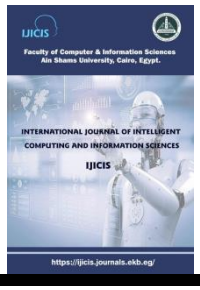
IJICIS, Vol.25, No.1, 74-88
DOI: 10.21608/ijicis.2025.375275.1389

International Journal of Intelligent Computing and Information Sciences

https://ijicis.journals.ekb.eg/



AUTOMATIC SUMMARIZATION TECHNIQUES FOR ARABIC TEXT

Karim Morsi*

Information Systems Departement
Faculty of Computer and Information Science, Ain-Shams University,
Cario,Egypt
karim.mohamed.fcis@cis.asu.edu.eg

Fatma Najib

Information Systems Departement
Faculty of Computer and Information Science, Ain-Shams University,
Cario,Egypt
fatma_mohamed@cis.asu.edu.eg

Wedad Hussein

Information Systems Departement
Faculty of Computer and Information Science, Ain-Shams University,
Cario,Egypt
wedad.hussein@cis.asu.edu.eg

Rasha Ismail

Information Systems Departement
Faculty of Computer and Information Science, Ain-Shams University,
Cario,Egypt
rashaismail@cis.asu.edu.eg

Received 2025-04-13; Revised 2025-04-13; Accepted 2025-04-16

Abstract: *The fast growth of data has transformed text processing, making it challenging to extract key information efficiently. Text summarization techniques address this by reducing lengthy documents while retaining essential content. Automatic text summarization can be broadly categorized into two main types, extractive and abstractive summarization. In extractive summarization, the final summaries are constructed by selecting and extracting content directly from the source text. On the other hand, abstractive summarization takes a different approach. It aims to understand the source text and convey its core ideas in a more concise form using linguistic techniques. Arabic is spoken by over 300 million people and serves as the official language in 22 countries. There is a growing demand for effective Arabic summarization systems to facilitate efficient information processing and retrieval in the Arab-speaking world. Transformers revolutionize NLP by using self-attention to capture long-range dependencies and process input sequences at the same time, improving efficiency. In abstractive summarization,Transformers play an important role because they produce clear, logical summaries that go beyond simply selecting important passages and rewriting the text in a way that is human-like. In this paper, we present a comprehensive investigation of Arabic summarization datasets and techniques introduced to date, with a focus on fine-tuning and using pre-trained transformer models for Arabic summarization, such as AraT5 and AraBERT. We compare their performance using the ROUGE metric on the Wikilingua multi-sentence dataset and find that AraT5 outperforms AraBERT, showing its effectiveness in abstractive summarization tasks.*

Keywords: *Natural language processing, text summarization, abstractive Arabic summarization, Deep learning, and transformers.*

***Corresponding Author**: Karim Morsi

Information Systems Department, Faculty of Computer and Information Science, Ain Shams University, Cairo, Egypt

Email address: karim.mohamed.fcis@cis.asu.edu.eg

1. Introduction

In fact, text summarization has grown considerable importance recently. This is primarily due to the vast volume of text generated daily on the internet in diverse formats, making pre-existing text data readily accessible through online sources or personal and corporate computers. Text summarization serves the essential purpose of reducing lengthy texts into shorter versions while retaining the core ideas, thus significantly saving time during the reading process [1]. It effectively addresses the challenge of having to go through extensive literatures on the specific topic to get the main ideas. Moreover, in comparison to a human-generated summary, automated text summarization not only saves time but also offers potential cost savings.

Furthermore, automatic text summarization plays an essential role in various applications. It uses intelligent filtering techniques to ease the extraction of essential information from digital documents, facilitating the discovery of embedded knowledge within these documents This technology also helps with coping with the vast quantity of textual data available. By reducing, categorizing, and retrieving information as needed, document summarization helps address the challenges created by the vast amount of information available on the Internet [1]. Moreover, text summarization has a wide range of practical uses, including summarizing articles for websites, enhancing search engine results, and simplifying the comprehension of theses and dissertations. It is also important in developing solutions for managing and filtering information resources, ensuring that only relevant data is retrieved from them. This versatility makes text summarization a valuable tool in streamlining information access and improving efficiency across numerous domains.

The complexity of the Arabic language presents significant challenges for the text summarizing area, which has primarily concentrated on the English language. Arabic has special difficulties because of its complex morphology, diglossia, and variety of dialects. Because of these linguistic features, automatic summarization in Arabic is more difficult than in English. Furthermore, extractive summary techniques which choose and link together preexisting sentences or phrases from the source text are the foundation of most text summarization solutions currently in use, including those for Arabic. Particularly in the Arabic context, abstractive summarization which means generating new, condensed text that communicates the key ideas is less common [1]. The significance of developing text summarizing methods that are relevant to the distinct linguistic characteristics and various requirements of the Arabic language is being acknowledged by researchers and developers [2].

In this paper, we present a comprehensive investigation of Arabic summarization datasets and techniques introduced to date, with a particular focus on the fine-tuning and use of pre-trained transformer models such as AraT5 and AraBERT for Arabic abstractive summarization. We evaluate their performance on the multi-sentence Wikilingua dataset using the ROUGE metric and show that AraT5 outperforms AraBERT, highlighting its effectiveness in this domain. The remainder of this paper is organized as follows: Section 2 reviews related work in the field. Section 3 provides an overview of automatic text summarization system classifications and approaches. Section 4 describes commonly used datasets and evaluation methods. Section 5 details the proposed methodology and experimental setup. Section 6 presents the results and discussion, and finally, Section 7 concludes the paper, and Section 8 outlines potential directions for future work.

2. Related Work

Deep learning, characterized by cascading nonlinear processing units, is a powerful method for feature extraction and representation learning. It operates through multiple feature layers, learning from input data either in a supervised or unsupervised manner. In contrast to traditional text summarization methods, which often involve manually extracting terms and suffer from the inclusion of unnecessary words, deep learning models, such as recurrent neural networks (RNNs) and Long Short-Term Memory (LSTM) networks, address these drawbacks. LSTMs are effective for understanding context and have applications in sequence-to- sequence tasks. The Bidirectional LSTM (Bi-LSTM) approach reduces sequence-to-sequence challenges, and attention mechanisms further enhance model accuracy. Transformers, as seen in models like BERT, have revolutionized language representation learning [2]. BERT, a bidirectional pre-trained model, uses self-attention mechanisms and fine-tuning to adapt to specific tasks. Pre-trained word embeddings, including word2vec and Glove, are also crucial in representing text data. These techniques, particularly deep learning and transformer-based models, have advanced the field of Arabic text summarization, offering improved extraction and quality. Table 1 shows Comparison among different techniques for Arabic summarization.

The authors in [3] proposed an automatic and extractive method for single-document summarization in the Arabic language. The proposed method aims to create informative summaries by evaluating each sentence's importance based on a combination of statistical and semantic features. Two summarization techniques, score-based and supervised machine learning, are utilized to generate the summary. The proposed method is extractive, meaning it selects sentences from the original document to create the summary. The second summarization technique is based on supervised machine learning. It also uses a set of features to determine sentence importance. Statistical Features, the features used include statistical measures that assess the importance of sentences. Semantic-Based Features, Semantic features capture the meaning and context of sentences, enabling the method to select sentences that cover the entire document's ideas. The method is evaluated using the EASC dataset, which is likely a corpus of Arabic text documents. The evaluation of the proposed method is performed using the ROUGE-2 measure, a common metric for assessing the quality of text summarization. The F-score of machine learning approach was 0.524 and an F-score of score-based approach was 0.617. But the model only selects and includes sentences from the original document in the summary. It may not be capable of generating abstractive summaries, which could be a limitation if abstractive summarization is required. The success of the model relies on the selection and formulation of appropriate features.

The authors in [4] proposed AraBART, the first Arabic model that is pretrained end-to-end with both the encoder and decoder components. AraBART is designed for abstractive summarization tasks and is evaluated on multiple datasets. AraBART is based on the architecture of BART Base, which includes 6 encoder and 6 decoder layers and has 768 hidden dimensions. An extra layer normalization layer is incorporated above both the encoder and decoder. AraBART is evaluated on multiple abstractive summarization datasets, most of which contain news articles with annotated summaries of varying abstractedness levels. AraBART outperforms several strong baseline models, such as a pretrained Arabic BERT-based model, multilingual BART, and multilingual T5. The models perform well particularly on the XL-Sum dataset, which is described as the "most abstractive dataset." And subset of Arabic Gigaword. for XL-Sum The F-score for ROUG1 was 34.5, ROUG2 was 14.6, ROUGL was 30.5 and for BS was 67.0. Macro averages are computed over all datasets and the F-score for ROUG1 was 42.4, ROUG2 was 28.8, ROUGL was 40.3 and for BS was 69.8. But AraBART is a large model with 139

million parameters, which makes it computationally expensive to train and use. This can be a disadvantage for users with limited computational resources.

In [5], the authors introduced an abstractive summarization system based on a sequence-to-sequence (seq2seq) framework that integrates Gated Recurrent Units (GRU), Long Short-Term Memory (LSTM), and Bidirectional LSTM (BiLSTM) networks. The model employed global attention mechanisms in both the encoder and decoder components. To improve the understanding of Arabic words and achieve better results, the AraBERT preprocessing stage was utilized. Additionally, the authors compared two word embedding models, skip-gram and continuous bag of words (CBOW) Word2Vec models. The experimental results were evaluated using several metrics. The author uses two different datasets The Arabic Headline Summary (AHS) dataset 1 and The Arabic Mogalad_Ndeef (AMN) dataset 2 The F-score obtained in dataset 1 for ROUGE-1 was 51.49, for ROUGE-2 it was 12.27, for ROUGE-L it was 34.37, and for BLEU it was 0.41. The F-score obtained in dataset 2 for ROUGE-1 was 44.28, for ROUGE-2 it was 18.35, for ROUGE-L it was 32.46, and for BLEU it was 0.41. These results indicate that the BiLSTM architecture achieved the best performance among the tested models. Furthermore, using the skip-gram Word2Vec model performed better than using the CBOW Word2Vec model. But Words in Arabic has a wide range of vocabulary, and seq2seq models may struggle with out-of-vocabulary words that are not present in the training data.

In [6], the authors introduced a model for abstractive Arabic text summarization that utilizes sequence-to-sequence recurrent neural networks. The encoder is composed of multiple layers (Input text, keywords of the text, text name entities) and utilizes bidirectional LSTMs to capture information from both the past and future context of the input text. In contrast, the decoder employs a single-layer and unidirectional LSTM to generate the abstractive summary. The dataset used for training and evaluation consists of 79,965 documents. The F-score for ROUGE1 was 38.4, ROUGE1-NOORDER was 46.2, ROUGE1-CONTEXT was 58.1 and for ROUGE1-STEM was 52.6. In addition to quantitative measures, a qualitative evaluation was performed. The qualitative measure assessed the readability and relevance of the generated summaries for 50 randomly selected documents. But the proposed model can be get overfitting deep neural networks, especially those with a large number of layers and parameters, are responsible to overfitting the training data. Overfitting can lead to poor generalization, where the model performs well on the training data but poorly on unseen data.

The research in [7] introduced a new abstractive text summarization approach for the Arabic language, called SemG-TS, which is based on semantic graph embeddings and employs a deep neural network for generating abstractive summaries. The study provides a comprehensive evaluation of SemG-TS, comparing its performance to a popular baseline technique, word2vec, using both automatic and human evaluation methodologies. The proposed SemG-TS technique is designed to create abstractive summaries by first representing the original text as a semantic graph, leveraging the unique characteristics of the Arabic language. The graph is then embedded using the SemanticGraph2Vec approach and embedding text is as input to model consisted of Long Short-Term Memory (LSTM) in the Encoder, LSTM Basic Decoder. Articles for experimentation were collected from the AlJazeera.net website. The dataset contained a total of 16,770 paragraphs, with an average of 204 words per paragraph. The ROUGE evaluation measure was used to automatically evaluate the performance of SemG-TS compared to two versions of word2vec, one trained on the dataset and the other with random initial vectors. SemG-TS outperformed both versions of word2vec across all evaluation measures. Specifically, it achieved a 15.8% improvement in precision, a 29.5% improvement in recall, and a 21.4% improvement in F-measure compared to the best version of word2vec (random-based). Human experts conducted a manual evaluation

to assess the relevancy, similarity, readability, and overall satisfaction of the summaries generated by SemG-TS and word2vec. SemG-TS model shows better performance than word2vec, evident in ROUGE scores and overall text quality in terms of relevance, readability, and satisfaction. The F-score for ROUGE was 0.047. But semantic graph embeddings can be complex and might not be as clear as word embeddings. Understanding and fine-tuning the embedding process can be challenging.

The authors in [8] proposed an Extractive summarization system. The model relies on a textual graph, where the original text is transformed into a graph using a novel formulation. This formulation takes into account factors like sentence relevance, coverage, and diversity to evaluate each sentence in the text. Both statistical and semantic criteria are employed to determine the importance of each sentence. A sub-graph is then constructed to reduce the size of the original text. This step typically involves choosing a subset of the most relevant and significant sentences from the textual graph to form a more concise summary of the original text. Less important or irrelevant phrases are eliminated from the summarized sentences. The system achieved an F-score of 0.617 using the ROUGE-2 metric to evaluate performance. But the model described appears to focus on extractive summarization, where the summary is composed of selected sentences from the original text. While extractive summarization can be effective in capturing important information, it may not be able to generate summaries that go beyond the content of the original text.

A comparative study was performed in [9] that compares the performance of models based on RNN and Transformer architectures, specifically mBERT, AraBERT, AraGPT2, and AraT5 (pre-trained language models), for abstractive summarization in Arabic. These models are known for their ability to understand and generate text in Arabic. The researchers built a large Arabic summarization dataset comprising 84,764 high-quality pairs of text and summaries. This dataset serves as the training and evaluation data for the models used in the research. To reduce the under-fitting problem, they extracted an extra dataset comprising 280,000 examples. As a result of retraining the models on the combined dataset, they named the new models "Seq2Seq-LSTM+" and "Transformer+". The "+" designation indicates that these models are trained on the expanded dataset and are expected to show improved performance compared to their previous versions. The system achieved an F-score of 33.04 for Seq2Seq-LSTM, 32.12 for Transformer, 37.57 for Seq2Seq-LSTM+, 39.61 Transformer+, 42.96 mBERT2-mBERT, 40.48 AraGPT2, 44.02 BERT2BERT, 46.87 AraT5 using ROUGE-L. But Transformer-based models typically require a large amount of training data to achieve optimal performance. Although the researchers addressed the under-fitting issue by adding an additional dataset, the models may still show limitations if the training data does not capture the full diversity and complexity of the target domain or if the dataset is imbalanced or contains biases. While the models may perform well on in-domain data, their performance may decrease when applied to out-of-domain text.

A hybrid approach for Arabic summarization was proposed in [10], which combines a Modified Sequence-To-Sequence (MSTS) model with a transformer-based model. The MSTS model is adapted by adding multi-layer encoders and a single-layer decoder. This model is employed for extractive summarization, generating summaries by selecting and reordering sentences from the input text. The output of the extractive summarization is then processed by the transformer-based model to produce abstractive summaries. Additionally, they introduce a new Arabic benchmark dataset, the HASD (Arabic Summarization Dataset), and modify the widely-used extractive EASC benchmarks by adding abstractive summaries to each text in the EASC dataset. They also propose a novel evaluation metric called the Arabic-ROUGE measure, which evaluates the quality of abstractive summaries based on their structure and word similarity. It is specifically designed for evaluating the effectiveness of abstractive

summarization in the Arabic language. The F-score obtained in EASC for ROUGE-1 was 0.587, for ROUGE-2 it was 0.48, for ROUGE-L it was 0.56, BLEU it was 0.42 and for Arabic Rouge was 0.652. The F-score obtained in HASD for ROUGE-1 was 0.6374, for ROUGE-2 it was 0.4908, for ROUGE-L it was 0.6047, BLEU it was 0.44, and for Arabic Rouge was 0.713. But Combining a Modified Sequence-To-Sequence (MSTS) model with a transformer-based model can lead to increased model complexity. This complexity may result in longer training times, increased resource requirements.

TABLE 1. Comparison between different Arabic summarization methods

Model	Summary Approach	Methodology	Dataset	ROUGE			BLEU
				1	2	L	
Machine learning-Model [3]	Extractive summary	Score-Based Algorithm. Machine learning-based method.	Essex Arabic Summaries Corpus (EASC)	0.643	0.617	—	—
Transformer Model [4]	Abstractive summary	AraBART Model.	Arabic Gigaword XL-Sum	averages over all dataset: 42.4	averages over all dataset: 28.8	averages over all dataset: 40.3	—
Seq2Seq Model [5]	Abstractive summary	Seq2Seq-BiLSTM-GRU model with global attention	The Arabic Headline Summary (AHS) The Arabic Mogalad_ Ndeef (AMN).	AHS : 51.49 AMN : 44.28	AHS : 12.27 AMN : 18.35	AHS : 34.37 AMN : 32.46	AHS : 0.41 AMN : 0.41
Seq2Seq Model [6]	Abstractive summary	Seq2Seq-BiLSTM model with global attention	SANAD_ SUBSET and other resources	38.4	—	—	—
Seq2Seq Model [7]	Abstractive summary	Seq2Seq-BiLSTM model with global attention using SemanticGraph2vec embedding.	8385 documents collected from the AlJazeera. net website and CNN-Arabic news.	0.047	—	—	—
Graph-Based Approach [8]	Extractive summary	Graph-Based Text Summarization.	Essex Arabic Summaries Corpus (EASC)	—	0.617	—	—
Seq2Seq Model Transformer Model [9]	Abstractive summary	Seq2Seq-BiLSTM model with global attention. Transformer Model.	SumArabic	49.06	30.81	46.87	—
Seq2Seq and Transformer Model [10]	Abstractive summary	Seq2Seq-BiLSTM model with global attention integrated with mT5.	Hybrid Arabic text summarization dataset (HASD) which contain extractive and abstractive summary Essex Arabic Summaries Corpus (EASC)	EASC : 0.587 HASD: 0.6374	EASC : 0.48 HASD: 0.4908	EASC : 0.56 HASD: 0.6047	EASC: 0.42 HASD 0.44

3. Automatic Text Summarization Systems Classifications and Approaches

3.1. Classifications of the Automatic Text Summarization Systems

In Figure 1, a variety of classifications for automatic text summarization systems are illustrated. These systems can be categorized based on several criteria, each highlighting a different aspect of the summarization process. First, according to the summarization approach, systems can be extractive, abstractive, or hybrid. Second, based on the input size, summarization systems are classified into single-document and multi-document summarization. Third, classification based on the nature of the output summary includes generic and query-based summarization. Fourth, depending on the learning paradigm, summarization algorithms can be supervised or unsupervised. Fifth, considering the content of the summary, systems are divided into informative and indicative summarization. Sixth, based on the type of summary generated, categories include headline generation, sentence-level summarization, highlights, and full summaries. Seventh, with respect to the summarization domain, systems may be general-purpose or domain-specific. Finally, summaries can also be classified by language into monolingual, multilingual, and cross-lingual summarization. These classification criteria offer a structured perspective for analyzing the various dimensions and functionalities of automatic text summarization systems.

The two main categories of automatic text summarization based on the approach, are extractive and abstractive [11]. When summaries are created using only content that has been extracted, the final sentences produced are phrases or words that were taken from the source text. This process is known as extractive summarization, while abstractive summarization aims to understand the text and express its main ideas in a shorter form using linguistic techniques [11].

According to input size Automatic text summarization systems can be categorized into two main types, single document summarization and multi-document summarization. Single-document summarization systems focus on generating a summary from a single source information, main ideas, and important points within that single document. This type of summarization is commonly used for tasks like summarizing news articles, research papers, or blog posts [12]. Multi-document summarization systems, on the other hand, create summaries by processing and summarizing multiple source documents. These systems are designed to extract and reduce relevant information from a set of documents covering the same or related topics.

Classification based on the nature of the output summary can be further divided into two subcategories, generic summarization and query-based summarization. Generic text summarization involves identifying and extracting key information from one or more documents to produce a concise summary that captures the overall meaning or main ideas of the content. The generated summary in query-based summarization includes content that is directly related to the original search query. It focuses on providing information that answers the specific query and meets the user's information needs [13].

Automatic text summarization algorithms can be categorized as either supervised or unsupervised. In supervised summarization, the algorithm requires an annotated dataset during the training phase. This training data consists of examples of source documents and their corresponding human-generated summaries [14]. Unsupervised algorithms use various statistical, linguistic, and machine learning techniques to extract or generate summaries directly from the source documents.

Automatic text summarization can also be categorized based on the characteristics of the summary's content into two main types, informative and indicative. An indicative summary omits specific details and only communicates the main ideas or topics of the original material. An informative summary, on the other hand, includes the essential details and main concepts from the original text. Without going into the specifics, it covers all the important subjects and information [15].

Automatic text summarization based on the type of summary generated by the system can include several categories, headline, sentence-level, highlights, and full summary. Headline generation creates a brief title or headline, typically shorter than a complete sentence [16]. Sentence level summarization aims to generate a single sentence summary from the input text [16]. These summaries are often abstractive in nature, Highlights Summarization are often presented in the form of bullet points or key points. Full summary generation aims to create a more comprehensive summary that may vary in length based on specific requirements.

Classification based on the summarization domain can be categorized into two main types, general or domain-specific summarization. General summarization, also known as domain-independent summarization, is designed to summarize documents that belong to various domains and cover a wide range of topics [17]. In contrast, domain-specific summarization focuses on generating summaries for documents within a specific domain or area of expertise, such as medical documents, legal documents, financial reports, or scientific literature.

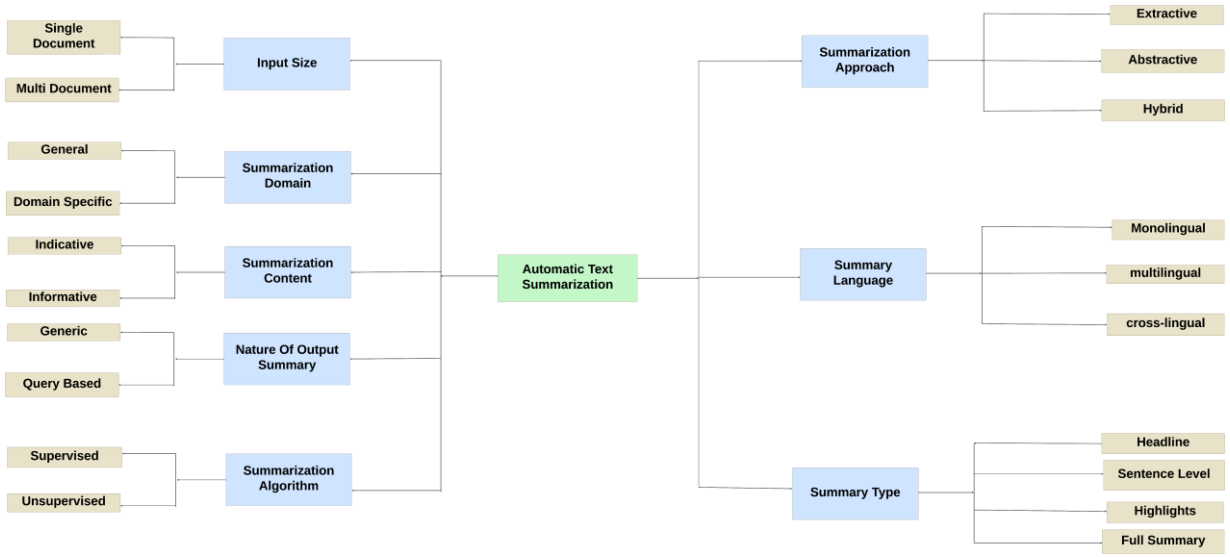


Figure 1 : Classifications of automatic text summarization systems.

Summaries can be classified according to their language into three main types: monolingual, multilingual, and cross-lingual summarization. Monolingual summarization systems operate when the language of the source document and the language of the generated summary are the same, and they aim to extract or generate summaries in that language [17]. Multilingual summarization systems are used when the source text is written in multiple languages, and the system can generate summaries in those same languages. When the original text is written in one language and the system produces the summary in another, this is known as cross-lingual summarizing.

3.2. Approaches and Techniques for Automatic Text Summarization

There are three main types to text summarization: extractive, abstractive, and hybrid. Each approach uses different techniques. Statistical-based methods, concept-based methods, topic-based methods, sentence centrality or clustering-based methods, graph-based methods, semantic-based methods, machine learning-based methods, deep learning-based methods, and fuzzy-logic-based methods are those used in extractive summarization [17]. Three categories can also be used to group abstractive methods: Structure-based methods rely on pre-established frameworks like graphs, trees, rules, templates, and ontologies, semantic-based methods use natural language generation techniques involving information units, predicate-argument structures, and semantic graphs, and deep-learning-based methods. Neural-based and classical methods are two further classifications for abstractive techniques, with the term "classical methods" generally referring to non-neural-based approaches. Furthermore, hybrid approaches combine abstractive and extractive techniques. Extractive-to-abstractive and extractive-to-shallow-abstractive approaches are the two main categories of hybrid summarization.

4. Text Summarization Datasets and Evaluation

4.1. Text Summarization Datasets

Arabic datasets are collections of texts or documents in Arabic that are used for training and evaluating models for various NLP tasks. Datasets are important for developing and benchmarking algorithms. Finding standard testing datasets and their associated human-created "golden summaries" for the Arabic language can be challenging. This is a common issue for languages with fewer resources and less extensive research compared to English.

The large number of sentences in the text, text size, and data collection all affect how accurate automatic text summarization becomes. These factors play a significant role in determining how well a summarization system can perform.

One well-known source for abstractive single-document summarization in Arabic is the "Document Understanding Conference Datasets of Arabic" (DUC2004). This dataset includes 100 Arabic news stories with four handwritten summaries for each article [18]. How-ever, it's important to note that the summaries are written in English, presenting a language challenge for training models.

The "Essex Arabic Summaries Corpus" is a valuable linguistic resource, specifically focused on Arabic text summarization. This corpus consists of a collection of Arabic documents alongside human-generated summaries [19]. The corpus is important in advancing research and developments in the field of Arabic natural language processing, offering valuable insights and benchmarking capabilities.

The "WikiLingua Arabic Text Summarization Dataset" is a significant resource for the development and evaluation of text summarization models in the Arabic language [20]. It is a collection of 770,000 news articles and summary pairings in from WikiHow in 18 languages including, 29,229 Arabic texts and an abstractive document summaries.

The "XL_Sum Arabic Text Summarization Dataset" is a valuable resource for Arabic text summarization research. This dataset encompasses a diverse collection of Arabic articles along with corresponding summaries, facilitating the development and evaluation of summarization models [21].

The "SANAD Dataset" is a substantial resource for single-document extractive summarization in the Arabic language. This dataset includes an extensive amount of Arabic news articles from the three well-known news websites Akhbarona, Al-Arabiya, and Al-Khaleej. The SANAD Dataset includes 190,000 articles in total [22].

"GigaWord 5 Arabic Text Summarization Dataset" offers a comprehensive collection of Arabic news articles, making it a flexible choice for research in extractive and abstractive summarization [23]. With its vast corpus of news articles, it serves as a crucial benchmark for evaluating summarization models in the Arabic language.

4.2. Summary Evaluation

There are two main approaches to evaluating automatically generated summaries: manual and automatic. Each has its own set of difficulties [17]. To generate a concise and readable summary, it should cover key ideas, address significant topics, minimize redundancy, and maintain coherence.

4.2.1. Manual Evaluation

Manual evaluation, which involves human annotators review both the original text and the summary and provide subjective ratings based on predetermined criteria, is essential for evaluating the quality of text summarization. Judges can assess the quality of summaries with the help of criteria and guidelines provided by the National Institute of Standards and Technology (NIST) [17]. The criteria include linguistic conciseness, referential clarity, relevance, organizational structure, and overall coherence, among others. A qualitative scale with points ranging from 1 (worst) to 5 (best) is also defined. Nevertheless, no summary is perfect, and every assessment is personal.

4.2.2. Automatic Evaluating

Because summaries and original documents must be read by humans, manual summaries evaluation and analysis takes a lot of time and effort. On the other side, automatic text summarization can be evaluated using a variety of metrics [24]. These metrics aim to assess both the accuracy of the generated summary and the quality of the selected content. ROUGE is one of the most widely used evaluation frameworks, including variants such as ROUGE-N, ROUGE-L, and ROUGE-W. These metrics measure the overlap of words or n-grams between the generated summaries and reference (gold) summaries.

- ROUGE-N measures n-gram recall between a candidate summary and reference summaries as in Equation 1.

- ROUGE-L (R-L): The longest common subsequence between a reference summary and a candidate summary is measured by ROUGE-L, making it particularly useful for evaluating summaries that differ in word order but convey similar meanings.
- ROUGE-W: measures word overlap, with a weighting mechanism that emphasizes longer n-grams.

$$\boldsymbol{ROUGE-N} = \frac{\sum_{S \in Refrence\ Texts} \sum_{n-gram \in S} \boldsymbol{Match}(n-gram)}{\sum_{S \in Refrence\ Texts} \sum_{n-gram \in S} \boldsymbol{Count}(n-gram)} \qquad (1)$$

where the maximum number of n-grams that occur simultaneously in a candidate text and a set of reference texts is represented by Match(n-gram).

Precision in summarization evaluation is calculated as the proportion of sentences shared between the candidate summary and the reference summaries to the total number of sentences in the candidate summary, as shown in Equation 2. Recall, as defined in Equation 3, is the ratio of overlapping sentences between the candidate and reference summaries to the total number of sentences in the reference summary. F-Measure integrates recall and precision as in Equation 4 [24].

$$\text{Precision} = \frac{\boldsymbol{Sref} \cap \boldsymbol{Scan}}{\boldsymbol{Scand}} \quad (2)$$

$$\text{Recall} = \frac{\boldsymbol{Sref} \cap \boldsymbol{Scan}}{\boldsymbol{Sref}} \quad (3)$$

$$\text{F-Measure} = \frac{\mathbf{2}\,(\boldsymbol{Precision})(\boldsymbol{Recall})}{\boldsymbol{Precision}+\boldsymbol{Recall}} \quad (4)$$

where " $\boldsymbol{Sref}$ " indicates the number of sentences that appear in reference summaries, and " $\boldsymbol{Scan}$" indicates the count of sentences present in the summary generated by the system.

5. Methodology And Experimental Setup

Transformers represent a powerful neural network architecture specifically designed for handling sequential data. By using self-attention mechanisms, they enable models to capture long-range dependencies between tokens within a sequence. In contrast to traditional recurrent networks, transformers process inputs in parallel, which significantly enhances their efficiency and scalability for tasks such as translation, summarization, and text generation. AraT5 is a transformer-based model derived from Google's T5 (Text-To-Text Transfer Transformer) architecture and is specifically pre-trained for the Arabic language. It approaches all NLP tasks in a text-to-text format, treating both inputs and outputs as text sequences. In contrast, AraBERT is built on the BERT (Bidirectional Encoder Representations from Transformers) architecture and is pre-trained specifically for Arabic. However, since summarization typically requires handling variable-length inputs and outputs through an encoder-decoder structure, AraBERT is less appropriate for this task because it is an encoder-only model.

We use a BERT2BERT encoder-decoder architecture and initialize it using AraBERT's pre-trained weights where the encoder captures the contextual representation of the input text, and the decoder generates the abstractive summary.

5.1. Models

5.1.1 AraT5

The T5 model is an encoder-decoder architecture that combines all natural language processing activities into a single text-to-text framework. AraT5 is the Arabic adaption of this model. With this method, the model can use the same structure to do many tasks, such as classification, machine translation, and text summarization. The training of AraT5 is based on Masked Language Modeling (MLM) the model understands the structure and context of language by requiring it to predict the missing or masked tokens.

In this study, we fine-tuned the pre-trained version provided by [9]. They used $\text{AraT5}_{\text{Base}}$ version [8] which is derived from the T5_{Base} model architecture. Twitter and Modern Standard Arabic (MSA) data

were used to train this version. The Twitter dataset includes 1.5 billion tweets, selected to contain a minimum of three Arabic words, whereas the MSA dataset consists of 70 GB of text taken from diverse Arabic sources. The encoder and decoder AraT5 consist of 12 layers and 12 attention heads for each. We employed the Adam optimizer (Adaptive Moment Estimation), with a learning rate of 0.00002 and a batch size of 2 and gradient accumulation of 8 steps. We evaluated the model using standard evaluation metrics for abstractive summarization, including accuracy, recall, and F-measure scores for each ROUGE metric.

5.2.2 AraBERT

AraBERT is a pre-trained transformer model designed specifically for Arabic, based on the BERT architecture. Unlike models designed for sequence-to-sequence tasks, AraBERT is an encoder-only model, making it well-suited for tasks such as text classification, named entity recognition, and question answering. However, for summarization tasks, which typically require both encoding and decoding, a BERT-to-BERT approach can be employed. In this study, we fine-tuned an enhanced version of AraBERT, which had been modified by [9] to fit text generation tasks, particularly for abstractive summarization. In this version, AraBERT's weights are used to initialize the encoder component of a BERT2BERT encoder-decoder architecture. The decoder, which is also based on BERT2BERT, is then fine-tuned for the summarization task. With this configuration, AraBERT's powerful encoding powers are extended to tasks involving text production. The encoder processes the input text, capturing the contextual nuances and representations necessary for summarization. The decoder, initialized with AraBERT weights, generates the summary by predicting the most likely sequence of words given the encoded input. This combination allows for effective text summarization by leveraging AraBERT's pre-trained knowledge and adapting it to a sequence-to-sequence framework. In the decoder, we incorporated a cross-attention layer, initialized with random weights, placed between the self-attention and feed-forward layers in each block. We also modified the self-attention mechanism in the decoder to be unidirectional, ensuring that it only considers previously generated tokens and does not access future tokens. Lastly, a language modeling (LM) head was added after the final block of the decoder to aid in generating summary tokens. We employed the Adam optimizer, with a learning rate of 0.00002 and a batch size of 2 and gradient accumulation of 8 steps. We evaluated the model using accuracy, recall, and F-measure values for each ROUGE metric. Before calculating these ROUGE metrics, we applied stemming to both the predicted and reference texts to normalize the words and enhance the consistency of the evaluation.

5.2. Dataset

We used the Wikilingua dataset [20], , which contains approximately 29,229 news articles along with their corresponding abstractive summaries. Tokenization plays an important role in filling the gap between raw text data and the numerical input required for machine learning models, making it an essential tool in natural language processing research. To enable effective model training, evaluation, and data preprocessing, The Hugging Face Transformers' AutoTokenizer was employed to tokenize the data for both AraT5 and AraBERT models. We limited the input text to 512 tokens, padding shorter sentences to this length. For the summarization output, we restricted the length to 128 tokens. The dataset was split into 19,992 records (70%) for training, 2,859 records (10%) for validation, and 5,711 records (20%) for testing.

6. Results and Discussion

When applying the previously mentioned fine-tuning models for multi-sentence abstractive summarization, AraT5 shows better performance than AraBERT. AraT5 (an encoder-decoder model), which is particularly well-suited for complex text generation tasks like summarization. This architecture allows for effective handling of both the encoding of the input text and the generation of the summary, providing a more coherent and contextually accurate output. In contrast, AraBERT (an encoder-only model), which is primarily designed for tasks that involve understanding and encoding text rather than generating new text. Although AraBERT excels in tasks like classification and extraction, its lack of a dedicated decoding component limits its effectiveness in generating summaries. Moreover, using an encoder-decoder model with pre-initialized weights from an encoder-only model, like AraBERT, does not leverage the full potential of the encoder-decoder architecture. The original encoder-decoder architecture, such as that found in AraT5 models, is built to handle the end-to-end process of text generation from scratch. This native architecture ensures that both the encoding and decoding processes are optimally aligned for generating high-quality text outputs, such as summaries. In contrast, while initializing a decoder with weights from an encoder-only model, it often results in ineffective performance compared to using a model where the encoder and decoder are designed to work together smoothly from the beginning. Figure 2 shows that AraT5 outperforms AraBERT in ROUGE metrics, generating more accurate and coherent summaries

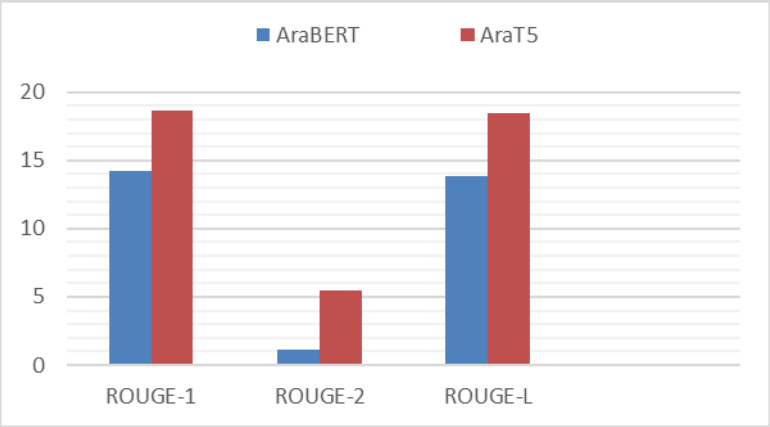


Figure 2: Test set scores based on the ROUGE F1 evaluation.

7. Conclusion

The need for Arabic natural language processing increases as more Arabic documents are becoming available online. But compared to English, this field's research has been comparatively lacking, posing challenges. The unique features of the Arabic language, such as morphological richness and orthographic ambiguity, create more homographs and ambiguity compared to English. It is important for reducing the quality gap between summaries written by humans and automatically. In this study, we conduct a comprehensive review of Arabic summarization datasets and techniques introduced to date. We explored two types of transformer-based architectures for Arabic text summarization, the encoder-decoder architecture like AraT5 and the encoder-only architecture like AraBERT. To implement AraBERT model, we used a BERT-to-BERT encoder-decoder architecture, and initialize it with AraBERT weights. We used Wikilingua multi-sentence dataset to train these modes. Our results indicate that AraT5's encoder-decoder architecture is particularly effective for abstractive summarization tasks, as it easily integrates encoding and generation processes. This leads to higher-quality and contextually relevant summaries compared to the AraBert model. These findings highlight the importance of model architecture in Arabic

text summarization, showing that encoder-decoder frameworks like AraT5 offer significant advantages for generating coherent and informative summaries.

8. Future Work

The future work in Arabic text summarization involves several areas of focus and improvement. Semantic Understanding, Enhancing the understanding of semantic meaning in Arabic text to create more informative and contextually relevant summaries. Multi lingual Summarization, Adapting and extending summarization models to work with multiple languages, allowing for cross-lingual summarization, especially in multilingual regions. Developing solutions specific to different Arabic dialects and writing styles to ensure more accurate and relevant summarization. Real-time Summarization, exploring real-time or dynamic summarization methods that can quickly summarize continuously changing content, such as news or social media. Future studies should explore advanced fine-tuning strategies, hybrid modeling approaches, and the incorporation of additional linguistic resources. Investigating the impact of diverse embedding techniques, reinforcement learning-based training, and transformer-based architecture.

References

1. M. Allahyari et al., “Text summarization techniques: a brief survey,” arXiv Prepr. arXiv1707.02268, 2017.
2. A. Elsaid, A. Mohammed, L. F. Ibrahim, and M. M. Sakre, “A comprehensive review of arabic text summarization,” IEEE Access, vol. 10, pp. 38012–38030, 2022.
3. A. Qaroush, I. A. Farha, W. Ghanem, M. Washaha, and E. Maali, “An efficient single document Arabic text summarization using a combination of statistical and semantic features,” J. King Saud Univ. Inf. Sci., vol. 33, no. 6, pp. 677–692, 2021.
4. M. K. Eddine, N. Tomeh, N. Habash, J. Le Roux, and M. Vazirgiannis, “Arabart: a pretrained arabic sequence-to-sequence model for abstractive summarization,” arXiv Prepr. arXiv2203.10945, 2022.
5. Y. M. Wazery, M. E. Saleh, A. Alharbi, and A. A. Ali, “Abstractive Arabic text summarization based on deep learning,” Comput. Intell. Neurosci., vol. 2022, 2022.
6. D. Suleiman and A. Awajan, “Multilayer encoder and single-layer decoder for abstractive Arabic text summarization,” Knowledge-Based Syst., vol. 237, p. 107791, 2022.
7. W. Etaiwi and A. Awajan, “SemG-TS: Abstractive arabic text summarization using semantic graph embedding,” Mathematics, vol. 10, no. 18, p. 3225, 2022.
8. Y. A. AL-Khassawneh and E. S. Hanandeh, “Extractive Arabic text summarization-graph-based approach,” Electronics, vol. 12, no. 2, p. 437, 2023.
9. M. Bani-Almarjeh and M.-B. Kurdy, “Arabic abstractive text summarization using RNN-based and transformer-based architectures,” Inf. Process. Manag., vol. 60, no. 2, p. 103227, 2023.
10. A. ELSAID, A. MOHAMMED, L. Fattouh, and M. SAKRE, “Hybrid Arabic text summarization Approach based on Seq-to-seq and Transformer,” 2023.
11. L. H. Belguith, M. Ellouze, M. H. Maaloul, M. Jaoua, F. K. Jaoua, and P. Blache, “Automatic summarization,” Nat. Lang. Process. Semit. Lang., pp. 371–408, 2014.
12. M. Joshi, H. Wang, and S. McClean, “Dense semantic graph and its application in single document summarisation,” Emerg. ideas Inf. Filter. Retr. DART 2013 Revis. Invit. Pap., pp. 55–67, 2018.
13. D. Sahoo, R. Balabantaray, M. Phukon, and S. Saikia, “Aspect based multi-document summarization,” in 2016 international conference on computing, communication and automation (ICCCA), IEEE, 2016, pp. 873–877.

14. M. Mohd, R. Jan, and M. Shah, “Text document summarization using word embedding,” Expert Syst. Appl., vol. 143, p. 112958, 2020.
15. V. Gupta and G. S. Lehal, “A survey of text summarization extractive techniques,” J. Emerg. Technol. web Intell., vol. 2, no. 3, pp. 258–268, 2010.
16. F. Dernoncourt, M. Ghassemi, and W. Chang, “A repository of corpora for summarization,” in Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018), 2018.
17. W. S. El-Kassas, C. R. Salama, A. A. Rafea, and H. K. Mohamed, “Automatic text summarization: A comprehensive survey,” Expert Syst. Appl., vol. 165, p. 113679, 2021.
18. D. Harman and P. Over, “The effects of human variation in duc summarization evaluation,” in Text Summarization Branches Out, 2004, pp. 10–17.
19. M. El-Haj, U. Kruschwitz, and C. Fox, “Creating language resources for under-resourced languages: methodologies, and experiments with Arabic,” Lang. Resour. Eval., vol. 49, pp. 549–580, 2015.
20. F. Ladhak, E. Durmus, C. Cardie, and K. McKeown, “WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization,” arXiv Prepr. arXiv2010.03093, 2020.
21. T. Hasan et al., “XL-sum: Large-scale multilingual abstractive summarization for 44 languages,” arXiv Prepr. arXiv2106.13822, 2021.
22. O. Einea, A. Elnagar, and R. Al Debsi, “Sanad: Single-label arabic news articles dataset for automatic text categorization,” Data Br., vol. 25, p. 104076, 2019.
23. C. Napoles, M. R. Gormley, and B. Van Durme, “Annotated gigaword,” in Proceedings of the joint workshop on automatic knowledge base construction and web-scale knowledge extraction (AKBC-WEKEX), 2012, pp. 95–100.
24. C.-Y. Lin, “Rouge: A package for automatic evaluation of summaries,” in Text summarization branches out, 2004, pp. 74–81.