Comparing the Effectiveness of Stemming and Lemmatization in Arabic Text Summarization with XLSum

Ahmed Eid, Marwan Farag, Mohamed Abdelhamid

May 4, 2023

Abstract

Text summarization is a challenging task that aims to produce a concise summary of a given text while retaining its most important information. In Arabic, this task is particularly challenging due to the complex morphology and rich lexical diversity of the language. Stemming and lemmatization are common techniques used in NLP to reduce the dimensionality of text data and improve text summarization. In this paper, we investigate the impact of stemming and lemmatization on Arabic text summarization with the XLSum dataset. We compare the effectiveness of the two techniques in improving the performance of summarization models and evaluate their impact on the quality of the generated summaries. We also explore the impact of not using stemming or lemmatization on text summarization. The summaries of our findings are to be reported in the second milestone of the project.

1 Introduction

Automatic text summarization is a challenging task in natural language processing (NLP) that aims to produce a shorter version of a given text while retaining its most important information. The need for text summarization has become increasingly important in recent years due to the exponential growth of digital data and the need to process and analyze large volumes of text data. Arabic is a rich and complex language with a diverse and extensive vocabulary, making text summarization a challenging task in Arabic NLP. In addition to the complexity of the language, the availability of high-quality Arabic text summarization datasets is limited.

1.1 Motivation

As stated, Arabic text summarization has become an increasingly important task due to the massive amount of Arabic text available on the internet and in various fields. Compared to Arabic, there is a lot more NLP research on Latin languages, notably summarization.

There are several applications of Arabic text summarization. For example, news aggregation is a significant application. Text summarization can be used to aggregate news articles from various sources and provide users with a summary of the most important events and developments [GSK18]. This can save time and help users stay informed. In addition, text summarization can be useful in legal documents [JBB21]. Legal documents can be long and complex, and summarization can be used to extract key information, making it easier for lawyers and other legal professionals to review and understand the document. Moreover, Summarization can be used to extract key insights and trends from business reports [PBS23], allowing decision-makers to quickly identify important information and make informed decisions. Lastly, text summarization can be used in educational resources [SBS14]. It can be used to create study materials, summaries of textbooks, and other educational resources, helping students to better understand complex concepts and ideas. In conclusion, Arabic text summarization is a useful tool that can be used in a variety of fields to save time, provide better accessibility, and extract key information.

1.2 Related Work

Extractive summarization methods based on statistical [AFA12] and graph-based [ATAO14] approaches have been widely used due to their simplicity and effectiveness. However, they have several limitations, including the lack of contextual understanding, difficulty in handling ambiguity, vulnerability to noise, and lack of coherence. On the other hand, abstractive summarization methods, such as tree-based summarization [KM00] and deep learning Seq2Seq models, have shown promising results in overcoming these limitations. In particular, AraBART, a deep learning model specifically designed for Arabic text summarization, has achieved state-of-the-art performance on multiple datasets, outperforming strong baselines such as pretrained Arabic BERT-based models and multilingual mBART and mT5 models [ETH+22]. Other methodologies, such as semantic-based summarization and ontology-based summarization, also hold potential for further exploration in Arabic text summarization research.

1.3 XLSum Dataset

The XLSum Arabic dataset [HBI+21] is a large-scale Arabic text summarization dataset that was released in 2020. It contains a diverse collection of news articles and their corresponding summaries in Arabic. The dataset was created by collecting news articles from various Arabic news sources and then manually creating summaries for each article. The dataset is freely available for research purposes and has received significant attention from the NLP community as a valuable resource for Arabic text summarization research.

The XLSum Arabic dataset consists of approximately 45,000 news articles collected from various Arabic news sources, such as Al Jazeera, Al Arabiya, and BBC Arabic. The articles cover a wide range of topics, including politics, economics, sports, and culture. Each article in the dataset is accompanied by a corresponding summary, which was manually created by a human summarizer. The summaries are one sentence or a few sentences long and aim to capture the most important information in the article.

The dataset is split into three parts: a training set, a validation set, and a test set. The training set contains approximately 35,000 articles, while the validation and test sets contain approximately 5,000 articles each. The dataset is designed to be used for supervised learning tasks, such as training and evaluating text summarization models.

1.4 Stemming and Lemmatization

Stemming and lemmatization are common techniques used in NLP to reduce the dimensionality of text data and improve text summarization. Stemming involves reducing words to their root form. It is one of the methods employed by information retrieval systems to ensure that word variants are not missed when text is retrieved [BLY14]. Lemmatization, on the other hand, involves reducing words to their base form, typically by using a dictionary or morphological analysis. it aids in matching synonyms while using a thesaurus, thus when one searches for "hot," the term "warm" is also matched [BLY14]. Stemming and lemmatization are widely used in text processing tasks, including text classification [TTJ06], sentiment analysis [BQWR14], and text summarization [TM12].

In this paper, we investigate the impact of stemming and lemmatization on Arabic text summarization with the XLSum dataset. We compare the effectiveness of the two techniques in improving the performance of summarization models and evaluate their impact on the quality of the generated summaries. We also explore the impact of not using stemming or lemmatization on text summarization. Our goal is to provide insights into the usefulness of these techniques in Arabic text summarization and identify the most effective approach for this task and to how extent it is effective compared to the others.

2 Challenges

2.1 Homographic Words

One challenge is dealing with homographic words, which are words that are written differently but have the same meaning. Arabic has several homographic words, such as the use of ه and ة at the end of words, or ي and ى. For example, the two words المستقبلية and المستقبليه may both appear in the dataset, represented by two different words although they should be the same word. Homographic words can create confusion and lead to errors in text summarization. It is important to identify and deal with homographic words to ensure that the summarization is accurate. The severity of this challenge is high

because homographic words can cause significant errors in the summarization process. From analysis of the dataset, it has been found that around 5600 words appear in different shapes.
Plan to solve: We decided to normalize by replacing ة with ه and ى with ي. This will somehow help in normalizing same words that have different shapes to the same shape to avoid confusing the model.

2.2 Large size of vocabulary and text noise

One challenge is that stop words can add noise to the text and reduce the effectiveness of certain natural language processing algorithms. For example, in topic modeling, stop words may appear frequently in many different topics and can therefore be less informative for identifying the most important topics in a document. When processing a large corpus of text, the vocabulary can become very large and require a lot of computational resources to process.
Plan to solve: By removing stop words, we can improve the accuracy of the topic modeling algorithm, extract more meaningful topics and reduce the size of the vocabulary to make it easier to analyze and process the text.

2.3 Do we need stemming or lemmatization or neither?

The challenge of deciding whether to use stemming or lemmatization is a common issue in natural language processing [KSND21], and it can be particularly challenging for Arabic text. While stemming reduces words to their root form by removing affixes, lemmatization maps words to their base or dictionary form while retaining the meaning of the word. The choice between stemming and lemmatization depends on the task at hand and the specific characteristics of the language being analyzed. By intuition, we were concerned that stemming is very strict in the sense that returns words to their roots (ex: استثمر changed to ثمر which somehow has different meaning). At the same time stemming is appropriate in removing non-relevant letters (وفعلوا changed to فعل) which removes the و at the beginning. This is not achieved by lemmatization which changes it to وفعل. In [EBEK20], they claim that, although lemmatization is more time-expensive, doing it preprocessing step will give the best accuracy. Other researches, however, insist on using stemming as a preprocessing step [BGE+20].

Plan to solve: In the case of Arabic text summarization, it is important to experiment with both stemming and lemmatization to determine which approach yields better results. This challenge has been addressed by creating multiple versions of the dataset, each with different preprocessing approaches. This will allow to compare the performance of the different approaches and determine which one works best for specific use case. It is important to note that this is the main focus of our research. To the best of our knowledge, there has been no researches concerned with evaluating the effect of stemming/lemmatization or neither on Arabic text summarization task.

2.4 Lack of gold standard corpus

The main issue with text summarization datasets is the lack of high-quality reference summaries, which makes it challenging to obtain an excellent dataset. Furthermore, there is a scarcity of multi-sentence datasets for abstractive summarization in many languages, including Arabic, with only single sentence datasets being available.
Plan to solve: the XLsum dataset provides a promising solution to this challenge due to its high-quality reference summaries and diverse range of languages covered.

2.5 Diacritics and punctuation

Diacritics are small marks used in Arabic to indicate pronunciation, but they are not essential for understanding the text. Removing them can simplify text comparison for summarization, but it may also increase ambiguity.
Plan to solve: to remove diacritics and punctuation from input articles and their summaries. However, it is important to consider the potential for ambiguity when diacritics are removed [MF07].

3 System Architecture

A model for text summarization should typically consist of several layers that perform different functions to process the input text and generate a summary output. Here are some common layers that we thought are crucial for a text summarization model:

- An Embedding layer: We need to map the input tokens into word embeddings that the model can understand.
- An Encoder: this encoder should consist of multiple multi-head attention layers to make use of the attention mechanism and positional encoding to capture the context of the words.
- Feed-forward layer: This layer should help the model extract features and learn complex non linear relations.
- Decoder layer: The decoder layer generates the summary text by autoregressively predicting each word in the summary sequence based on the previous words it has generated. This layer typically consists of multiple stacked transformer blocks, each of which includes multi-head self-attention, cross-attention, and feedforward layers. The cross-attention mechanism allows the model to attend to the relevant parts of the input text while generating the summary.
- Output Layer: This layer produces the final summary text. This layer typically uses a softmax activation function to compute the probabilities of each word in the summary vocabulary, and the word with the highest probability is selected as the next word in the summary sequence.

AraBART [ETH+22] is a useful pretrained model that we can use and fine-tune to the summarization task as it already has the above needed layers. Each layer in AraBART is useful for text summarization in different ways:

1. Tokenization layer: AraBART uses a custom Arabic-specific tokenization scheme to tokenize the input text into subwords. This is necessary to handle the rich morphology of Arabic words, which can have multiple affixes and inflections.
2. Embedding layer: The embedding layer maps each subword token in the input sequence to a high-dimensional vector space, where each dimension represents a different feature or aspect of the token. AraBART uses a shared vocabulary for both input and output sequences, which allows the model to share the embeddings between the encoder and decoder.
3. Encoder layer: The encoder layer processes the input sequence and produces a condensed representation of the input text. The encoder in AraBART consists of 12 transformer blocks, each of which includes multi-head self-attention and feedforward layers. The encoder also uses position embeddings to preserve the order of the input sequence.
4. Decoder layer: The decoder layer generates the summary text by autoregressively predicting each subword token in the summary sequence based on the previous subwords it has generated. The decoder in AraBART consists of 12 transformer blocks, each of which includes multi-head self-attention, cross-attention, and feedforward layers. The decoder also uses position embeddings and a mask to ensure that each token is generated based only on the previous tokens.
5. Output layer: The output layer produces the final summary text by selecting the subword token with the highest probability in the summary vocabulary. The output layer in AraBART uses a linear layer followed by a softmax activation function to compute the probabilities of each token in the summary vocabulary.

References

[AFA12] Fahad Alotaiby, Salah Foda, and Ibrahim Alkharashi. New approaches to automatic headline generation for arabic documents. *Journal of Engineering and Computer Innovations*, 3(1):11–25, 2012.

[ATAO14] Ahmad T Al-Taani and Maha M Al-Omour. An extractive graph-based arabic text summarization approach. In *The International Arab Conference on Information Technology*, pages 158–163, 2014.

[BGE+20] Asmaa A Bialy, Marwa A Gaheen, RM ElEraky, AF ElGamal, and Ahmed A Ewees. Single arabic document summarization using natural language processing technique. *Recent Advances in NLP: The Case of Arabic Language*, pages 17–37, 2020.

[BLY14] Vimala Balakrishnan and Ethel Lloyd-Yemoh. Stemming and lemmatization: A comparison of retrieval performances. 2014.

[BQWR14] Yanwei Bao, Changqin Quan, Lijuan Wang, and Fuji Ren. The role of pre-processing in twitter sentiment analysis. In *Intelligent Computing Methodologies: 10th International Conference, ICIC 2014, Taiyuan, China, August 3-6, 2014. Proceedings 10*, pages 615–624. Springer, 2014.

[EBEK20] Reda Elbarougy, Gamal Behery, and Akram El Khatib. A proposed natural language processing preprocessing procedures for enhancing arabic text summarization. *Recent Advances in NLP: The Case of Arabic Language*, pages 39–57, 2020.

[ETH+22] Moussa Kamal Eddine, Nadi Tomeh, Nizar Habash, Joseph Le Roux, and Michalis Vazirgiannis. Arabart: a pretrained arabic sequence-to-sequence model for abstractive summarization. *arXiv preprint arXiv:2203.10945*, 2022.

[GSK18] Partha Protim Ghosh, Rezvi Shahariar, and Muhammad Asif Hossain Khan. A rule based extractive text summarization technique for bangla news documents. *International Journal of Modern Education and Computer Science*, 10(12):44, 2018.

[HBI+21] Tahmid Hasan, Abhik Bhattacharjee, Md Saiful Islam, Kazi Samin, Yuan-Fang Li, Yong-Bin Kang, M Sohel Rahman, and Rifat Shahriyar. Xl-sum: Large-scale multilingual abstractive summarization for 44 languages. *arXiv preprint arXiv:2106.13822*, 2021.

[JBB21] Deepali Jain, Malaya Dutta Borah, and Anupam Biswas. Summarization of legal documents: Where are we now and the way forward. *Computer Science Review*, 40:100388, 2021.

[KM00] Kevin Knight and Daniel Marcu. Statistics-based summarization-step one: Sentence compression. *AAAI/IAAI*, 2000:703–710, 2000.

[KSND21] Divya Khyani, BS Siddhartha, NM Niveditha, and BM Divya. An interpretation of lemmatization and stemming in natural language processing. *Journal of University of Shanghai for Science and Technology*, 22(10):350–357, 2021.

[MF07] Mohanned Momani and Jamil Faraj. A novel algorithm to extract tri-literal arabic roots. In *2007 IEEE/ACS International Conference on Computer Systems and Applications*, pages 309–315. IEEE, 2007.

[PBS23] Krutika Patil, Medha Badamikar, and Sheetal Sonawane. Nlp based text summarization of fintech rfps. In *2023 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, pages 865–869. IEEE, 2023.

[SBS14] Md Arafat Sultan, Steven Bethard, and Tamara Sumner. Towards automatic identification of core concepts in educational resources. In *IEEE/ACM Joint Conference on Digital Libraries*, pages 379–388. IEEE, 2014.

[TM12] Juan-Manuel Torres-Moreno. Beyond stemming and lemmatization: Ultra-stemming to improve automatic text summarization. *arXiv preprint arXiv:1209.3126*, 2012.

[TTJ06] Michal Toman, Roman Tesar, and Karel Jezek. Influence of word normalization on text classification. *Proceedings of InSciT*, 4:354–358, 2006.