Extract with Order for Coherent Multi-Document Summarization

Mir Tafseer Nayeem
University of Lethbridge
Lethbridge, AB, Canada
mir.nayeem@uleth.ca

Yllias Chali
University of Lethbridge
Lethbridge, AB, Canada
chali@cs.uleth.ca

Abstract

In this work, we aim at developing an extractive summarizer in the multi-document setting. We implement a rank based sentence selection using continuous vector representations along with key-phrases. Furthermore, we propose a model to tackle summary coherence for increasing readability. We conduct experiments on the Document Understanding Conference (DUC) 2004 datasets using ROUGE toolkit. Our experiments demonstrate that the methods bring significant improvements over the state of the art methods in terms of informativity and coherence.

1 Introduction

The task of automatic document summarization aims at finding the most relevant informations in a text and presenting them in a condensed form. A good summary should retain the most important contents of the original document or a cluster of documents, while being coherent, non-redundant and grammatically readable. There are two types of summarizations: abstractive summarization and extractive summarization. Abstractive methods, which are still a growing field are highly complex as they need extensive natural language generation to rewrite the sentences. Therefore, research community is focusing more on extractive summaries, which selects salient (important) sentences from the source document without any modification to create a summary. Summarization is classified as single-document or multi-document based upon the number of source document. The information overlap between the documents from the same topic makes the multi-document summarization more challenging than the task of summarizing single documents.

One crucial step in generating a coherent summary is to order the sentences in a logical manner to increase the readability. A wrong order of sentences convey entirely different idea to the reader of the summary and also make it difficult to understand. In a single document, summary information can be presented by preserving the sentence position in the original document. In multi-document summarization, the sentence position in the original document does not provide clue to the sentence arrangement. Hence it is a very challenging task to perform the arrangement of sentences in the summary.

2 Related Work

During a decade, several extractive approaches have been developed for automatic summary generation that implement a number of machine learning, graph-based and optimization techniques. LexRank (Erkan and Radev, 2004) and TextRank (Mihalcea and Tarau, 2004) are graph-based methods of computing sentence importance for text summarization. The RegSum system (Hong and Nenkova, 2014) employs a supervised model for predicting word importance. Treating multi-document summarization as a submodular maximization problem has proven successful by (Lin and Bilmes, 2011). Unfortunately, none of the above systems care about the coherence of the final extracted summary.

In very recent works using neural network, (Cheng and Lapata, 2016) proposed an attentional encoder-decoder and (Nallapati et al., 2017) used a simple recurrent network based sequence classifier to solve the problem of extractive summarization. However, they are limited to single document settings, where sentences are implicitly ordered according to the sentence position. (Parveen and Strube, 2015; Parveen et al., 2015)

Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing, ACL 2017, pages 51–56,
Vancouver, Canada, August 3, 2017.

proposed graph-based techniques to tackle coherence, which is also limited to single document summarization. Moreover, a recent work (Wang et al., 2016) actually proposed a multi-document summarization system that combines both coherence and informativeness but this system is limited to syntactic linkages between entities.

In this paper, we implement a rank based sentence selection using continuous vector representations along with key-phrases. We also model the coherence using semantic relations between entities and sentences to increase the readability.

3 Sentence Extraction

We here successively describe each of the steps involved in the sentence extraction process such as sentence ranking, sentence clustering, and sentence selection.

3.1 Preprocessing

Our system first takes a set of related texts as input and preprocesses them which includes tokenization, Part-Of-Speech (POS) tagging, removal of stopwords and Lemmatization. We use **NLTK** toolkit[1] to preprocess each sentence to obtain a more accurate representation of the information.

3.2 Sentence Similarity

We take the pre-trained word embeddings[2] (Mikolov et al., 2013) of all the non stopwords in a sentence and take the weighted vector sum according to the term-frequency (TF) of a word(w) in a sentence(S). Where, E is the word embedding model and $idx(w)$ is the index of the word w. More formally, for a given sentence S in the document D, the weighted sum becomes,

$$S = \sum_{w \in S} TF(w, S) \cdot E[idx(w)]$$

Then we calculate cosine similarity between the sentence vectors obtained from the above equation to find the relative distance between S_i and S_j. We also calculate $NESim(S_i, S_j)$ by finding the Named Entities present in S_i and S_j using NLTK Toolkit, then calculating their overlap.

$$CosSim(S_i, S_j) = \frac{S_i \cdot S_j}{||S_i||\ ||S_j||}$$

$$NESim(S_i, S_j) = \frac{|NE(S_i) \cap NE(S_j)|}{min(|NE(S_i)|, |NE(S_j)|)}$$

$$Sim(S_i, S_j) = \lambda \cdot NESim(S_i, S_j) + (1 - \lambda) \cdot CosSim(S_i, S_j) \quad (1)$$

The overall similarity calculation involves both $CosSim(S_i, S_j)$ and $NESim(S_i, S_j)$ where, $0 \leq \lambda \leq 1$ decides the relative contributions of them to the overall similarity computation. This standalone similarity function will be used in this work with different λ values to accomplish different tasks.

3.3 Sentence Ranking

In this section, we rank the sentences by applying TextRank algorithm (Mihalcea and Tarau, 2004) which involves constructing an undirected graph where sentences are vertices, and weighted edges are formed connecting sentences by a similarity metric. TextRank determines the similarity based on the lexical overlap between two sentences. However, this algorithm has a serious drawback: If two sentences are talking about the same topic without using any overlapped words, there will be no edge between them. Instead, we use the continuous skip-gram model introduced by (Mikolov et al., 2013) to measure the semantic similarity along with the entity overlap. We use the similarity function described in Equation (1) by setting $\lambda = 0.3$.

After we have our graph, we can run the main algorithm on it. This involves initializing a score of 1 for each vertex, and repeatedly applying the TextRank update rule until convergence. The update rule is:

$$Rank(S_i) = (1 - d) + d * \sum_{S_j \in N(S_i)} \frac{Sim(S_i, S_j)}{\sum_{S_k \in N(S_j)} Sim(S_j, S_k)} Rank(S_j)$$

Where, $Rank(S_i)$ indicates the importance score assigned to sentence S_i. $N(S_i)$ is the set of neighboring sentences of S_i, and $0 \leq d \leq 1$ is a dampening factor, which the literature suggests its setting to 0.85. After reaching convergence, we extract the sentences along with TextRank scores.

[1] http://www.nltk.org/

[2] https://code.google.com/archive/p/word2vec/

3.4 Sentence Clustering

The sentence clustering step allows us to group similar sentences. We use a hierarchical agglomerative clustering (Murtagh and Legendre, 2014) with a complete linkage criteria. This method proceeds incrementally, starting with each sentence considered as a cluster, and merging the pair of similar clusters after each step using bottom up approach. The complete linkage criteria determines the metric used for the merge strategy. In computing the clusters, we use the similarity function described in Equation (1) with $\lambda = 0.4$. We set a similarity threshold ($\tau = 0.5$) to stop the clustering process. If we cannot find any cluster pair with a similarity above the threshold, the process stops, and the clusters are released. The clusters may be small, but are highly coherent as each sentence they contain must be similar to every other sentence in the same cluster.

This sentence clustering step is very important due to two main reasons, (1) Selecting at most one sentence from each cluster of related sentences will decrease redundancy from the summary side (2) Selecting sentences from the diverse set of clusters will increase the information coverage from the document side as well.

3.5 Sentence Selection

In this work, we use the concept-based ILP framework introduced in (Gillick and Favre, 2009) with some suitable changes to select the best subset of sentences. This approach aims to extract sentences that cover as many important concepts as possible, while ensuring the summary length is within a given budgeted constraint. Unlike (Gillick and Favre, 2009) which uses bigrams as concepts, we use keyphrases as concepts. Keyphrases are the words or phrases that represent the main topics of a document. Sentences containing the most relevant keyphrases are important for the summary generation. We extracted the keyphrases from the document cluster using RAKE[3] (Rose et al., 2010). We assign a weight to each keyphrase using the score returned by RAKE.

Let w_i be the weight of keyphrase i and k_i a binary variable that indicates the presence of keyphrase i in the extracted sentences. Let l_j be the number of words in sentence j, s_j a binary variable that indicates the presence of sentence j in the extracted sentence set and L the length limit for the set. Let Occ_{ij} indicate the occurrence of keyphrase i in sentence j, the ILP formulation is,

$$Maximize : (\sum_i w_i k_i + \sum_j Rank(S_j) \cdot s_j) \quad (2)$$

$$Subject\ to : \sum_j l_j s_j \leq L \quad (3)$$

$$s_j Occ_{ij} \leq k_i, \quad \forall i, j \quad (4)$$

$$\sum_j s_j Occ_{ij} \geq k_i, \quad \forall i \quad (5)$$

$$\sum_{j \in g_c} s_j \leq 1, \quad \forall g_c \quad (6)$$

$$k_i \in \{0, 1\} \quad \forall i \quad (7)$$

$$s_j \in \{0, 1\} \quad \forall j \quad (8)$$

We try to maximize the weight of the keyphrases (2) in the extracted sentences, while avoiding repetition of those keyphrases (4, 5) and staying under the maximum number of words allowed for the sentence extraction (3).

In addition to (Gillick and Favre, 2009), we put some extra features like maximizing the sentence rank scores returned from the sentence ranking section. In order to ensure only one sentence per cluster in the extracted sentences we add an extra constraint (6). In this process, we extract the optimal combination of sentences that maximize informativity while minimizing redundancy (Figure 1 illustrates our sentence extraction process in brief).

4 Sentence Ordering

Classic reordering approaches include inferring order from weighted sentence graph (Barzilay et al., 2002), or perform a chronological ordering algorithm (Cohen et al., 1999) that sorts sentences based on timestamp and position.

We here propose a simple greedy approach to sentence ordering in multi-document settings. Our assumption is that a good sentence order implies the similarity between all adjacent sentences since word repetition (more specifically, named entity repetition) is one of the formal sign of text coherence (Barzilay et al., 2002). We define coherence of document D which consists of sentences from

[3]https://github.com/aneesha/RAKE

System	Models	R-1	R-2	R-SU4	Coherence
Baseline	LexRank	35.95	7.47	12.48	0.39
	GreedyKL	37.98	8.53	13.25	0.46
State-of-the-art	Submodular	39.18	9.35	**14.22**	0.51
	ICSISumm	38.41	9.78	13.31	0.44
Proposed System	ILPRankSumm	**39.45**	**10.12**	14.09	**0.68**

Table 1: Results on DUC 2004 (Task-2) for the baseline, state-of-the-art and our system.

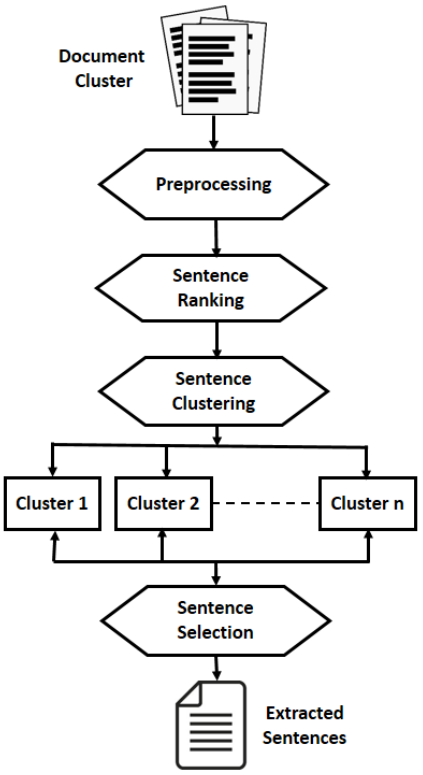

Figure 1: Sentence Extraction Process

S_1 to S_n in the following equation. For calculating $Sim(S_i, S_{i+1})$, we use the similarity function described in equation (1) with $\lambda = 0.5$, giving the named entities a little more preference.

$$Coherence(D) = \frac{\sum_{i=1}^{n-1} Sim(S_i, S_{i+1})}{n-1}$$

We propose a greedy algorithm for placing a sentence in a document based on the coherence score we discussed above[4]. At the beginning, we randomly select a sentence from the extracted sentences without any position information and place the sentence in the ordered set D. We then incrementally add each extracted sentences to the document set D using Algorithm (1) to get the final order of summary sentences.

[4]Note that, we didn't take any position information of the original sentences to be extracted from the document.

Algorithm 1: Place a sentence to a document

Procedure `SentencePositioning`(D, S_n)
  Data: Input document D which is assumed sorted. New sentence S_n which we will place in the document D.
  Result: Return new document D_n after placing the sentence S_n.
  $t \leftarrow 1$;
  $Coh_{max} \leftarrow 0$;
  $D_{tmp} \leftarrow D$;
  $l \leftarrow DocLength(D)$;
  while $t \leq l + 1$ **do**
    $\Rightarrow$*Place the* S_n *in* t^{th} *position of* D_{tmp} ;
    $Coh_{tmp} \leftarrow Coherence(D_{tmp})$;
    if $Coh_{tmp} > Coh_{max}$ **then**
      $D_n \leftarrow D_{tmp}$;
      $Coh_{max} \leftarrow Coh_{tmp}$;
      $\Rightarrow$ *Remove* S_n *from the* t^{th} *position of the document* D_{tmp} ;
    end
    $t \leftarrow t + 1$;
  end
  return D_n;

5 Evaluation

We evaluate our system **ILPRankSumm** (**ILP** based sentence selection with Text**Rank** for Extractive **Summ**arization) using **ROUGE**[5] (Lin, 2004) on DUC 2004 (Task-2, Length limit(L) = 100 words). However, ROUGE scores are biased towards lexical overlap at surface level and insensitive to summary coherence. Moreover, sophisticated coherence evaluation metrics are seldom adopted for summarization thus many of the previous systems used human evaluation for measuring readability. For this reason, we evaluate our summary coherence using (Lapata and Barzilay, 2005) (Barzilay and Lapata, 2008) which defines coherence probabilities for an ordered set of sentences.

5.1 Baseline Systems

We compare our system with baseline (LexRank, GreedyKL) and state of the art systems (Submodular, ICSISumm). **LexRank**(Erkan and Radev, 2004) represents input texts as graph where nodes

[5]ROUGE-1.5.5 with options: -n 2 -m -u -c 95 -x -r 1000 -f A -p 0.5 -t 0

are the sentences and the edges are formed between two sentences if the cosine similarity is above a certain threshold. Sentence importance is calculated by running the PageRank algorithm on the graph. **GreedyKL** (Haghighi and Vanderwende, 2009) iteratively selects the next sentence for the summary that will minimize the KL divergence between the estimated word distributions. (Lin and Bilmes, 2011) treat the document summarization problem as maximizing a **Submodular** function under a budget constraint. They achieved a near-optimal information coverage and non-redundancy using a modified greedy algorithm. On the other hand, **ICSISumm** (Gillick and Favre, 2009) employs a global linear optimization framework, finding the globally optimal summary rather than choosing sentences according to their importance in a greedy fashion.

The summaries generated by the baselines and the state-of-the-art extractive summarizers on the DUC 2004 dataset were collected from (Hong et al., 2014).

5.2 Results

Our results include R-1, R-2, and R-SU4, which counts matches in unigrams, bigrams, and skip-bigrams respectively. The skip-bigrams allow four words in between. According to Table 1, R-1, R-2 scores obtained by our system outperform all the baselines and state of the art systems on DUC 2004 datasets. One of the main reasons of getting the improved R-1 and R-2 score is the use of keyphrases. Moreover, there is no significant difference between our proposed system and submodular in case of R-SU4. We also get better coherence probability because of our sentence ordering technique. The system's output for a randomly selected document set (e.g. d30015t) from DUC 2004 is shown in Table 2.

5.3 Limitations

One of the essential properties of the text summarization systems is the ability to generate a summary with a fixed length (DUC 2004, Task-2: Length limit = 100 words). According to (Hong et al., 2014) all the summarizer from the previous research either truncated the summary to 100^{th} word, or removed the last sentence from the summary set. In this paper, we follow the second one to produce grammatical summary. However, the first one produces a certain ungrammatical sentence, later one can lose a lot of information in the worst case, if the sentences are long. We more focus on the grammaticality of the final summary.

Summary Generated (After Sentence Extraction)
But U.S. special envoy Richard Holbrooke said the situation in the southern Serbian province was as bad now as two weeks ago. A Western diplomat said up to 120 Yugoslav army armored vehicles, including tanks, have been pulled out. On Sunday,Milosevic met with Russian Foreign Minister Igor Ivanov and Defense Minister Igor Sergeyev, Serbian President Milan Milutinovic and Yugoslavia's top defense officials. To avoid such an attack, Yugoslavia must end the hostilities, withdraw army and security forces, take urgent measures to overcome the humanitarian crisis, ensure that refugees can return home and take part in peace talks, he said.
Summary Generated (After Sentence Ordering)
On Sunday, Milosevic met with Russian Foreign Minister Igor Ivanov and Defense Minister Igor Sergeyev, Serbian President Milan Milutinovic and Yugoslavia's top defense officials. But U.S. special envoy Richard Holbrooke said the situation in the southern Serbian province was as bad now as two weeks ago. A Western diplomat said up to 120 Yugoslav army armored vehicles, including tanks, have been pulled out. To avoid such an attack, Yugoslavia must end the hostilities, withdraw army and security forces, take urgent measures to overcome the humanitarian crisis, ensure that refugees can return home and take part in peace talks, he said.

Table 2: System's output (100 words) for the document set **d30015t** from DUC 2004.

6 Conclusion and Future Work

In this work, we implemented an ILP based sentence selection along with TextRank scores and key phrases for extractive multi-document summarization. We further model the coherence to increase the readability of the generated summary. Evaluation results strongly indicate the benefits of using continuous word vector representations in all the steps involved in the overall system. In future, we will focus on jointly extracting the sentences to maximize informativity and readability while minimizing redundancy using the same ILP model. Moreover, we will also try to propose a solution for the length limit problem.

Acknowledgments

We would like to thank the anonymous reviewers for their useful comments. The research reported in this paper was conducted at the University of Lethbridge and supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada discovery grant and the University of Lethbridge.

References

Regina Barzilay, Noemie Elhadad, and Kathleen R. McKeown. 2002. Inferring strategies for sentence ordering in multidocument news summarization. *J. Artif. Int. Res.* 17(1):35–55.

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Comput. Linguist.* 34(1):1–34.

Jianpeng Cheng and Mirella Lapata. 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)*. Association for Computational Linguistics, Berlin, Germany, pages 484–494.

William W. Cohen, Robert E. Schapire, and Yoram Singer. 1999. Learning to order things. *J. Artif. Int. Res.* 10(1):243–270.

Günes Erkan and Dragomir R. Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *J. Artif. Int. Res.* 22(1):457–479.

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*. Association for Computational Linguistics, Stroudsburg, PA, USA, ILP '09, pages 10–18.

Aria Haghighi and Lucy Vanderwende. 2009. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, NAACL '09, pages 362–370.

Kai Hong, John Conroy, Benoit Favre, Alex Kulesza, Hui Lin, and Ani Nenkova. 2014. A repository of state of the art and competitive baseline summaries for generic news summarization. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. European Language Resources Association (ELRA), Reykjavik, Iceland, pages 1608–1616. ACL Anthology Identifier: L14-1070.

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Gothenburg, Sweden, pages 712–721.

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, IJCAI'05, pages 1085–1090.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. Association for Computational Linguistics, Barcelona, Spain, pages 74–81.

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '11, pages 510–520.

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into texts. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*. Association for Computational Linguistics, Barcelona, Spain, pages 404–411.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems*. Curran Associates Inc., USA, NIPS'13, pages 3111–3119.

Fionn Murtagh and Pierre Legendre. 2014. Ward's hierarchical agglomerative clustering method: Which algorithms implement ward's criterion? *J. Classif.* 31(3):274–295.

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*. pages 3075–3081.

Daraksha Parveen, Hans-Martin Ramsl, and Michael Strube. 2015. Topical coherence for graph-based extractive summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1949–1954.

Daraksha Parveen and Michael Strube. 2015. Integrating importance, non-redundancy and coherence in graph-based extractive summarization. In *Proceedings of the 24th International Conference on Artificial Intelligence*. AAAI Press, IJCAI'15, pages 1298–1304.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining* pages 1–20.

Xun Wang, Masaaki Nishino, Tsutomu Hirao, Katsuhito Sudoh, and Masaaki Nagata. 2016. Exploring text links for coherent multi-document summarization. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 213–223.