

Thesis submitted to the
UNIVERSITY OF MOHAMED BOUDIAF – MSILA



FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
DEPARTMENT OF COMPUTER SCIENCE

In partial fulfillment of the requirements for the degree of

Master in Computer science

Specialty: Artificial Intelligence

By

Allal Radhia

Title of the thesis

Arabic Text Summarization using Transformer Models

Under the supervision of

Dr: Rahima Bentercia

Composition of the jury

Dr. Saoudi Laliyah

University of Msila

President

Dr. Bentercia Rahima

University of Msila

Reporter

Dr. Brahimi Mahmoud

University of Msila

Examiner

Juin, 2026.

Abstract

This work presents an Arabic text summarization system based on Transformer models. The system was developed using Python, Streamlit, PyTorch, and the Hugging Face Transformers library, and supports abstractive, extractive, and hybrid summarization approaches.

The experimental evaluation was conducted using Arabic texts from the XL-Sum dataset and two Transformer-based models: mT5 and AraT5. The generated summaries were evaluated using ROUGE metrics and human assessment criteria. The results showed that both models were capable of producing coherent and informative summaries. However, mT5 achieved better overall performance in terms of ROUGE scores, while AraT5 generated summaries faster. These findings demonstrate the effectiveness of Transformer models for Arabic text summarization and highlight the trade-off between summary quality and computational efficiency.

Key words: Abstractive Summarization, Transformer models, mT5, AraT5, ROUGE, Arabic Text, NLP.

الملخص

يقدم هذا العمل نظامًا لتلخيص النصوص العربية يعتمد على نماذج المحولات (Transformer). تم تطوير النظام باستخدام لغات وأدوات: Python وStreamlit وPyTorch ومكتبة Hugging Face Transformers، وهو يدعم أساليب التلخيص التجريدي، الاستخراجي، والهجين.

أُجري التقييم التجريبي باستخدام نصوص عربية من مجموعة بيانات XL-Sum ونموذجين يعتمدان على المحولات وهما: mT5 وAraT5. وقُيِّمت الملخصات الناتجة باستخدام مقاييس ROUGE ومعايير التقييم البشري. وأظهرت النتائج أن كلا النموذجين قادران على إنتاج ملخصات متماسكة ومثيرة. ومع ذلك، حقق نموذج mT5 أداءً عامًا أفضل من حيث درجات ROUGE، بينما قام نموذج AraT5 بتوليد الملخصات بشكل أسرع. وتثبتت هذه النتائج فعالية نماذج المحولات في تلخيص النصوص العربية، وتسلبت الضوء على المقايضة بين جودة الملخص والكفاءة الحسابية.

الكلمات المفتاحية: لتلخيص التجريدي، نماذج المحولات، mT5، AraT5، ROUGE، النص العربي، معالجة اللغات الطبيعية.

Dedications

I dedicate this work to the one who raised me, struggled for me, and supported me throughout my journey — my beloved father, my strength and my inspiration.

To the one beneath whose feet Allah placed Paradise, my source of love, comfort, and strength — my dearest mother, guardian angel.

To my two beloved sisters, whose presence in my life is one of God's greatest blessings.

To my little brother, my support and my source of strength.

To all the members of my family and relatives, each one by name.

To all my friends, near and dear to my heart.

To all my teachers and professors who taught and guided me throughout my academic journey.

And finally, to everyone who holds a special place in my memory, even if my dissertation could not contain all their names.

Acknowledgments

First and foremost, all praise and gratitude are due to Allah Almighty, who granted me the strength, wisdom, patience, and perseverance to complete this work. To Him belongs all thanks for blessing me with the opportunity and success to achieve this accomplishment.

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Bentercia Rahima, for her valuable guidance, continuous support, insightful advice, and encouragement throughout this research journey.

My heartfelt thanks also go to the members of the examination committee for dedicating their time and effort to read, evaluate, and provide constructive feedback on this dissertation.

Finally, I would like to extend my gratitude to everyone who contributed, directly or indirectly, to the success and completion of this work.

Thank you all.

Table of Contents

List of figures	v
List of tables	vi
Introduction	1
Arabic Text Summarization	4
1.1. Introduction	4
1.2. Natural Language Processing (NLP)	4
1.2.1. Definition	4
1.2.2. Natural Language Processing Steps	4
1.3. Arabic Language Processing	5
1.3.1. Definition	5
1.3.2. Linguistic Characteristics of the Arabic Language	5
1.3.3. Preprocessing Techniques for Arabic Text	6
1.4. Text Summarization	7
1.4.1. Extractive Summarization	8
1.4.2. Abstractive Summarization	8
1.4.3. Hybrid Approaches	9
1.5. Challenges in Arabic Text Summarization	9
1.5.1. Rich Morphology	9
1.5.2. Limited Resources	10
1.5.3. Cross-domain Generalization and Topic Variations	10
1.5.4. Evaluation Metrics Complexity	10
1.6. Conclusion	10
Related Work	11
2.1. Introduction	11
2.2. Extractive Summarization Approaches	11
2.3. Abstractive Summarization Approaches	12
2.4. Hybrid Summarization Approaches	14
2.5. Conclusion	15
The Proposed Arabic Text Summarization System	16
3.1. Introduction	16
3.2. General Architecture of the Proposed System	16
3.3. Work Environment	17
3.3.1. Hardware	17
3.3.2. Software	17

3.4.	Dataset Description.....	19
3.4.1.	Selection of Dataset	20
3.4.2.	Dataset Conversion.....	20
3.5.	Pre-Processing.....	21
3.6.	The Proposed Models	21
3.6.1.	mT5 Multilingual-XLSum.....	21
3.6.2.	AraT5 Arabic Summarizer	22
3.7.	Inference and Generation Strategy	23
3.8.	Evaluation Metrics.....	24
3.9.	Graphical User Interface	24
3.9.1.	Single Text Summarization	25
3.9.2.	Model Comparison	26
3.9.3.	Batch CSV Evaluation	27
3.9.4.	Evaluation and Statistics	28
3.9.5.	Out-of-Domain Testing.....	29
3.9.6.	Experiment History.....	30
3.10.	Conclusion.....	31
	Experimental Results and Analysis	32
4.1.	Introduction	32
4.2.	Performance Evaluation	32
4.2.1.	ROUGE Metrics.....	32
4.2.2.	Compression Ratio	32
4.2.3.	Human Evaluation.....	33
4.3.	Results and Analysis	34
4.3.1.	mT5	34
4.3.2.	AraT5.....	36
4.4.	Comparative Analysis	37
4.5.	Qualitative Analysis.....	39
4.6.	Conclusion.....	40
	Conclusion	41
	References	43

List of figures

Figure 1-1: Types of text summarization.....	8
Figure 1-2: Extractive Summarization.....	8
Figure 1-3: Abstractive Summarization.....	9
Figure 1-4: Hybrid text summarization system	9
Figure 3-1: General Architecture of the Proposed System	16
Figure 3-2: Python Logo.....	18
Figure 3-3: Streamlit Logo	18
Figure 3-4: Pandas Logo.....	18
Figure 3-5: PyTorch Logo	19
Figure 3-6: Transformers Logo.....	19
Figure 3-7: PyCharm Logo	19
Figure 3-9: Graphical User Interface	25
Figure 3-10: Single Text Summarization.....	26
Figure 3-11: Model Comparison.....	27
Figure 3-12: Batch CSV Evaluation	28
Figure 3-13: Evaluation and Statistics	29
Figure 3-14: Out-of-Domain Testing.....	30
Figure 3-15: Experiment History	31
Figure 4-1: Human Evaluation	33
Figure 4-2: Qualitative summarization sample using mT5 in Abstractive Mode.....	34
Figure 4-3: Qualitative summarization sample using AraT5 in Abstractive Mode...	36
Figure 4-4: Syrian military article summaries with mT5 and AraT5	40

List of tables

Table 1-1: Structure of an Arabic word.....	5
Table 1-2: Segmentation of word (فسيأكلونها).....	6
Table 2-1: Sample of previous studies that belong to Extractive Summarization Approache	12
Table 2-2: Sample of previous studies that belong to Abstractive summarization Approach	14
Table 2-3: Sample of previous studies that belong to Hybrid summarization Approaches....	15
Table 3-1: Original XL-Sum Dataset Fields.....	20
Table 3-2: Dataset Format Conversion and File Roles.....	20
Table 3-3: Generation Parameters	24
Table 4-1: Preliminary Results for mT5 with Abstractive Mode	35
Table 4-2: Preliminary Results for mT5 with Hybrid mode.....	35
Table 4-3: Preliminary Results for AraT5 with Abstractive mode.....	36
Table 4-4: Preliminary Results for AraT5 with Hybrid mode.....	37
Table 4-5: Comparative Analysis of mT5 and AraT5 on 50 Articles	37
Table 4-6: Article "The Syrian Army Regaining Control of Al-Khalidiya"	39

Introduction

The rapid growth of digital information has created an increasing need for intelligent systems capable of processing and understanding large amounts of textual data efficiently. In this context, Artificial Intelligence (AI) has become one of the most influential fields in modern computing, offering advanced techniques that enable machines to analyze, learn from, and generate human language with remarkable accuracy.

Natural Language Processing (NLP) as a major branch of Artificial Intelligence, focuses on the interaction between computers and human language. With the continuous expansion of online content, the volume of Arabic textual data has grown significantly. This growth has highlighted the importance of developing automated tools that can help users access essential information quickly and effectively.

Among the most important NLP applications is text summarization, which aims to reduce long texts into shorter versions. Arabic text summarization represents a challenging task due to the linguistic characteristics of the Arabic language, including complex morphology, rich vocabulary, syntactic variations, and semantic ambiguity. Traditional approaches are unable to handle this complexity, creating a need for advanced models.

Recent advancements in deep learning and transformer-based architectures have greatly enhanced the performance of Natural Language Processing systems. Transformer models have shown remarkable effectiveness in capturing contextual and semantic information from Arabic texts despite the linguistic complexities of the language. In this context, the present research focuses on the development and evaluation of transformer-based models for Arabic text summarization.

Problem Statement

Despite recent advances in natural language processing, especially with the introduction of deep learning methods, the field of "Arabic text summarization" is still mostly unexplored. Abstractive text summarizing is more complicated than extractive summarization since it calls for the creation of original, cohesive phrases that convey the main ideas of the source material rather than just choosing preexisting sentences. This is especially difficult for Arabic literature because of the language's vast vocabulary, intricate morphology, and lack of high-quality, extensive summary datasets. In order to close this gap, the current study assesses transformer-

based models like mT5, in order to create an end-to-end system that can efficiently perform high-quality abstractive summarization on Arabic texts.

Motivation

While Arabic digital content is rapidly increasing, individuals and organizations face significant challenges in processing large volumes of text. Information overload and the inefficiency of manual reading make it difficult to extract relevant insights in a timely manner. Existing systems often fail to effectively summarize Arabic texts due to the complexity of the language and the limitations of current approaches. This circumstance inspired us to create a more sophisticated and precise text summarizing system, especially one that makes use of transformer-based models and deep learning.

Objective

The principal aim of this study is to develop and evaluate deep learning models for Arabic text summarization. The specific goals are outlined as follows:

- To design and implement an automatic system for Arabic text summarization capable of reducing large texts into concise and meaningful summaries.
- To investigate the challenges associated with the Arabic language in the context of text summarization.
- To apply and compare transformer-based models (such as AraT5 and mT5) in order to improve the quality and accuracy of generated summaries.
- To generate coherent, fluent, and semantically meaningful summaries that reflect the original content effectively.
- To evaluate the performance of the proposed models using appropriate evaluation metrics for summarization tasks, such as ROUGE scores, to measure content relevance and quality of the generated summaries.

Organization of The Manuscript

The organization of this manuscript is presented as follows:

Chapter1: Arabic Text Summarization

This chapter covers the theoretical aspects required by presenting typical NLP concepts, examining the linguistic characteristics of Arabic, discussing the techniques of text summarization, and the main challenges faced within Arabic text summarization.

Chapter2: Related Work

This chapter reviews the most relevant and recent studies and categorizes the latest methods into three main approaches: extractive, abstractive, and hybrid summarization approaches.

Chapter3: The Proposed Arabic Text Summarization System

This chapter outlines the study's chosen methodology, text summarizing system, tools, datasets, and execution strategy.

Chapter4: Experimental Results and Analysis

This chapter presents the results obtained using different transformer models, comparing them through different evaluation metrics.

Conclusion

This last section provides an overview of the research findings, the work's limitations, and potential future prospects for Arabic text summarization using transformer mode

Arabic Text Summarization

1.1. Introduction

Text summarization reduces the size of a document while preserving its essential information, which is crucial for handling large volumes of Arabic text data. This chapter presents the theoretical aspects necessary for a deep understanding of Arabic text summarization. It discusses the main approaches, including natural language processing, as well as the linguistic features of Arabic and the necessary preprocessing techniques. It also addresses the differences between extractive, abstractive, and hybrid summarization. Finally, it highlights the key challenges in summarizing Arabic text.

1.2. Natural Language Processing (NLP)

1.2.1. Definition

Natural Language Processing is a theoretically motivated range of computational techniques for analyzing and representing naturally occurring texts at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications [1].

1.2.2. Natural Language Processing Steps

1.2.2.1. Data Collection and Preprocessing

NLP starts with collecting text data from various sources. this data is then preprocessed through steps like tokenization, lowercasing, and stop-word removal, in order to clean and standardize the text for better analysis.

1.2.2.2. Feature Extraction and Representation

After preprocessing, the text is converted into numerical representations that machines can understand. Techniques such as Bag of Words, TF-IDF, and advanced embeddings like Word2Vec and Glove are used to capture both the importance and meaning of words.

1.2.2.3. Text Analysis and Understanding

At this stage, NLP techniques are applied to extract meaningful information from text. Tasks such as part-of-speech tagging, sentiment analysis, and topic modeling help in understanding the structure, context, and intent of the text.

1.2.2.4. Model Selection and Training

An appropriate model is selected depending on the task, and then trained using the processed data. The model learns patterns and relationships within the text while its performance is improved through parameter tuning.

1.2.2.5. Evaluation and Deployment

Finally, the model is evaluated using metrics like accuracy, precision, and recall to ensure its effectiveness. Once validated, it is deployed in real-world applications to analyze and process new text data efficiently.

1.3. Arabic Language Processing

1.3.1. Definition

Arabic is one of the five most widely spoken languages in the world, with over 300 million native speakers. Its complex morphological structure distinguishes it significantly from IndoEuropean languages like English. The Arabic alphabet consists of 28 letters: 3 long vowels and 25 consonants. Short vowels are represented by diacritical marks above or below the consonants, not as individual letters. Arabic has no capital or lowercase letters; all letters are the same size. Because Arabic texts lack long vowels, they can sometimes be difficult to understand due to lexical ambiguity. Arabic is written from right to left, and the shapes of the letters vary depending on their position in a word – whether at the beginning, middle, or end [2].

1.3.2. Linguistic Characteristics of the Arabic Language

1.3.2.1. Morphology

It means that how words are formed and their internal structure, this includes the analysis of the smallest units of meaning in a language known as "morphemes". And in Arabic language it is (root, prefixes, affixes, suffixes, patterns). The following representation schematizes a possible structure of a word.

Table 1-1: Structure of an Arabic word [3]

Post fixed	Suffix	Schematic body	Prefix	Antefix
------------	--------	----------------	--------	---------

- ❖ Post fixed are personal pronouns.
- ❖ Suffixes are the conjugation ending of verbs and masks which/plural/females for nouns including adverbals.

- ❖ Prefixes represented by a single letter, indicate the conjugating person of the verbs in present tense
- ❖ Antefixes are prepositions or conjunctions at the beginning of words [3].

Example: The word (فسيأكلونها)

In English it means: "and they will eat it" when analyzing and breaking down this word, it consists the following parts:

Table 1-2: Segmentation of word (فسيأكلونها)

Post fixed	Suffix	Schematic body	Prefix	Antefix
ها	ونـ	أكل	سيـ	فـ
Pronoun suffix complement	Verbal suffix expressing the plural	Derived from the root	Verbal prefix expressing the future	Conjunction

1.3.2.2. Diacritics

Diacritics (Tashkeel) are marks added to letters that have phonetic or semantic value, so we can think of them as vowels in English. In Arabic, diacritics (Harakat) are:

- Fatha (َ): a small diagonal line above a letter, representing a short "a" sound.
- Kasra (ِ): a small diagonal line below a letter, representing a short "i" sound.
- Damma (ُ): a small comma-like hook above a letter, representing a short "u" sound.
- Sukun (◌ْ): a small circle above a letter, indicating the absence of a vowel (a pause).

1.3.3. Preprocessing Techniques for Arabic Text

Preprocessing text is important step in NLP tasks because it allows for the cleaning and transformation of raw text data into a format that is suitable for analysis in Arabic datasets. The Arabic language is unique a Semitic language, possesses definite characteristics and challenges that need specific preprocessing techniques. These challenges including processing diacritics, spellings, abbreviations, and dialectal variations [4].

1.3.3.1. Normalization

Normalization is the process of converting text into a standard and consistent format, and it involves simplifying variations of letters and symbols in order to alleviate sparsity in your data. Typical procedures include the following:

- ❖ Remove non-letters and special characters, such as \$, &, and %.
- ❖ Remove non-Arabic letters.
- ❖ Replace initial ("إ", "آ" or "أ") with bar alif ("ا").
- ❖ Replace Ta'a marbota ("ة") with Ha'a ("ه").
- ❖ Remove the ("ال") from the beginning of the word.
- ❖ Replace the final ("ى") with ("ا").
- ❖ Remove the diacritics [5].

1.3.3.2. Tokenization

Tokenization is an essential preprocessing step for any language because it is the first step that splits the long text into a small part called a "token", making it easy to use in text analysis algorithms. These words were then separated into numerous delimiters, including white spaces, tabs, and punctuation marks. Moreover, a sub-word tokenizer allows splitting a dataset into units called "sub-words" [5].

1.3.3.3. Stop Word Removal

Removing common words is essential for the preprocessing of all languages because it eliminates unimportant words with very limited semantic meaning, thus reducing the text dimensions used in the parser. Common words include prepositions, pronouns, and articles (such as "من", "على", "في") that appear frequently and do not help in distinguishing documents [5].

1.3.3.4. Stemming and lemmatization

Stemming refers to the reduction of words to their stems or roots that are used to fit different word variants while text lemmatization seeks to regroup semantically associated words. This is to reduce the size of the vocabulary [5].

1.4. Text Summarization

Text summarization is a NLP task that involves breaking down lengthy texts into shorter versions while retaining and valuable information and preserving the meaning of the text. Text summarization is particularly useful in comprehending lengthy textual documents, and it also helps to reduce computational resources and time. There are 3 main types for summarizing text: extractive, abstractive, and hybrid. Each approach use different techniques:

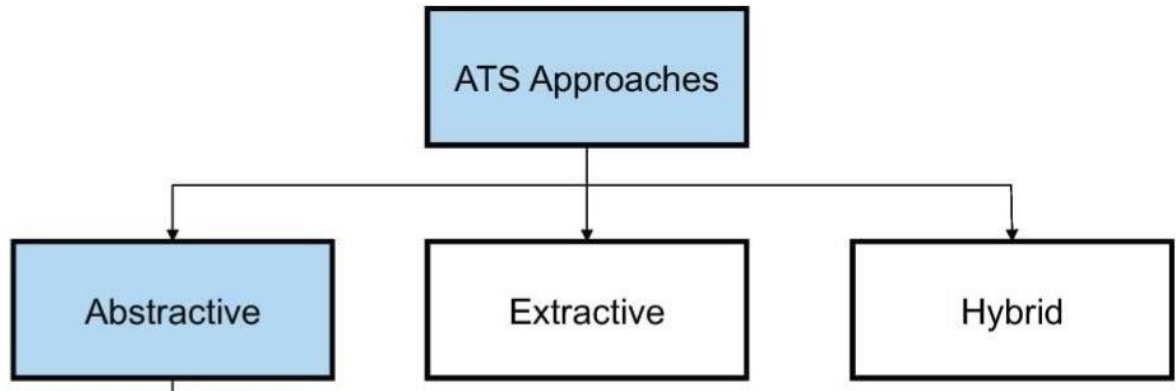


Figure 1-1: Types of text summarization

1.4.1. Extractive Summarization

Extractive summarization selects the most relevant sentences directly from the original text without modification. The main distinction between extractive approaches lies in how they assess sentence importance while minimizing redundancy across topics. These scoring strategies ultimately determine which sentences are selected and retained in the final summary.

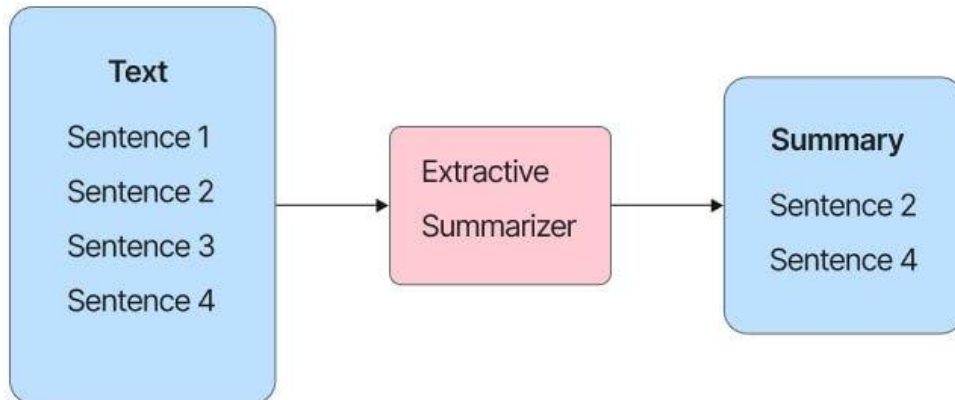


Figure 1-2: Extractive Summarization

1.4.2. Abstractive Summarization

Abstractive summarization produces summaries using sentences not found in the original texts that express their core meaning. It requires a deep understanding of the document's main ideas and the ability to paraphrase them in a clear and concise manner. Models like AraT5 and Pegasus are popular here.

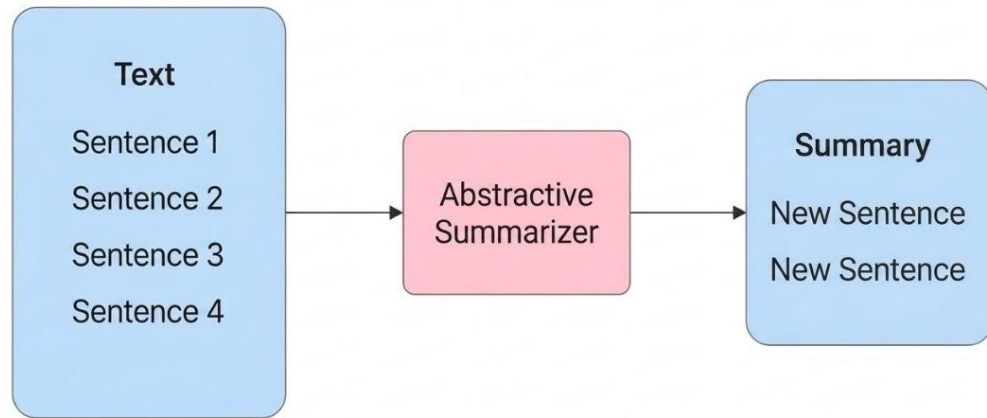


Figure 1-3: Abstractive Summarization

1.4.3. Hybrid Approaches

Hybrid methods combining integrates extractive and abstractive approaches by selecting important content from the source text and then refining it through paraphrasing to produce coherent and concise summaries.

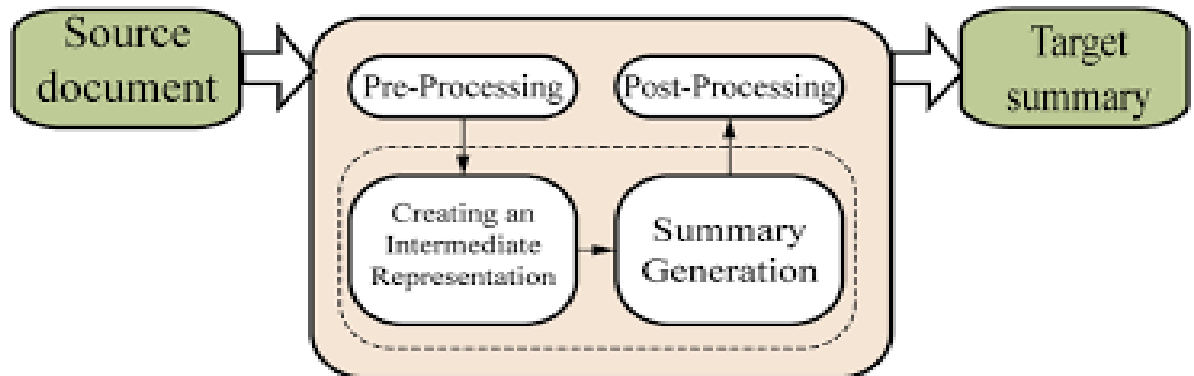


Figure 1-4: Hybrid text summarization system

1.5. Challenges in Arabic Text Summarization

Arabic text summarization presents a number of difficulties and challenges because of the peculiarities of the Arabic language and the complexities of summarizing Arabic text. Some of them can be listed as follows:

1.5.1. Rich Morphology

Arabic has a rich morphological structure, with extensive use of prefixes, suffixes, and root-based word formation. This complexity adds challenges to the identification and extraction of important content for summarization [6].

1.5.2. Limited Resources

Compared to some other languages, there is a relatively limited availability of high-quality Arabic summarization datasets, linguistic resources, and tools. The scarcity of resources makes it challenging to train and evaluate robust summarization models [6].

1.5.3. Cross-domain Generalization and Topic Variations

Arabic text summarization models often struggle to generalize across different domains, as they must handle a wide range of topics and varying levels of text complexity, such as news, social media, legal, and scientific texts [6].

1.5.4. Evaluation Metrics Complexity

Evaluating Arabic text summarization remains challenging due to the absence of widely accepted metrics specifically designed for Arabic. Common evaluation methods, such as ROUGE, often fail to fully capture the linguistic richness and nuances of the Arabic language [6].

1.6. Conclusion

In summary, this chapter provides the theoretical background for summarizing Arabic text. It presents the main approaches, including the linguistic characteristics of Arabic along with the necessary preprocessing steps, and highlights both extractive and abstract methods. It also discusses key challenges, such as the richness of morphology, limited resources, and evaluation metrics. These insights provide the foundation for the methodology that will be developed in subsequent chapters on Arabic text summarization.

Related Work

2.1. Introduction

With the rapid growth of Arabic digital content in recent years, there has been an increasing need for automated systems that can efficiently process and summarize large volumes of text. Arabic text summarization has become an important research area, as it helps reduce information overload. This chapter reviews the most relevant and recent studies in the field of text summarization on Arabic language. It covers different approaches, including Extractive, Abstractive, and Hybrid summarization techniques. Each study is presented by describing the proposed approach, the dataset used, and the evaluation metrics applied.

2.2. Extractive Summarization Approaches

The authors in [7] 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 method is evaluated using the EASC dataset and the ROUGE-2 measure. The F-score of score-based approach was 0.617 and an F-score of machine learning approach was 0.524. However, in the summary the model only chooses and incorporates sentences from the original content. If abstractive summarizing is needed, its inability to provide abstractive summaries could be a drawback. The right features must be chosen and developed for the model to be successful.

The research in [8] presented a system for extractive summarization. The model is based on text graphing, where the original text is transformed into a graph using an innovative formula. This formula considers factors such as sentence relevance, coverage, and variety to evaluate each sentence in the text. Statistical and semantic criteria are used to determine the significance of each sentence. A subgraph is then generated to reduce the size of the original text. The system achieved an F-score of 0.617 using the ROUGE-2. However, extractive summarization in which the summary is made up of chosen sentences from the original text seems to be the main

focus of the approach presented. Extractive summarization may not be able to produce summaries that go beyond the original text's substance, even though it can be useful in catching crucial information.

Table 2-1: Sample of previous studies that belong to Extractive Summarization Approaches [9]

Model	Summary Approach	Methodology	Dataset	ROUGE		
				1	2	L
Machine learning-Model [7]	Extractive summary	Score-Based Algorithm. Machine learning-based method.	Essex Arabic Summaries Corpus (EASC)	0.643	0.617	—
Graph-Based Approach [8]	Extractive summary	Graph-Based Text Summarization.	Essex Arabic Summaries Corpus (EASC)	—	0.617	—

2.3. Abstractive Summarization Approaches

In [10], authors 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. The models perform well particularly on the XL-Sum dataset, the F-score for ROUGE-1 was 34.5, for ROUGE-2 was 14.6, and for ROUGE-L was 30.5. With 139 million parameters, AraBART's large size makes it computationally expensive to train and deploy, which may disadvantage users with limited hardware resources.

The research described in [11] 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. Two distinct datasets are used by the authors 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, and for ROUGE-L it was 34.37. The F-score obtained in dataset 2 for ROUGE-1 was 44.28, for ROUGE-2 it was 18.35, and for ROUGE-L it was 32.46. However, Arabic possesses an exceptionally rich vocabulary,

meaning that seq2seq models often struggle with out-of-vocabulary words that were absent during the training phase.

The study in [12] introduced a model for abstractive Arabic text summarization that utilizes sequence-to-sequence recurrent neural networks. The encoder is composed of multiple layers 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. But the proposed model remains susceptible to overfitting. Deep neural networks, particularly those with extensive layers and parameters, are highly prone to memorizing the training data. This compromises generalization, causing the model to perform exceptionally well on training sets but poorly on unseen data.

SemG-TS, a novel abstractive text summary method for Arabic, was presented in [13]. It is based on semantic graph embeddings and uses a deep neural network to produce abstractive summaries. The report offers a thorough assessment of SemG-TS, utilizing both automatic and human evaluation approaches to compare its performance to word2vec, a widely used baseline technique. The suggested SemG-TS method uses the special qualities of the Arabic language to first describe the original text as a semantic graph in order to provide abstractive summaries. The AlJazeera.net website was used to gather articles for the experiment. There were 16,770 paragraphs in the dataset, with an average of 204 words per paragraph. SemG-TS's performance was automatically compared to two word2vec versions one trained on the dataset and the other using random initial vectors using the ROUGE evaluation metric. SemG-TS exceeded both word2vec versions in every evaluation metric. In terms of ROUGE scores and overall text quality in terms of relevance, readability, and satisfaction, the SemG-TS model outperforms word2vec. ROUGE's F-score was 0.047. However, semantic graph embeddings are inherently complex and often lack the interpretability of standard word embeddings. Consequently, analyzing and fine-tuning the embedding process can pose a significant challenge.

Table 2-2: Sample of previous studies that belong to Abstractive summarization Approaches [9]

Model	Summary Approach	Methodology	Dataset	ROUGE		
				1	2	L
Transformer Model [10]	Abstractive summary	AraBART Model.	Arabic Gigaword XL-Sum	42.4	28.8	40.3
Seq2Seq Model [11]	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
Seq2Seq Model [12]	Abstractive summary	Seq2Seq-BiLSTM model with global attention	SANAD_ SUBSET and other resources	38.4	—	—
Seq2Seq Model [13]	Abstractive summary	Seq2Seq-BiLSTM model with global attention using SemanticGraph2vec embedding.	8385documents collected from the AlJazeera. net website and CNN-Arabic news.	0.047	—	—

2.4. Hybrid Summarization Approaches

The work presented in [14] proposed a hybrid approach for Arabic summarization 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. They also present the HASD, a new Arabic benchmark dataset, and modify the widely-used extractive EASC benchmarks by adding abstractive summaries to each text in the EASC dataset. Additionally, they suggest a brand-new assessment metric known as the Arabic-ROUGE measure, which assesses abstractive summaries' quality according to their word similarity and structure. 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, 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, and for Arabic Rouge was 0.713. But integrating a Modified Sequence-to-Sequence (MSTS) model with a

transformer-based architecture inherently increases model complexity. This heightened complexity often translates to prolonged training durations and elevated computational resource requirements.

Table 2-3: Sample of previous studies that belong to Hybrid summarization Approaches [9]

Model	Summary Approach	Methodology	Dataset	ROUGE		
				1	2	L
Seq2Seq and Transformer Model [14]	Hybrid summary	Seq2Seq-BiLSTM model with global attention integrated with mT5.	Hybrid Arabic text summarization dataset (HASD) which contain extractive and abstractive summary	EASC: 0.587	EASC: 0.48	EASC: 0.56
			Essex Arabic Summaries Corpus (EASC)	HASD: 0.6374	HASD: 0.4908	HASD: 0.6047

2.5. Conclusion

In summary, the reviewed studies show that Arabic text summarization has evolved from traditional extractive methods, such as score-based algorithms and graph-based techniques, to more advanced abstractive approaches based on Seq2Seq architectures, BiLSTM-GRU networks, attention mechanisms, and transformer models like AraBART. In addition, hybrid approaches combining extractive and abstractive techniques demonstrated promising performance by benefiting from both methods. The studies also highlighted the importance of preprocessing and dataset quality in improving summarization results. Overall, transformer-based and hybrid models appear to be the most effective approaches for Arabic text summarization. Therefore, the next chapter presents the methodology and implementation adopted in this work.

The Proposed Arabic Text Summarization System

3.1. Introduction

This chapter presents an Arabic text summarization system based on advanced transformer-based models, designed to generate concise and meaningful summaries. The chapter describes the methodology, tools, and the technical framework employed in the development of the system.

3.2. General Architecture of the Proposed System

The proposed system follows a modular pipeline that begins with Arabic text input and ends with summary generation and evaluation. The general workflow of the system is shown in Figure 3-1.

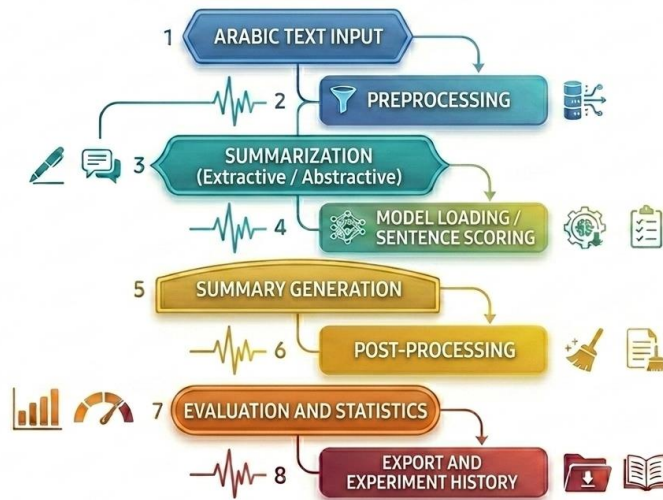


Figure 3-1: General Architecture of the Proposed System

The system receives Arabic text either manually from the user or through a CSV dataset. The input text is cleaned using light Arabic preprocessing steps, then the user selects the summarization mode. In the abstractive mode, the text is passed to a transformer encoder-decoder model. In the extractive mode, the system scores sentence and selects the most relevant ones. In the hybrid mode, the system first

extracts important sentences and then sends them to a transformer model for abstractive rephrasing.

The generated summary is then evaluated using word statistics, compression ratio, ROUGE metrics, and optional human evaluation criteria. The results can be saved and exported in CSV format.

3.3. Work Environment

3.3.1. Hardware

In this research, the development and training phases of the models were conducted locally within a dedicated computing environment utilizing the PyCharm IDE. The hardware configuration used to support the computational demands of the experiments, including data preprocessing, tokenization, and model evaluation, consisted of the following local resources:

- ❖ **Processor (CPU):** Intel Core i5 processor, which handled the execution of the training scripts and text processing pipelines.
- ❖ **Memory (RAM):** 8 GB of RAM, utilized for managing dataset loading, batching, and memory allocation during the experimentation process.

3.3.2. Software

Furthermore, a suite of software and tools was employed to facilitate system development and training.

3.3.2.1. Programming Language

- ❖ **Python:** Python is a high-level, object-oriented, interpreted programming language with dynamic semantics. It is highly appealing for Rapid Application Development and for usage as a scripting or glue language to join existing components together because of its high-level built-in data structures, dynamic typing, and dynamic binding. Python's easy-to-learn syntax prioritizes readability, which lowers software maintenance costs. Python promotes code reuse and software modularity by supporting modules and packages. For all major platforms, the comprehensive standard library and the Python interpreter are freely distributable and available in source or binary form [15].



Figure 3-2: Python Logo

3.3.2.2. Libraries and Frameworks

- ❖ **Streamlit:** streamlit is an open-source Python framework for data scientists and AI/ML engineers to deliver dynamic data apps with only a few lines of code [16]. In short streamlit is a Python library that converts scripts into web applications.



Figure 3-3: Streamlit Logo

- ❖ **Pandas:** pandas is an open-source Python package for cleaning, analyzing, and manipulating data. Like spreadsheets or SQL tables, it offers quick and adaptable tools for working with tabular data [17].



Figure 3-4: Pandas Logo

- ❖ **PyTorch:** PyTorch is a Python-based deep learning library that runs on CPU by default and supports GPU acceleration using CUDA. It follows a define by run approach, creating dynamic computation graphs during execution, which makes debugging and customization easier [18].



Figure 3-5: PyTorch Logo

- ❖ **Transformers:** Hugging Face's Transformers library is a well-known open-source Python framework that offers thousands of machine learning models that have already been trained. It makes it simple for developers to integrate AI features like computer vision, audio processing, and natural language processing by abstracting away the complexity of training and implementing models [19].



Figure 3-6: Transformers Logo

3.3.2.3. Development Environments

- ❖ **PyCharm:** PyCharm is a dedicated Integrated Development Environment (IDE) used for programming in Python. Developed by JetBrains, it provides a comprehensive suite of tools to write, debug, and manage Python code including smart code completion, syntax highlighting, version control integration, and built-in support for web development and data science [20].



Figure 3-7: PyCharm Logo

3.4. Dataset Description

3.4.1. Selection of Dataset

The dataset used in this project is derived from the Arabic XL-Sum benchmark, which is a comprehensive and diverse dataset comprising 1.35 million professionally annotated article-summary pairs from BBC. The dataset covers 45 languages ranging from low to high-resource, for many of which no public dataset is currently available. XL-Sum is highly abstractive, concise, and of high quality, as indicated by human and intrinsic evaluation [21]. It is considered suitable for abstractive summarization because the summaries are concise and human-written, making them appropriate as reference summaries for evaluation.

In this project, the Arabic XL-Sum data was selected because it provides real Arabic news articles together with reference summaries. This structure is directly compatible with the objective of the project, which is to evaluate Arabic text summarization using transformer-based models.

The original dataset records contain several fields, as illustrated in Table 3-1:

Table 3-1: Original XL-Sum Dataset Fields

Field	Description
Id	Unique identifier of the article
Url	Original BBC article URL
Title	Article title
Summary	Human-written reference summary
Text	Full article text

3.4.2. Dataset Conversion

The original dataset was provided in JSONL format. Since the implemented application supports batch evaluation through CSV files, the JSONL files were converted into structured CSV format using a Python script.

The original files were organized as follows:

Table 3-2: Dataset Format Conversion and File Roles

File	Role	Original format	Converted format
arabic_train.jsonl	Training data	JSONL	CSV
arabic_val.jsonl	Validation data	JSONL	CSV
arabic_test.jsonl	Testing data	JSONL	CSV

During the conversion process, the text field was mapped to the article column, while the summary field was mapped to the reference_summary column. Additional

metadata such as id, title, url, and a fixed domain label were preserved to enrich the dataset.

This conversion made the dataset directly usable in the batch evaluation module of the application.

3.5. Pre-Processing

In NLP tasks, data preprocessing plays a crucial role in improving the quality of textual data and enhancing model performance. In this project, preprocessing was applied to Arabic texts to prepare them for the summarization process. Data cleaning refers to the process of identifying and correcting or removing inaccurate, noisy, irrelevant data, improperly formatted, or redundant data (as discussed in chapter 1). However, transformer models usually rely on their own tokenizers; therefore, excessive preprocessing may remove useful information from the input text.

For this reason, the system applies light and optional preprocessing rather than aggressive normalization. The preprocessing options implemented in the application include:

- ❖ Remove URLs.
- ❖ Remove the diacritics.
- ❖ Normalize Alef forms.
- ❖ Normalize Ya.
- ❖ Punctuation cleaning.
- ❖ Remove Tatweel.
- ❖ Unnecessary spaces removal.

3.6. The Proposed Models

In this section, we present our research and the models employed in this study. We implemented two transformer-based models: mT5 and AraT5, each model and its main characteristics will be discussed in detail in the following subsections:

3.6.1. mT5 Multilingual-XLSum

mT5 (Massively Multilingual Text-to-Text Transfer Transformer) is a multilingual variant of Google's T5 model. By reframing all natural language

processing tasks into a unified "text-to-text" format, it can understand, translate, and generate text across 101 languages using a single pre-trained model [22].

The implemented model is: csebuetnlp/mT5_multilingual_XLSum, a multilingual Transformer model fine-tuned on the XL-Sum dataset for abstractive summarization.

The model contains the following components:

- **mT5 Tokenizer** for multilingual text tokenization.
- **Transformer Encoder** for extracting contextual representations from the input text.
- **Transformer Decoder** for generating the summary sequence.
- **Generation Module** using beam search and decoding constraints.
- **Long Text Chunking Module** integrated into the application to process texts exceeding the model input limit.

During inference, the input text is tokenized, padded, and truncated according to the selected maximum input length. For long documents, the system automatically splits the text into smaller chunks, summarizes each chunk independently, and then generates a final summary from the combined intermediate summaries. The model produces the final summary using beam search, repetition penalty, and n-gram repetition control mechanisms.

The mT5 model was selected because it showed stable behavior with Arabic news texts and was directly compatible with the Arabic XL-Sum dataset used in the project.

3.6.2. AraT5 Arabic Summarizer

The T5 (Text-to-Text Transfer Transformer) 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. [23]

In this project, the implemented Arabic-specific model is: malmarjeh/t5-arabic-text-summarization. This model is based on the T5 encoder–decoder Transformer architecture and was fine-tuned for Arabic abstractive text summarization using a dataset containing approximately 84,764 Arabic paragraph-summary pairs.

The following main components are included in the model:

- **Tokenizer:** Converts Arabic text into subword tokens compatible with the model vocabulary.
- **Encoder:** Processes the input sequence and generates contextual representations of the text.
- **Decoder:** Produces the summary token by token based on the encoder outputs.
- **AraBERT Preprocessor:** Applies Arabic-specific preprocessing operations before tokenization to improve text consistency and model performance.
- **Generation Module:** Controls summary generation using decoding parameters such as beam search, repetition penalty, and n-gram repetition constraints.

During inference, the input Arabic text is first preprocessed using the AraBERT preprocessor. The cleaned text is then tokenized and converted into numerical representations. These representations are passed through the encoder-decoder Transformer architecture, where the encoder captures the semantic information of the source text and the decoder generates a condensed summary. The generation process is guided by configurable parameters including maximum summary length, beam search width, repetition penalty, and no-repeat n-gram size to improve summary quality and reduce redundancy.

The AraT5 model was selected because it is specifically optimized for Arabic language processing and demonstrated strong performance in generating fluent and coherent Arabic summaries within the proposed system.

3.7. Inference and Generation Strategy

The main application does not train transformer models directly inside the graphical interface. Instead, it uses pre-trained or previously fine-tuned models for inference. This

design was chosen because full fine-tuning requires considerable computational resources, especially GPU memory and long training time.

The implemented strategy focuses on:

- Loading pre-trained/fine-tuned transformer models.
- Tokenizing Arabic input texts.
- Generating summaries using controlled decoding.
- Evaluating generated summaries.
- Comparing different summarization modes and models.

The generation parameters used in the application are summarized in Table 3-3.

Table 3-3: Generation Parameters

Parameter	Default / Used Value	Role
Max_input_length	512 tokens	Maximum input length
Max_summary_length	Depends on short/medium/long	Maximum generated summary length
Min_summary_length	Depends on short/medium/long	Minimum generated summary length
Num_beams	4	Beam search decoding
Repetition_penalty	3.0	Reduces repetitive output
No_repeat_ngram_size	3	Prevents repeated n-grams
Chunk_words	400 words	Splits long texts into chunks

The application allows the user to change these parameters from the sidebar. For weak devices, smaller values such as max_input_length = 256 and num_beams = 1 can be used to reduce computational cost.

3.8. Evaluation Metrics

The evaluation strategy of the proposed system combines automatic and manual evaluation. Automatic evaluation is mainly based on ROUGE, while manual evaluation is used to assess qualitative aspects such as fluency and faithfulness.

- ❖ ROUGE Metrics: The custom function "compute_rouge_scores" calculates: ROUGE-1, ROUGE-2, and ROUGE-L.
- ❖ Compression Ratio.
- ❖ Human Evaluation.

3.9. Graphical User Interface

The graphical user interface was developed using Streamlit. It was designed in Arabic with right-to-left layout support in order to make the application more accessible for Arabic users.

The interface contains several tabs, each corresponding to a specific function of the system, as shown in the Figure 3-8.



Figure 3-8: Graphical User Interface

3.9.1. Single Text Summarization

This tab allows the user to input an Arabic text manually or upload a TXT file. The user can choose the model, summarization mode, summary length, and generation parameters. After clicking the summarization button, the generated summary is displayed with statistics such as word count, compression ratio, generation time, and number of chunks.

A copy button is also provided to allow the user to copy the generated summary directly.



Figure 3-9:Single Text Summarization

3.9.2. Model Comparison

The model comparison tab allows the user to run the same input text through multiple models, such as mT5 and AraT5. The system displays the generated summaries and compares them according to:

- Number of words in the summary.
- Compression ratio.
- Generation time.
- ROUGE scores, when a reference summary is provided.

This module helps evaluate the differences between multilingual and Arabic-specific transformer models.



Figure 3-10: Model Comparison

3.9.3. Batch CSV Evaluation

The batch evaluation tab allows the user to upload a CSV dataset containing multiple Arabic articles and reference summaries. The required columns are:

- article
- reference_summary

For each article, the system generates a summary, compares it with the reference summary, calculates ROUGE scores, and stores the results in a table. The final results can be exported as a CSV file.

This module is particularly useful for experimental evaluation because it allows the system to be tested on several articles instead of a single example.

تقييم جماعي باستخدام CSV

رفع ملف CSV على عمود للأصلي، ويفضل أن يحتوي على عمود للتلخيص المرجعي لحساب ROUGE. مثال: article, reference_summary

رفع ملف CSV: xsum_a_or_app.csv 2.1MB

معالجة البيانات:

	id	title	article	reference_summary
0	130806_nidhal_hassan_trial		تدخل حسن وناظر طالبان في	مسكري الأمريكي، تدخل حسن، الذي يدافع عن نفسه، بقتل الجور، محتججا بجماعية المسلمين وتجاوز طالبان في
1	160129_germany_asylum_seekers		الأمم المتحدة، وتلقه في ظل مساعيها للحد	الأمم المتحدة، وتلقه في ظل مساعيها للحد
2	130729_syria_homs_area_retaken		بذرة التي كان أحد المواقع الحصينة	بذرة التي كان أحد المواقع الحصينة
3	140517_arsenal_fa_cup_winners		ين لاشي إلى فوز بثلاثة أهداف	ين لاشي إلى فوز بثلاثة أهداف
4	140722_iraq_minorities		زاد، لكن هذا البلد يضم عددا من الإ	زاد، لكن هذا البلد يضم عددا من الإ

أكثر عمود النص الأصلي: article

Figure 3-11: Batch CSV Evaluation

3.9.4. Evaluation and Statistics

This tab displays detailed statistics for the latest generated summary. It includes:

- Number of words in the original text.
- Number of words in the generated summary.
- Number of sentences.
- Compression ratio.
- ROUGE scores.
- Human evaluation scores.

The tab also provides a simple visualization comparing the length of the original text and the generated summary.

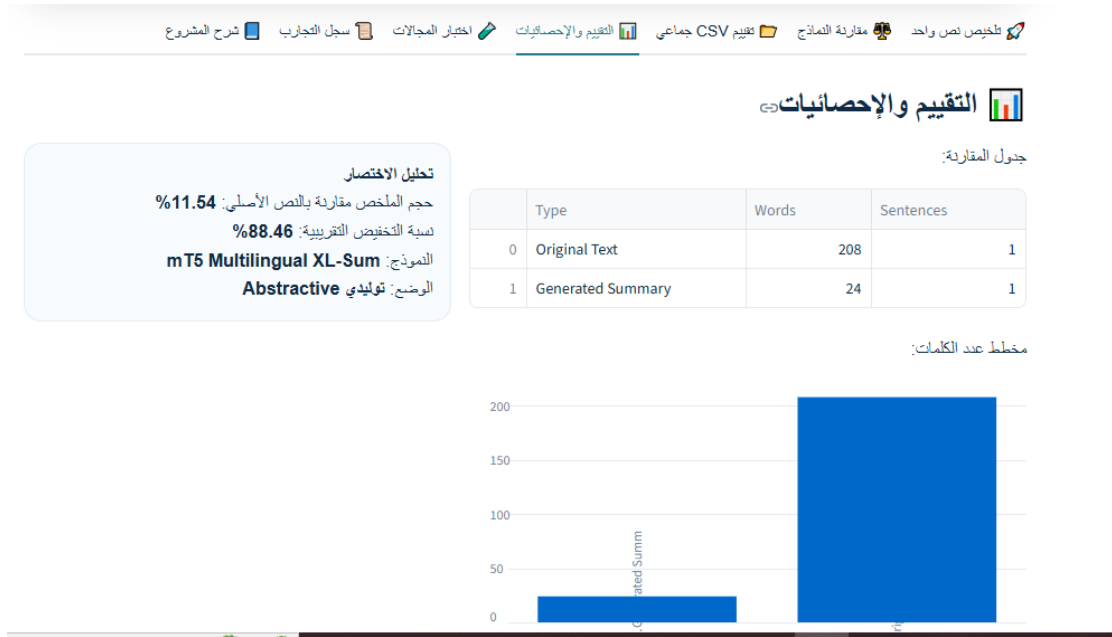


Figure 3-12: Evaluation and Statistics

3.9.5. Out-of-Domain Testing

The out-of-domain testing tab was added to examine whether the model performs well outside the news domain. The system includes example texts from different domains, such as:

- News texts.
- Academic texts.
- Educational texts.
- Scientific texts.
- Short social-media-like posts.

This module is important because models trained or fine-tuned on news data may not perform equally well on academic or specialized texts.



Figure 3-13: Out-of-Domain Testing

3.9.6. Experiment History

The experiment history tab records the summaries generated during the session. It stores information such as:

- Date and time.
- Model name.
- Summarization mode.
- Number of original words.
- Number of summary words.
- Compression ratio.
- Generation time.
- ROUGE scores.
- Generated summary.

The user can export the experiment history as a CSV file for later analysis and inclusion in the research report.



Figure 3-14: Experiment History

3.10. Conclusion

This chapter presented the proposed Arabic text summarization system based on transformer models. The system was implemented as a graphical application using Streamlit and integrates mT5 and AraT5 models for abstractive summarization. It also supports extractive and hybrid summarization modes, batch CSV evaluation, Arabic-adapted ROUGE metrics, human evaluation, experiment history, and out-of-domain testing. Overall, the proposed application provides a practical and extensible framework for Arabic text summarization using transformer-based models. The next chapter, Chapter 4, will discuss the experimental results and evaluate the performance of the proposed summarization models.

Experimental Results and Analysis

4.1. Introduction

This chapter details the experimental evaluation conducted on an Arabic text summarization system using two Transformer-based models, mT5 and AraT5. The evaluation aims to measure the ability of these models to generate accurate and coherent summaries from Arabic texts extracted from the XLSum dataset. The performance analysis is based on ROUGE evaluation metrics and generation time measurements. In addition, a comparative study is carried out to identify the relative strengths and weaknesses of each model. The results are presented through structured tables and quantitative metrics, providing a comprehensive assessment of Transformer-based approaches for Arabic text summarization.

4.2. Performance Evaluation

4.2.1. ROUGE Metrics

ROUGE, which stands for Recall-Oriented Understudy for Gisting Evaluation, is one of the most widely used automatic metrics for summarization evaluation. It compares a generated summary with one or more human-written reference summaries by measuring overlapping units such as words, n-grams, and word sequences.

In this project, three ROUGE variants were used:

- ROUGE-1: measures the overlap of individual words (unigrams) between an AI-generated text and a human-created reference text.
- ROUGE-2: measures the overlap of bigrams (two-word sequences) between a machine-generated text and a reference text
- ROUGE-L: is based on the longest common subsequence (LCS) between our model output and reference. the longest sequence of words that is shared between both. A longer shared sequence should indicate more similarity between the two sequences [24].

4.2.2. Compression Ratio

In addition to ROUGE, the system calculates the compression ratio, which measures the relative length of the generated summary compared with the original article.

The compression ratio is calculated as follows:

$$\text{Compression Ratio} = (\text{Summary Words} / \text{Original Text Words}) \times 100$$

A low compression ratio indicates that the generated summary is very short compared with the original document. However, an extremely low value may also indicate that the summary omits important information.

4.2.3. Human Evaluation

Because ROUGE mainly measures lexical overlap and does not fully capture semantic correctness, the system includes a human evaluation module. The user can evaluate the generated summary according to five criteria:

- Coherence: Logical organization of the summary
- Fluency: Linguistic quality and readability
- Relevance: Coverage of important information
- Non-redundancy: Absence of unnecessary repetition
- Faithfulness: Preservation of the original meaning

Each criterion is scored from 1 to 5. The system then calculates the average human evaluation score, as shown in the Figure 4-1.

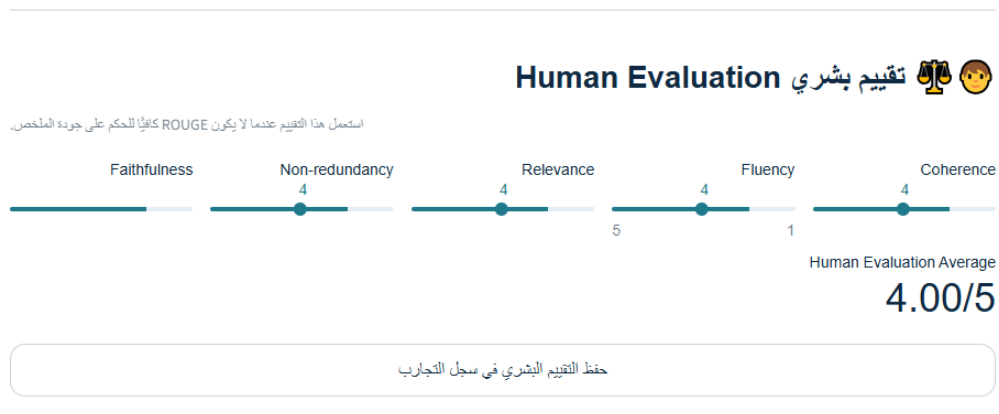


Figure 4-1: Human Evaluation

4.3. Results and Analysis

The performance of this Arabic text summarization system was examined by running the transformer-based models (mT5 Multilingual XL-Sum and AraT5 Arabic Summarizer) on a subset extracted from the XL-Sum Arabic dataset, which contains human-annotated Arabic articles across various news categories. The whole analysis is based on test metrics computed across different system modes (Abstractive and Hybrid). The results are presented in tabular form for further clarity, with a detailed interpretation of performance.

4.3.1. mT5

❖ Abstractive Mode

Initially, a preliminary qualitative analysis was conducted by testing the model on a single Arabic article. This initial step aimed to observe how the model processes the text and generates the summary, as displayed in the Figure 4-2. This allowed for an immediate evaluation of the system's interface and the model's textual output.



Figure 4-2: Qualitative summarization sample using mT5 in Abstractive Mode

Following this single-text trial, the quantitative experiment was expanded to include a dataset sample of five Arabic news articles from the Arabic XL-Sum test set. The main objective of this step was to examine the general performance metrics and evaluate the generated results. The model used in this phase was mT5 Multilingual XL-Sum, operating under the abstractive summarization mode.

The obtained results are presented in Table 4-1.

Table 4-1: Preliminary Results for mT5 with Abstractive Mode

Model	Mode	Articles	ROUGE-1	ROUGE-2	ROUGE-L	Compression Ratio	Avg. Time
mT5 Multilingual XL-Sum	Abstractive	5	0.2124	0.0433	0.1419	8,28%	52.2 s

The results show that the model was able to generate concise Arabic summaries. The average ROUGE-1 score of 0.2124 indicates partial lexical overlap between the generated summaries and the reference summaries. The lower ROUGE-2 score suggests that the model often uses different phrasing from the human summaries. The ROUGE-L score also indicates limited structural similarity between the generated and reference summaries.

The compression ratio of 8.28% shows that the generated summaries were very short compared with the original articles. This strong compression can be useful for quickly extracting the main idea, but it may also lead to the loss of important details.

❖ Hybrid Mode

The same number of articles from the dataset was used; however, the summarization approach was changed to hybrid summarization.

The obtained outcomes are displayed in Table 4-2.

Table 4-2: Preliminary Results for mT5 with Hybrid mode

Model	Mode	Articles	ROUGE-1	ROUGE-2	ROUGE-L	Compression Ratio	Avg. Time
mT5 Multilingual XL-Sum	Hybrid	5	0.2124	0.0433	0.1419	8.28%	31.1 s

The summarization results remained identical in terms of ROUGE scores. However, a noticeable improvement was observed in the generation time, which decreased from approximately 52.2 seconds to 31.1 seconds. This indicates that the optimization mainly affected the computational efficiency of the system without altering the quality of the generated summaries.

4.3.2. AraT5

Similarly, the same experimental procedure was applied to the AraT5 model to evaluate its performance across both abstractive and hybrid summarization modes.

❖ Abstractive Mode

First, the same single Arabic news article underwent a qualitative evaluation. This approach enabled a head-to-head visual and textual comparison between AraT5 and mT5, highlighting how AraT5 processes input text and generates sentences compared to mT5, as illustrated in Figure 4-3.



Figure 4-3: Qualitative summarization sample using AraT5 in Abstractive Mode

Subsequently, the quantitative evaluation was conducted using the same sample of five Arabic news articles with AraT5 models.

The results are shown in Table 4-3.

Table 4-3: Preliminary Results for AraT5 with Abstractive mode

Model	Mode	Articles	ROUGE-1	ROUGE-2	ROUGE-L	Compression Ratio	Avg. Time
AraT5 Arabic Summarizer	Abstractive	5	0.1099	0.0083	0.0859	9.8%	23.2s

The obtained results show that the AraT5 model was able to generate highly compressed Arabic summaries. The average ROUGE-1 score of 0.1099 indicates a modest lexical overlap between the generated summaries and the human reference summaries. The exceptionally low ROUGE-2 score of 0.0083 strongly suggests that

the model frequently utilizes entirely different phrasing, alternative vocabulary, or structural variations compared to the human summaries. Additionally, the ROUGE-L score of 0.0859 indicates limited structural and sequential similarity between the generated outputs and their reference counterparts.

The compression ratio of 9.8% confirms that the generated summaries were short and highly distilled relative to the original text documents.

❖ Hybrid Mode

Table 4-4: Preliminary Results for AraT5 with Hybrid mode

Model	Mode	Articles	ROUGE-1	ROUGE-2	ROUGE-L	Compression Ratio	Avg. Time
AraT5 Arabic Summarizer	Hybrid	5	0.1099	0.0083	0.0859	9.8%	25.1 s

🌈 The similarity of the values in the hybrid mode across both models is due to the limited number of articles used. However, when the number of articles was increased to 50, the values changed. The primary focus of the project was placed on the abstractive mode.

4.4. Comparative Analysis

The comparative evaluation was conducted using two Transformer-based models, namely mT5 Multilingual XLSum and AraT5 Arabic Summarizer, on 50 Arabic news articles extracted from the XLSum dataset. The evaluation relied on ROUGE metrics and generation time measurements.

The results are shown in Table 4-5.

Table 4-5: Comparative Analysis of mT5 and AraT5 on 50 Articles

Model	Avg. ROUGE-1	Avg. ROUGE-2	Avg. ROUGE-L	Avg. Time (s)
mT5	0.2656	0.1162	0.2375	41.21 s
AraT5	0.1188	0.0208	0.0991	26.82 s

The average experimental results demonstrate that the mT5 model consistently outperformed AraT5 across all ROUGE evaluation metrics. Specifically, mT5 achieved an average ROUGE-1 score of 0.2656, an average ROUGE-2 score of 0.1162, and an average ROUGE-L score of 0.2375. In contrast, AraT5 obtained

lower average scores of 0.1188, 0.0208, and 0.0991 for ROUGE-1, ROUGE-2, and ROUGE-L, respectively.

These findings indicate that mT5 was more effective at preserving the important information contained in the original texts and generating summaries that were closer to the reference summaries. The higher ROUGE scores suggest a better overlap between the generated and reference summaries, reflecting superior summarization quality. This performance can be attributed to the fact that mT5 was originally trained on the XLSum dataset, which includes multilingual news summarization data and therefore aligns well with the evaluation corpus used in this study.

Regarding computational efficiency, AraT5 required less processing time than mT5. The average generation time of AraT5 was 26.82 seconds, whereas mT5 required an average of 41.21 seconds to generate summaries. Therefore, AraT5 offers faster inference and lower computational cost, while mT5 provides significantly better summarization quality according to the ROUGE evaluation metrics.

Overall, the comparative analysis confirms that mT5 is the most effective model among the evaluated systems for Arabic text summarization, achieving the best balance between content preservation and summary relevance. Although the formal comparative analysis was structured around 50 designated articles, the models were further evaluated using numerous additional, randomly selected external texts to verify generalizability. Across all these supplementary trials, mT5 consistently maintained its superiority over AraT5, confirming the reliability of the observed patterns. While expanding the formal evaluation framework to 100 or more articles would provide more data, it was strictly constrained by time limitations, as qualitative verification of Arabic summaries demands extensive processing time. Consequently, mT5 remains the preferred choice for high-quality results, whereas AraT5 may be preferred in scenarios where faster summary generation is required.

4.5. Qualitative Analysis

In addition to numerical evaluation, qualitative inspection was performed to analyze the semantic quality of the generated summaries.

Some generated summaries successfully captured the main idea of the original article. For example, in one article about the Syrian army regaining control of Al-Khalidiya district in Homs (as presented in the Table 4-6).

Table 4-6: Article "The Syrian Army Regaining Control of Al-Khalidiya"

Title	Article
الخالدية: التلفزيون السوري يعلن استعادة الجيش السيطرة على الحي من أيدي المعارضة المسلحة	"وأكدت وسائل الإعلام السورية أن الجيش "استعاد الأمن والاستقرار عبر حي الخالدية بالكامل". ولم ترد تأكيدات لهذا النبأ من جانب المعارضة، لكن المرصد السوري لحقوق الإنسان كان قد قال الاثنين إن الاشتباكات مستمرة في حي الخالدية. لكنه قال إن قوات الحكومة استعادت أغلب أرجاء الحي وأحكمت حصارها على المناطق المحدودة التي ما زالت متبقية تحت سيطرة المعارضة في وسط المدينة. رمز للمعارضة دبابات سورية شوهدت داخل الحي. ويعد حي الخالدية أحد الأحياء الرمزية للمقاتلين المعارضين للنظام السوري. وتعني السيطرة عليه عزل الأحياء التي يسيطر عليها المعارضون والمحصنة منذ أكثر من عام، ويمهد هذا الطريق لسيطرة الجيش السوري على مدينة حمص بأكملها. مواضع قد تهلك نهاية وقال المرصد السوري الاثنين إن الطيران الحربي السوري نفذ غارتين على حي باب هود الواقع جنوب الخالدية. وتأتي المكاسب التي حققتها قوات الجيش بعد شهر من بدء هجوما في حمص في إطار حملة لتكوين محور يربط بين دمشق ومناطق ساحلية على البحر المتوسط. وكان الجيش السوري قد استعاد قبل نحو شهرين السيطرة على منطقة القصير الاستراتيجية في ريف حمص، التي بقيت تحت سيطرة المعارضين لأكثر من عام. وقد لقي 100 ألف شخص حتفهم في الصراع الدائر في سوريا منذ عامين، والذي بدأ في صورة احتجاجات سلمية على حكم الرئيس السوري بشار الأسد في مارس/آذار 2011. وفر نحو مليوني سوري من الحرب."

The generated summaries from both models (mT5 and AraT5) preserved the central event and produced a fluent Arabic sentence, as displayed in the Figure 4-4.



Figure 4-4: Syrian military article summaries with mT5 and AraT5

However, some summaries contained factual inconsistencies. For instance, in a sports-related article, the generated summary confused the winner of the match. This type of error shows that a fluent generated summary is not always factually correct.

This observation highlights an important limitation of abstractive summarization models: they may generate grammatically correct summaries while altering or distorting factual information. Therefore, automatic evaluation should be complemented with human evaluation, especially when factual accuracy is important.

4.6. Conclusion

The experimental results demonstrated that both mT5 and AraT5 are capable of generating meaningful and coherent summaries for Arabic texts. However, the comparative analysis revealed that mT5 consistently achieved higher ROUGE scores, indicating a better ability to preserve the essential information contained in the original texts and produce summaries that were closer to the reference summaries. In contrast, AraT5 achieved lower ROUGE scores but required less processing time, making it more computationally efficient. Overall, the experiments validate the proposed system and show that Transformer architectures can successfully address the challenges of Arabic text summarization.

Conclusion

This work focused on the development and evaluation of an Arabic text summarization system based on Transformer models. The proposed system was implemented as an interactive graphical application using Python, Streamlit, PyTorch, and the Hugging Face Transformers library. It integrates different summarization approaches, including abstractive, extractive, and hybrid summarization, while providing additional functionalities such as model comparison, ROUGE evaluation, human evaluation, experiment tracking, and out-of-domain testing.

The experimental study was conducted using two Transformer-based models, namely mT5 Multilingual XLSum and AraT5 Arabic Summarizer. The obtained results demonstrated that both models are capable of generating coherent and informative summaries from Arabic texts while significantly reducing the length of the original documents. The evaluation process, based on ROUGE metrics and human assessment criteria confirmed the effectiveness of Transformer architectures for Arabic text summarization tasks.

The comparative analysis showed that mT5 achieved the best overall performance, obtaining higher average ROUGE scores (ROUGE-1 = 0.2656, ROUGE-2 = 0.1162, and ROUGE-L = 0.2375), indicating a better ability to preserve important information and generate summaries closer to the reference summaries. In contrast, AraT5 achieved lower ROUGE scores but required less processing time, with an average generation time of 26.82 seconds compared to 41.21 seconds for mT5. These findings highlight the trade-off between summarization quality and computational efficiency.

Several challenges were encountered during this work, including the limited availability of high-quality Arabic summarization datasets, the computational requirements of Transformer models, and the difficulty of evaluating generated summaries using automatic metrics alone. Furthermore, the system showed higher performance on news articles than on specialized academic or scientific texts. Other limitations include the large size of Transformer models, the need to download model files during the first execution, the possibility of generating fluent but factually inaccurate summaries, and the limitations of ROUGE in capturing semantic correctness. In addition, the current application mainly relies on pre-trained or previously fine-tuned models, while full fine-tuning requires more powerful hardware resources.

Overall, the proposed system demonstrates the potential of Transformer-based models for Arabic automatic text summarization and provides a practical platform for experimentation and evaluation. Future work may focus on performing full fine-tuning on larger Arabic summarization datasets such as XL-Sum, integrating additional models such as AraBART, improving long-document processing through more advanced chunking strategies, and adopting semantic evaluation metrics such as BERTScore. Further enhancements could also include supporting PDF and Word document summarization, optimizing the system for resource-constrained devices, extending evaluation to a wider range of domains, and deploying the application as a real-time web service.

References

- [1] E. LIDDY, "Natural language processing, NY: In Encyclopedia of Library and Information Science", 2001.
- [2] A.Khouloud, "Multi-label classification of Arabic text using deep learning models," Master's thesis, University of MOHAMED BOUDIAF, MSILA, 2025.
- [3] K. Ayoub, S. Ramzi, "Deep learning models applied to arabic quranic text," Master's thesis, University of MOHAMED BOUDIAF, MSILA, 2024.
- [4] A. A. Nafea, M. S. Muayad, R. R. Majeed, A. Ali, O. M. Bashaddadh, M. A. Khalf, A. N. Sami, and A. Steiti, "A Brief Review on Preprocessing Text in Arabic Language Dataset: Techniques ana Challenges," *Babylonian Journal of Artificial Intelligence*, pp. 46-53, 2024.
- [5] A. Tahani, S. Shireen, A. Reem, S. Shahendah, and N. Sarah, "Preprocessing Techniques for Clustering Arabic Text: Challenges and Future Directions," *International Journal of Advanced computer Science and Applications*, vol. 15, no. 8, 2024.
- [6] A. Souiri, M. Al Achhab, B. E. EL Mohajir, M. Naoum, O. EL Hichami, and A. Zbakh "Arabic Text Summarization Challenges using Deep Learning Techniques: A Review," *International Journal on Recent and Innovation Trends in Computing and Communications*, vol. 11, no. 11, 2023.
- [7] A. Qaroush, I. A. Farha, W. Ghanem, "An efficient single document Arabic text summarization using a combination of statistical and semantic features," *King Saud University- computer and Information Sciences*, vol. 33, no. 6, pp. 677-692, 2021.
- [8] Y. A. AL-Khassawneh, E. S. Hanandeh , "Extractive Arabic text summarization-graph-based approach," *Electronics*, vol. 12, no. 2, p. 437, 2023.
- [9] K. Morsi, F. Najib, W. Hussein, and R. Ismail, "AUTOMATIC SUMMARIZATION TECHNIQUES FOR ARABIC TEXT," *International Journal of Intelligent Computing and Information Sciences*, vol. 25, no. 1, pp. 74-88, 2025.
- [10] M. K. Eddine, N. Tomeh, N. Habash, J. Le Roux, and M. Vazirgiannis, "Arabart: a pretrained arabic sequence-to-sequence model for abstractive summarization," pp. 31-42, 2022.
- [11] Y. M. Wazery, M. E. Saleh, A. Alharbi, and A. A. Ali, "Abstractive Arabic text summarization based on deep learning," *Computational Intelligence and Neuroscience*, 2022.

- [12] D. Suleiman, and A. Awajan, "Multilayer encoder and single-layer decoder for abstractive Arabic text summarization," *Knowledge-Based systems*, vol. 237, p. 107791, 2022.
- [13] W. Etaiwi, and A. Awajan, "SemG-TS: Abstractive arabic text summarization using semantic graph embedding," *Mathematics*, vol. 10, no. 18, p. 3225, 2022.
- [14] A. ELSAID, A. MOHAMMED, L. Fattouh, and M. SAKRE, "Hybrid Arabic text summarization Approach based on Seq-to-seq and Transformer," 2023.
- [15] "what is Python? Executive Summary,"[online]. Availables: <https://www.python.org/doc/essays/blurb/>, accessed on: 19-05-2026
- [16] "Streamlit documentation," [online]. Availables: <https://docs.streamlit.io/>, accessed on: 19-05-2026
- [17] "Pandas Introduction," [online]. Availables: <https://www.geeksforgeeks.org/pandas/introduction-to-pandas-in-python/>, accessed on: 19-05-2026
- [18] "What is PyTorch," [online]. Availables: <https://www.geeksforgeeks.org/deep-learning/getting-started-with-pytorch/>, accessed on: 19-05-2026
- [19] "Total noob's intro to Hugging Face Transformers," [online]. Availables: https://huggingface.co/blog/noob_intro_transformers#what-is-hugging-face-transformers, accessed on: 19-05-2026
- [20] "Quick start guide," [online]. Availables: <https://www.jetbrains.com/help/pycharm/quick-start-guide.html>, accessed on: 19-05-2026
- [21] "Dataset Card for "XL-Sum"," [online]. Availables: <https://huggingface.co/datasets/csebuetnlp/xlsum>, accessed on: 19-05-2026
- [22] "mT5: A massively multilingual pre-trained text-to-text transformer," [online]. Availables: <https://arxiv.org/abs/2010.11934> , accessed on: 19-05-2026
- [23] E.B. Nagoudi, A. Elmadany, and M. Abdul-Magied "AraT5: Text-to-Text Transformers for Arabic Language Generation," *Deep Learning and Natural Language Processing Group*, vol. 1, p. 628–647, 2022.
- [24] "Two minutes NLP — Learn the ROUGE metric by examples," [online]. Availables: <https://medium.com/nlplanet/two-minutes-nlp-learn-the-rouge-metric-by-examples-f179cc285499>, accessed on: 19-05-2026