International Journal of Psychosocial Rehabilitation, Vol. 24, Issue 06, 2020
ISSN: 1475-7192

Query-based Text Summarization using Averaged Query

[1]Abhinandh Ajay, [2]Shravan V, [3]R. Srinivasan

Abstract

Automatic text summarization is one of the most common problems in natural language processing and machine learning.Text summarization usually works by shortening a given passage and conveying the general meaning of the passage. There are two approaches to this: extraction-based summarization and abstraction based summarization. Extraction - based summarization, while easier to implement, is usually grammatically incorrect. Abstraction based summarization overcomes this by framing its own sentences using grammatical knowledge of the language and is therefore much harder to implement. There is an ever-increasing amount of unstructured data in the world, so situations can arise where it is only necessary to extract the summary of a part of the given data. This is the case when a person wants to learn something about a certain topic but often has to skim through a lot of unrelated or unnecessary information. While this is not a problem for small amounts of data, it can quickly turn into a burden as the size of the data increases. This can lead to a loss of focus or interest in the topic. In this work, the proposed technique allows the user to enter a keyword and get a summary related to say query using the word frequencies to find the most relevant words. We also use cosine similarities to remove redundant sentences from the summary. This query based text summarization technique produces a unique summary for every unique keyword. Better readability can be achieved by using abstractive text summarization.

Keywords: text summarization, data, abstraction

I. INTRODUCTION

With the information boom at the turn of the century, it has become increasingly important to have a method of extracting relevant information for a certain use case from a plethora of available sources. Automatic text summarization is one such method that allows us to take a large amount of data and condense it into a much shorter form which contains the most relevant and important information for users.

There are two prominent types of text summarization:

- Generic[1]
- Query based[2]

[1] *Department of Computer Science and Engineering SRM Institute of Science and Technology Kattankulathur, Tamil Nadu - 603203*
[2] *Department of Computer Science and Engineering SRM Institute of Science and Technology Kattankulathur, Tamil Nadu - 603203*
[3] *Department of Computer Science and Engineering SRM Institute of Science and Technology Kattankulathur, Tamil Nadu - 603203*

Received: 22 Dec 2019 | Revised: 13 Jan 2020 | Accepted: 05 Feb 2020

Generic text summarization extracts important information from the given data without any prior knowledge. Query based, on the other hand, takes in a query or a keyword input from the user and creates a summary with relevant information pertaining to that keyword.

With the growing pace of development in the field of natural language processing. It is inevitable to exploit its development to device algorithms to produce clean and concise texts from the long documents that we come across ever so often[3].

The current era is the era of endless learning and with the help of text summarization, text can be more easily understood by scholars and researchers trying to explore the depths of a passage.

In this work, we introduce a technique known as

AQSum Algorithm and evaluate the results that we obtain using this algorithm using the ROUGE

evaluation metric[4].

The paper is organized as follows. Section 2 examines the literature survey. Section 3 gives a brief overview of the proposed work for this paper. Section 4 explains the implementation which is evaluated and tested in Section 5 and 6. The paper is concluded and has future enhancements in Section 7.

II. STATE OF THE ART

This work is based on the strategy employed by Yutong Wu et al. for obtaining coherent summaries that is relevant to a certain query that exists within the document. The paper implements extractive text summarization, where the summaries are uniquely tailored to the query. The authors use dual pattern- enhanced representation model, which extracts sentences in the documents which consist of the query and extracts the words that are before and after the query in that sentence. The sentences are then ranked based on the extracted words from the document.

Our paper, however, extracts the most similar words that are related to the query from the document and extract the whole sentence that contains each of the words to increase the accuracy and the coherence of the summaries.

There are various algorithms currently under research for the accurate summarization of text. Extraction based models use various approaches to summarize text which includes and is not limited to clustering, pattern-based, machine learning, and term frequency.

Yutong Wu, Yuefeng Li, Yue Xu[5] proposes an unsupervised pattern-enhanced approach for representation of topics across documents (and query relevance), to generate summaries that adhere to the user needs. The model can be used in SDS (Single Document Summarization) and can be extended to aid MDS (Multi-Document Summarization) as well. The model integrates pattern-mining, topic modeling and query expansion techniques which adds the advantage of producing discriminative patterns and punishing patterns that are not related to the given query. The paper talks about an efficient sentence ranking algorithm which

appears to exploit the benefits of query relevance and topic modeling to provide sentences that adhere to the query, nevertheless it still stays within the context of the topic within the documents in which it appears. Query expansion is optimized by using re- weighting technique. The limitation of this model is that it only works in an extractive summarization model.

Hal Daume III and Daniel Marcu[6] proposes an algorithm that uses the Bayesian statistical model in order to optimize information retrieval. It utilizes multiple documents that are relevant to the given query and generates a D X Q binary matrix 'r' where r dq = 1 if and only if d is the document set that is relevant to the query q. The advantages of this algorithm is that it produces relevant documents from search engines and returns a short summary based on the entered query. The model is trained using the data from Text REtrieval Conference Competitions (TREC). It is evaluated manually by seven human judges who perform the sentence extraction task. They were supplied with the query and a single document relevant to that query, and were asked to select up to four sentences from the document that best fit the context of the query. The limitation arises from the fact that the query expansion is based on the relevance of each document to the query, which means that query irrelevant document sets play no part in the summaries.

Sheng-Tang Wu, Yuefeng Li, Yue Xu[7] employs text mining using techniques such as, the pattern based model containing frequent sequential patterns, instead of the keyword-based approach which is more commonly seen in this field. In term frequency and inverse document frequency (tfidf) weighting scheme is used for representing text in Rocchio classifiers. In addition to tfidf, the global idf and entropy weighting scheme is proposed and improves the performance by an average of 30%. The paper's primary objective is to use the Reuters text collection to test the different techniques such as Pr, PTM(Pattern Text Mining), PDR (Pattern Deploying with Relevance function) and the Rocchio Algorithm and the results show that the PDR outperforms all the other algorithms.

Ercan Canhasia, Igor Kononenkoa[8] propose a matrix factorization method, known as weighted

Archetypal Analysis. A graph model is used to select sentences weighted by the similarity to the given query. This results in positively and/or negatively related sentences being assigned values on the weighted dataset boundary. The advantage of using the wAA is to compute the values in the extremes, archetypes, thereby allowing the estimation of the importance of sentences in target documents set. It also increases the variability and diversity of the query-focused summary. In addition, the wAA integrates the combined benefits of clustering and archetypes. The limitations, however, is that in order to provide the summary the query should be modeled jointly with the docume nt to be summarized, which is not a severe limitation in SDS as it is in MDS. The wAA method is evaluated by comparing it with other proficient summarizers using the ROUGE scores and it is shown to outperform 80% of the summarizers.

Lu Wang, Hema Raghavan, Vittorio Castelli, Radu Florian, Claire Cardie[9] propose a sentence-compression-based framework, and design learning- based compression models built on parse trees for the summarization task. The paper investigates the role of learning-based sentence compression in query focused MDS. They designed three ways to approach sentence-compression—sequence-based, rule-based and tree-based. The framework consists of three steps: Sentence Ranking, Sentence Compression and Post-Processing. The importance of each sentence for the given query is determined by sentence ranking. Next, the sentence compressor iteratively generates the most concise versions of the ranked sentences, which are then added to the summary. The finally stage applies coreference resolution and sentence reordering to build a more efficient and concise summary. The

evaluation using ROUGE shows a considerable improvement over pure extraction-based methods and the state of the art. The advantage of the proposed model is that some of the disadvantages of extractive text summarization such as lengthy sentences that are partially relevant to the query being excluded from the actual summaries because of restrictions in the length of the summary. That being said, the limitations of this model is that it still does not outperform certain aspect of the extractive text summarization approach.

Wenjuan Luo, Fuzhen Zhuang, Qing He, Zhongzhi Shi[10] propose a novel Probabilistic - model Relevance, Coverage, and Novelty (PRCN) framework, which exploits a reference topic model incorporating user query for dependent relevance measurement. The paper constructs features that pertain to sentence features regarding relevance and novelty, while a greedy algorithm maintains topic coverage for topic balance. This greedy algorithm is used to select rational coverage on different topics. The advantage in this approach is that, it's the only algorithm that explicitly considers coverage, novelty and balance in query-focused multi-document summarization. The limitation to this approach is that it fails to consider taking into account the novelty, coverage, and balance from query-aware angle. The evaluation of the algorithm using the ROUGE shows that the algorithms such as NA and NK outperform the PCRN simply because they are able to account for this limitation.

Guangbing Yang, Dunwei Wen, Kinshuk, Nian- Shing Chen, Erkki Sutinen[11] proposes an approach based on hierarchical Bayesian topic models, by incorporating a set of n-grams into hidden topics to capture word dependencies that appear close to the word. In the Bayesian topic model used, the similarity is analyzed using advanced methods keeping in mind the probabilistic distribution of topics. Evaluation metrics show a significant improvement when compared the baseline. But a major limitation is that the model takes a long time to be train a large dataset which is required to maintain a desirable accuracy. It also uses a Bernoulli distribution which is poor at indicating word co-occurrence among multiple words.

III. PROPOSED WORK

Our paper uses the Topically Diverse Query Focus Summarization (TD-QFS) dataset[12] which was made to be an improvement over the DUC dataset. The dataset consists of topics under Alzheimers, asthma, cancer and obesity out of which we used the obesity data. The dataset consists of raw data obtained from various sources, queries, and manually made summaries for the given queries. We clean the data to remove unwanted parts of speech and only consider the tokens that contribute towards the semantics of the document. The most similar words that exist in the document for the query is searched and the sentences are added to list of sentence tokens. The sentences are then ranked based on sentence scores, which we get by tokenizing the sentences and finding the maximum occurring token in the sentences. We also use cosine similarity matrix in order to make sure we do not have redundant sentences in the summary. The summaries are then evaluated using the ROUGE evaluation metric.

IV. IMPLEMENTATION

A. Data Preprocessing

a) Data Cleaning

Since the information sources have a lot of reference tags that do not contribute towards the context of an article, and so

we remove these tags from the text. We also remove the stop words[13] and punctuations and symbols to retain words that give meaning to the text.

b) Lemmatization

Using an NLP library called spaCy we use a method called lemmatization[14] which converts the inflected forms of the words to a normalized word form. This step ensures that there is no redundancy in counting the frequency of the words. We employ lemmatization over stemming[15] as the latter only removes the suffix from words while the former know when the base word is different from the conjugated word and does not blindly remove suffix.

eg. Stemming: was —> wa

Lemmatization: was —> is

b) Tokenization

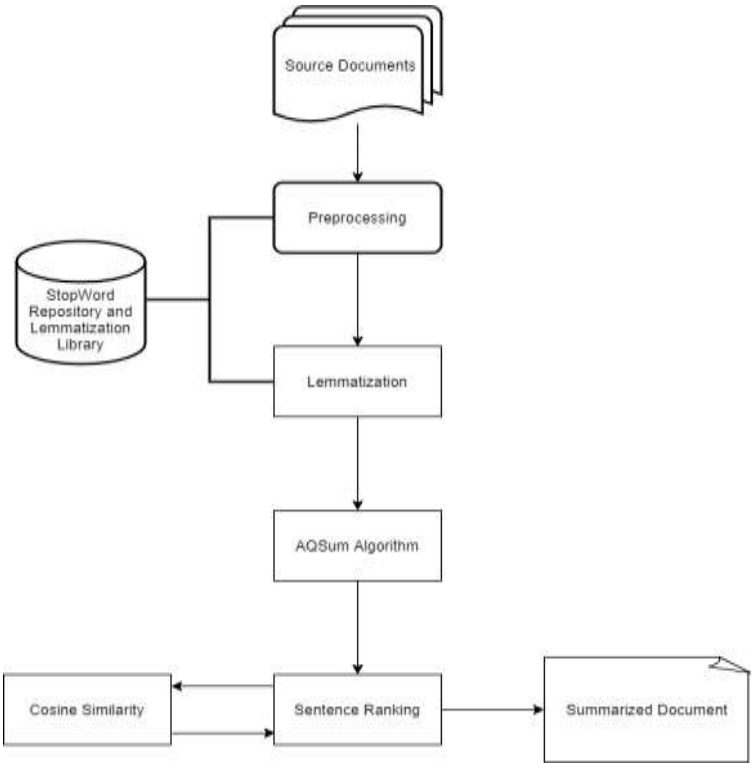
Tokenization[16] is the splitting of content into sentences or words. In this paper, we make use of both sentence and word tokens, where the former is used for the final summary and the latter for the algorithm.

B. Summarization

a) Word-Frequencies

In this paper, we implement query based summarization, and so we count the word tokens that appear in the sentences where the query is present. We consider the equations to set the frequency for the input query as using the original frequency can lead to undesirably large value which tends to form a bias with the scoring mechanism. We then normalize the frequency by generating weighted frequency values using

Fig.1. Architecture Diagram



$$firstAverage = firstMaxValue + secondMaxValue \quad / \quad 2$$

$$secondAverage = secondMaxValue + thirdMaxValue \quad / \quad 2$$

$$wordFrequencies[inputQuery] = firstAverage + secondAverage \quad / \quad 2$$

$$wordFrequencies[word] = \frac{wordFrequencies[word]}{maximumFrequencyWord} \quad / \quad 2$$

b) Sentence Scoring

In this step, we scan through the words in each of the sentences[17], and we assign scores based on their weighted frequency.

We also eliminate large sentences because they show bias towards the calculation. If the word frequency of the query is not modified in the previous step then it results in the sentences with the query being giving much higher scores when compared to the other sentences. This results in sentences with query having more value even if it might not actually be relevant to the query.

c) Cosine Similarity

Cosine Similarity[18][19] is implemented to find sentences that are very similar in order to reduce redundancy[20] in summaries. We implement cosine similarity by generating vectors for each of the sentences that we want to compare. Each sentence vector consist of the frequencies of words present in that sentence from the pool of words present in both sentences.

$$similarity = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A^2} \sqrt{\sum_{i=1}^{n} B^2}}$$

d) Summary Generation

The summary is generated by taking the top

10 highest ranked sentences which are not similar to one another and arranging them in the order that it was found in the passage. The higher ranking words, according to the algorithm, are supposed to be more relevant to the query. But at the same time, since averaging was done, there are also sentences which don't include the query that have a high score.

V. EVALUATION

A. Base Algorithm

The base algorithm makes use of only the word frequencies which results in the frequency of the query being much greater than the frequencies of the other words. Therefore, the when the weighted frequencies are generated, the score of the query is much greater than any other word thereby making the sentence scoring mechanism unfair. It is observed that in a majority of cases, the most relevant sentences selected with this value for the frequency are almost always are sentences which have the query present in it even though the sentence might not be as important to the summary as a sentence without the query.

When we use ROUGE on the TD-QFS dataset we get the following results (Table 1) for the summary generated by using the query "body mass index"

Table 1 - ROUGE - Base Algorithm

B. Averaged Query Summarization (AQSum) Algorithm

The AQSum algorithm makes use of the second highest and third highest word frequency to modify the frequency of the query. This way the new "frequency" value is more in line with the rest of the frequencies thereby reducing its strength while scoring sentences. Therefore the sentences scored this way might also lead to high ranking sentences where the query is not present but is still of relevance.

When we use ROUGE on the TD-QFS dataset we get the following results (Table 2) for the summary generated by using the query "body mass index"

Table 2 - ROUGE - AQSum Algorithm

VI. RESULTS

While evaluating the summaries generated using different queries from examples present in the TD- QFS dataset we got precision, recall and f measure values in a somewhat inconsistent manner. While evaluation did lead to high values (Table 3), we thought it would be sound to also used a completely different evaluation system to check discrepancies. We also performed this summarization on data directly obtained from wikipedia to test the system against larger volumes of data.

Table 3 - Best results during evaluation

Therefore we resorted to using a human centric evaluation system where we asked different people to evaluate the summaries generated from the base algorithm and the ones generated using the AQSum algorithm on a scale of 1 to 5. We observed that 4 out of 5 people rated the summary generated by AQSum (Fig. 3) higher than the summary generated by the base algorithm (Fig. 2). Therefore while the algorithm proposed in this work is not perfect, it still performs better than the general method of calculating the query frequency.

A. Summary Generated by Base Algorithm

The summary generated by the base algorithm mostly consisted of sentences which contained the query. Since the text for this example was taken from a large corpus, it resulted in the frequency of the query being much greater than the frequency of the rest of the words. Therefore, while there were some sentences which was relevant to the query, there were also many unnecessary sentences thereby resulting in an inaccurate summary.

Received: 22 Dec 2019 | Revised: 13 Jan 2020 | Accepted: 05 Feb 2020

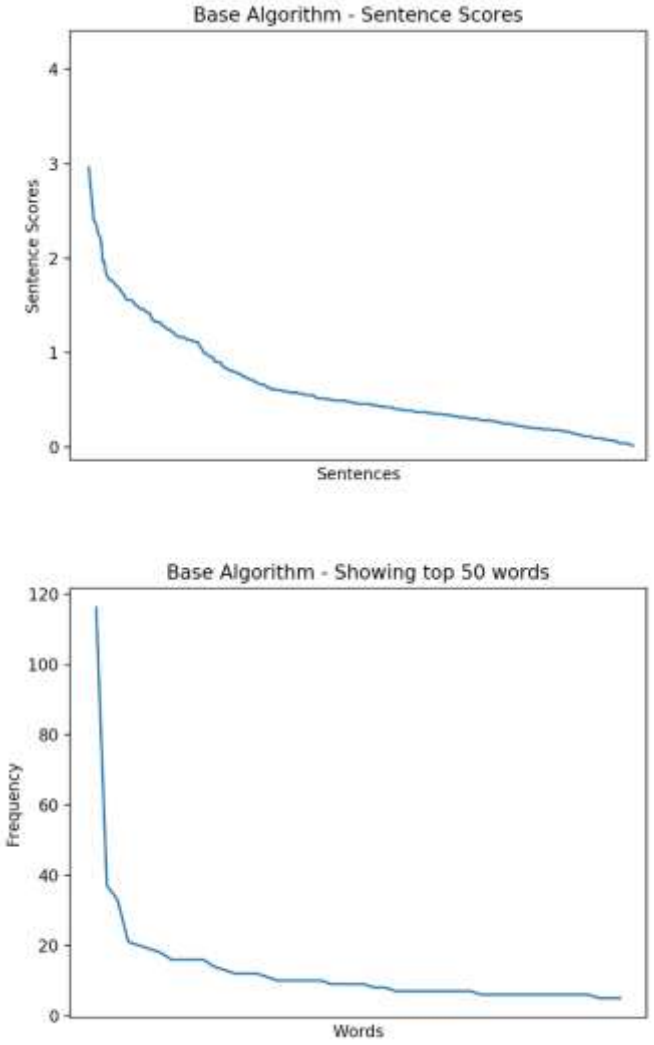


Fig. 2. Base Algorithm Curve

B. Summary Generated by AQSum Algorithm

The summary generated by the AQSum algorithm consists of both sentences with the query and sentences without the query. Since the text for this example was taken from a large corpus, the frequency of the query was large. This however was fixed by the AQSum algorithm thereby normalizing the frequency to a value more in line with the other frequencies. Therefore the summary generated had more sentences with were related to the query even if they didn't necessarily contain the query.

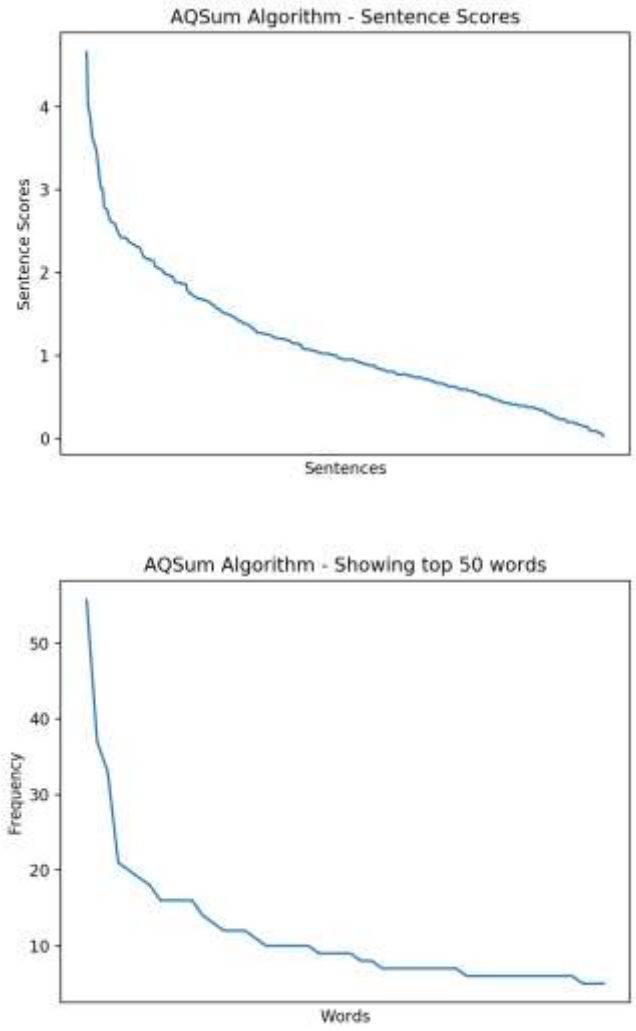


Fig. 3. AQSum Algorithm Curve

The tests and evaluations done on different sets of data show that the algorithm tends to perform better on much larger datasets when compared to smaller ones. Larger datasets have more words so averaging the query frequency has a larger impact on the result obtained.

VII. CONCLUSION

This work was done using data obtained from TD- QFS dataset. The data was not exactly as per requirements for analysis and there had to be pre- processed to make ready for use. The objective was to provide a concise summary of a topic related to the query supplied by the user. The generated summary is 10 lines long and has the sentences most relevant to the query provided. The AQSum algorithm modifies the score of the query so that its weightage is reduced in order to make it more in line with the rest of the scores so as to not provide a bias to the query term while performing sentence ranking.

Received: 22 Dec 2019 | Revised: 13 Jan 2020 | Accepted: 05 Feb 2020

Future research could enhance the accuracy by first performing coreference resolution before applying the algorithm. If this is done then the pronouns can be linked to their respective subjects or objects and frequency calculation will result in more accurate weighted score generation. It allows for more sentences to be considered which previously might not have been due to the absence of the query Word vector models like words2vec can also be used to gauge the similarity between sentences in order to eliminate redundancies.

VIII. REFERENCES

(1) Gong, Yihong and Liu, Xin, Generic Text Summarization Using Relevance Measure and Laten Semantic Analysis, Association for Computing Machinery, 2001, pages 19-25

(2) N. Rahman and B. Borah, A survey on existing extractive techniques for query-based text summarization, International Symposium on Advanced Computing and Communication (ISACC), 2015, pages 98-102

(3) Ghambir, Mahak and Gupta, Vishal, Recent Automatic Text Summarization Techniques: A Survey, Kluwer Academic Publishers, 2017, pages 1-66

(4) Lin, Chin-Yew, {ROUGE}: A package for Automatic Evaluation of Summaries, Association for Computational Linguistics, 2004, pages 74-81

(5) Yutong Wu, Yuefeng Li and Yue Xu, Dual pattern-enhanced representations model for query-focused multi-document summarization, Knowledge-Based Systems, Volume 163, 1 January 2019, pages 736-748

(6) H. Daumé III, D. Marcu, Bayesian query- focused summarization, in: Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics, ACL-44, Association for Computational Linguistics, Stroudsburg, PA, USA, 2006, pp. 305–312

(7) S.T. Wu, Y. Li, Y. Xu, Deploying approaches for pattern refinement in text mining, in: Proceedings of the Sixth IEEE International Conference on Data Mining, ICDM 2006, (ISSN: 1550-4786) 2006, pp. 1157–116

(8) E. Canhasi, I. Kononenko, Weighted archetypal analysis of the multi-element graph for query- focused multi-document summarization, Expert Syst. Appl. 41 (2) (2014) 535–543

(9) L. Wang, H. Raghavan, V. Castelli, R. Florian, C. Cardie, A sentence compression based framework to query-focused multi-document summarization, 2016, arXiv preprint arXiv:1606.07548.

(10)W. Luo, F. Zhuang, Q. He, Z. Shi, Exploiting relevance, coverage, and novelty for query- focused multi-document summarization, Knowl.-Based System. 46 (2013) 33–42

(11)Y. Guangbing, W. Dunwei, Kinshuk, C. Nian- Shing, S. Erkki, A novel contextual topic model for multi-document summarization, Expert Syst. Appl. 42 (3) (2015) 1340–1352

(12)Baumel, Tal and Cohen, Raphael and Elhadad, Michael, Topic Concentration in Query Focused Summarization Datasets, Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, 2016, Pages 2573-2579

(13)Schofield, Alexandra and Magnusson, Maans and Mimno, David, Pulling Out the Stops: Rethinking Stopword Removal for Topic Models, Association for Computational Linguistics, 2017, pages 432-436

(14)Jenna Kanerva and Filip Ginter and Tapio Salakoski, Universal Lemmatizer: A sequence to Sequence Mo del for Lemmatizing Universal Dependencies Treebanks, ArXiv, 2019

(15)Singh, Jasmeet and Gupta, Vishal, A Systematic Review of Text Stemming Techniques, Kluwer Academic Publishers, 2017, pages 157-217

(16)Webster, Jonathan J. and Kit, Chunyu, Tokenization as the Initial Phase in NLP,