Uncover COVID-19 Patterns from Open Research Articles by Text Summarization

Qianlin Liang
University of Massachusetts, Amherst
qliang@cs.umass.edu

Abstract

COVID-19 pandemic is an emerging and rapidly evolving situation. In response to this situation, research groups have collected an enormous volume of scholarly related articles. However, accessing such vast body of biomedical literature may be inconvenience and ineffective without using summarize tools. In this project, we develop a graph-based algorithm to extractively summarize biomedical research literatures. We employ biomedical domain-specific knowledge by incorporating the Unified Medical Language System (UMLS) into our algorithm. We evaluate our algorithm using the CORD-19 dataset. The result shows that our algorithm can successfully identify sentences cover the main topic of the document.

1 Introduction

In the past 3 months, Coronavirus disease 2019 (COVID-19) has spread globally and resulting in the coronavirus pandemic. By the end of May 1st, 2020, more than 3.34 million cases have been reported across 187 countries, resulting in more than 238,000 deaths(WHO, 2020).

In response to this pandemic, the White House and a coalition of leading research groups have prepared the COVID-19 Open Research Dataset (CORD-19), which contains over 59,000 biomedical research literatures, including 47,000 with full text, about COVID-19, SARS-CoV-2 and related coronavirus (Wang et al., 2020). New insights may be generated from these literatures in support of the ongoing fight against the disease. However, the enormous volume of literatures makes it inconvenience and ineffective to access the key information of the papers. Text summarization may help to solve this problem.

Automatic text summarization is a promising method for helping researchers to efficiently obtain the key information in a given topic (Mishra et al., 2014).The goal of text summarization is to present a subset of the source text, which expresses the most important points. The reduction of data accomplished by text summarization aims to allow users to identify and process relevant information more quickly and accurately without having to read the entire document. Moreover, automatic summaries have been shown to improve indexing and categorization of biomedical literature(Gay et al., 2005).

Text summarization has been studied for a long time by NLP community(Nazari and Mahdavi, 2019). There exist many effective text summarization algorithms. However, most of them only use information that can be directly extracted from documents, such as terms, part-of-speech tags and sentence positions. Recent studies have shown the benefits of summarizing based on richer representations that take advantage of domain-specific knowledge source (Plaza et al., 2011). They retrieve concepts and relations which are derived from Unified Medical language System (UMLS) to construct a semantic graph and then apply a clustering algorithm to identify themes and topics within the text. Finally, they assign sentences to each of the cluster and extract salient sentences.

In this project, our goal is to reproduce the result demonstrated in (Plaza et al., 2011). We study how to incorporate domain-specific knowledge with NLP technique to improve summarization process and solve read world problems. We define our task as follow: given a biomedical research literature with full text, retrieve origin salient sentences that represent the main points of the literature. This report is organized as follows: Section 2 describe the background and of text summarization and UMLS. Section 3 shows details of our algorithm. Section 4 presents experiment and evaluation results. Section 5 includes discussion and limitation of the methods.

Input: Boris Johnson has said the UK is "past" the peak of its coronavirus outbreak and is looking towards a lockdown exit strategy, as he led the British government's coronavirus briefing for the first time since recovering from Covid-19.

Abstract: The UK has passed its coronavirus peak, Boris Johnson says.

Extract: Boris Johnson has said the UK is "past" the peak of its coronavirus outbreak and is looking towards a lockdown exit strategy

Figure 1: Example of text text summarization outputs. *extractive* methods output salient sentences from input and *abstractive* methods generate new representative content.

The final section concludes our project.

2 Background

In this section, we first demonstrate related work in text summarization then we introduce the Unified Medical Language System (UMLS).

2.1 Related work

Text summarization methods can be classified as two categories, *abstract* and *extract*. An *abstract* summary generates new content inferred from the input document(s) while an extractive summary verbatim fragment from the input document(s). Figure 1 shows an example of outputs of these methods. Although human summaries are generally *abstract* and it is the preferred output of text summarization, few studies based on abstractive methods in biomedical text summarization domain. This may because of the difficulties of identifying the salient concepts in the source, constructing the semantic representation and rewriting the summary. It might also due to the lack of annotated dataset. Abstractive methods are generally supervised, which requires a labeled training set. However, summarizing biomedical research literatures requires expert knowledge and is extremely time-consuming. Thus, collecting such dataset seems impractical. As a result, extractive summarization has been shown to outperform abstractive summarization in many cases(Mani and Maybury, 1999).

There are a variety of extractive text summarization methods in biomedical domain. (Sarkar, 2009) employ features, such as term frequency, title and position, and mathematical formula to produce scores for each sentences and extract the sentences with highest scores. (Chuang and Yang, 2000) employ a supervised learning approach. The authors train a classifier to predict if a sentence is salient based on feature vectors. (Plaza et al., 2011) extract salient sentences using feature vectors with extra knowledge source, UMLS, which we will discuss in next subsection.

2.2 Unified Medical Language System

The Unified Medical Language System (UMLS) (Bodenreider, 2004) is a set of files and software that brings together many health and biomedical vocabularies and standards to enable interoperability between computer systems. The UMLS integrates over 2 million names for some 900,000 concepts from more than 60 families of biomedical vocabularies as well as 12 millions relations among these concepts. The UMLS is consisted of three parts: SPECIALIST Lexicon and Lexical Tools, Metathesaurus and Semantic Network.

The SPECIALIST Lexicon and Lexical Tools(NLM, 2020a) is a large syntactic lexicon of biomedical and general English and tools for normalizing strings, generating lexical variants and creating indexes. It allows users to generate lexical variants based on inflection (e.g. plural vs. singular) and derivation (e.g. the adjectival form of a noun), as well as to perform other tasks useful for term mapping such as removing stop words.

The UMLS Metathesaurus(NLM, 2020b) is the major component of UMLS. It provides not only terms and codes from more than 100 vocabularies, but also hierarchies, definitions and other relationships and attributes among the concepts.

The UMLS Semantic Network(NLM, 2020c) consists of a set of broad subject categories, or Semantic Types that provide a consistent categorization of all concepts represented in the UMLS Metathesaurus and a set of useful and important relationships, or Semantic Relations, that exist between Semantic Types.

The National Library of Medicine also provides a series of related tools to make UMLS more handy. In this project, we use MetaMap(Aronson, 2001) to extract Metathesaurus concepts from input text. The input of MetaMap can be text of variable length and its output is a ranked list of Metathesaurus concepts associated with each piece of text. Figure 2 shows an example of MetaMap mapping for the input *coronavirus disease*. The phrase is mapped to 6 biomedical concepts in Metathesaurus. Each of them is assigned a score based on the closeness of the match. Three of these concepts are excluded. Since there exist two concepts map to the same term (i.e. *coronavirus*) and both of them have the

```
Phrase: "coronavirus disease"
Meta Candidates (Total=6; Excluded=3; Pruned=0; Remaining=3)
   861   Disease [Disease or Syndrome]
   805 E MALS (MALARIA, MILD, SUSCEPTIBILITY TO) [Finding]
   805 E MALS (NCR3 wt Allele) [Gene or Genome]
   805 E MAL (CD8A wt Allele) [Gene or Genome]
   694   Coronavirus (Genus: Coronavirus) [Virus]
   694   Corona virus (Coronavirus Infections) [Disease or Syndrome]
Meta Mapping (888):
   694   Coronavirus (Genus: Coronavirus) [Virus]
   861   Disease [Disease or Syndrome]
Meta Mapping (888):
   694   Corona virus (Coronavirus Infections) [Disease or Syndrome]
   861   Disease [Disease or Syndrome]
```

Figure 2: An example of MetaMap mapping for the input text *coronavirus disease*. The concept names are shown in parentheses and semantic types are shown in brackets.

same score, MetaMap return two mappings.

3 Methods

In this section, we describe our method in details. Similar to the origin paper, our algorithm has 7 steps: 1) preprocessing; 2) concept mapping; 3) sentence representation; 4) document representation; 5) concept clustering; 6) sentence-to-cluster assignment; 7) sentence selection. The overview of our algorithm is shown in figure 3.

3.1 Preprocessing

In this step, we perform the following operations to make the document ready for subsequent steps:

1. Extract the title, abstract and body text from the document. The input document may have other sections such as an acknowledgement section. But these sections generally contain little information about the main content of the article. Thus, we decide to drop them.

2. Remove the reference and citation spans. Biomedical research literatures often include citation and reference in their body paragraphs. These spans provide little information, and thus we remove them.

3. Remove non ASCII tokens. Since MetaMap and our algorithm only support English text, we remove non ASCII tokens from the document. We use (Honnibal and Montani, 2017) for tokenization and non ASCII token removing.

4. Different from the origin paper, instead of performing sentence splitting and stop word removing in this phase, we leave them for MetaMap to make them be consistent with the system.

After preprocessing, only the body text is fed into the subsequent steps.

3.2 Concept Mapping

In this step, we feed our preprocessed document into MetaMap, which maps phrases to Concept Unique Identifiers (CUI) and their corresponding semantic types. MetaMap also split document into sentences and return mappings corresponding to each sentences.

In this project, we use 2018 version MetaMap and 2019AB version UMLS, which are the latest version to date. We use MetaMap with a `--word_sense_disambiguation` option, which force the program returns the most likely semantic type for given context. However, MetaMap may still fail to return a single mapping when multiple semantic types have the same score. This is exactly the case shown in figure 2. The phrase *coronavirus* can be mapped equally likely to `[Virus]` or `[Disease or Syndrome]`. In this case, we choose the first mapping. We also use `--ignore_stop_phrases` option to remove stop words in input document.

Moreover, similar to the origin paper, some semantic types are too broad to be counted. Thus, we ignore these concepts in this project. Ignored concepts include: *quantitative concept, qualitative concept, tempo- ral concept, functional concept,*

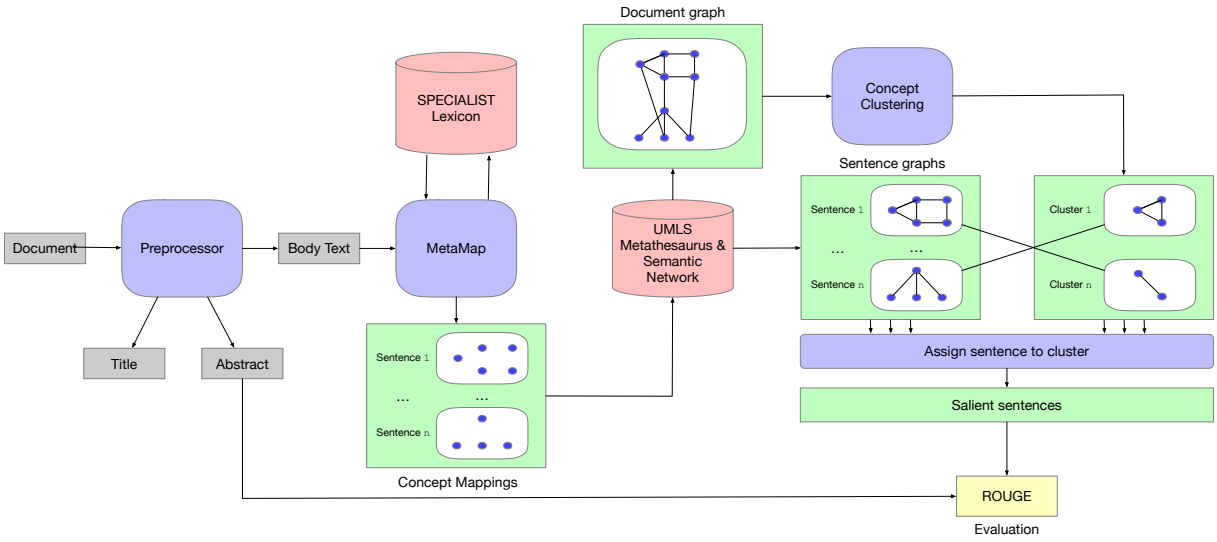

Figure 3: Overview of the entire algorithm

Phrases	Concept	Semantic Types
Entry of these viruses	Virus Entry	Biologic Function
mediated by the viral spike glycoprotein S	Mediated	Organic Chemical
mediated by the viral spike glycoprotein S	viral glycoprotein	Cell Component
a receptor on the target cell	Receptor Cell	Cell

Table 1: MetaMap mappings for input phrase *Entry of these viruses is mediated by the viral spike glycoprotein S and a receptor on the target cell*

idea or concept, intellectual product, mental process, spatial concept and language. Table 1 shows example MetaMap mappings for input phrase *Entry of these viruses is mediated by the viral spike glycoprotein S and a receptor on the target cell*.

3.3 Sentence representation

Next, we construct a graph-based sentence representation using the concepts we retrieved. To construct sentence graph, we first retrieve relations of mapped concepts from UMLS Metathesaurus. The relationships we are interested in are as follows: 1) hypernym (`RB` in UMLS) relations between **concepts**; 2) related to (`related_to` in UMLS) relations between **concepts**; 3) associated with (`associated_with` in UMLS) relations between **semantic types**. Table 2 shows an example of relations retrieved from UMLS Metathesaurus for concepts in table 1.

Then, we recursively add the hypernym concepts to the concept set until we have retrieve the entire concept hierarchy. We compute the depth of each concept and drop the concepts with depth less than 2 as they are too broad. The remaining concepts are the nodes of our sentence graph. Finally, we add edge (v_i, v_j) if and only if these is a hypernym relation between v_i and v_j. The sentence graph for the example sentence is shown in figure 4. The red nodes are leaf concepts we get from the MetaMap. The blues nodes are hypernyms. The edges represent hypernym (or "is a") relation. For example, virus entry *is a* virus infection mechanism. Virus infection mechanism *is a* virus characteristic and so on. The concepts *Mediated* and *Receptor Cell* are excluded because they are the root of their hierarchies and thus too broad to be counted.

3.4 Document representation

So far, our sentence graphs are unweighted and only have hypernym relation edges. To construct a document graph representation, We merge our sentence graphs, add related and associated relations as edges and add weights. Repeated concepts are merged to a single node. We only add related and associated relations between leaf concepts because we want the directly mapped concepts gain more attentions. We use the following formula to

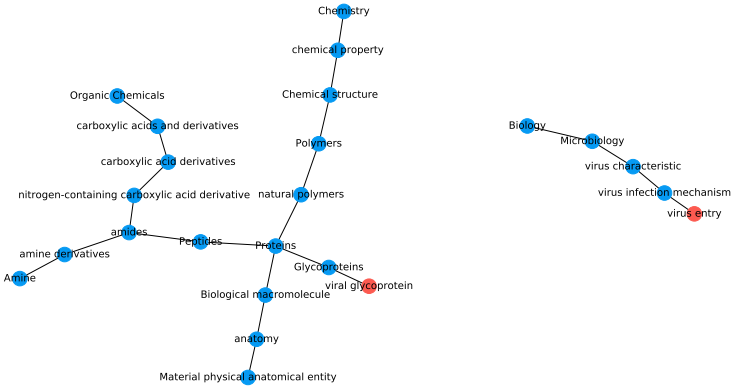

Figure 4: Final sentence graph for concepts in table 1

compute weight w_{ij} for edge (v_i, v_j):

$$w(v_i, v_j) = \begin{cases} \frac{\text{depth}(v_i)}{\text{depth}(v_j)} & \text{if } (v_i, v_j) \text{ is hypernym} \\ 1.0 & \text{otherwise} \end{cases} \tag{1}$$

The intuition is that the edges closer to the direct mapped concepts should have higher weight. The final document representation is a undirected weighted graph. Note that the graph is not necessarily connected.

3.5 Concept Clustering

Next, we retrieve the main concepts of the document by clustering the document graph. We first extract n salient concept vertices from the graph. n is a parameter that can be configured. The salience of concepts are sum of weight of edges connected to them:

$$\text{salience}(v_i) = \sum_j w(v_i, v_j) \tag{2}$$

We follow the origin paper and named this vertices *hub vertices*. Then we compute the closeness of each hub vertices. the closeness is just the weight of the edge between the pair of vertices. If not edge between the pair, the closeness is 0:

$$\text{closeness}(v_i, v_j) = \begin{cases} w(v_i, v_j) & \text{if } (v_i, v_j) \in \text{edges} \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Then, for each hub vertex, we find the closest hub vertex with it and form a *hub vertices set* (or HVS). If no other hub vertices connected to it, then it form a HVS with itself. Note that a hub vertex may appear in different HVS. Next, we compute the intra connectivity of each HVS and the inter connectivity of each pairs of HVS. The intra connectivity is the sum of all edges within the HVS. The inter connectivity is sum of all edges between two HVSs:

$$\text{intra-connectivity}(h) = \sum_{(v_i, v_j) \in h} w(v_i, v_j) \tag{4}$$

Concepts	Hypernyms	Related	Associated
Virus Entry	virus infection mechanism	[]	[]
Mediated	[]	[]	[]
viral glycoprotein	Glycoproteins	[]	[]
Receptor Cell	[]	[]	[]

Table 2: Concept relations for concept in table 1. From the table, we see that *Mediated* and *Receptor Cell* do not have any hypernyms. This means these two concepts are too broad. Thus, we will exclude these two concepts when constructing the semantic graph

$$\text{inter-connectivity}(h_i, h_j) = \sum_{v_i \in h_i, v_j \in h_j} w(v_i, v_j) \tag{5}$$

Finally, for each pair of HVSs, we compare the intra connectivity and inter connectivity between them. If the inter connectivity is larger than the intra connectivity, we merge the two HVSs. The intuition behind this is similar to the Sihouettes (Rousseeuw, 1987), which propose that an object should be more similar to its own cluster than other clusters. Then we iterate through all non hub vertex nodes and assign them to the closest cluster. The overall clustering algorithm is shown in algorithm 1.

Algorithm 1: concept_clustering(G, V, E, n)

```
Result: Concept clusters
hv = top n vertices in V with highest salience ;
hvs = [] ;
for v ∈ hv do
    if v is not connected with any other hv
     then
        hvs.append([v])
    else
        hvs.append([v, closest(v)])
    end
end
for h_i ∈ hvs do
    intra_score = intra_conn(h_i) ;
    for h_j ∈ hvs and i ≠ j do
        inter_score = inter_conn(h_i, h_j) ;
        if intra_score < inter_score then
            Merge(h_i, h_j) ;
        end
    end
end
for v ∈ V and v ∉ hv do
    if v connect to any vertices in hv then
        i = closest hvs ;
        hvs[i].append(v) ;
    end
end
return hvs
```

Note the not all concepts are assigned to a cluster as the the graph may not be connected.

3.6 Sentence-to-cluster assignment

Now we have extracted main concepts of the document. Our next goal is assign a match score for each cluster-sentence pair. Specifically, we compare the similarity between the vertices in sentence graph and the vertices in clusters. If we a node v in sentence graph S_i is also in HVS of cluster C_j, we add a vote (S_i, C_j) pair. If s_i is in C_j but not in its HVS, we add half a vote to the pair. Otherwise we do not vote for the pair. The formula for calculating the similarity is shown as follow:

$$\text{vote}(v, C_j) = \begin{cases} 1.0 & \text{if} v \in HVS(C_j) \\ 0.5 & \text{if} v \in C_j \\ 0.0 & \text{otherwise} \end{cases} \tag{6}$$

$$\text{similarity}(S_i, C_j) = \sum_{v \in S_i} \text{vote}(v, C_j) \tag{7}$$

3.7 Sentence selection

Now, we have scores for each cluster-sentence pair. We sort sentence match scores decreasingly for each cluster and use the following methods to select sentences:

1. **Main concept method**: Select salient sentences based only on the ranking of main concept cluster, the cluster with the highest number of concepts.

2. **Proportional method**: Select sentences with size proportional to the size of concept cluster.

3. **Hybrid method**: Aggregate sentence score by normalizing and summing the similarity with each clusters. Then we sorted the sentence scores and pick the sentences with highest scores. The formula used for aggregation is as follow:

$$\text{hybrid}(S_i) = \sum_{C_i} \frac{\text{similarity}(S_i, C_i)}{|C_i|} \tag{8}$$

4 Experiment

In this section, we will describe the methods to evaluate our algorithm, the dataset we use for evaluation and present the final result.

4.1 Metrics

The evaluation of summaries can be broadly classified into two categories: *Intrinsic* and *extrinsic* (Jones and Galliers, 1995). Intrinsic methods evaluate the summarization out based on certain metrics. Extrinsic methods evaluate the impact of the summarization system on specific tasks such as decision making accuracy. In this project, we employ

	α	BLEU	ROUGE-1	ROUGE-2
Random	0.10	0.1816	0.3065	0.0964
	0.15	0.1463	0.4090	0.1214
	0.30	0.1224	0.5756	0.2009
Main Concept	0.10	0.1764	0.3848	**0.1178**
	0.15	0.1575	0.4595	0.1468
	0.30	**0.1317**	0.6022	0.2277
Proportional	0.10	**0.1852**	0.3698	0.1172
	0.15	**0.1641**	0.4487	**0.1532**
	0.30	0.1303	0.5944	0.2212
Hybrid	0.10	0.1773	**0.3880**	0.1167
	0.15	0.1631	**0.4690**	0.1489
	0.30	0.1315	**0.6029**	**0.2297**

Table 3: Overall experiment result. α is the fraction of the length of input document. The highest scores are marked in bold.

intrinsic methods for evaluation. Specifically, we use the recall-oriented understudy for gisting evaluation (ROUGE)(Lin, 2004) and bilingual evaluation understudy as our evaluation metrics. ROUGE and BLEU are metrics generally used for evaluation automatic summarization and machine translation in NLP. They return scores in range $[0, 1]$. The higher the better. BLEU shows the accuracy of summaries and ROUGE shows the informativeness of summaries. We also compute the ROUGE-1 and ROUGE-2, which are ROUGE for unigram and bigrams respectively.

Using BLEU and ROUGE means we have to compare our summarization output to some reference standard developed by experts. However, developing such reference standard is expensive and time consuming. Thus, most of biomedical literature dataset do not contain such reference. To address this problem, we use abstract as our reference standard.

4.2 Dataset

We use the CORD-19 (Wang et al., 2020) dataset, which contains over 59,000 scholarly articles, including over 47,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses, for evaluation. We drop records without full text or abstract. Figure 5 shows the distribution of word count of remaining articles.

4.3 Evaluation

We implement our algorithm in Python. We download UMLS Metathesaurus and run it locally using MySQL server. Our program communicate with MetaMap and UMLS Metathesaurus using

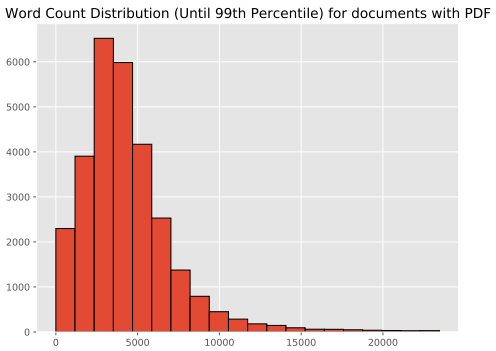

Figure 5: Word count distribution fo articles with full text in CORD-19 dataset. The mean value is 4870.71.

`subprocess` and `pymysql` packages. It takes about 2 minutes to process a single document.The main bottleneck is the communication cost between main program and MetaMap. We try running the algorithm concurrently using multithreading technique. However, MetaMap becomes unstable in this situation and sometimes get stuck. Fortunately, there is not real-time requirements for text summarization. We only need to run the program once for each document and store the result for future reference. In this case, the process time seems reasonable. Nevertheless, due to the time limit of this project, we decide to evaluate our algorithm on a small subset of CORD-19. Specifically, we randomly select 100 sample from CORD-19 dataset and process these documents in sequence.

We have two parameters in our algorithm, the number of salient concepts n and the length of output summaries, L. We set n and L proportional

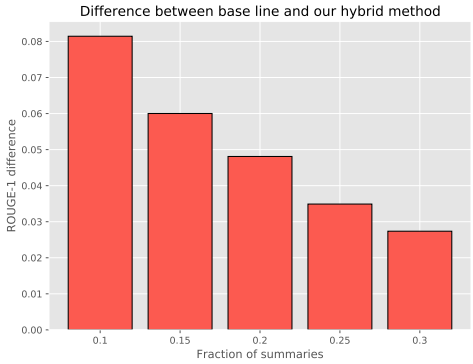

Figure 6: ROUGE score difference between random baseline and hybrid method.

to the total number of concepts and the length of input document respectively. Specifically, we set n to 10 percent of total number of concepts and L to α percent of input document length. We try three α, 0.1, 0.15 and 0.3 corresponding to small, medium and large output size. We also try the three sentence selection methods mentions in last section. The results are presented in table 3.

From the table, we get the following insights. First, all of our three sentence selection methods outperform the baseline. The hybrid method has the best ROUGE-1 score for all output sizes and the best ROUGE-2 score for large output size. The proportional method has the best BLEU score for small and medium output sizes, and best ROUGE-2 score for medium output size. The main concept method has the best BLEU score for large output size and best ROUGE-2 score for small output size.

Moreover, as α increase, all ROGUE scores increase but BLEU scores decrease. This make sense because as α increase, we select more sentences from the document and thus the summaries become more informative. When $\alpha = 1$, the output contain the complete information as well as the length of input document, but this is trivial for text summarization.

We also notice that the smaller the output size, the ROUGE score difference between our algorithm and the random baseline becomes larger. To further verify this fact, we try more α values and compare the ROUGE-1 difference between random baseline and our hybrid method. The result is shown in figure 6. The result verify that as the outsize become smaller, the ROUGE difference between our algorithm and random baseline becomes larger. This indicates that our algorithm works good especially when the required output size is small as it always returns the most representative sentences first.

5 Discussion and Limitation

From the last section, we see that our algorithm extracts key information from biomedical literatures successfully and outperforms the baseline under all circumstances. However, our algorithm exist the following limitations and we can further improve the algorithm by addressing these limitations:

1. Our algorithm does not take repetitions of concepts into account. Intuitively, if a concept appears in a document many times, we should pay more attention to it. However, recall that, our algorithm merge same concepts into a single node when creating the document graph. This may lose the information of repetition.

2. The salience of concept depends more on the UMLS concept network instead of the document itself. Consider the graph shown in figure 4, the most salient concepts in this case are the the concepts with most number of hypernyms in stead of the leaf concepts.

6 Conclusion

In this project, we implement a graph-based extractive summarization algorithm for biomedical research literatures. We employ domain-specific knowledge by mapping text into biomedical concepts using Unified Medical Language System (UMLS). We evaluate our method using the CORD-19 dataset and show that our algorithm extracts salient information from the input document successfully and outperforms the baseline. Finally, we discuss the limitations and potential optimizing points.

References

Alan R. Aronson. 2001. Effective mapping of biomedical text to the umls metathesaurus: the metamap program. *Proceedings. AMIA Symposium*, pages 17–21.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32 Database issue:D267–70.

Wesley T. Chuang and Jihoon Yang. 2000. Extracting sentence segments for text summarization: a machine learning approach. In *SIGIR '00*.

Clifford W. Gay, Mehmet Kayaalp, and Alan R. Aronson. 2005. Semi-automatic indexing of full text biomedical articles. *AMIA ... Annual Symposium proceedings. AMIA Symposium*, pages 271–5.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Karen Sparck Jones and Julia Galliers. 1995. Evaluating natural language processing systems: An analysis and review. *Evaluating Natural Language Processing Systems*.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *ACL 2004*.

Inderjeet Mani and Mark T. Maybury. 1999. Advances in automatic text summarization. *Computational Linguistics*, 26:280–281.

Rashmi Mishra, Jiantao Bian, Marcelo Fiszman, Charlene R. Weir, Siddhartha Jonnalagadda, Javed Mostafa, and Guilherme Del Fiol. 2014. Text summarization in the biomedical domain: A systematic review of recent research. *Journal of biomedical informatics*, 52:457–67.

Neda Nazari and Mohammad Amin Mahdavi. 2019. A survey on automatic text summarization. *Journal of AI and Data Mining*, 7:121–135.

NLM. 2020a. The specialist lexicon.

NLM. 2020b. Umls metathesaurus.

NLM. 2020c. The umls semantic network.

Laura Plaza, Alberto Díaz, and Pablo Gervás. 2011. A semantic graph-based approach to biomedical summarisation. *Artificial intelligence in medicine*, 53 1:1–14.

Peter Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis.

Kamal Sarkar. 2009. Using domain knowledge for text summarization in medical domain.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Darrin Eide, Kathryn Funk, Rodney Kinney, Ziyang Liu, William Merrill, Paul Mooney, Dewey Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Chris Wilhelm, Boya Xie, Douglas Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. Cord-19: The covid-19 open research dataset.

WHO. 2020. Coronavirus disease (covid-19) situation report 102.