estimators belong to the large class of regularized kernel based methods over a reproducing kernel hilbert space.
CoreNLP Ours	Not available. (Examples, of, such estimators), (Examples, belong to, large class), (large class, of, regularized kernel), . . .
Case 2	It is also a minimizer of the following optimization problem involving the original loss function .
CoreNLP Ours	(It, is minimizer of, optimization problem), (It, is minimizer of, following optimization problem), (It, is also minimizer of, optimization problem), . . . (It, is minimizer), (minimizer, of, following optimization problem), (following optimization problem, involving, original loss function)

Table 6: Phrase Relation Extraction Comparison

though such duplication could be solved by post-processing, missing relations (such as the relation between a noun in a main clause and a noun in an adjective clause) were still not found. In addition, our framework could extract more useful triples from these cases.

6 Conclusion

In this research, a fully unsupervised framework is proposed for extractive summarization. The proposed method addresses the common word domination issue from a graph-based approach by using a phrase-level graph that focuses on key noun phrases based on grammar dependencies. The extracted key noun phrases effectively capture the major concepts of a document and can be used to construct an extractive summary. Experiments showed that the proposed method outperformed all the extractive baselines, even for supervised methods. A human evaluation also showed that the use of keyphrases and sentence reordering successfully benefited the coherence of the summaries. In the future, we aim to adapt the proposed key noun-phrases for unsupervised abstractive summarization.

References

- Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Yue Dong, Yikang Shen, Eric Crawford, Herke van Hoof, and Jackie Chi Kit Cheung. 2018. **Bandit-Sum: Extractive summarization as a contextual bandit**. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3739–3748, Brussels, Belgium.
- Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research*, 22:457–479.
- Yen-Hao Huang, Ratana Pornvattanavichai, Fernando Henrique Calderon Alvarado, and Yi-Shin Chen. 2021. **Unsupervised multi-document summarization for news corpus with key synonyms and contextual embeddings**. In *Proceedings of the 33rd Conference on Computational Linguistics and Speech Processing (ROCLING)*, pages 192–201.
- Tho Thi Ngoc Le, Minh Le Nguyen, and Akira Shimazu. 2016. Unsupervised keyphrase extraction: Introducing new kinds of words to keyphrases. In *Australasian Joint Conference on Artificial Intelligence*, pages 665–671. Springer.
- Chin-Yew Lin and FJ Och. 2004. Looking for a few good metrics: Rouge and its evaluation. In *Ntcir Workshop*.
- Yang Liu and Mirella Lapata. 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)*, pages 3730–3740, Hong Kong, China. Association for Computational Linguistics.
- Chirantana Mallick, Ajit Kumar Das, Madhurima Dutta, Asit Kumar Das, and Apurba Sarkar. 2019. Graph-based text summarization using modified textrank. In *Soft computing in data analytics*, pages 137–146. Springer Singapore, Singapore.
- Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.
- Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*, pages 404–411.
- Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Ranking sentences for extractive summarization with reinforcement learning. 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 1747–1759.
- Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.
- Abigail See, Peter J Liu, and Christopher D Manning. 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)*, pages 1073–1083.
- Paul Tarau and Eduardo Blanco. 2019. Dependency-based text graphs for keyphrase and summary extraction with applications to interactive content retrieval. *arXiv preprint arXiv:1909.09742*.
- Yuxiang Wu and Baotian Hu. 2018. Learning to extract coherent summary via deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Kaichun Yao, Libo Zhang, Tiejian Luo, and Yanjun Wu. 2018. Deep reinforcement learning for extractive document summarization. *Neurocomputing*, 284:52–62.
- Hao Zheng and Mirella Lapata. 2019. Sentence centrality revisited for unsupervised summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6236–6247.
- Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. 2020. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208.