Arabic Snippet Generation Using Extractive Query-based Summarization

Mariam Safieldin
mas177@mail.aub.edu

Supervisor: Dr. Shady Elbassiouni

This document is a report for the final project of Information Retrieval and Web Search (CMPS 391) course.

***Abstract*—In this project, an Arabic Query-based Text Summarization System was implemented. The target of the system is to produce an extractive summary that includes the relevant parts of a document to a given user information need in Arabic. To satisfy this objective, the system encompasses three main components, a query expansion component, a document selector component, and a document summarizer component which ranks document sentences based on extracted query-based features. A benchmark data collection IV was used to evaluate the output of the implemented system. The findings reveal that the derived features were capable of rating the sentences based on the search query. The query expansion component did not enhance the system significantly.**

Keywords: Query-based Summarization, Feature Extraction, Text Ranking Model, Arabic Text Summarization, Snippet Generation, Arabic Text Retrieval.

I. Introduction

Using summaries is an action that almost everyone does on a daily basis. Summaries are commonly used in a variety of settings and at all times. Paper abstracts, TV news headlines, book reviews, store shopping recommendations, search results snippets, and even movie trailers are all examples of summaries. With the extensive use of search engines and the increase of resources on the web, the need of summarization and information need specific search results became very important. Document summarizing can be a useful approach for dealing with this problem. The goal of query-based document summarization is to provide a brief summary of the source material that includes relevant information for the user's information need.

Text summarization can be categorized into 3 categories, each with two sub-categories. First, documents can be summarized with an extractive approach, where the most important and relevant sentences are extracted from the document, or an abstractive approach, where the generated summary is a condensed paraphrased version of the original document, thus it is close to human-generated summaries. Second, summaries can be generated from a single document or multiple documents. Third, summaries can be generic, thus encompassing the most dominant and important sentences or outlining the main idea of the information in the document, or query-based, which means that it highlights certain information in the document related to the user's information need [1].

Researchers in the field of Arabic natural language processing encounter a variety of challenges while working with the Arabic language. Tasks like word stemming, machine translation, automated summarization, and even sentence segmentation more difficult when perfomed on Arabic language due to the prevalence of Arabic diglossia, high rates of ambiguity, and the language's highly derivational and inflectional structure of the language. Unfortunately, there are no resourceful advances in research in Arabic text summarization. The majority of research in this area is done on generic summarization rather than query-based summarization. Automatic summarizing is an active area, particularly in the Arabic language, where the amount of research on automatic Arabic text summary is limited and growing slowly in comparison to the amount of literature on other languages like English [2]. The aim of this project is to implement an Arabic query based extractive summarization system which can provide users with snippets of documents based on their information need. Figure 1 below clarifies our target case showing the input and output of our proposed approach.

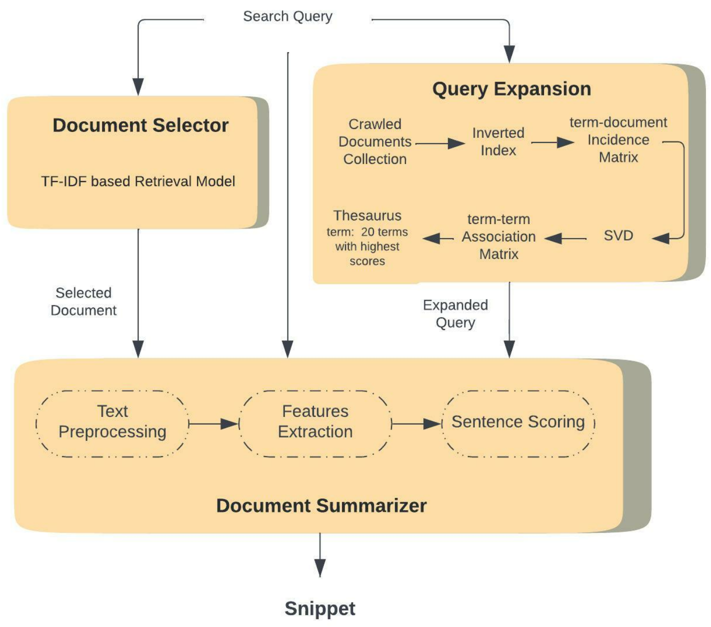

Fig. 1: Arabic Query-based Summarization System

The remainder of the report is organized as follows: In Section II a subset of the previous work is reported and compared to our work. Section III reports the proposed methodology. In Section IV, benchmark used, experimental setup, and comparison to previous work are discussed. Finally, the conclusion is presented in Section V.

II. Related Work

The previous work can be categorized into four main categories based on the methodology used for Arabic automatic text summarization. The approaches used are statistical-based approaches, graph-based approaches, evolutionary-based approaches, and machine learning-based approaches. Statistical-based approaches generate summaries using a ranking methodology of the sentences that allows for the selection of the most related sentences from the document. Features are extracted from the sentences so that relevant sentences can be identified. Features extracted encompass features such as keywords, sentence length, sentence title, sentence position, term frequency (TF) and inverse-document frequency (IDF) [1].

For Arabic single-document summary, Bialy et al. [3] presented a statistically-based generic extractive technique. The suggested method is divided into three stages: pre-processing, sentence scoring, and summary generation. To evaluate their method, the authors collected 33 Arabic short articles (one to three paragraphs each) from Wikipedia and had two human experts summarize them. The results are compared to summaries made by two human experts.

Qaroush et al. [4] proposed a generic extractive Arabic single-document summarization system that uses supervised machine learning algorithms and a combination of statistical and semantic features. The features used include key-phrases, sentence location, similarity with title, sentence centrality, sentence length, cue-words, positive key-words, sentence inclusion of numeric data. The authors evaluated their approach using EASC corpus and the ROUGE measure tool.

Afsharizadeh et al. [5] proposed a query-oriented extractive summarization method in which they extract features from sentence to identify and select important sentences in a document through ranking the sentences based on the scores given to the extracted features. In their work, they extend a generic extractive summarization approach proposed by Ahuja and Anand [6]. The proposed summarization approach encompasses 5 stages, which include, data preparation, text pre-processing, feature extraction, sentence scoring, and summary generation. They use five features proposed in [6], and they additionally add other six features. The features used include both query dependent and independent features such as the normalized sentence length, sentence position, and topic token frequency. It is worth noting that the proposed system in [5] and [6] is implemented to summarize English documents.

El-Haj et al. [7] proposed a query-oriented summary for a single Arabic document system. They employ a simple system to find the relevant documents to an input query which relies on matching between the query's bag-of-words and the documents collection. Afterwards, the user selects the document to be summarized. A vector space model is ued to generate the document summary where the document sentences and the query are viewed as vectors. Every distinct term that appears in the whole document collection has one component in each vector. Each term in a document is automatically given a weight to represent its relative importance. The document is broken down into paragraphs. Then the query's bag-of-words and the cosine similarity measure are used to rank the paragraphs and identify those that are closely relative to the query. The summary is limited to encompass 50% of the original document listed so that their original positions are preserved. The authors apply their system on 251 Arabic articles downloaded from the Wikipedia website. The authors evaluate their system manually through a set of five groups from different backgrounds have participated in testing and evaluating our AQBTSS system. The evaluators vary in their educational levels. The selected groups include: Arabic literature, humanities, computer science, and K-12 school students, and K-12 school teachers.

Graph-based approaches focus on the semantic relations between the sentences. The document's sentences are represented as a graph, with nodes representing sentences and arcs representing relationships between them [1]. Al-Taani et al. [8] propose a graph-based approach where they represent a document as weighted directed graph with nodes representing sentences and edge weights representing sentence similarity. This resemblance is established by rating the sentences based on statistical characteristics. The cosine similarity metric is determined based on the TF-IDF word weighting strategy (Term Frequency-Inverse Document Frequency). The summary is obtained by calculating the shortest path between the first and last nodes in the graph while taking into account the user compression ratio. EASC corpus and intrinsic assessment technique are used to evaluate the proposed methodology.

III. Methodologies

In this section, the methodology applied to generate the extractive summary based on the user's given information need is discussed.

A. *Data Pre-processing*

Data pre-processing is an essential step in natural language processing tasks such as text summarization. The chosen document's text and the input query go through data pre-processing steps before extracting features from the sentences to relate them to the query.

The first step performed is removing the punctuation, such as question marks, exclamation marks, and commas, from the text. Secondly, the Arabic diacritics were also removed from the text, since they might be in a document, yet they are not commonly used when querying for a certain information need. Letters that exist in different shapes such as إ أ ا and ى ي are unified as the diversity in a single letter's shape would affect the matching process. Afterwards, the stop words are removed from the text using a predefined stop words list from Natural Language Toolkit (NLTK). Finally, the text is stemmed using Tashaphyne library's stemmer which was used to extract the root of each word.

B. *Document Selection*

To automate the process of generating the summary a document selector module was used to find the relevant document from a document collection based on a user's query. The document selector is a simple retrieval model based on vector space model which uses a TF-IDF index built using the term frequency (TF) and inverse document frequency (IDF) to score the documents.

Firstly, an inverted index was created through saving a posting list for every word in our benchmark data collection after stemming and removing the stop words from them, the posting list has the ids of the documents that a certain word occurs in them. Afterwards, this inverted index was used to construct the TF-IDF index through stemming the documents and counting the occurrences c of every word in the document that it occurs in, and then using the formula

1 to compute the TF-IDF score of every word and its corresponding documents, where *dft* is the document term frequency.

$$tf - idf = (1 + \log_2(c))/\log_2(N/dft) \tag{1}$$

Given an input search query, the terms of the query go through the text pre-processing steps in III-A, then the TF-IDF scores of every term within each document are used to compute scores for the available documents in our benchmark. The document with the highest score is retrieved by the document selector and passed to the document summarizer component III-C.

C. Document Summarizer

Features Extraction

The relevancy of the sentences were assessed through extracting query based features that would reflect the similarity of every sentence from the selected document to the information need given as an input to the system.

In this project, extractive summarization strategy is used, hence the most informative sentences with respect to a certain query are identified and extracted from the text. Determining which sentences in the document are the most relevant is the most essential step in the extractive summarization process. Accordingly, useful features are be extracted from each sentence, and a score is allocated to it depending on its feature values. The highest-ranking sentences are then chosen for inclusion in the summary.

Four query based features, discussed below, were extracted for every sentence in the selected document. Each feature was assigned a weight which was used to contribute in scoring the sentences. The fourth feature is derived using 4 sub-features based on text similarity algorithms, each sub-feature is further assigned a weight during the sentence scoring step.

1 - Query Sentence Unigram Overlap

In natural language processing, an n-gram is a sequence of n words. Accordingly, a unigram is the basic text unit. A unigram model's textual units are the single words. The overlap between the query words and a sentence words were counted after stemming the query words to their roots.

2 - Query Sentence Bigram Overlap

Since standalone words cannot convey semantic information, the sequence and collocation of words must be considered. Query sentence bigram overlap feature measures the number of occurrences of two consecutive words from the query in every sentence. Similar to the previous feature, the occurrences of query bigrams were counted after stemming the query words to their roots and creating a list of all bigrams in both the query and the document.

3 - Part-of-speech (POS) Tagging with Query Comparison

Stanford's Arabic Part-of-speech tagger was used to specify the words categories. This feature is used to identify sentences which specifically contain proper nouns that occur in the query as well, since sentences with proper nouns are more likely to include useful information [5]. This feature is calculated by dividing the number of total proper nouns contained in the sentence and the search query by the sentence's length. POS tagging was performed on a version of the document which did not go through the stemming phase during the pre-processing step.

4 - Similarity Measures

A set of text similarity measures based on text distance and text representation were used to evaluate the similarity between the input search query and the document sentences. From the viewpoint of distance, text distance similarity measures reflect the semantic proximity of two text terms. The Manhattan distance is the sum of the absolute differences between two real-valued vectors and is used to compute the distance between them. After representing text with one-hot encoding, Manhattan distance is used to determine the similarity of two texts [10].

Lexical similarity of the sentences and the query was assessed using two character-based measures, longest common substring (LCS) similarity and Jaro similarity, and one phrase-based measure which is the Jaccard similarity. The phrase-based approach varies from the character-based method in that the fundamental unit of the phrase-based method is a phrase word [10]. LCS reflects the length of the longest substring that is the same in the provided two text to compare. Jaro similarity between two strings *S1* and *S2* is represented by the following mathematical formula 2, where m is the number of characters in two strings that match, and t is half of the transposition number [10].

$$Sim = \begin{cases} 0 & m = 0 \\ \frac{1}{3}\left(\frac{m}{|S_1|} + \frac{m}{|S_2|} + \frac{m-t}{m}\right) & \end{cases} \tag{2}$$

Jaccard similarity is a phrase-based methodology which measures similarity through dividing the size of the intersection by the size of the union of two sets.

Sentence Scoring

A weight was assigned for each extracted and computed feature mentioned in III-C after tuning the weights through performing a randomized search to obtain the weights that would maximize the system's precision and recall. The weights for the four feature discussed in III-C, Query Sentence Unigram Overlap, Query Sentence Bigram Overlap, Part-of-speech Tagging with Query Comparison, and Similarity Measures,that maximized the overall system's precision and recall are 1, 0.75, 0.65, and 1, respectively. While the weights for the four sub-features the similarity measures feature, Manhattan Distance, Jaccard Similarity, Jaro Similarity, and Longest Common Substring (LCS), are 0.70, 0.75, 0.95, and 0.85, respectively. The number of sentences extracted depends on the number of sentences available in the original document. For documents that contain more than 20 sentences, the top 30% ranked sentences of the document were extracted. While, if the document has more than 20 sentences, the top 20% ranked sentences were extracted.

Corpus Collection and Query Expansion

To the aim of improving the retrieval of the most relevant sentences to the given search query, the query was expanded through building a thesaurus that would include terms and their most relevant terms based on extracting the 20 terms with the highest score from a term-term association matrix for every term in our dictionary. To build the thesaurus, a corpus was collected through crawling several websites, namely, BBC, arageek, Almrsal, and Al-ain, reaching a number of 30K+ documents. Every document was pre-processed and converted into a text file to build an inverted index that would encompass every term and the documents it appears in. The first 8K values in the built inverted index were used to generate a term-document incidence matrix A. Afterwards, the Singular Value Decomposition (SVD) of matrix A wass computed, which was then used to compute the term-term association matrix U. Finally, matrix U was used to extract the 20 terms with highest scores for every present term in the dictionary [11].

The query was expanded through looking for every term in the query if it exists in the thesaurus, and adding the top 3 terms in this term's terms list.

IV. Experiment and Result Evaluation

Benchmark

The data collection used for evaluating the system is a collection of 26 articles, 33 queries and their corresponding snippets. 14 articles were crawled from google along with their snippets after searching for a certain information need, and saving this information need as the query to be used while evaluating the system. The majority of snippets when a certain query is typed were the first two or three sentences from the corresponding webpage which in many cases contain the query terms yet they do not provide the information needed. To solve this issue, the *people also ask for* set of questions provided by Google were used instead to form a query based on the question asked and take the answer as the snippet, this way the answer to the query is no more limited to the first couple of lines in a webpage that contain the query terms, yet the snippet actually contain the information requested. Another approach followed was to crawl articles that contain information about several related ideas or topics, and query different motifs within the same document. 12 articles were used from the EASC dataset, and the corresponding snippet to a certain query was manually extracted through skimming the article and extracting the sentences that are relevant to the query. Some of these articles were used to provide different snippets for multiple different queries.

The length of the articles vary, yet they range from 2-5 paragraphs. The length of the snippet vary based on the article and the information need, yet they range from 2-6 sentences.

Results

The built system was evaluated using the benchmark introduced in IV. ROUGE python library was used to compute the recall, precision, and F1-score for ROUGE-1, ROUGE-2, and ROUGE-L to evaluate the system's extracted snippets against the human-extracted snippets in the benchmark. ROUGE is a widely established standard for summarization evaluation that is based on an n-gram co-occurrence between machine and human summaries [12].

The overlap of unigrams between the system and reference summaries is referred to as ROUGE-1. The overlap of bigrams between the system and reference summaries is referred to as ROUGE-2. The longest common subsequence (LCS) between the model output and the reference is measured by ROUGE-L, which implies counting the longest common token sequence between the two, with the notion that a longer common sequence indicates that the two summaries are more comparable. The recall counts the number of overlapping n-grams found in both the model output and reference then divides this number by the total number of n-grams in the reference summary. Precision metric is calculated by dividing the number of overlapping n-grams by the model n-gram count. While the F1-score combines both the recall and precision through dividing the multiplication of the recall with the precision with their sum and multiplying this number by 2 [13].

The system's performance was evaluated once with the query expansion component III-C included and other time without expanding the queries. The results are shown in tables I and II. The results show that the features extracted were capable of ranking the sentences in accordance to the given search query. The query expansion component did not show a significant improvement to the system.

Metric	Recall	Precision	F1 Score
ROUGE-1	0.843	0.438	0.538
ROUGE-2	0.810	0.405	0.489
ROUGE-L	0.834	0.431	0.531

TABLE I: System's ROUGE Evaluation Results on The Benchmark

Metric	Recall	Precision	F1 Score
ROUGE-1	0.845	0.424	0.523
ROUGE-2	0.809	0.378	0.477
ROUGE-L	0.838	0.418	0.522

TABLE II: System's ROUGE Evaluation Results on The Benchmark with Query Expansion

V. Conclusion

An Arabic Query-based Text Summarization System was built in this project. The system's goal is to generate an extracted summary in Arabic that incorporates the relevant parts of a document for a specific user information demand. The system included three key components to achieve this goal: a query expansion component, a document selection component, and a document summarizer component that ranked document sentences based on extracted query-based attributes. The findings reveal that the derived features were capable of rating the sentences based on the search query. The query expansion component did not enhance the system significantly.

References

[1] Al-Taani, A. T. (2021). Recent Advances in Arabic Automatic Text Summarization. International Journal of Advances in Soft Computing & Its Applications, 13(3).

[2] Al-Saleh, A. B., & Menai, M. E. B. (2016). Automatic Arabic text summarization: a survey. Artificial Intelligence Review, 45(2), 203-234.

[3] Bialy, A. A., Gaheen, M. A., ElEraky, R. M., ElGamal, A. F., & Ewees, A. A. (2020). Single arabic document summarization using natural language processing technique. In Recent Advances in NLP: The Case of Arabic Language (pp. 17-37). Springer, Cham.

[4] Qaroush, A., Farha, I. A., Ghanem, W., Washaha, M., & Maali, E. (2021). An efficient single document Arabic text summarization using a combination of statistical and semantic features. Journal of King Saud University-Computer and Information Sciences, 33(6), 677-692.

[5] Afsharizadeh, M., Ebrahimpour-Komleh, H., & Bagheri, A. (2018, April). Query-oriented text summarization using sentence extraction technique. In 2018 4th international conference on web research (ICWR) (pp. 128-132). IEEE.

[6] Ahuja, R., & Anand, W. (2017). Multi-document text summarization using sentence extraction. In Artificial Intelligence and Evolutionary Computations in Engineering Systems (pp. 235-242). Springer, Singapore.

[7] El-Haj, M. O., & Hammo, B. H. (2008, October). Evaluation of query-based Arabic text summarization system. In 2008 International Conference on Natural Language Processing and Knowledge Engineering (pp. 1-7). IEEE.

[8] Al-Taani, A. T., & Al-Omour, M. M. (2014). An extractive graph-based Arabic text summarization approach. In The International Arab Conference on Information Technology.

[9] Ma, Y., & Wu, J. (2014, December). Combining n-gram and dependency word pair for multi-document summarization. In 2014 IEEE 17th International Conference on Computational Science and Engineering (pp. 27-31). IEEE.

[10] Wang, J., & Dong, Y. (2020). Measurement of text similarity: a survey. Information, 11(9), 421.
[11] Efron, M. (2008). Query expansion and dimensionality reduction: Notions of optimality in rocchio relevance feedback and latent semantic indexing. Information processing & management, 44(1), 163-180.
[12] Imam, I., Nounou, N., Hamouda, A., Allah, H., & Khalek, A. (2013). Query Based Arabic Text Summarization 1.
[13] Lin, C. Y. (2004, July). Rouge: A package for automatic evaluation of summaries. In Text summarization branches out (pp. 74-81).