Omar Ahmed

Semantic search engine for the holy Quran

Metropolia University of Applied Sciences

Master of Engineering

Information Technology

Master’s Thesis

9 May 2024

PREFACE

بِسْــــــــــمِ اللهِ الرَّحْمَنِ الرَّحِيْمِ

When I started to search for a topic for my thesis, I decided to choose something that has meaning and value. So, the meaning would be a topic where I can learn something new, the value would be something that can help others in real life or inspire others to build on top of that. I ended up choosing to develop an AI solution to help everyone fetches and understand the holy Quran and to support my nation with my knowledge.

The biggest challenge for this journey was the passing away of my father. Alhamdulillah, thank you Dad for all the love and support you gave me, you are always the best and I'll never forget you.

To my mom, thanks for all the love and support you have given me, I love you, Mom.

To my beautiful wife Tota and my children Laila and Zain that wouldn't be possible Without your support, love, and encouragement.

I need to also thank my family -**Hassan's Omar Family**- Ali, Pedo for being in my life.

Lastly, I need to extend my gratitude to my supervisors Ville Jääskeläinen and Peter Hjort for all their support and kindness.

Espoo - Finland, 9 May 2024
Omar Ahmed

Abstract

Author:	Omar Ahmed
Title:	Semantic search engine for the holy Quran
Number of Pages:	71 pages + 2 appendices
Date:	9 May 2024
Degree:	Master of Engineering
Degree Programme:	Information Technology
Professional Major:	Networking and Services
Supervisors:	Peter Hjort, Advisor – Main evaluator
	Ville Jääskeläinen, Program manager

Fetching verses from the holy Quran is not an easy process, even using Google or ChatGPT will not help that much in this task and the user will get either ton of articles or untrusted quality data because ChatGPT still can make mistakes. This tool is important especially for Muslims to search verses by using the natural language and for everyone to know more about the Quran.

From theory to practice, the biggest challenge was which path needed to be used to solve this task. From using pre-trained models to topic modeling to word2vec that was the key point to the solution with cosine similarity. In the end, a web UI was built on top of that, that takes user queries and returns relevant Quranic verses.

Keywords: Semantic search, Cosine similarity, Word2vec, Transformers, Topic modelling, NLP.

Contents

List of Abbreviations

BERT	Bidirectional Encoder Representations from Transformers
GloVe	Global Vectors for Word Representation
ML	Machine Learning
NLP	Natural Language Processing
LDA	Latent Dirichlet Allocation
SVD	Singular Value Decomposition
Word2vec	Word to Vector

1 Introduction

Nowadays, In the age where everything is rounding around information technology, the digital world become a vital part of our daily life in all fields from banks to hospitals to education to a wide variety of fields. It was wise to transform our ordinary way of reading, understanding, and exploring religious books with the power of transformative technology.

As the Holy Quran is the main and the only book around the Islamic world that almost 2 billion people are interested in it, so it was a chance to pick up this topic that will not only help Muslims in their normal usage of the Holy Quran but it will be a trusted and an illuminative guide for non-Muslims who are interested to dive deeper and fetch information from this book for hundreds of different topics.

This research is in pursuit of leveraging the process of exploring the holy Quran and unraveling the profound meanings within its verses which may require experts to dig and retrieve, thanks to the power of modern computational techniques that launched a new era of exploration and analysis.

The focus of this thesis and this scholarly journey is to develop a simple Semantic Search Engine for the Holy Quran, aiming to bypass the boundaries of traditional textual searches and unlock a deeper comprehension of its sacred content.

The Holy Quran is the book that has been read by all Muslims around the world as the literal word of Allah which was first given to Prophet Muhammad and then to the whole Islamic world. It can be considered an illuminative guide that contains a comprehensive set of verses covering diverse aspects of life in different situations, morality, and spirituality. As an example, Muslims are getting the rules and guidance of trading from the holy Quran.

Traditional exact word search engines often fall into big and popular trouble in capturing the understandable meanings and the interconnections present between these verses, as a conclusion that leads to limiting the depth of the exploration and understanding.

Recognizing this issue and starting from this point, the Semantic Search Engine was the solution appropriate proposition in this thesis seeks to bridge the gap between the impressive depth of the Quranic knowledge and the user's need for getting precise and meaningful information.

The concept of the semantic search technique goes beyond the traditional concept of keyword matching, diving into the semantic relationships and contextual intricacies embedded within the text. By making use of advanced natural language processing (NLP) techniques, ontologies, and machine learning algorithms, the proposed search engine aims to interpret and comprehend the semantic nuances of Quranic verses.

This innovative approach allows users to explore, diver deeper and get a better understanding of the Holy Quran not merely as a collection of isolated verses but as an interconnected and harmonious body of knowledge, revealing the rich interconnectivity and relativity of meanings throughout the sacred text.

Furthermore, this thesis endeavors to address the challenges inherent in the linguistic complexities of the Arabic language, the contextual richness of Quranic verses, and the diverse interpretations offered by classical interpretation. The Semantic Search Engine aims to simplify the search process and provide users with a focused search experience based on their specific questions and levels of expertise.

As a starting point for this scholarly endeavour, the intention is not to replace the role of religious scholars and experts but to complement their efforts by providing a powerful tool for in-depth exploration and analysis. This thesis involves a harmonious combination of technology and spirituality, linking a deeper connection between individuals and the sacred teachings of the Holy Quran. Through the development of a Semantic Search Engine for the Holy Quran, this research seeks to apply the technology to the religious field, introducing a new creative way for the exploration of getting impressive knowledge in the digital age.

1.1 Motivation: The Need for Semantic Search in the Holy Quran

Searching in the holy Quran is not an easy process, it requires some previous experience such as being familiar with the Arabic language and the meaning of the verses to start searching for a specific topic. To start with, the holy Quran includes thousands of verses [1] and a manual search in this case will be very difficult because verses are not sorted by topic.

As an example, to get verses from the holy Quran that can answer the question **“What are the rights of women after divorce in Islam?”**. Answering this question will require a lot of research and verse collection which may take many hours and will require an expert in the Quran to do that. From this point using modern technologies such as **ML** and **NLP** came into action to solve this issue by implementing a semantic search [2] engine that can get the natural language as a text input and return related verses as a text output. The main motivation items can be listed as follows:

1.1.1 Complexity of Quranic Text

The Holy Quran is considered the literal word of Allah in Islam and presents an impressive linguistic and literary within its verses. Its verses are intricately formed and wisely sorted with no faults or grammar problems, meanings within its verses bypassed the conventional modes of expression.

The main language is the classical Arabic language in which the Quran is written adds another dimension of complexity, with its rich and deep vocabulary and complex grammatical structures challenging even proficient readers. Consequently, the conventional keyword-based search approaches often fail to capture the depth, nuance, and interconnectedness of Quranic verses, a more sophisticated method is necessary for exploration.

1.1.2 Diverse Interpretations

Throughout history, Islamic scholars and researchers have offered diverse interpretations of the Quran, reflecting the deepness and richness of Islamic content especially what you can find in the Holy Quran and the complexity of human understanding. These interpretations reveal legal and spiritual perspectives, reflecting the nature the deep layers and the knowledge of the Quranic message.

As a result, users who are seeking to explore and get a better understanding of a specific topic, concept, or passage may encounter a variety of interpretations, each offering unique insights and perspectives. A Semantic Search Engine has the potential to facilitate the navigation of this diverse interpretation, allowing users to explore the Quranic text easily and in a nuanced and contextualized manner.

1.1.3 Seek for Deeper Understanding

Nowadays, in current modern society, where there is a spread for knowledge everywhere in addition to a growing thirst for deeper spiritual understanding and knowledge gain engagement with religious texts. The Holy Quran, as a source of guidance and illumination from Allah, serves as a lighthouse for seekers of truth and enlightenment. However, the impressive volume and complexity of its content and verses can pose challenges to those seeking to reveal its secrets and ambiguity.

A Semantic Search Engine offers a pathway to deeper exploration and understanding by enabling users to surf the Quranic corpus [3] with precision and insight, uncovering hidden interconnections between topics and verses, and layers of hidden meanings that lie in the deep of the context.

1.1.4 Enhanced User Experience

Regular keyword-matching search engines often provide a specific legacy approach to information retrieval, focusing primarily on keyword matching and relevance ranking. While effective for many types of content, this approach may fall short when applied to the intricacies of religious texts that are rich with different meanings and a variety of topics like the holy Quran.

A Semantic Search Engine, by contrast, offers a more intuitive and well-functionally implemented user experience, allowing users to engage with the Quranic text in a manner that is reflective of its linguistic and semantic richness. Through features such as semantic analysis, contextual understanding, users can have a journey of exploration that is tailored to their individual interests and inquiries.

1.1.5 Integration of Technology and Faith

The intersection of technology and faith in the modern ages represents a fertile ground for innovations and explorations in order to reach a better world that has answers to all questions especially in the field of religion that can influence billions of people around the world for all religions not just the Islamic world but for everyone and that can lead to more peace and having a better understanding for all religions without the need of experts but with the power of the AI.

In recent years, advancements in computational linguistics, artificial intelligence, and data analytics have opened new boundaries in the study of religious texts, enabling scholars and practitioners to uncover insights and patterns that were previously inaccessible. A Semantic Search Engine for the Holy Quran is a great example of this convergence between technology and faith, making use of the power of modern computing technologies to facilitate a deeper engagement with sacred texts such as what exists in the holy Quran.

1.1.6 Global Accessibility

From reading paper books to digital views with a variety of devices and different monitor sizes, the process of digitalization the religious texts have transformed the way that individuals can access, study, and engage with sacred texts in addition to the ability to put notes or highlight or even setting bookmarks where you can leave and save your reading progress.

In an increasingly interconnected world, digital platforms provide a gateway to global audiences, no matter where you are because they can bypass geographical restrictions, linguistic, and cultural boundaries. A Semantic Search Engine for the Holy Quran extends this accessibility by offering users the ability to explore the Quranic text in their preferred language “mostly it will be Arabic in the first iteration and more languages will be added later” while retaining the integrity and authenticity of its original Arabic form.

This democratization that makes it easy for access from anywhere enables individuals from diverse backgrounds and communities or even from different religions to engage and interact with the Quranic message through this search engine on their own terms, which makes it a more inclusive and diverse message for everyone.

1.1.7 Scholarly Inquiry and Research

For scholars and researchers engaged in Quranic studies, a Semantic Search Engine represents a powerful tool for data analysis, comparative analysis, and thematic exploration.

Whether exploring linguistic variations, tracking thematic exploration, or conducting cross-referential studies, scholars can make use of the capabilities of a Semantic Search Engine to advance the boundaries of Quranic scholarship and contribute to the global dialogue on Islam and its sacred texts.

1.2 Background: Understanding Semantic Search and Its Applications

Semantic search represents an industrial revolution in the field of information technology in addition to being a pivotal advancement in information retrieval, making use of the most sophisticated algorithms to realize and capture the meaning of the user’s query using his own local language even if the search term is not that much formal instead of the legacy keyword matching techniques which may require to change your search query many times to satisfy the search algorithm.

This exactly looks like an investigator who is navigating through a very crowded area, semantic search in a nutshell can understand the essence of user queries regardless of each quality whether it is a slang or formal language, thereby leading to a revolution in the landscape of online search methodologies.

At its core from the heart of the main idea, semantic search bypasses the boundaries of traditional keyword-based systems by diving into the semantic nuances of language. Unlike conventional search engines, which rely solely on textual similarity that may require the user to input a specific word to get the approximate correct results and may not get what he is looking for in the end.

Semantic search algorithms make use of the power of artificial intelligence to ease the understanding of the complexity of the contextual meaning, synonyms, and related concepts that lie down in the deeper layers within user queries and his intent. Consequently, semantic search generates a more intuitive and search experience that is all about the user and his intent with his simple language regardless of its quality, effectively bridging the gap between user intent and search results.

Operationalizing semantic search involves the intricate interaction of diverse computational techniques, including knowledge representation, machine learning and natural language processing which is a field of artificial intelligence and computational linguistics that focuses on the interaction between humans and computers through natural language in addition to enabling computers to understand, interpret, and generate human language in a way that is both meaningful and contextually relevant.

These methods help search engines understand human language better. They can figure out the hidden meanings and connections between words in what people search for. This makes it easier for search engines to give more accurate results that match what users are looking for.

In real life, semantic search isn't just for regular web searches. It's used in many parts of the digital world today. It's not just about searching the internet. It's about how we find things and understand information across different digital platforms and tools we use every day.

Social media platforms in the modern platforms are making use of the most beneficial algorithms nowadays such as that one regarding semantic search algorithms to personalize content recommendations [4], increasing the possibility of the user interacting and doing more engagement and adding more community interaction.

The modern and popular E-commerce platforms such as Amazon, Alibaba, or any other competitor in this field are focusing on the core idea of how to keep loyal customers which can happen thanks to the semantic search to deliver tailored product recommendations, thereby enhancing user experience and facilitating informed purchase decisions.

Semantic search serves as the foundational mechanism enabling intelligent virtual assistants, such as Siri and Alexa, to understand user inquiries and execute commands with precision and efficacy, by trying to understand the user’s intent and the contextual nuances.

Virtual assistants with efficiency handle intricate queries and strengthen seamless human-computer interaction across diverse domains. This advanced capability allows users to engage in natural-language interactions with virtual assistants, thereby enhancing user experience and facilitating efficient task execution.

However, despite its promising possibilities in the modern technology research field, semantic search techniques face challenges for a variety of reasons [5]. Human language for example can be tricky and misleading to be understood because it's not always clear or consistent. This makes it difficult for the machine to interpret the intent of the user.

So, researchers and developers must keep working hard to improve how computers understand and respond to language because language is growing very fast, and every day mostly new words appear at least in slang. It's like a puzzle that they're always trying to solve so that we can get better results when we ask our virtual assistants for help.

As a result, semantic search represents a paradigm shift in information retrieval, embodying the convergence of artificial intelligence, linguistics, and cognitive science. As the digital landscape continues to evolve, semantic search stands as a prove to the modern science research march of technological progress, reshaping the features of human interaction with information in profound and unmatched ways, this technique can be involved in several industries such as:

- **Search Engines:** it helps users to get relevant search results based on the context of their search query not with key matching.
- **E-Commerce:** it can help to get a better understanding of what the user is looking for and return the relevant products.
- **Virtual Assistants:** Google Assistant as an example uses semantic techniques to understand the context from the user input to do the correct task.
- **Healthcare:** Medical bots are using semantic methods heavily to fully understand the context of the problem and based on that it can move the case in the correct direction.

1.3 Research Objectives: Addressing the Challenges of Quran Search and Retrieval

This study aims to use the power of AI/ML to tackle the obstacles related to searching and retrieving information using a simple natural language from the Quran, the holy book of Islam. In doing so, we are trying to make the process more accessible, easier, and user-friendly for individuals seeking guidance, knowledge, and inspiration from this revered text.

Many people today are exploring Islam to understand its views on specific topics. This helps them avoid relying on potentially misleading information from various sources.

As AI is very popular, it was a nice opportunity to automate the manual process of searching the holy Quran by implementing a smart semantic search engine for the holy Quran that can make it easy for everyone to use and use his own words to retrieve the relevant verses that can answer his questions and support my people with my knowledge. Objectives and Challenges can be as the following:

- Objectives:
    - Enhancing Search Efficiency: the main goal is to develop techniques and tools that help users find and locate specific verses, themes, or topics within the Quran concisely, quickly, and efficiently. This involves improving search algorithms and designing intuitive interfaces to streamline the search process.
    - Improving Retrieval Accuracy: One of the targets is to refine the accuracy of search results by implementing advanced linguistic and semantic analysis techniques. By understanding the context and meaning of verses, the tool should deliver more relevant and precise information to users.
    - Facilitating Cross-Referencing: Enabling users to explore interconnections between different verses, chapters, and themes within the Quran. This involves developing features that allow for seamless navigation and cross-referencing of related content.

  - Ensuring Accessibility: It’s very important in this research to ensure that our search and retrieval solutions are accessible to a diverse range of users, including those with varying levels of technological proficiency and linguistic abilities. This includes designing interfaces that are intuitive, user-friendly, and culturally sensitive.
  - Promoting Understanding and Reflection: Behind facilitating search and retrieval for verses, there is aimed to promote deeper understanding and reflection on the teachings and messages of the Quran. Features such as commentary, interpretation aids, and educational resources, enrich users' engagement with the text.
- Challenges:
  - Data collection and annotation: in the first step, data should be collected and annotated. For example, the Quran should be divided into verses based on the topic.
  - Linguistic Complexity: The Quran is written in classical Arabic, which presents linguistic challenges for modern readers. The text contains archaic vocabulary, complex grammatical structures, and rhetorical devices that may be unfamiliar to contemporary users.
  - Data cleaning: data should be normalized and tokenized.
  - Ambiguity and Interpretation: The Quranic text is rich in symbolism, which can lead to multiple interpretations of the same verses. Navigating through these interpretations while maintaining accuracy and authenticity poses a significant challenge.
  - Achieve the embedding representation of the Quran verses.
  - Choose the best algorithm to find the semantic similarity.
  - Semantic Complexity: Understanding the semantic nuances and contextual meanings of Quranic verses requires deep knowledge of Islamic theology, jurisprudence, and linguistic traditions. Capturing these nuances in search and retrieval algorithms is a challenging task.
  - Cross-Referencing and Contextualization: The Quranic text is highly interconnected, with verses often referring to events, concepts, and themes mentioned elsewhere in the scripture.

Creating mechanisms for effective cross-referencing and contextualization poses technical and interpretative challenges.

- Cultural Sensitivity: The Quran is not only a religious text but also a cultural and historical artefact with profound significance for Muslims worldwide. Developing search and retrieval systems that reflect diverse cultural contexts is essential.
- Technological Limitations: Existing search and retrieval technologies may not be well-suited to handle the unique linguistic and semantic characteristics of the Quran. Adapting and customizing these technologies to effectively process and analyze Quranic texts presents technical challenges.
- User Accessibility: Ensuring that Quranic search and retrieval tools are accessible to users with diverse linguistic backgrounds, educational levels, and technological proficiencies requires careful interface design and user experience considerations.
- Authenticity and Reliability: Verifying the authenticity and reliability of Quranic texts and interpretations is crucial for maintaining the integrity of search and retrieval results. Addressing issues related to variant readings, textual transmission, and scholarly consensus is a significant challenge.

2 Literature Review

2.1 Existing Quran Search Engines: Limitations of Keyword-Based Approaches

In the modern age, technology has greatly facilitated the ease of access to a variety of religious texts such as the holy Quran. With the invention and continuous development of Quran search engines, users can explore its verses with ease of access and quick retrieval for the desired info.

However, many of these search engines [6][7] rely heavily on the current legacy technique that is known with keyword-based approaches, which may present limitations and shortages in terms of accuracy and depth of understanding. This literature review aims to explore the existing Quran search engines and discuss the drawbacks associated with keyword-based methods.

Quran search engines are digital tools designed to assist users in locating specific verses or topics within the Quranic text. These platforms offer functionalities such as keyword search, verse lookup, enhancing accessibility and research capabilities for scholars, students, and general readers who are seeking to retrieve verses that can answer their questions or regarding religion comparison reasons.

While keyword-based approaches provide a convenient means of searching for specific terms or phrases within the Quran, they have several inherent limitations that may not be the best regarding information retrieval:

- **Contextual Understanding:**
  - Keyword-based searches may overlook the contextual nuances of Quranic verses.
  - The Quran is a highly nuanced text that is rich with deep meanings and a variety of words that can have the same meaning which is used in a different context, and the meaning of verses often depends on their surrounding context.
  - Keyword searches may fail to capture these subtleties, leading to misinterpretations or incomplete understandings of the text.
- **Linguistic Challenges:**
  - The Arabic language, in which the Quran is written, poses unique challenges for keyword-based searches. Arabic words often have multiple meanings and can vary depending on their context grammatical structure and diacritic.
  - As a result, simplistic keyword-matching algorithms may struggle to accurately interpret the intended meaning of Quranic verses as they can have different meanings and then it may return wrong results.
- **Semantic Complexity:**

    - The Quran employs rich semantic and rhetorical devices, including metaphors, allegories, and analogies, which may not be effectively captured through keyword-based searches alone.
    - Understanding the deeper layers of meaning in Quranic verses requires more sophisticated analysis techniques that go beyond simple keyword matching.
- **Cultural and Historical Context:**
    - Quranic interpretation often requires consideration of the cultural and historical context in which the text was revealed.
    - Keyword-based approaches may overlook these contextual factors, leading to interpretations that are disconnected from the broader of the Quranic revelation.

The primary problem with the current 'Keyword-based' approach is that it depends alone on finding the exact words from the search query. However, it overlooks the fact that different words with similar meanings can convey the same idea but in different contexts.

For instance, if you search for 'forgiveness', the system may not recognize 'mercy' as a relevant term, even though it could serve as a suitable alternative. This limitation restricts the effectiveness of the search process, as it fails to consider synonyms or related terms that could accurately reflect the user's intent.

To address the limitations of keyword-based Quran search engines, future research should focus on the development of more advanced computational methods that integrate linguistic analysis, semantic modelling, and contextual understanding.

By leveraging techniques from natural language processing, computational linguistics, and Quranic text, researchers can create more robust and comprehensive tools for exploring the depths of the Quranic text. As the exact match offers low-quality search results as illustrated (see Figure 1), the target is to make use of broad search techniques and what is beyond that such as semantic search techniques.

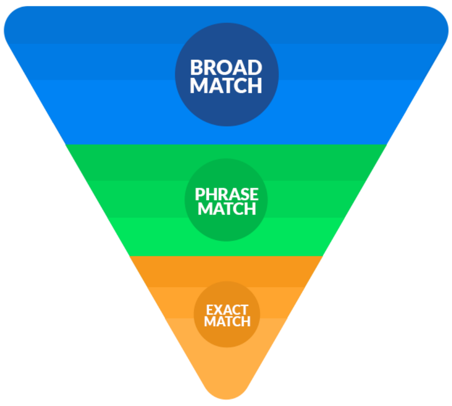


Figure 1. Search techniques triangle.

Another significant issue coming out from the system's incapacity to grasp contextual nuances and manage synonyms effectively. This shortage often results in the misinterpretation of user queries.

For instance, consider a scenario where a user seeks verses related to spiritual enlightenment or guidance, commonly referred to as "spiritual light." However, due to the system's limited ability to grasp the context and handle synonyms, it may retrieve verses that are related to physical light instead.

Although both concepts, spiritual and physical light, share an underlying meaning connection, the system fails to recognize the distinction, leading to erroneous search outcomes. This failure to comprehend the subtleties of language and context hinders the search engine's ability to accurately fulfil user queries, thereby diminishing its overall effectiveness and reliability.

Lastly, while existing Quran search engines have significantly enhanced accessibility to the Quranic text, their reliance on keyword-based approaches poses limitations in terms of accuracy, depth of understanding, and contextual relevance.

Addressing these limitations requires a multidisciplinary approach that combines computational techniques with insights from Quranic scholarship and linguistics. By overcoming the challenges associated with keyword-based methods, future Quran search engines can provide users with richer and more nuanced insights into the holy Quran.

2.2 Semantic Search Techniques: Exploring Concepts, Ontologies, and Embeddings

Semantic search techniques represent a significant advancement in information retrieval systems, enabling users to find relevant content based on the meaning and context of their queries rather than relying solely on keyword matching. This literature review dives deeper into the exploration of semantic search techniques and how they can be used to develop a smart retrieval system, particularly focusing on concepts, ontologies, and embeddings.

2.2.1 Understanding Semantic Search

Semantic search goes beyond traditional keyword-based approaches by considering the meaning, relationships, and context of words and phrases within a given text or document corpus such as the holy Quran. Unlike keyword matching, which relies on exact matches that may retrieve a piece of misleading information, semantic search techniques aim to understand the intent behind user queries and retrieve relevant information based on semantic similarity.

2.2.2 Concepts in Semantic Search

Concepts play a central role in semantic search, representing abstract ideas or categories that encapsulate related terms and entities. Semantic search systems leverage concept-based indexing and retrieval mechanisms to match user queries with documents containing relevant concepts, even if the exact keywords do not appear verbatim in the text.

2.2.3 Ontologies and Semantic Networks

Ontologies are formal representations of knowledge domains, typically structured as hierarchical networks of concepts and their relationships (see Figure 2).

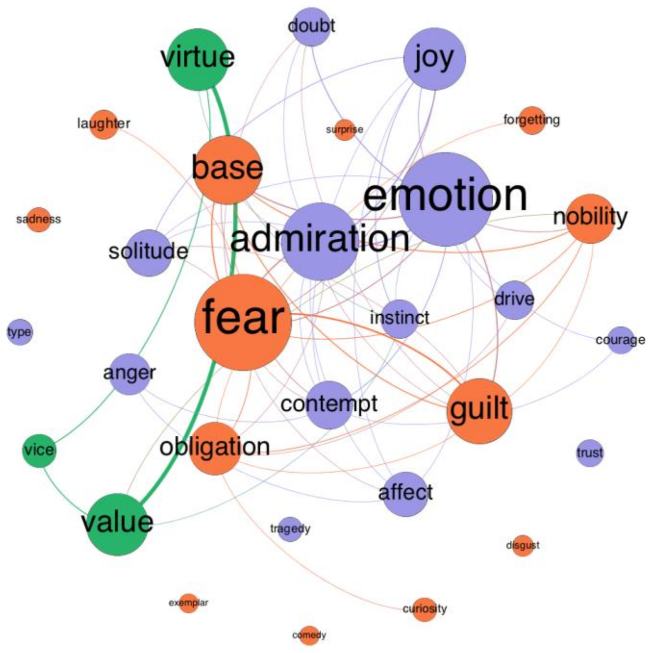


Figure 2. Semantic Networks.

In semantic search, ontologies provide a structured framework for organizing and categorizing information, enabling more precise and contextually relevant search results. By incorporating ontological knowledge into search algorithms, semantic search systems can infer semantic relationships between terms and improve the accuracy of search results.

2.2.4 Embeddings and Semantic Similarity

Word embeddings [8] are dense, low-dimensional vector representations of words that capture semantic relationships based on contextual usage patterns. Techniques such as word2vec and GloVe generate embeddings by training neural networks on large text corpora, other techniques for generating embeddings could be matrix factorization or SVD [9], enabling algorithms to compute semantic similarity between words and phrases.

In semantic search, embeddings facilitate the matching of user queries with relevant documents based on semantic proximity, even in the absence of exact keyword matches (see Figure 3).

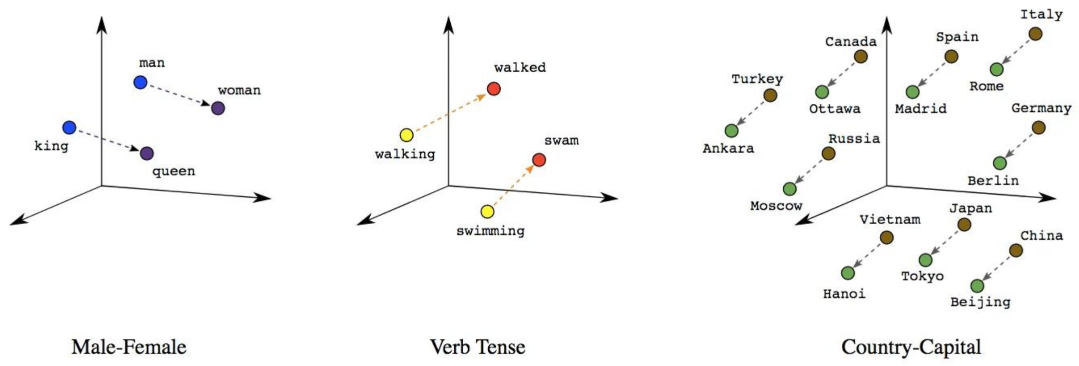


Figure 3. Word embeddings.

2.2.5 Challenges and Opportunities

While semantic search techniques offer promising benefits for information retrieval, they also present several challenges, including:

- **Scalability:** Processing and indexing large-scale datasets with semantic annotations can be computationally intensive, requiring efficient algorithms and infrastructure.
- **Ambiguity and Polysemy:**
    - Natural language is inherently ambiguous, with words often having multiple meanings depending on context.
    - Semantic search systems must address issues of ambiguity and polysemy to accurately interpret user queries and retrieve relevant results.

- **Integration with Existing Systems:** Integrating semantic search capabilities into existing search engines and applications may require significant technical expertise and architectural modifications.

Lastly, semantic search techniques represent a paradigm shift in information retrieval, enabling more accurate, contextually relevant, and intuitive search experiences for users. By leveraging concepts, ontologies, and embeddings, semantic search systems can overcome the limitations of traditional keyword-based approaches and unlock new possibilities for exploring and discovering knowledge in vast digital repositories.

As research and development in semantic search continue to evolve, the future holds exciting prospects for enhancing the efficiency and effectiveness of information discovery in the digital age.

In general, semantic search techniques make it possible to understand the meaning and the whole context of the user query that can facilitate retrieving contextually relevant search results, here are the anatomy items for the idea:

Concept-based search: instead of the old and legacy strategy of searching relying solely on keyword matching, semantic search considers the context, outlines and underlying concepts associated with the words. It aims to understand the user's intent in his natural language and the context in which terms are used.

As an example, when a user searches for "forgiveness," the system understands related concepts like mercy, pardon, and compassion, providing more comprehensive results.

Ontologies: this is the definition of the hierarchies and the relationships between concepts. It can help in organizing and representing knowledge, facilitating a profound understanding of the subject matter.

Word embeddings: this technique represents words as vectors in a multi-dimensional space, the embedding is used in text analysis. In that vector space, it is expected that words that are close to each other have similar meanings.

Context-aware search: this technique considers the surrounding context while searching and helping in uncovering the ambiguity of meanings.

NLP: this technique adds the ability for the computer to understand and process the human language in addition to analyzing the structure, the contextual meaning, and the interpretation of the user queries. It can understand the relationship between entities in the query and decide whether the user is asking about the knowledge or the exact word.

Query expansion: this technique helps in capturing a broader range of relevant words by taking care of a word synonym, such as expanding the word ‘right’ with ‘appropriate’ and ‘good.’

2.3 Arabic Natural Language Processing (NLP): Challenges and Opportunities

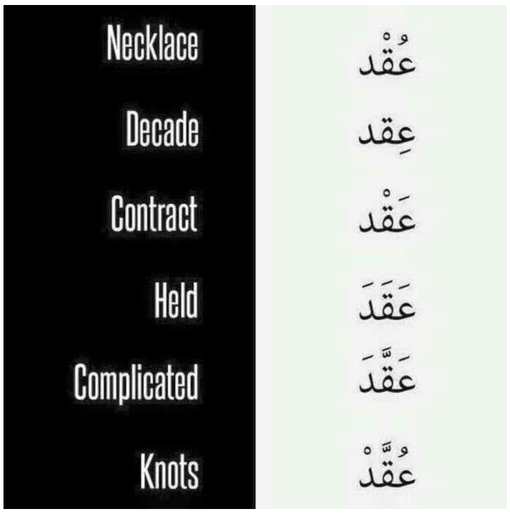
Arabic Natural Language Processing (NLP) refers to the field of computational linguistics that focuses on developing algorithms and techniques to enable computers to understand, interpret, and generate multiple languages including the Arabic language. This literature review explores the challenges and opportunities in Arabic NLP.

2.1 Challenges in Arabic NLP

Arabic presents unique challenges for natural language processing due to its rich morphology, complex grammar, and diverse dialects. Some of the key challenges include:

- **Morphological Complexity:** Arabic is characterized by a complex morphological structure, with words often containing roots, prefixes, suffixes, and infixes. This morphological richness poses challenges for tasks such as tokenization, stemming, and part-of-speech tagging.

- **Dialectal Variation:**
    - Arabic is spoken across a wide geographic area, leading to significant variation in vocabulary, pronunciation, and grammar across different dialects.
    - NLP systems must account for this dialectal diversity to ensure robustness and accuracy in text processing and understanding.
- **Limited Linguistic Resources:**
    - Compared to languages like English, Arabic has relatively fewer annotated corpora, lexicons, and linguistic resources available for training and evaluation purposes.
    - The scarcity of high-quality data presents challenges for developing accurate and reliable NLP models and algorithms.
- **Orthographic Variation:**
    - Arabic script allows for variation in spelling and diacritization, leading to inconsistencies in text representation.
    - NLP systems must account for orthographic variations to ensure robustness and accuracy in tasks such as named entity recognition and text normalization.
- **Semantic Ambiguity:**
    - Arabic, like many natural languages, shows semantic ambiguity, where words and phrases can have multiple meanings depending on context. (see Figure 4).



Necklace	عُقْد
Decade	عِقد
Contract	عَقْد
Held	عَقَدَ
Complicated	عَقَّدَ
Knots	عُقَّدْ

Figure 4. The same word with different meanings.

- Resolving semantic ambiguity poses challenges for tasks such as word sense disambiguation and semantic parsing.

2.2 Opportunities in Arabic NLP

Despite the challenges, Arabic NLP offers numerous opportunities for advancing research and applications in various domains. Some of the key opportunities include:

- **Language Preservation:** Arabic NLP plays a crucial role in preserving and promoting the Arabic language in the digital age by enabling the development of language technologies, educational resources, and cultural preservation initiatives.

- **Multilingual Applications:**

  - Arabic is one of the most widely natural languages in the world, with over 400 million speakers.

  - NLP technologies can facilitate multilingual communication, cross-cultural understanding, and access to Arabic-language content on the Internet.

- **Social and Economic Impact:** Arabic NLP has the potential to drive social and economic development by enabling access to information, empowering communities, and strengthening innovation in sectors such as education, healthcare, e-commerce, and government services.

- **Linguistic Research:** Arabic NLP provides valuable insights into the structure, evolution, and usage patterns of the Arabic language, contributing to linguistic research and language documentation efforts.

Lastly, Arabic NLP presents both challenges and opportunities for researchers, developers, and practitioners. Addressing the unique linguistic characteristics and cultural dimensions of Arabic requires collaborative efforts from academia, industry, and government stakeholders.

By overcoming challenges and capitalizing on opportunities, Arabic NLP can empower individuals, enrich communities, and promote linguistic diversity in the digital era.

2.3 Related Research: Applications of Semantic Search in Religious Texts

Semantic search, a powerful technology in information retrieval, is increasingly finding applications in analyzing and exploring religious texts. This literature review examines the diverse applications of semantic search in religious texts and its implications for scholars, practitioners, and the public.

2.4 Semantic Search in Religious Texts

Semantic search techniques enable a deeper understanding of religious texts by going beyond literal interpretations and uncovering layers of meaning and context. Some key applications include:

- **Contextual Understanding:**
    - Semantic search helps users grasp the contextual nuances of religious texts by analyzing relationships between words, phrases, and concepts.
    - It facilitates the exploration of themes, metaphors, and allegories embedded within religious text.
- **Comparative Analysis:**
    - Semantic search enables scholars to perform comparative analyses across different religious texts, identifying shared themes, motifs, and linguistic patterns.
    - It strengthens dialogue and understanding among diverse religious traditions by highlighting commonalities and differences.
- **Interpretation and Exegesis:**
    - Semantic search aids scholars and theologians in interpreting and exegeting religious texts by providing insights into historical, cultural, and theological contexts.
    - It supports the identification of textual variants, commentaries, and exegetical traditions relevant to the interpretation process.
- **Cross-Referencing and Citation:**
    - Semantic search facilitates cross-referencing and citation of religious texts, allowing users to trace the use of specific verses, concepts, or motifs across multiple scriptures and scholarly works.
    - It enhances the accuracy and reliability of textual citations in academic research and religious discourse.
- **Personalized Study and Reflection:** Semantic search empowers individuals to personalize their study and reflection of religious texts by

offering tailored search results, recommendations, and annotations based on their interests, beliefs, and preferences.

2.5 Implications and Future Directions

The application of semantic search in religious texts holds significant implications for scholarship, education, interfaith dialogue, and spiritual practice. Future research directions may include:

- **Development of Domain-Specific Semantic Models:** Researchers can develop domain-specific semantic models and ontologies tailored to the linguistic and conceptual characteristics of religious texts, enhancing the accuracy and relevance of semantic search results.
- **Integration with Digital Humanities Approaches:** Integration of semantic search with digital humanities approaches such as text mining, network analysis, and visualization can facilitate new insights and discoveries in religious studies and textual scholarship.
- **Ethical and Cultural Considerations:** Ethical and cultural considerations surrounding the use of semantic search in religious texts, including privacy, bias, and interpretation, warrant careful attention and interdisciplinary dialogue.
- **Accessibility and Outreach:** Efforts to make semantic search tools and resources accessible to diverse audiences, including scholars, clergy, educators, and normal people, can promote broader engagement with religious texts and foster informed dialogue and understanding.

Lastly, the application of semantic search in religious texts represents promising boundaries in interdisciplinary research, innovation, and dialogue. By making use of the power of semantic technologies, scholars and practitioners can unlock new dimensions of meaning, insight, and connection within the rich tapestry of religious traditions and scriptures.

As technology continues to evolve, the potential for semantic search to deepen our understanding of religious texts and inspire transformative insights into the human condition remains boundless.

3 Methodology

The methodology chapter of this research highlights the systematic approach that has been taken to develop a semantic search engine for the Holy Quran. This chapter can be considered as a roadmap, a step-by-step guideline process that is used to collect, preprocess, and analyze Quranic text to create a robust search system.

In this chapter, it was necessary to dive into the deep layers of data collection, preprocessing, embedding representation, and semantic similarity calculation. Each subsection of the methodology sheds light on the methods and techniques that have been used to ensure the accuracy, reliability, and relevance of the search engine.

By adhering to a structured methodology, transparency and reproducibility must be provided in that research process. Through data collection and deep analysis, it was mandatory to build a semantic search engine that effectively serves the needs of users seeking to explore the rich and profound meanings of the Holy Quran.

With simplicity and formality as our guiding principles, this chapter established a clear and fundamental approach to the development of the semantic search system.

This methodology chapter lays down a strong groundwork for the upcoming stages of implementation and evaluation. It sets a solid foundation for the subsequent phases of development, ensuring a reliable basis for both implementing and evaluating the semantic search.

3.1 Data Collection: Acquiring Annotated Quran Verses and Topics

The methodology for acquiring annotated Quran verses and topics involves a structured and precise process aimed at ensuring the high accuracy and reliability of the data gathered as the topic is around religion so data collection must be completely correct otherwise it may lead the users to wrong results, and this is not acceptable at all.

This phase of the research is fundamental as it forms the foundation upon which the semantic search engine for the Holy Quran will be built. Below is the outline of the detailed steps involved in this crucial stage.

3.1.1 Identification of Reliable Sources

The initial stage of data collection involves the crucial task of identifying trusted sources for Quranic texts and annotations [10][11].

From this point, it was required to collaborate with experts in religious studies who could provide in-depth explanations, valuable resources, and insightful interpretations of the Quranic verses to ensure access to a comprehensive understanding and good materials essential for our study's success ended up with the good 4 resources as the table below:

Table 1. Shows the in-detail comparison between the collected resources to decide later which of them will be used or a combination of them.

Source	Description
https://quranbysubject.com	it includes Quran verses divided according to the topic, so the user here needs to choose the topic first then it will bring all verses related to that topic but if the user doesn’t know which topic he is looking for it will be very difficult to find what he is looking for

Source	Description
https://surahquran.com/quran-search/Topic-page-1	This resource has the same strategy as the previous one which is topics then verses but this one looks better and richer with topics than the previous one and most probably will end up using this one
https://quranpedia.net/topics	This also has the same approach and a massive amount of data, but it hangs a lot which makes the search process a bit difficult so most probably exclude this resource.
https://modoee.com	This resource includes a lot of extra information that is divided into files, and it will be very difficult to use this resource for data collection so most probably will exclude this one also

3.1.2 Selection Criteria for Annotated Verses

Establishing clear criteria for selecting annotated Quranic verses is essential to maintain consistency and relevance in the dataset. Criteria may include the significance of the verse, the authenticity of the annotation, and the diversity of topics covered.

3.1.3 Compilation of Annotated Verses

Once the selection criteria are defined, annotated Quranic verses are compiled from various sources. Care is taken to ensure that annotations are accurately attributed to their respective scholars or commentators.

3.1.4 Verification and Cross-Referencing

To uphold the integrity of the dataset, a strict verification process is employed to cross-reference annotations with multiple sources that can be found in Table 1.

This helps identify contradictions or inconsistencies and ensures the reliability of the collected data.

3.1.5 Topic Extraction and Categorization

In addition to annotated verses, topics and themes addressed in the Quran are identified and extracted. This involves categorizing verses based on their subject matter, allowing for a structured organization of the data.

3.1.6 Annotation Standardization

An important aspect of the data collection process is standardizing annotations to maintain uniformity and clarity. This may involve reconciling variations in terminology or interpretations across different sources.

3.1.7 Ethical Considerations

Throughout the data collection phase, ethical considerations regarding the use of Quranic texts and annotations are essential. Respect for religious sensitivities and adherence to copyright laws are always upheld.

3.1.8 Documentation and Metadata

Detailed documentation of the collected data, including metadata such as source information, annotation references, and topic classifications, is maintained for transparency and reproducibility.

3.1.9 Quality Assurance

Quality assurance measures, such as peer review and expert consultation, are implemented to ensure the accuracy and completeness of the collected dataset.

3.1.10 Pilot Testing

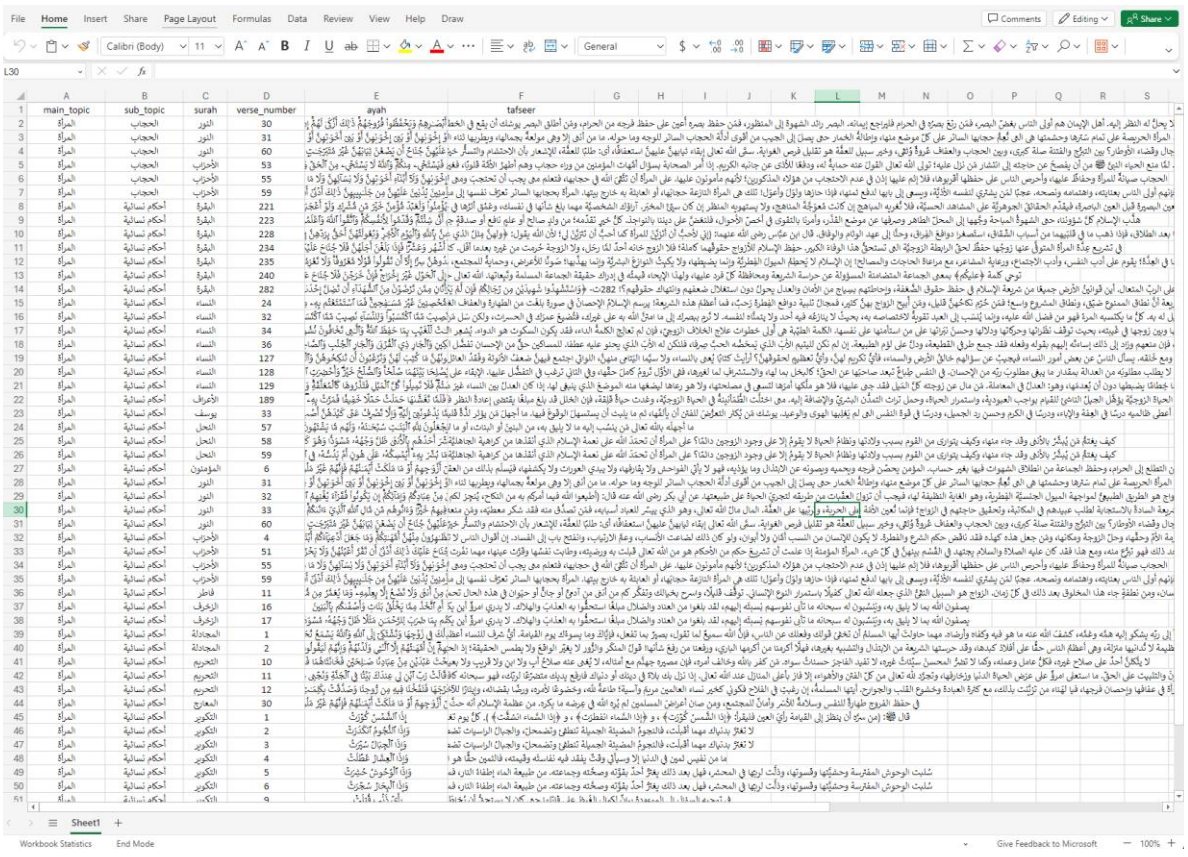
Before proceeding to the development phase of the semantic search engine, pilot testing of the collected data is conducted to identify any potential issues or areas for improvement.

3.1.11 Representation of Data Collection

After considering all the previous steps, the last step was to put the collected data together in a spreadsheet to start the real work. Data was collected to fit into 6 columns as follows:

- Main Topic: for example, **Women's** topic was chosen to start with
- Subtopic: from the main topic, 2 sub-topics were chosen such as **Hijab** and **Women's rulings**
- Surah
- Verse Number
- Ayah ‘Verse’
- Tafseer's “interpretation”

Data was stored in Microsoft Excel (see Figure 5).



	main_topic	sub_topic	surah	verse_number	ayah	tafseer
2	المرأة	الحجاب	النور	30	[illegible]	[illegible]
3	المرأة	الحجاب	النور	31	[illegible]	[illegible]
4	المرأة	الحجاب	النور	60	[illegible]	[illegible]
5	المرأة	الحجاب	الأحزاب	53	[illegible]	[illegible]
6	المرأة	الحجاب	الأحزاب	55	[illegible]	[illegible]
7	المرأة	الحجاب	الأحزاب	59	[illegible]	[illegible]
8	المرأة	أحكام نسائية	البقرة	221	[illegible]	[illegible]
9	المرأة	أحكام نسائية	البقرة	223	[illegible]	[illegible]
10	المرأة	أحكام نسائية	البقرة	228	[illegible]	[illegible]
11	المرأة	أحكام نسائية	البقرة	234	[illegible]	[illegible]
12	المرأة	أحكام نسائية	البقرة	235	[illegible]	[illegible]
13	المرأة	أحكام نسائية	البقرة	240	[illegible]	[illegible]
14	المرأة	أحكام نسائية	البقرة	282	[illegible]	[illegible]
15	المرأة	أحكام نسائية	النساء	24	[illegible]	[illegible]
16	المرأة	أحكام نسائية	النساء	32	[illegible]	[illegible]
17	المرأة	أحكام نسائية	النساء	34	[illegible]	[illegible]
18	المرأة	أحكام نسائية	النساء	36	[illegible]	[illegible]
19	المرأة	أحكام نسائية	النساء	127	[illegible]	[illegible]
20	المرأة	أحكام نسائية	النساء	128	[illegible]	[illegible]
21	المرأة	أحكام نسائية	النساء	129	[illegible]	[illegible]
22	المرأة	أحكام نسائية	الأعراف	189	[illegible]	[illegible]
23	المرأة	أحكام نسائية	يوسف	33	[illegible]	[illegible]
24	المرأة	أحكام نسائية	النحل	57	[illegible]	[illegible]
25	المرأة	أحكام نسائية	النحل	58	[illegible]	[illegible]
26	المرأة	أحكام نسائية	النحل	59	[illegible]	[illegible]
27	المرأة	أحكام نسائية	المؤمنون	6	[illegible]	[illegible]
28	المرأة	أحكام نسائية	النور	31	[illegible]	[illegible]
29	المرأة	أحكام نسائية	النور	32	[illegible]	[illegible]
30	المرأة	أحكام نسائية	النور	33	[illegible]	[illegible]
31	المرأة	أحكام نسائية	النور	60	[illegible]	[illegible]
32	المرأة	أحكام نسائية	الأحزاب	4	[illegible]	[illegible]
33	المرأة	أحكام نسائية	الأحزاب	51	[illegible]	[illegible]
34	المرأة	أحكام نسائية	الأحزاب	55	[illegible]	[illegible]
35	المرأة	أحكام نسائية	الأحزاب	59	[illegible]	[illegible]
36	المرأة	أحكام نسائية	فاطر	11	[illegible]	[illegible]
37	المرأة	أحكام نسائية	الزخرف	16	[illegible]	[illegible]
38	المرأة	أحكام نسائية	الزخرف	17	[illegible]	[illegible]
39	المرأة	أحكام نسائية	المجادلة	1	[illegible]	[illegible]
40	المرأة	أحكام نسائية	المجادلة	2	[illegible]	[illegible]
41	المرأة	أحكام نسائية	التحريم	10	[illegible]	[illegible]
42	المرأة	أحكام نسائية	التحريم	11	[illegible]	[illegible]
43	المرأة	أحكام نسائية	التحريم	12	[illegible]	[illegible]
44	المرأة	أحكام نسائية	المعارج	30	[illegible]	[illegible]
45	المرأة	أحكام نسائية	التكوير	1	[illegible]	[illegible]
46	المرأة	أحكام نسائية	التكوير	2	[illegible]	[illegible]
47	المرأة	أحكام نسائية	التكوير	3	[illegible]	[illegible]
48	المرأة	أحكام نسائية	التكوير	4	[illegible]	[illegible]
49	المرأة	أحكام نسائية	التكوير	5	[illegible]	[illegible]
50	المرأة	أحكام نسائية	التكوير	6	[illegible]	[illegible]
51	المرأة	أحكام نسائية	التكوير	9	[illegible]	[illegible]

Figure 5. Data collection representation.

Lastly, the data collection process for acquiring annotated Quranic verses and topics is a meticulous endeavour guided by clear criteria, ethical considerations, and quality assurance measures. By adhering to these principles, aiming to build a robust foundation for the development of an effective and reliable semantic search engine for the Holy Quran.

3.2 Data Preprocessing: Cleaning, Normalizing, and Tokenizing Arabic Text

Data preprocessing is a fundamental stage in this project to unravel the depths of meaning within the Quranic text. It's like tidying up a workspace before starting a project or tuning an instrument before playing music. This crucial step ensures that the text is in a format that tools can easily handle, allowing it to explore its meanings more effectively.

It's about getting rid of any unnecessary clutter and making sure the text is clear and understandable. Just as an artist prepares their canvas before painting or a musician tunes their instrument before performing, data preprocessing sets the stage for the analysis of the Quranic text. It's all about making sure everything is ready for the exploration of these sacred verses.

3.2.1 Cleaning

Data cleaning is an integral part of preparing the Quranic text for analysis. It's like sorting through a collection of precious gems to ensure only the most flawless ones are selected. In this process, a process such as a meticulous review of each verse, removing any inconsistencies, errors, or irrelevant information should be done.

This ensures that the data that the application will use is accurate, reliable, and free from any distractions that could skew our analysis. By meticulously cleaning the text, the process now can lay a solid foundation for our subsequent exploration, ensuring that every aspect of the Quranic verses is ready for scrutiny and interpretation.

Think of cleaning as tidying up a cluttered room before guests arrive. In the context of the Quranic text, cleaning involves removing any unnecessary elements that might introduce some struggles to the analysis.

What should be done is scrub away extra spaces, punctuation marks, special characters, and diacritics, ensuring that only the pure essence of the text remains. By doing this, a clean and organized foundation is created for further exploration.

In essence, data cleaning is about refining the raw material of the Quranic text to reveal its true essence and significance.

3.2.2 Normalizing

Data normalization is a crucial step in preparing the Quranic text for analysis, it’s like aligning a compass to true north for accurate navigation. This process involves standardizing the format and structure of the text to eliminate inconsistencies and facilitate meaningful comparisons across different verses and chapters.

By normalizing the data, we ensure that each verse follows a consistent pattern, making it easier to identify patterns and relationships within the text. Just as a well-organized library allows for efficient browsing and retrieval of information, data normalization ensures that the analysis tools can effectively process and interpret the Quranic verses.

The goal of data normalization is to reduce redundancy and complexity in the Quranic text, simplifying it for efficient analysis. It's like simplifying a complex mathematical equation to its most basic form for easier calculation. This process involves transforming the text into a standardized representation, where each verse adheres to a set of predefined rules and conventions. This could include converting numerical values to a common scale, standardizing units of measurement, or encoding categorical variables into a consistent format.

By normalizing the data in this way, we ensure that our analysis is based on uniform and comparable information, enabling us to draw meaningful conclusions with confidence.

Moreover, data normalization enhances the accuracy and reliability of our analysis by minimizing the impact of outliers and anomalies in the Quranic text. It's like calibrating a weighing scale to ensure precise measurements every time. By removing inconsistencies and contradictions, a level playing field for analysis is created, where each verse is treated with the same level of scrutiny and attention.

This allows us to focus on the underlying patterns and insights within the text, rather than being distracted by irregularities or outliers. In essence, data normalization is about creating a solid foundation for our analysis, where the integrity and coherence of the Quranic text are preserved and enhanced for deeper exploration.

3.2.3 Tokenizing

Tokenizing is a fundamental process in breaking down the Quranic text into smaller, more manageable units, akin to dissecting a complex puzzle into individual pieces for easier examination. This step involves dividing the text into discrete components, such as words, phrases, or even individual characters, each representing a unique unit of meaning.

By tokenizing the text, we create a structured framework that allows us to analyze and interpret the Quranic verses with greater precision and clarity. Just as assembling a jigsaw puzzle requires sorting its pieces into distinct groups, tokenizing enables us to organize the textual elements systematically, paving the way for deeper insights and understanding.

The goal of tokenizing is to extract meaningful units of information from the Quranic text while preserving their contextual significance and relationships. It's like breaking down a complex recipe into its ingredients to understand the dish's flavor profile. This process involves identifying boundaries between words, punctuation marks, and other linguistic elements, thereby delineating the structure of the text.

By tokenizing the Quranic verses in this manner, we create a roadmap for analysis, where each token serves as a building block for uncovering the text's underlying meaning and nuances. Just as decrypting an encrypted message requires breaking it down into its components, tokenizing enables us to decode the intricate language of the Quran with precision and clarity.

Furthermore, tokenizing enhances the efficiency and effectiveness of our analysis by enabling us to apply computational techniques to the Quranic text. It's like converting a manual task into an automated process for faster and more accurate results.

By tokenizing the text, we create a digital representation that can be processed by algorithms and analytical tools, allowing for systematic exploration and interpretation of the verses. This computational approach opens new avenues for research and discovery, enabling us to uncover hidden patterns, correlations, and insights within the Quranic text that may have remained difficult through manual analysis alone.

In essence, tokenizing empowers us to make use of the full potential of technology in understanding and unravelling the profound wisdom contained within the verses of the Quran.

3.3 Arabic Bert Embeddings: Representing Quran Verses Semantic Meaning

The journey into the realm of Arabic Bert Embeddings reveals a pathway to understanding the depths of Quranic verses with remarkable clarity. Although the concept may seem complex initially, it serves as a vital tool for unlocking the profound wisdom embedded within the holy text.

Imagine it as a guiding light leading us through the intricate maze of the Quran, revealing the hidden truths and profound insights along the way. Just as a lantern through the darkness to reveal hidden paths, Arabic Bert Embeddings illuminate the intricate meanings and nuances of the sacred verses, guiding seekers of knowledge towards deeper understanding and enlightenment.

3.3.1 Understanding Arabic Bert Embeddings

At its core, Arabic Bert Embeddings harness the power of advanced algorithms to transform Quranic verses into a language that computers can decipher.

They're akin to translators, converting the intricate beauty of each verse into mathematical representations. These representations capture the essence, context, and semantic nuances of the verses, providing a gateway to deeper comprehension.

3.3.2 Revealing the Semantic Significance

Each verse in the Quran is a treasure trove of wisdom, layered with multiple meanings and interpretations. Arabic Bert Embeddings act as our tools for diving and understanding these layers, revealing the intricate web of connections and insights hidden within. They enable us to distinguish the subtle nuances of language, context, and symbolism embedded in the verses, offering a richer understanding of their semantic depth.

3.3.3 The Importance of Semantic Analysis

In our pursuit to grasp the profound message of the Quran, it is imperative to delve into the semantic layers of its verses. Arabic Bert Embeddings serve as indispensable tools, empowering us to conduct thorough analyses, uncovering themes, and correlations tangled throughout the text. Through the meticulous exploration of semantic meanings within verses, we not only enhance our understanding but also gain profound insights into their relevance to our lives and the broader world. By decrypting the intricate layers of semantic significance, we unravel the profound wisdom encapsulated within each verse, enriching our spiritual journey, and deepening our connection to the divine message.

3.3.4 Utilizing Arabic Bert Embeddings

Arabic Bert Embeddings [12] serve as versatile tools in our exploration of the Quran. We can employ them to conduct comparative studies, identifying similarities and differences between verses. They also enable us to contextualize verses within the broader narrative of the Quran, shedding light on their significance within the larger framework of the scripture. Additionally, through machine learning techniques, we can leverage these embeddings to uncover novel insights and perspectives, enriching our understanding of the Quran's teachings.

In essence, Arabic Bert Embeddings serve as our companions on a transformative journey through the Quranic landscape. With their guidance, we navigate the complexities of the sacred text, seeking enlightenment and wisdom in its timeless verses. Through their lens, we gain a deeper understanding of the Quran's profound message, enriching our spiritual journey and fostering a deeper connection with the Quranic understanding.

3.4 Semantic Similarity Calculation: Measuring Relevance between Verses and Queries

Within the expansive realm of the Quran, distinguishing between the intricate connections between its verses stands as a fundamental pursuit. Semantic Similarity Calculation serves as a trusted guide, navigating through this maze of wisdom. Despite its seemingly complex name, its essence lies in simplifying the task of assessing the relevance between Quranic verses and queries, akin to a reliable map steering us through unfamiliar terrain.

Through this process, clarity and insight are gained in addition to uncovering the underlying threads that weave together the tapestry of revelation, enriching the understanding and deepening the spiritual journey.

3.4.1 Decrypting Semantic Similarity

At its core, Semantic Similarity Calculation is akin to a bridge connecting two distant shores. It quantifies the degree of similarity between Quranic verses, illuminating the threads that weave them together. Imagine it as a magnifying glass that zooms in on the subtle nuances of language, context, and meaning embedded within each verse.

3.4.2 The Quest for Relevance

In the journey through the Quran, relevance is the guiding star. Seek to uncover the connections between verses and queries, discerning patterns and insights that transcend mere words. Semantic Similarity Calculation serves as the beacon, illuminating the path to understanding by quantifying the relevance of verses to specific queries or themes.

3.4.3 Measuring the Magnitude of Meaning

Every verse in the Quran is a tank of meaning, rich with layers of interpretation and significance. Semantic Similarity Calculation empowers us to measure the magnitude of this meaning, quantifying the depth of connection between verses and queries. It allows us to gauge the relevance of each verse within the broader context of the Quran, shedding light on its significance and implications.

3.4.4 Navigating the Quranic Landscape

Armed with Semantic Similarity Calculation, navigating the vast landscape of the Quran with confidence and clarity. Using it to traverse the intricate web of connections between verses, decoupling the tapestry of wisdom throughout the scripture. Whether exploring themes of mercy, justice, or guidance, Semantic Similarity Calculation serves as the trusted guide, helping to distinguish between the relevance of each verse to the quest for understanding.

3.4.5 Unlocking Insights and Perspectives

Beyond its utility in measuring relevance, Semantic Similarity Calculation unlocks a treasure trove of insights and perspectives. It enables to identification of recurring themes, and patterns across the Quran, offering new ways for exploration and interpretation. Quantifying the connections between verses and queries enriches our understanding of the Quran's timeless message and its relevance to contemporary issues and challenges.

In essence, Semantic Similarity Calculation stands as a cornerstone of our journey through the Quran. It empowers us to navigate the complexities of the scripture with precision and insight, guiding us towards a deeper understanding of its profound teachings. With its assistance, we embark on a transformative quest for knowledge and enlightenment, seeking wisdom in the timeless verses of the Holy Quran.

4 Implementation

In the implementation chapter, the focus moves from planning and preparation to action. Here, ideas become reality as concepts are transformed into concrete outcomes. This section serves as a guide to the practical application of this project, providing a roadmap for turning theory into practice.

4.1 Data Preprocessing

Data preprocessing is like laying the groundwork for our project, the Semantic Search Engine for the Holy Quran. It's the first step where we clean up and organize the Quranic data to make it easier to understand and analyze. Think of it as tidying up before starting a big project – we want everything to be neat and organized so that we can find what we need quickly.

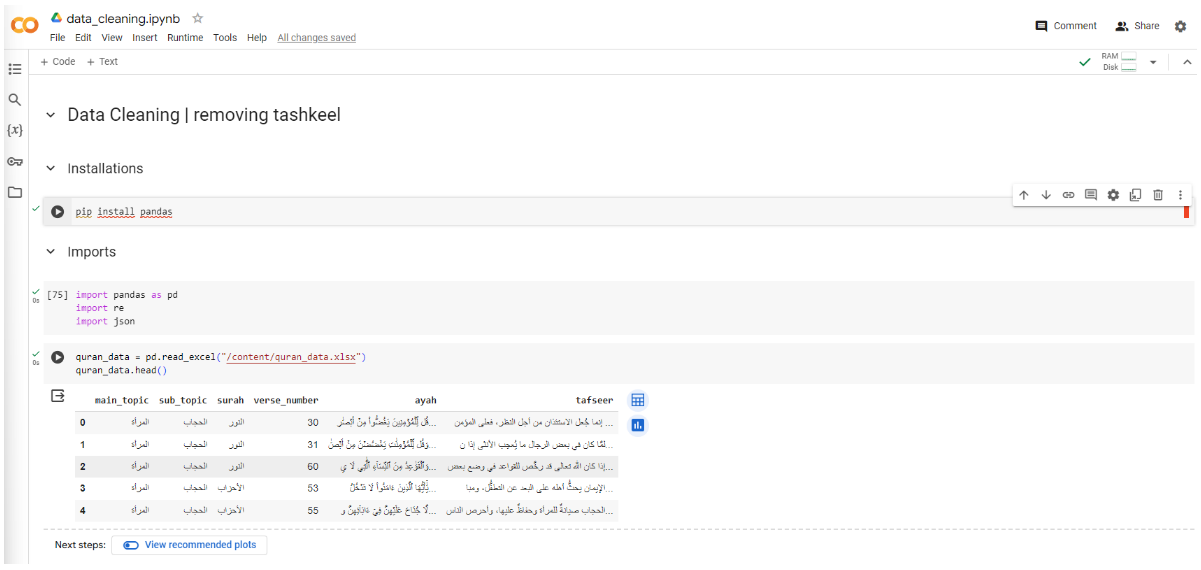
During data preprocessing, we go through the Quranic text, making sure everything is in the right format and easy to work with. We clean up any messy bits, standardize things, and break down the text into smaller parts, kind of like breaking a big task into smaller steps to make it more manageable. This way, when we move on to the next stages of our project, everything is set up and ready to go, making it smoother and more efficient.

4.1.1 Data Cleaning

Think of cleaning as tidying up a cluttered room before guests arrive. In the context of our Quranic text, cleaning involves removing any unnecessary elements that might hinder our analysis. We scrub away extra spaces, punctuation marks, special characters, and diacritics, ensuring that only the pure essence of the text remains. By doing this, we create a clean and organized foundation for further exploration.

From this point, the first task for this project is to remove Arabic diacritics for the Quranic text using **Python** and **Jupyter Notebook** and then to store the whole results in a JSON format file.

The first step was to install and import the required packages and then to explore the data to validate the data versus expectations as shown in the figure below (see Figure 6).



Data Cleaning | removing tashkeel

Installations

```
pip install pandas
```

Imports

```
[75] import pandas as pd
     import re
     import json
```

```
quran_data = pd.read_excel("/content/quran_data.xlsx")
quran_data.head()
```

	main_topic	sub_topic	surah	verse_number	ayah	tafseer
0	المرأة	الحجاب	النور	30	قُل لِّلْمُؤْمِنِينَ يَغُضُّوا۟ مِنْ أَبْصَٰرِ...	... إنما جُعل الاستئذان من أجل النظر، فعلى المؤمن
1	المرأة	الحجاب	النور	31	وَقُل لِّلْمُؤْمِنَٰتِ يَغْضُضْنَ مِنْ أَبْصَٰ...	لمّا كان في بعض الرجال ما يُعجب الأنثى إذا ن...
2	المرأة	الحجاب	النور	60	وَٱلْقَوَٰعِدُ مِنَ ٱلنِّسَآءِ ٱلَّٰتِى لَا ي...	إذا كان الله تعالى قد رخّص للقواعد في وضع بعض...
3	المرأة	الحجاب	الأحزاب	53	يَٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوا۟ لَا تَدْخُلُ...	الإيمان يحثُّ أهله على البعد عن التطفّل، ومبا...
4	المرأة	الحجاب	الأحزاب	55	لَّا جُنَاحَ عَلَيْهِنَّ فِىٓ ءَابَآئِهِنَّ و...	الحجاب صيانةٌ للمرأة وحفاظٌ عليها، وأحرص الناس...

Figure 6. Data cleaning preparation.

Step two is to define the function (Cleaning 1) to remove diacritics that should remove the diacritic from the Quranic text.

```
def remove_tashkeel(text):
    # Define a pattern to match Arabic diacritics and other related characters
    pattern = re.compile(r'[\u0617-\u061A\u064B-\u0652\u0657-
\u065F\u0670\u06D6-\u06ED]')
    # Replace diacritics with empty string
    text = re.sub(pattern, '', text)
    # Replace extra letters
    text = text.replace('ا' ,'ٱ')
    return text
```

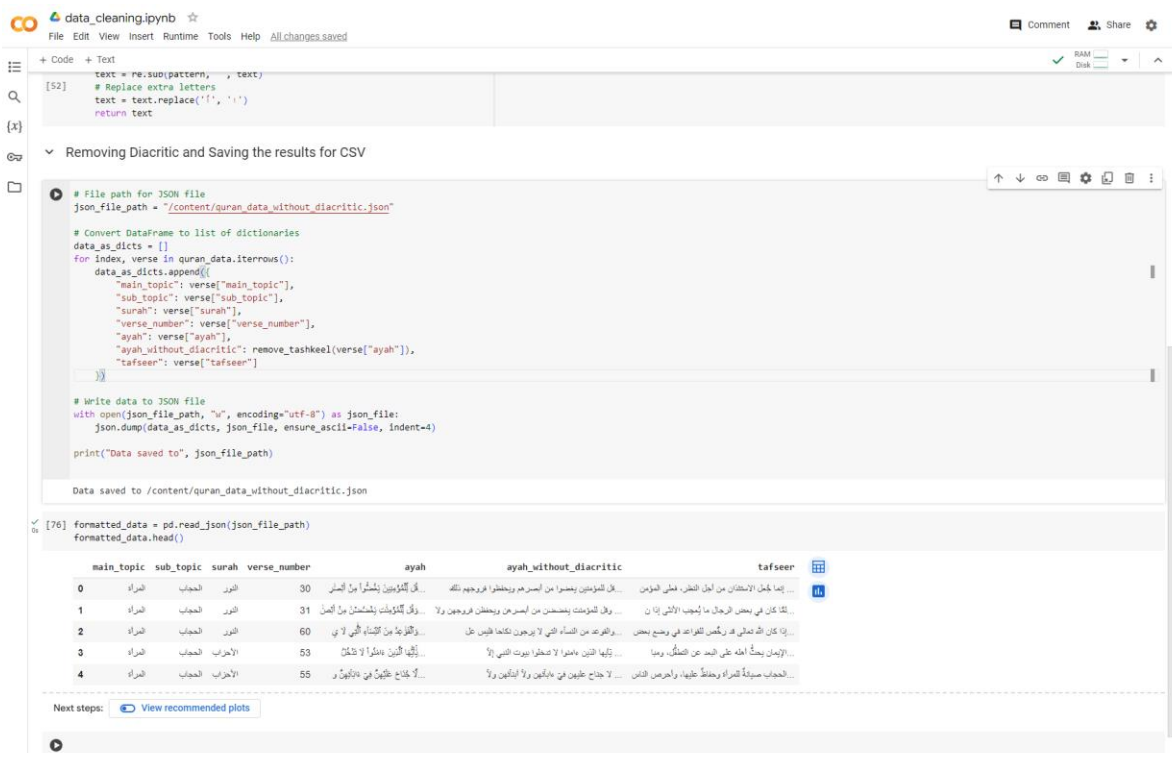
Cleaning 1. A Python function that removes diacritic from the text and returns the text after cleaning.

Step two is to loop throw the entire dataset, clean, and then export the data with an extra column **verse_without_diacritic** (Cleaning 2).

```
# File path for JSON file
json_file_path = "/content/quran_data_without_diacritic.json"

# Convert DataFrame to list of dictionaries
data_as_dicts = []
for index, verse in quran_data.iterrows():
    data_as_dicts.append({
        "main_topic": verse["main_topic"],
        "sub_topic": verse["sub_topic"],
        "surah": verse["surah"],
        "verse_number": verse["verse_number"],
        "ayah": verse["ayah"],
        "ayah_without_diacritic": remove_tashkeel(verse["ayah"]),
        "tafseer": verse["tafseer"]
    })

# Write data to JSON file
with open(json_file_path, "w", encoding="utf-8") as json_file:
    json.dump(data_as_dicts, json_file, ensure_ascii=False, indent=4)

print("Data saved to", json_file_path)
```

Cleaning 2. A Python function that removes diacritic from the text and returns the text after cleaning.

The last step in the cleaning process is to print and validate the output (see Figure 7).



```
[52]        text = re.sub(pattern, ' ', text)
            # Replace extra letters
            text = text.replace('ٱ', 'ا')
            return text
```

Removing Diacritic and Saving the results for CSV

```
# File path for JSON file
json_file_path = "/content/quran_data_without_diacritic.json"

# Convert DataFrame to list of dictionaries
data_as_dicts = []
for index, verse in quran_data.iterrows():
    data_as_dicts.append({
        "main_topic": verse["main_topic"],
        "sub_topic": verse["sub_topic"],
        "surah": verse["surah"],
        "verse_number": verse["verse_number"],
        "ayah": verse["ayah"],
        "ayah_without_diacritic": remove_tashkeel(verse["ayah"]),
        "tafseer": verse["tafseer"]
    })

# Write data to JSON file
with open(json_file_path, "w", encoding="utf-8") as json_file:
    json.dump(data_as_dicts, json_file, ensure_ascii=False, indent=4)

print("Data saved to", json_file_path)
```

```
Data saved to /content/quran_data_without_diacritic.json
```

```
[76] formatted_data = pd.read_json(json_file_path)
     formatted_data.head()
```

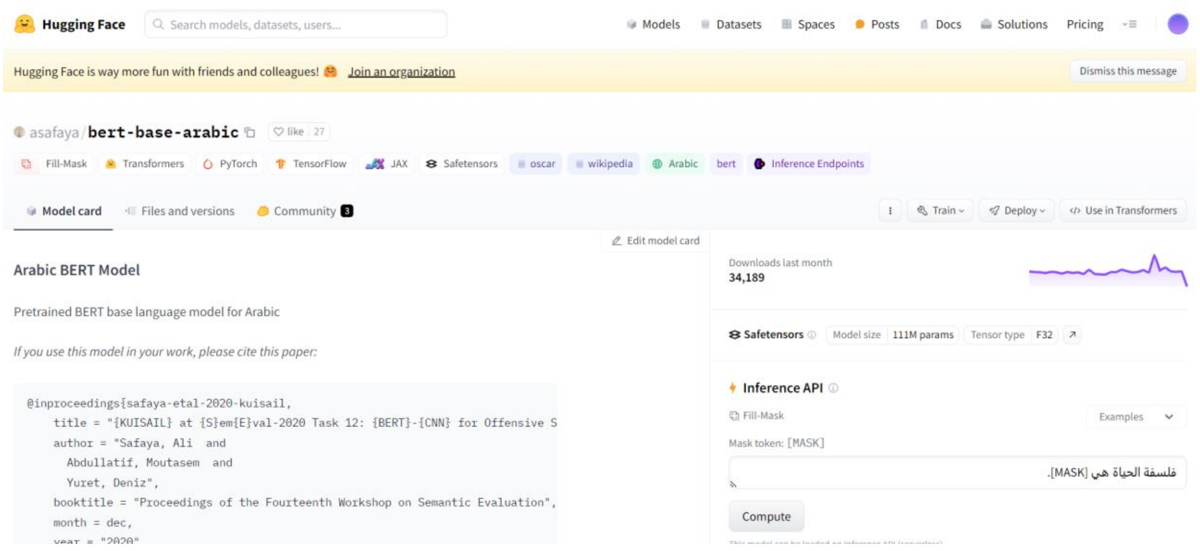
	main_topic	sub_topic	surah	verse_number	ayah	ayah_without_diacritic	tafseer
0	المرأة	الحجاب	النور	30	قُل لِّلْمُؤْمِنِينَ يَغُضُّوا۟ مِنْ أَبْصَٰرِ...	قل للمؤمنين يغضوا من أبصرهم ويحفظوا فروجهم ذلك...	إنما جُعل الاستئذان من أجل النظر، فعلى المؤمن ...
1	المرأة	الحجاب	النور	31	وَقُل لِّلْمُؤْمِنَٰتِ يَغْضُضْنَ مِنْ أَبْصَٰ...	وقل للمؤمنت يغضضن من أبصرهن ويحفظن فروجهن ولا ...	لمّا كان في بعض الرجال ما يُعجب الأنثى إذا ن...
2	المرأة	الحجاب	النور	60	وَٱلْقَوَٰعِدُ مِنَ ٱلنِّسَآءِ ٱلَّٰتِى لَا يَ...	والقوعد من النسآء التي لا يرجون نكاحا فليس عل...	إذا كان الله تعالى قد رخّص للقواعد في وضع بعض...
3	المرأة	الحجاب	الأحزاب	53	يَٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوا۟ لَا تَدْخُلُ...	يأيها الذين ءامنوا لا تدخلوا بيوت النبي إلا ...	الإيمان يحثُّ أهله على البعد عن التطفُّل، ومما...
4	المرأة	الحجاب	الأحزاب	55	لَّا جُنَاحَ عَلَيْهِنَّ فِىٓ ءَابَآئِهِنَّ وَ...	لا جناح عليهن فى ءابآئهن ولا أبنآئهن ولا ...	الحجاب صيانةٌ للمرأة وحفاظٌ عليها، واحرص الناس...

Figure 7. Data cleaning validation.

4.2 Experiment 1: Using a pre-trained model

The first idea that came to mind was to use a pre-trained Arabic BERT model and BERT is one of the language model types based on the BERT architecture, which was trained on Arabic data. BERT models use a transformer-based [13] architecture and are trained using a technique called masked language modelling, which helps the model understand the context and semantics of words in a sentence.

From Hugging Face, the first try was to use the **asafaya/bert-base-arabic** [14] model (see Figure 8).



Arabic BERT Model

Pretrained BERT base language model for Arabic

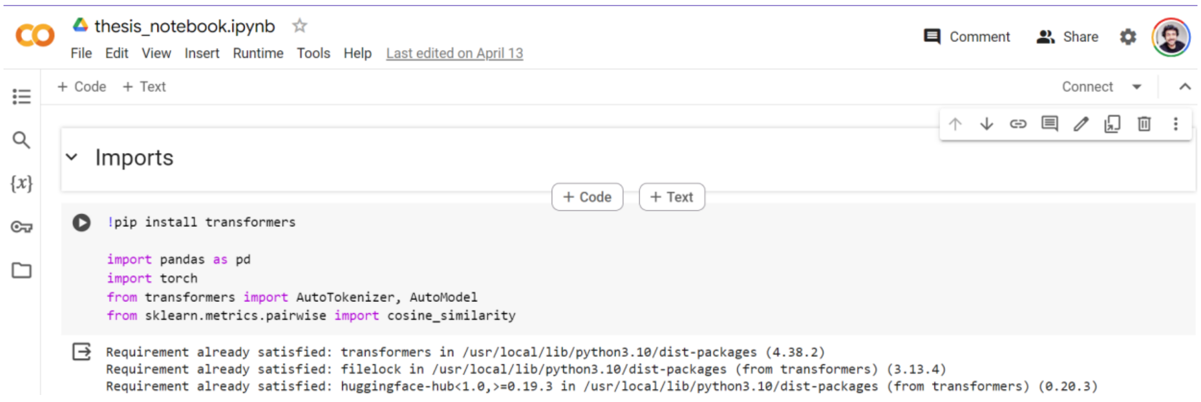
If you use this model in your work, please cite this paper:

```
@inproceedings{safaya-etal-2020-kuisail,
    title = "{KUISAIL} at {S}em{E}val-2020 Task 12: {BERT}-{CNN} for Offensive S
    author = "Safaya, Ali  and
      Abdullatif, Moutasem  and
      Yuret, Deniz",
    booktitle = "Proceedings of the Fourteenth Workshop on Semantic Evaluation",
    month = dec,
    year = "2020"
```

Figure 8. asafaya/bert-base-arabic model.

This model is very popular when you think about the Arabic pre-trained models, it has a model size of 111M params and was downloaded thousands of times during that month so that was a good indicator to start with

Back to Jupyter notebooks to check experiment 1, the are some packages that are essential for this experiment so that would be step 1 is to install and import the main packages which are Transformers, Pandas, and scikit-learn (see Figure 9).

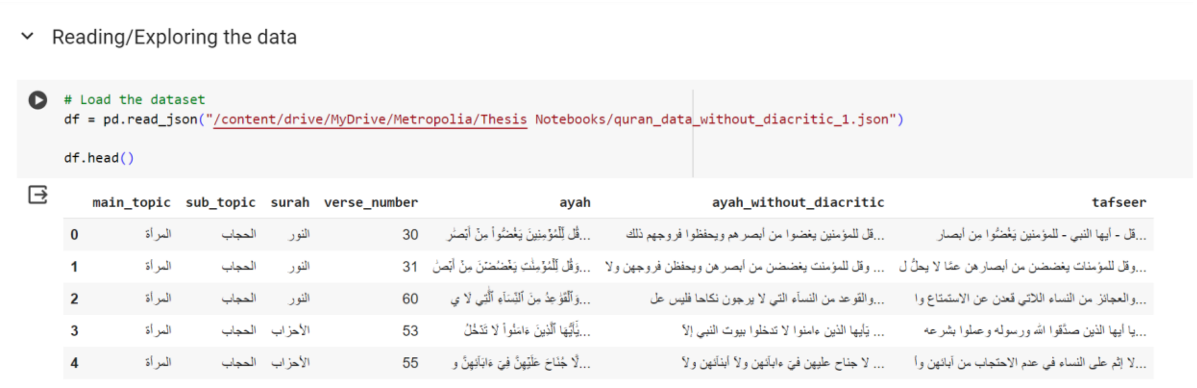


```
!pip install transformers

import pandas as pd
import torch
from transformers import AutoTokenizer, AutoModel
from sklearn.metrics.pairwise import cosine_similarity
```

```
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.38.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.13.4)
Requirement already satisfied: huggingface-hub<1.0,>=0.19.3 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.20.3)
```

Figure 9. Experiment 1 main imports.

Step 2 is to Load/Check the data and make sure that the data format is as expected (see Figure 10).

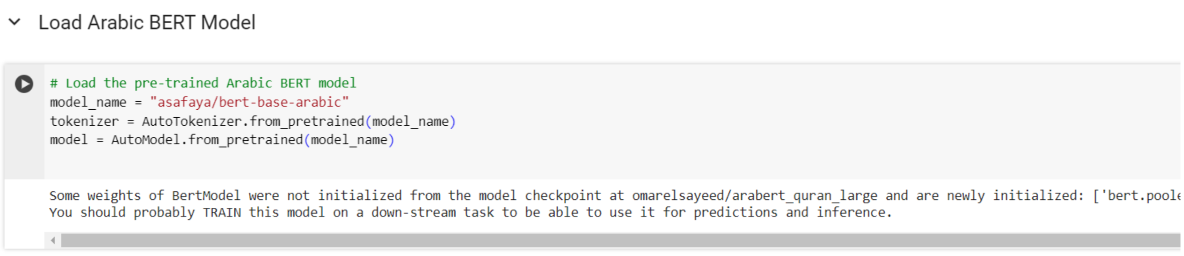

Reading/Exploring the data

```
# Load the dataset
df = pd.read_json("/content/drive/MyDrive/Metropolia/Thesis Notebooks/quran_data_without_diacritic_1.json")

df.head()
```

	main_topic	sub_topic	surah	verse_number	ayah	ayah_without_diacritic	tafseer
0	المرأة	الحجاب	النور	30	قُل لِّلْمُؤْمِنِينَ يَغُضُّوا۟ مِنْ أَبْصَٰرِ...	قل للمؤمنين يغضوا من أبصرهم ويحفظوا فروجهم ذلك...	قل - أيها النبي - للمؤمنين يَغُضُّوا مِن أبصار...
1	المرأة	الحجاب	النور	31	وَقُل لِّلْمُؤْمِنَٰتِ يَغْضُضْنَ مِنْ أَبْصَٰ...	وقل للمؤمنت يغضضن من أبصرهن ويحفظن فروجهن ولا ...	وقل للمؤمنات يغضضن من أبصارهن عمّا لا يحلُّ ل...
2	المرأة	الحجاب	النور	60	وَٱلْقَوَٰعِدُ مِنَ ٱلنِّسَآءِ ٱلَّٰتِى لَا ي...	والقوعد من النسآء التي لا يرجون نكاحا فليس عل...	والعجائز من النساء اللاتي قعدن عن الاستمتاع وا...
3	المرأة	الحجاب	الأحزاب	53	يَٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوا۟ لَا تَدْخُلُ...	يأيها الذين ءامنوا لا تدخلوا بيوت النبي إلا ...	يا أيها الذين صدّقوا الله ورسوله وعملوا بشرعه...
4	المرأة	الحجاب	الأحزاب	55	لَّا جُنَاحَ عَلَيْهِنَّ فِىٓ ءَابَآئِهِنَّ و...	لا جناح عليهن فيٓ ءابآئهن ولآ أبنآئهن ولا ...	لا إثم على النساء في عدم الاحتجاب من آبائهن وا...

Figure 10. Data loading/checking.

Step 3 is to load the model and to make sure that it works with no problem (see Figure 11).


Load Arabic BERT Model

```
# Load the pre-trained Arabic BERT model
model_name = "asafaya/bert-base-arabic"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModel.from_pretrained(model_name)
```

```
Some weights of BertModel were not initialized from the model checkpoint at omarelsayeed/arabert_quran_large and are newly initialized: ['bert.poole
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

Figure 11. Loading the Arabic BERT model.

Step 4 and the most important one is to define the semantic search function that will take the user query and then return similar matches from the annotated data with the score of the cosine similarity (Similarity 1).

```
def semantic_search(query, df, model, tokenizer, threshold=0.5,
num_results=3):
    # Tokenize the query
    encoded_query = tokenizer(query, return_tensors="pt", padding=True,
truncation=True)

    # Forward pass through the model
    with torch.no_grad():
        outputs = model(**encoded_query)
        query_embedding = outputs.last_hidden_state.mean(dim=1).squeeze()

    # Encode all Tafseer texts
    tafseer_embeddings = []
    for tafseer_text in df['tafseer']:
        encoded_tafseer = tokenizer(tafseer_text, return_tensors="pt",
padding=True, truncation=True)
        with torch.no_grad():
            outputs = model(**encoded_tafseer)
            tafseer_embedding =
outputs.last_hidden_state.mean(dim=1).squeeze().detach().cpu().numpy()
            tafseer_embeddings.append(tafseer_embedding)

    tafseer_embeddings = torch.tensor(tafseer_embeddings)

    # Calculate cosine similarity between query and Tafseer embeddings
    similarities = cosine_similarity(query_embedding.unsqueeze(0),
tafseer_embeddings)

    # Find indices of results above the threshold
    result_list = []

    # Iterate over similarity scores and indices
    for index, score in enumerate(similarities[0]):
        if score > threshold:
            result_list.append({"index": index, "score": score})

    # Sort indices by similarity score
    sorted_result_list = sorted(result_list, key=lambda x: x["score"],
reverse=True)

    return sorted_result_list[:num_results]
```

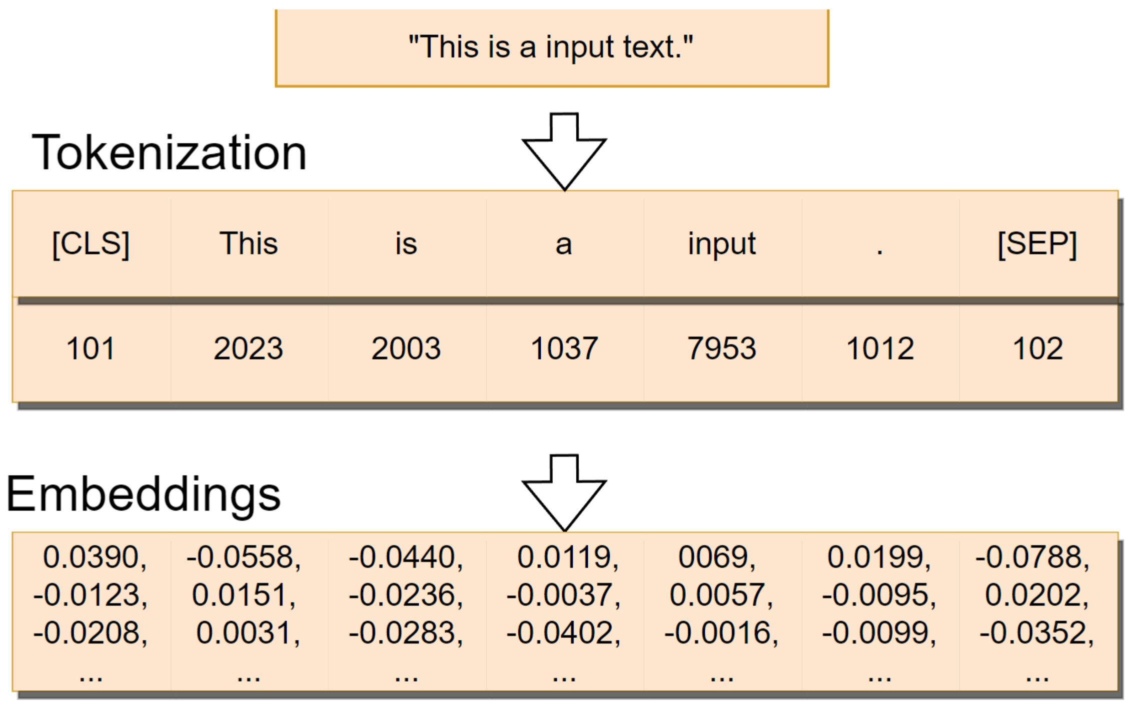
Similarity 1. A function that takes user query and returns similarly relevant results.

And what this function does is perform semantic search calculations to find the most similar text in an annotated data frame related to the user query using a pre-trained model and tokenizer, below is an elaboration of each part.

- **Function Inputs**
  - User query: the input that the user needs to get verses related to
  - df: the annotated data frame to use during the cosine similarity [15] calculations.
  - model: the pre-trained Arabic model to use as a main component.

  - tokenizer: this is a tool that converts the text into a specific format [16][17] that can be used by a language model.
  - Threshold: this is the minimum score for the cosine similarity to be considered
  - num_results: how many top scores above the threshold need to be listed.
- **Query Tokenization:**
  - It divides the text into tokens or parts, such as words or subwords, and converts them into numerical representations (token IDs) based on the model's vocabulary. The tokenizer can also handle other tasks like adding special tokens, padding, truncation, and creating attention masks, preparing the text for model input.
- **Compute Query Embedding:**
  - Then we take the tokenized query and pass it through the model in a forward pass process using **torch.no_grad()** to avoid updating model weights and then to extract the query embedding from the model's output.
- **Encode the Tafseer column.**
  - This step is the same as the previous one but for the Tafseer “Quranic interpretation” column in the annotated data frame.
- **Calculate similarities.**
  - In this step we Use the **cosine_similarity** function to compute the similarity between the query embedding and each text embedding. The function returns a similarity score for each pair.
- **Filter, Ranking, and returning the top results.**

More elaboration on how query tokenization works (see Figure 12).



[CLS]	This	is	a	input	.	[SEP]
101	2023	2003	1037	7953	1012	102
0.0390, -0.0123, -0.0208, ...	-0.0558, 0.0151, 0.0031, ...	-0.0440, -0.0236, -0.0283, ...	0.0119, -0.0037, -0.0402, ...	0069, 0.0057, -0.0016, ...	0.0199, -0.0095, -0.0099, ...	-0.0788, 0.0202, -0.0352, ...

Figure 12. Tokenization Process.

More elaboration on how **Cosine Similarity** works (see Figure 13).

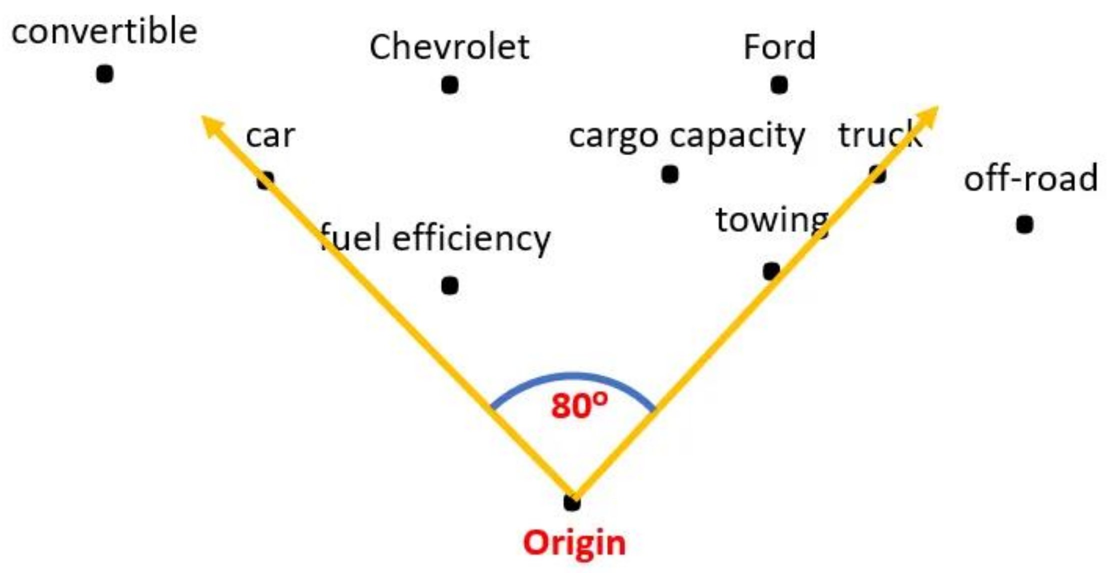
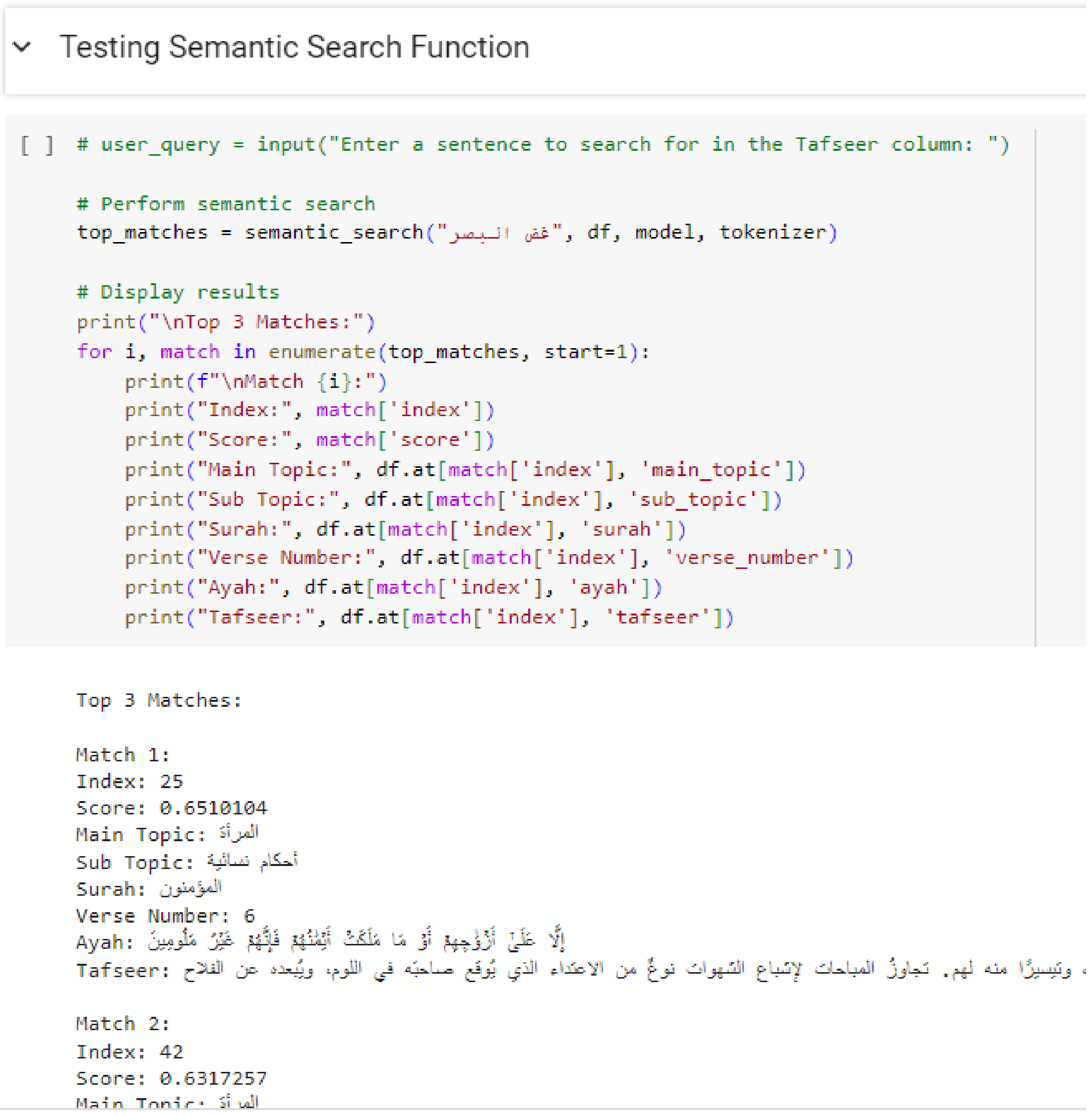


Figure 13. cosine similarity visualization.

The last step in the cleaning process is to print and validate the output (see Figure 14).



Testing Semantic Search Function

```
[ ] # user_query = input("Enter a sentence to search for in the Tafseer column: ")

    # Perform semantic search
    top_matches = semantic_search("غض البصر", df, model, tokenizer)

    # Display results
    print("\nTop 3 Matches:")
    for i, match in enumerate(top_matches, start=1):
        print(f"\nMatch {i}:")
        print("Index:", match['index'])
        print("Score:", match['score'])
        print("Main Topic:", df.at[match['index'], 'main_topic'])
        print("Sub Topic:", df.at[match['index'], 'sub_topic'])
        print("Surah:", df.at[match['index'], 'surah'])
        print("Verse Number:", df.at[match['index'], 'verse_number'])
        print("Ayah:", df.at[match['index'], 'ayah'])
        print("Tafseer:", df.at[match['index'], 'tafseer'])
```

```
Top 3 Matches:

Match 1:
Index: 25
Score: 0.6510104
Main Topic: المرأة
Sub Topic: أحكام نسائية
Surah: المؤمنون
Verse Number: 6
Ayah: إِلَّا عَلَىٰٓ أَزْوَٰجِهِمْ أَوْ مَا مَلَكَتْ أَيْمَٰنُهُمْ فَإِنَّهُمْ غَيْرُ مَلُومِينَ
Tafseer: ، وتيسيرًا منه لهم. تجاوزُ المباحات لإتباع الشهوات نوعٌ من الاعتداء الذي يُوقع صاحبه في اللوم، ويُبعده عن الفلاح

Match 2:
Index: 42
Score: 0.6317257
Main Topic: المرأة
```

Figure 14. Semantic similarity validation.

4.3 Analyzing Experiment 1

After running the previous code that returned the high-scored rows from the annotated data and starting to analyze the results with eyes something was interesting to mention.

However, the scores look good to consider but the results weren't good enough to consider because this is a sensitive topic related to religion and must be in high quality then from this point I suspected 3 main reasons behind the issue.

- **Quality of the embeddings**
    - The quality of the embeddings generated by the model might not capture the semantics of the text accurately, leading to high similarity scores that don't translate to meaningful results.
- **Overfitting or Underfitting**
    - The model may overfit or underfit its training data
- **Contextual understanding**
    - The model may not have good training data that covers the Quranic interpretation terms.

The last try for experiment 1 was to use another model from the hugging Face platform and instead of using **asafaya/bert-base-arabic** I tried **omarelsayeed/arabert_quran_large** [18] and honestly, it had better results but still not so good to consider.

Evaluating experiment 1 and the upcoming experiments will be a human evaluation in the early steps then this evaluation should follow the same measurements and guidelines as the base threshold will be 0.6 and 3 randomly queries will be tested such as:

1. Keeping the eye downward
2. Hijab in Islam
3. The final day

Table 2. Experiment 1 evaluation.

Query	Top 3 cosine similarity Scores	Evaluation
Keeping the eye downward	0.58, 0.56, and 0.55	All results all far away of the meaning of the query

Query	Top 3 cosine similarity Scores	Evaluation
Hijab in Islam	0.64, 0.62, and 0.61	The first result only is good, others are away from the target
The final day	0.61, 0.58, and 0.85	All results are away of the query

So, this approach achieved 1 correct out of 9 for the 3 queries.

The previous metrics were for 1 person but what about surveying another 5 people to have better metrics? Collected 10 questions for 5 different people as follows:

1. How Allah proved his existence
2. Why did Allah create hell?
3. What is death?
4. Why should women wear hijab?
5. What will happen after the death?
6. What kind of torture happens in hell?
7. Why hijab is mandatory?
8. What are the profit Muhammed characteristics?
9. What are the death signs?
10. Why should people thank Allah?

Table 3. Experiment 1 evaluation against surveying 5 different persons.

Query	Top 3 cosine similarity Scores	Evaluation
How Allah proved his existence	0.65, 0.64, and 0.64	All results are away of the query
Why did Allah create hell?	0.64, 0.63, and 0.63	All results are away of the query
What is death?	0.62, 0.62, and 0.60	All results are away of the query

Query	Top 3 cosine similarity Scores	Evaluation
Why should women wear hijab?	0.71, 0.71, and 0.68	All results are away of the query
What will happen after the death?	0.65, 0.64, and 0.64	All results are away of the query
What kind of torture happens in hell?	0.64, 0.64, and 0.62	All results are away of the query
Why hijab is mandatory?	0.59, 0.59, and 0.57	The last result is very good
What are the profit Muhammed characteristics?	0.56, 0.54, and 0.53	All results are away of the query
What are the death signs?	0.56, 0.56, and 0.56	All results are away of the query
Why should people thank Allah?	0.7, 0.69, and 0.69	All results are away of the query

So, this approach achieved 1 correct out of 30 for the 10 queries.

4.4 Experiment 2: Topic modelling

Topic modelling [19] is a text analysis technique that is used to discover hidden trends or patterns within a collection of documents. The main idea is to cluster similar words together into topics by organizing and summarizing large sets of textual data. It allows users to explore the data and understand the main topic/pattern within a lot of documents. It is useful for summarizing, organizing, and analyzing unstructured text data in various fields, such as social media analysis, research, and content categorization. More elaboration on how **Topic Modelling** works (see Figure 15).

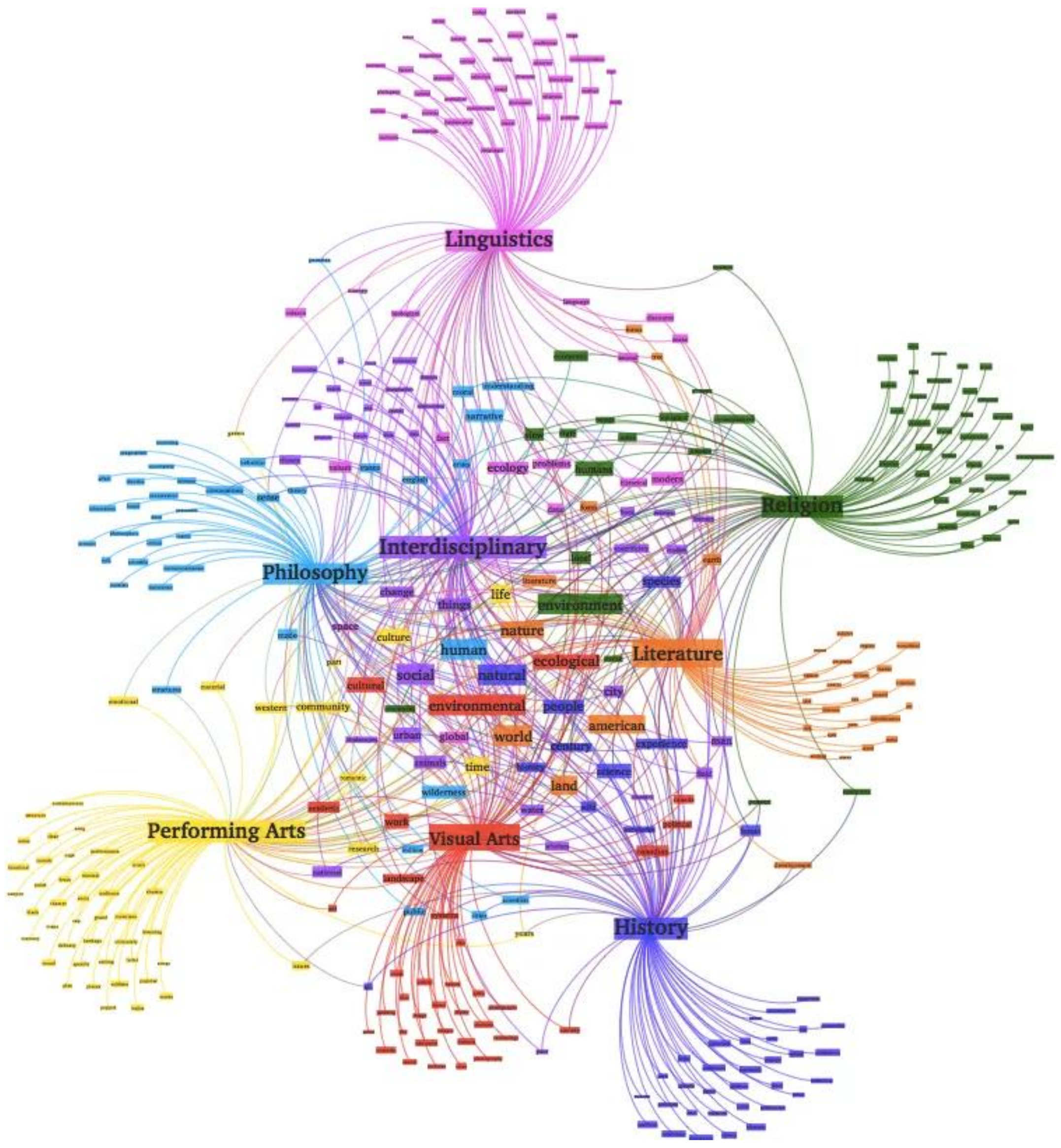


Figure 15. Topic modeling visualization.

For **experiment 2**, a technique called **LDA** [20][21] will be used. It is a statistical method used in natural language processing and text analysis to identify the underlying topics in a collection of documents. It assumes that each document is a mixture of topics, and each topic is a mixture of words. The method is based on a probabilistic model that represents documents as distributions of topics and topics as distributions of words.

In LDA, each document is associated with a set of topic proportions, which define the likelihood that the document contains content related to each topic. Similarly, each topic is associated with a distribution of words, which indicates the probability of each word occurring in that topic. By learning these distributions from a dataset, LDA can discover the hidden themes in a set of documents and organize them according to these topics. LDA is widely used for tasks such as topic modeling, text summarization, and information retrieval.

For this approach, some packages are required such as sklearn, pandas, NumPy, re. Below is the implemented code for experiment 2 (Similarity 2)

```
# Import necessary libraries
...

# Load Quranic verses data
...

# Vectorize the text data
...

# Initialize and fit LDA model
...

# Retrieve the topics and associated words
def get_topics(model, feature_names, num_words=10):
    topics = {}
    for idx, topic in enumerate(model.components_):
        topics[idx] = [feature_names[i] for i in topic.argsort()[:-num_words -
1:-1]]
    return topics

# Display topics and associated words
feature_names = vectorizer.get_feature_names_out()
topics = get_topics(lda, feature_names)
for topic, words in topics.items():
    print(f"Topic {topic}: {', '.join(words)}")

# Associate verses with topics based on topic distributions
topic_distributions = lda.transform(X)
df['topic_distribution'] = list(topic_distributions)

# Perform semantic search with topics
def semantic_search(query, df, model, tokenizer, threshold=0.5,
num_results=3):
    # Vectorize the query
    query_vector = vectorizer.transform([query])
    # Transform the query vector using the LDA model
    query_topic_distribution = lda.transform(query_vector)
    # Calculate cosine similarity between query topic distribution and verses'
topic distributions
    similarities = cosine_similarity(query_topic_distribution,
df['topic_distribution'].tolist())
    # Retrieve indices of results above the threshold
    above_threshold_indices = np.where(similarities > threshold)[1]
    # Sort indices by similarity score
    sorted_indices = above_threshold_indices[np.argsort(-similarities[0,
above_threshold_indices])]
    # Get top N results
    top_indices = sorted_indices[:num_results]
    # Retrieve top matches
    top_matches = df.iloc[top_indices]
    return top_matches
```

Similarity 2. Experiment 2

In a nutshell, this code performs topic modeling on Quranic verses data using **LDA** and then performs a semantic search based on topics. Let’s elaborate on what this code does.

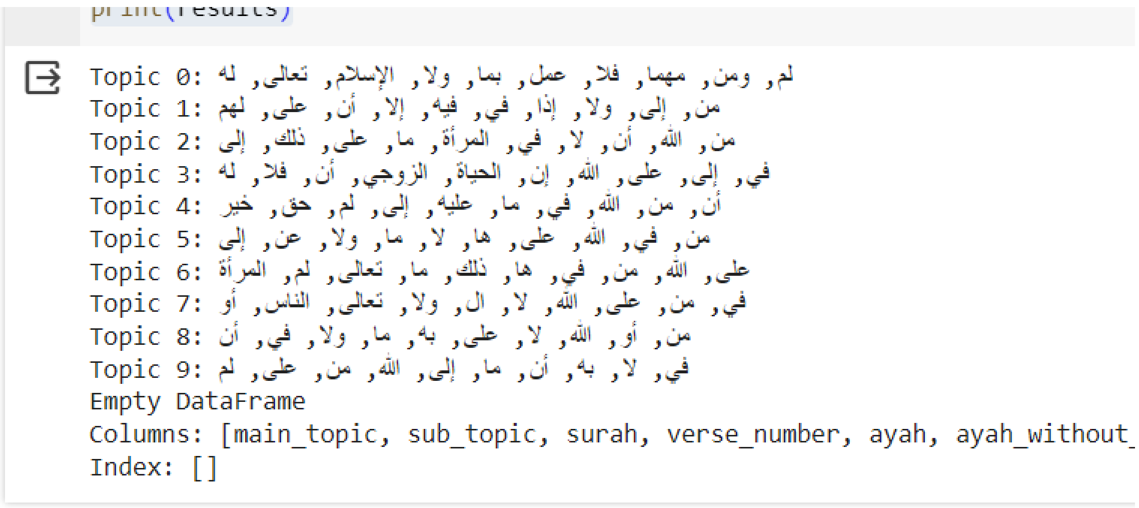
- Importing required packages

- Loading Quranic annotated data
- Vectorize the data
- Initialize and fit LDA modal.
- Retrieve the topics and associated words.
- Associate verses with topics
- Perform semantic search within the topics.
- Print the final results.

4.5 Analyzing Experiment 2

This approach wasn’t promising at all; however, it should be powerful for such a task but sounds like it requires more experience to analyze and use this technique wisely.

It couldn’t find patterns as expected and returned an empty list of relevant verses (see Figure 16).



```
print(results)
```

```
Topic 0: لم, ومن, مهما, فلا, عمل, بما, ولا, الإسلام, تعالى, له
Topic 1: من, إلى, ولا, إذا, في, فيه, إلا, أن, على, لهم
Topic 2: من, الله, أن, لا, في, المرأة, ما, على, ذلك, إلى
Topic 3: في, إلى, على, الله, إن, الحياة, الزوجي, أن, فلا, له
Topic 4: أن, من, الله, في, ما, عليه, إلى, لم, حق, خير
Topic 5: من, في, الله, على, ها, لا, ما, ولا, عن, إلى
Topic 6: على, الله, من, في, ها, ذلك, ما, تعالى, لم, المرأة
Topic 7: في, من, على, الله, لا, ال, ولا, تعالى, الناس, أو
Topic 8: من, أو, الله, لا, على, به, ما, ولا, في, أن
Topic 9: في, لا, به, أن, ما, إلى, الله, من, على, لم
Empty DataFrame
Columns: [main_topic, sub_topic, surah, verse_number, ayah, ayah_without_
Index: []
```

Figure 16. LDA results.

Let’s evaluate experiment 2 against the same 3 queries.

Table 4. Experiment 2 evaluation.

Query	Top 3 cosine similarity Scores	Evaluation
Keeping the eye downward	-	No results were found above the threshold
Hijab in Islam	-	No results were found above the threshold
The final day	0.99, 0.99, and 0.99	Regardless of the high scores, only the first result is promising

So, this approach achieved 1 correct out of 9 for the 3 queries.

Table 5. Experiment 2 evaluation against surveying 5 different persons.

Query	Evaluation
How Allah proved his existence	The first result is good
Why did Allah create hell?	All results are away of the query
What is death?	All results are away of the query
Why should women wear hijab?	The First result is good
What will happen after the death?	All results are away of the query
What kind of torture happens in hell?	All results are away of the query
Why hijab is mandatory?	The last result is very good
What are the profit Muhammed characteristics?	All results are away of the query
What are the death signs?	No results above the threshold

Query	Evaluation
Why should people thank Allah?	The last result is good

So, this approach achieved 4 correct out of 30 for the 10 queries.

4.6 Experiment 3: word2vec

Word2Vec [22] is a set of algorithms that learns word embeddings, which are continuous vector representations of words. It uses approaches such as neural networks to train word embeddings based on the context in which words appear in the text. There are two main algorithms: Skip-gram and Continuous Bag of Words (CBOW).

Each word is represented by a fixed-length vector. Words with similar meanings or usage patterns tend to have similar vectors. It can be used for tasks such as finding similar words, word analogy tasks, and as input for more complex NLP models.

In Experiment 1 we used a transformers model and simply **Transformers models**, such as BERT and GPT, are advanced neural network architectures that process text bidirectionally (both forwards and backwards).

They use attention mechanisms to learn the context and relationships between words in a sentence. They can be trained in large corpora of text and fine-tuned for specific tasks.

These models provide contextualized word embeddings, which means each word's representation depends on the words surrounding it in the sentence. This allows the models to capture subtle nuances and meanings.

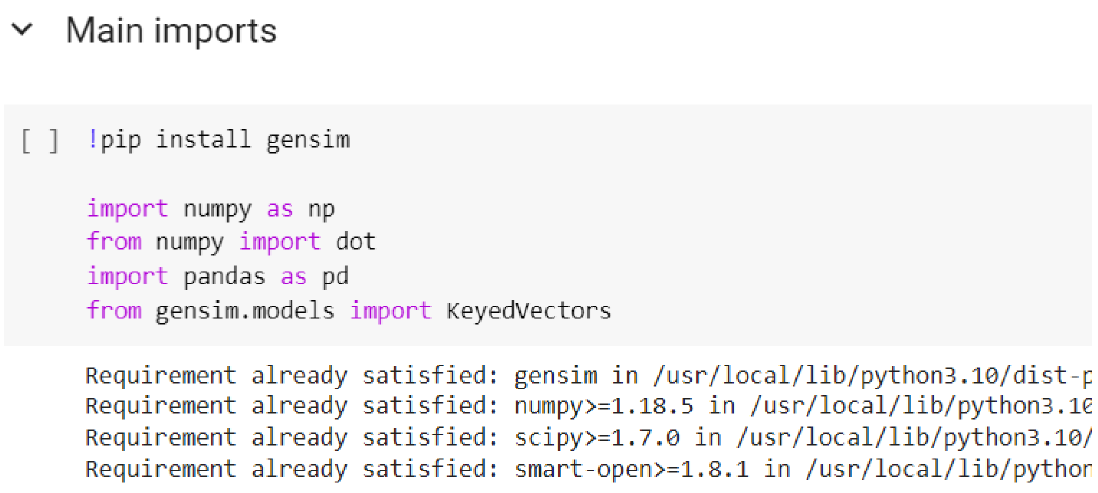
They are used for a wide range of NLP tasks, such as sentiment analysis, machine translation, text generation, and question-answering.

Before moving forward let’s compare word2vec against transformers models which exist in Experiment 1

Table 6. word2vec vs transformers.

	Word2vec	transformers
Contextualization	produces static word embeddings, meaning each word has a fixed representation regardless of its context	produce contextualized embeddings, where each word's representation can vary depending on the context in which it appears
Model complexity	simpler, shallower model	more complex and can handle a broader range of tasks
Training data	can work well with less data	require large amounts of data and computing power to train effectively
Applications	finding similar words	natural language understanding and generation

Back to Jupyter notebooks to check experiment 3, there are some packages that are essential for this experiment so that would be step 1 is to install and import the main packages which are gensim, Pandas, and NumPy (see Figure 16).

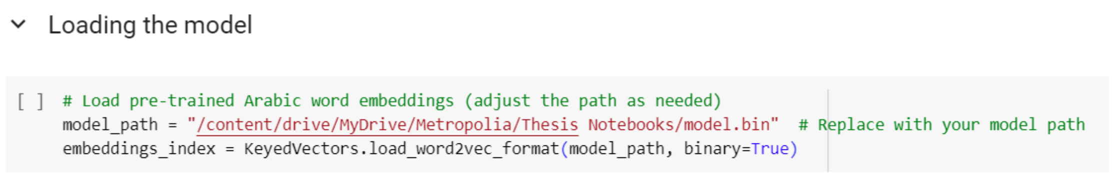


Main imports

```
[ ] !pip install gensim

    import numpy as np
    from numpy import dot
    import pandas as pd
    from gensim.models import KeyedVectors

    Requirement already satisfied: gensim in /usr/local/lib/python3.10/dist-p
    Requirement already satisfied: numpy>=1.18.5 in /usr/local/lib/python3.10
    Requirement already satisfied: scipy>=1.7.0 in /usr/local/lib/python3.10/
    Requirement already satisfied: smart-open>=1.8.1 in /usr/local/lib/pythor
```

Figure 16. Experiment 3 main imports.

Then reading and loading the model [23] (see Figure 17).

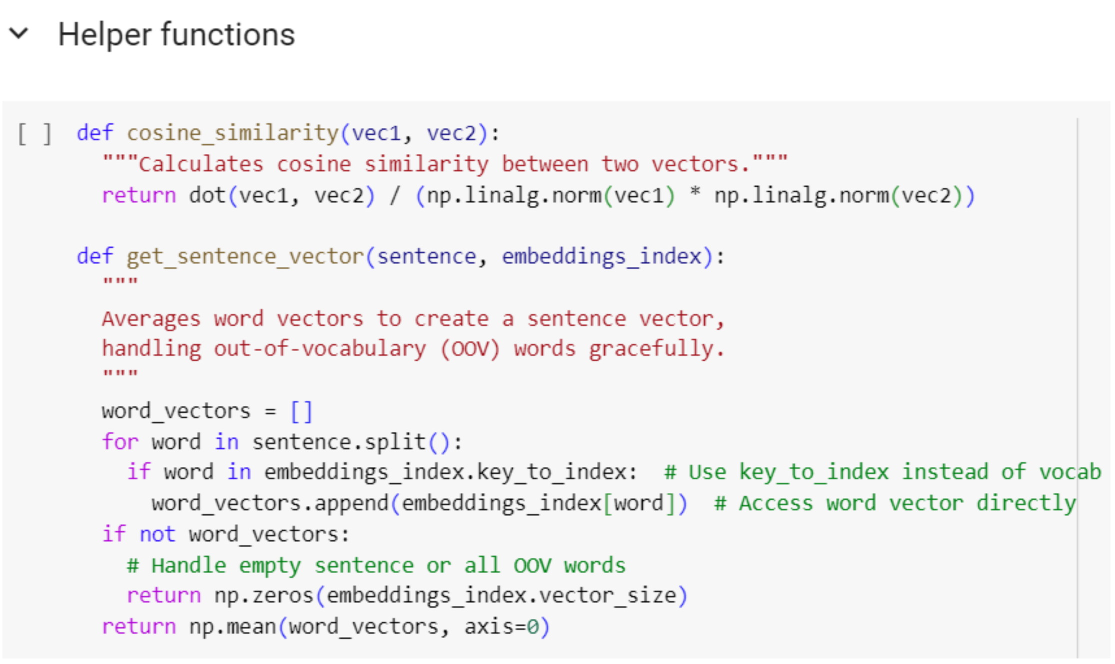


Loading the model

```
[ ] # Load pre-trained Arabic word embeddings (adjust the path as needed)
    model_path = "/content/drive/MyDrive/Metropolia/Thesis Notebooks/model.bin"  # Replace with your model path
    embeddings_index = KeyedVectors.load_word2vec_format(model_path, binary=True)
```

Figure 17. Loading the model.

Then to define helper functions such as cosine similarity and get sentence vector (see Figure 18).

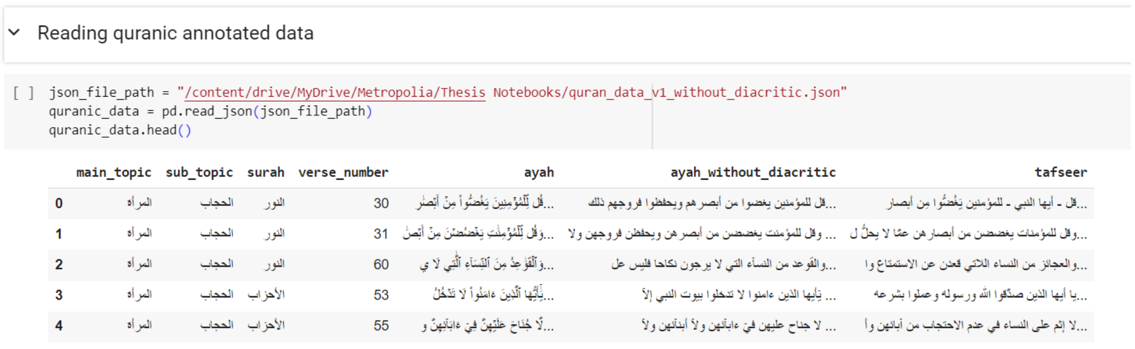


Helper functions

```
[ ] def cosine_similarity(vec1, vec2):
      """Calculates cosine similarity between two vectors."""
      return dot(vec1, vec2) / (np.linalg.norm(vec1) * np.linalg.norm(vec2))

    def get_sentence_vector(sentence, embeddings_index):
      """
      Averages word vectors to create a sentence vector,
      handling out-of-vocabulary (OOV) words gracefully.
      """
      word_vectors = []
      for word in sentence.split():
        if word in embeddings_index.key_to_index:  # Use key_to_index instead of vocab
          word_vectors.append(embeddings_index[word])  # Access word vector directly
      if not word_vectors:
        # Handle empty sentence or all OOV words
        return np.zeros(embeddings_index.vector_size)
      return np.mean(word_vectors, axis=0)
```

Figure 18. Helper functions.

Then read and check the annotated data to make sure that it matches our expectations (see Figure 19).



Reading quranic annotated data

```
[ ] json_file_path = "/content/drive/MyDrive/Metropolia/Thesis Notebooks/quran_data_v1_without_diacritic.json"
    quranic_data = pd.read_json(json_file_path)
    quranic_data.head()
```

	main_topic	sub_topic	surah	verse_number	ayah	ayah_without_diacritic	tafseer
0	المرأة	الحجاب	النور	30	...قُل لِّلْمُؤْمِنِينَ يَغُضُّوا۟ مِنْ أَبْصَٰرِ	...قل للمؤمنين يغضوا من أبصرهم ويحفظوا فروجهم ذلك	...قل - أيها النبي - للمؤمنين يَغُضُّوا مِن أبصار
1	المرأة	الحجاب	النور	31	...وَقُل لِّلْمُؤْمِنَٰتِ يَغْضُضْنَ مِنْ أَبْصَٰ	... وقل للمؤمنت يغضضن من أبصرهن ويحفظن فروجهن ولا	...وقل للمؤمنات يغضضن من أبصارهن عمّا لا يحلُّ ل
2	المرأة	الحجاب	النور	60	...وَٱلْقَوَٰعِدُ مِنَ ٱلنِّسَآءِ ٱلَّٰتِي لَا ي	...والقوعد من النسآء التي لا يرجون نكاحا فليس عل	...والعجائز من النساء اللاتي قعدن عن الاستمتاع وا
3	المرأة	الحجاب	الأحزاب	53	...يَٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُوا۟ لَا تَدْخُلُ	... يٰأيها الذين ءامنوا لا تدخلوا بيوت النبي إلآ	...يا أيها الذين صدّقوا الله ورسوله وعملوا بشرعه
4	المرأة	الحجاب	الأحزاب	55	...لَّا جُنَاحَ عَلَيْهِنَّ فِيٓ ءَابَآئِهِنَّ و	... لا جناح عليهن فيٓ ءابآئهن ولآ أبنآئهن ولآ	...لا إثم على النساء في عدم الاحتجاب من آبائهن وأ

Figure 19. Helper functions.

Then to define the main part of the code that does the actual job (Similarity 3)

```
# user_query = input("Enter a sentence to search for in the Tafseer column: ")
user_query = "الموت"
quranic_searching_column = quranic_data["tafseer"]

# Get sentence vectors for the list and query
list_vectors = [get_sentence_vector(sentence, embeddings_index) for sentence
in quranic_searching_column]
query_vector = get_sentence_vector(user_query, embeddings_index)

# Calculate cosine similarities between the query and each list item
similarities = [cosine_similarity(query_vector, list_vector) for list_vector
in list_vectors]
print("similarities",similarities)

# Convert the list of similarities to a NumPy array
similarities_array = np.array(similarities)

# Filter indices of values in similarities array that are greater than 0.61
indices = np.where(similarities_array > 0.61)[0]

# Filter the similarities array using the indices
filtered_similarities = similarities_array[indices]

# Sort the filtered similarities in descending order and get the sorted
indices
sorted_indices = np.argsort(filtered_similarities)[::-1]

# Get the top 3 highest similarities and their corresponding indices
top_3_indices = indices[sorted_indices[:3]]

# Print the rows from quranic_data that match the top 3 indices
print("Top 3 closest matches:")
pd.set_option('display.max_colwidth', None)
print(quranic_data.iloc[top_3_indices]['ayah'])
```

Similarity 3. Experiment 3

Let's elaborate on what this code does.

- **Mimic user input**
  - The user_query is defined as "الموت" (which means "death" in Arabic), representing the text that the user wants to search for in the Quranic data.
  - The code is set up to search for similar verses in the tafseer column of the quranic_data DataFrame.
- **Calculate Sentence Vectors**
  - The code uses a function (get_sentence_vector) to convert each sentence in the quranic_searching_column and the user_query into sentence vectors using a pre-trained embeddings index (embeddings_index).

    - Sentence vectors represent the sentences as numerical arrays (embeddings) based on word embeddings from the embeddings_index.
- **Calculate Similarities**
    - Calculates cosine similarity between the sentence vector for the user query (query_vector) and each sentence vector in the Quranic data (list_vectors).
    - Cosine similarity measures the cosine of the angle between two vectors, providing a similarity score between -1 and 1 (closer to 1 indicates higher similarity).
- **Filter Similarities**
    - Converts the list of similarities into a NumPy array.
    - Filters the indices of similarities that are greater than 0.61 (a specified threshold).
- **Sort and Get Top 3 Matches**
    - Sorts the filtered similarities in descending order and gets the indices of the top 3 highest similarities.
    - Retrieves the top 3 closest matches based on the indices.
- **Print Results**
    - Prints the top 3 closest matches (verses from the Quranic data) based on the user's query.

4.7 Analyzing Experiment 3

This approach works very fine with promising results, as an elaboration for this judge the annotated data has mainly 6 main topics but each verse can be involved in multiple topics not only one. So, let’s say that one verse can mention how women do Ramadan so this verse can be involved in topics such as “Ramadan, women fasting, etc.”.

When testing this approach with a query such as “What are the characteristics of the prophet Muhammed”, it returned results that included “he got the knowledge from Allah, he is a real prophet” and those are what the user is looking for.

Testing with a more complex sentence, let’s try something like “Avoid looking at Haram scenes is obliged for all Muslims.”, the results were very impressive such as **“And tell the believing women to lower their gaze and to be mindful of their chastity, and not to display their charms [in public] beyond what may [decently]”** partially translated for illustrating purposes.

Table 7. Experiment 3 evaluation.

Query	Top 3 cosine similarity Scores	Evaluation
Keeping the eye downward	0.7, 0.68, and 0.66	All results all good
Hijab in Islam	0.67, 0.66, and 0.65	The last 2 only is correct
The final day	0.66, 0.65, and 0.65	All results are good

So, this approach achieved 8 correct out of 9 for the 3 queries.

Table 8. Experiment 3 evaluation against surveying 5 different persons.

Query	Evaluation
How Allah proved his existence	The second result only is good
Why did Allah create hell?	All results are away of the query
What is death?	The first two results are good
Why should women wear hijab?	All results are good
What will happen after the death?	All results are good
What kind of torture happens in hell?	All results are good
Why hijab is mandatory?	All results are good

Query	Evaluation
What are the profit Muhammed characteristics?	All results are good
What are the death signs?	The last two results are good
Why should people thank Allah?	All results are away

So, this approach achieved 20 correct out of 30 for the 10 queries.

4.8 Experiment 4: training embeddings

The last experiment would be mostly like experiment 3 but this time with training own embeddings and using this model instead of a ready-made word2vec model

Then to define the main part of the code that does the actual job (Similarity 4)

```
# Main imports
...

# Reading the data
...

# Prepare the corpus
def preprocess_text(text):
    return simple_preprocess(text)  # Tokenize and convert text to lowercase


corpus = quranic_data['tafseer'].apply(preprocess_text).tolist()

# Train own Word2Vec model
vector_size = 100  # Size of the word vectors
window_size = 5  # Context window size
min_count = 1  # Minimum count of words to be considered
sg = 0  # Use CBOW model (1 for skip-gram, 0 for CBOW)

# Saving the model
word2vec_model = Word2Vec(corpus, vector_size=vector_size, window=window_size,
min_count=min_count, sg=sg)

# Save the trained model
model_path = "trained_word2vec_model.model"
word2vec_model.wv.save_word2vec_format(model_path, binary=True)

# Loading the model
embeddings_index = KeyedVectors.load_word2vec_format(model_path, binary=True)

# Helper functions
def cosine_similarity(vec1, vec2):
  """Calculates cosine similarity between two vectors."""
  return np.dot(vec1, vec2) / (np.linalg.norm(vec1) * np.linalg.norm(vec2))

def get_sentence_vector(sentence, embeddings_index):
  """
  Averages word vectors to create a sentence vector,
  handling out-of-vocabulary (OOV) words gracefully.
  """
  word_vectors = []
  for word in sentence.split():
    if word in embeddings_index.key_to_index:  # Use key_to_index instead of
vocab
      word_vectors.append(embeddings_index[word])  # Access word vector
directly
  if not word_vectors:
    # Handle empty sentence or all OOV words
    return np.zeros(embeddings_index.vector_size)
  return np.mean(word_vectors, axis=0)

# Getting user query and evaluate similarities
...
```

Similarity 4. Experiment 4

4.9 Analyzing Experiment 4

The similarities for the experiment were very low due to several reasons such as the lack of sufficient training data and the lack of diverse vocabulary. In the end, there were no records over the threshold of 0.6 and the maximum score was 0.4 which is a low score to trust. However, in experiment 3 for the same query the experiment returned 3 good results above 0.6.

The next step is to implement a web app to visualize the usage of this topic.

Table 9. Experiment 4 evaluation.

Query	Top 3 cosine similarity Scores	Evaluation
Keeping the eye downward	-	No results above the threshold
Hijab in Islam	-	No results above the threshold
The final day	-	No results above the threshold

So, this approach achieved 0 correct out of 9 for the 3 queries.

4.10 Implement QSE ‘Quran semantic engine’ web app

On top of Experiment 3, the next step is to implement a web app using **Flask** on the backend side to use that code and create an endpoint that can be used in the frontend side that is implemented by **React** [24]

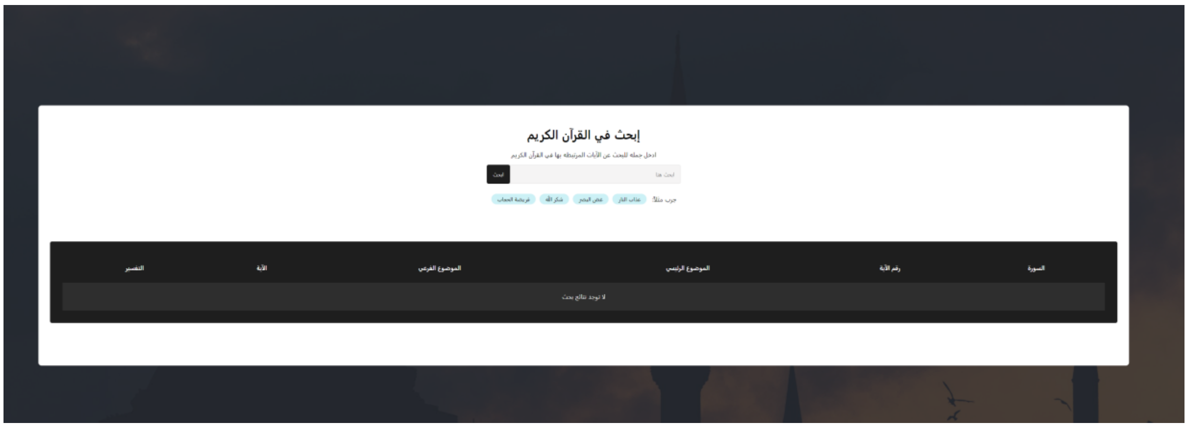
Below is the backend code (Similarity 5)

```
# Imports
...

# Load the model
...

# Helper Functions
# Calculates cosine similarity between two vectors.
def cosine_similarity(vec1, vec2):
    return dot(vec1, vec2) / (np.linalg.norm(vec1) * np.linalg.norm(vec2))


# Averages word vectors to create a sentence vector,
# handling out-of-vocabulary (OOV) words gracefully.
def get_sentence_vector(sentence, embeddings_index):
    word_vectors = []
    for word in sentence.split():
        if word in embeddings_index.key_to_index:  # Use key_to_index instead
of vocab
            word_vectors.append(embeddings_index[word])  # Access word vector
directly
    if not word_vectors:
        # Handle empty sentence or all OOV words
        return np.zeros(embeddings_index.vector_size)
    return np.mean(word_vectors, axis=0)


# Reading the Quranic date
...



# Define an endpoint that takes a query as input and returns a JSON response
@app.route('/search', methods=['GET'])
def query_endpoint():
    query = request.args.get('query')
    quranic_searching_column = quranic_data["tafseer"]

    # Get sentence vectors for the list and query
    list_vectors = [get_sentence_vector(sentence, embeddings_index) for
sentence in quranic_searching_column]
    query_vector = get_sentence_vector(query, embeddings_index)

    # Calculate cosine similarities between the query and each list item
    similarities = [cosine_similarity(query_vector, list_vector) for
list_vector in list_vectors]
    print("similarities", similarities)

    # Convert the list of similarities to a NumPy array
    similarities_array = np.array(similarities)

    # Filter indices of values in similarities array that are greater than
0.61
    indices = np.where(similarities_array > 0.61)[0]

    # Filter the similarities array using the indices
    filtered_similarities = similarities_array[indices]

    # Sort the filtered similarities in descending order and get the sorted
indices
    sorted_indices = np.argsort(filtered_similarities)[::-1]

    # Get the top 3 highest similarities and their corresponding indices
    top_3_indices = indices[sorted_indices[:3]]

    # Print the rows from quranic_data that match the top 3 indices
    ...
```

Similarity 5. A backend endpoint using Flask.

The whole project Frontend/Backend was published to GitHub [25].

Below are the final results (see Figure 20).



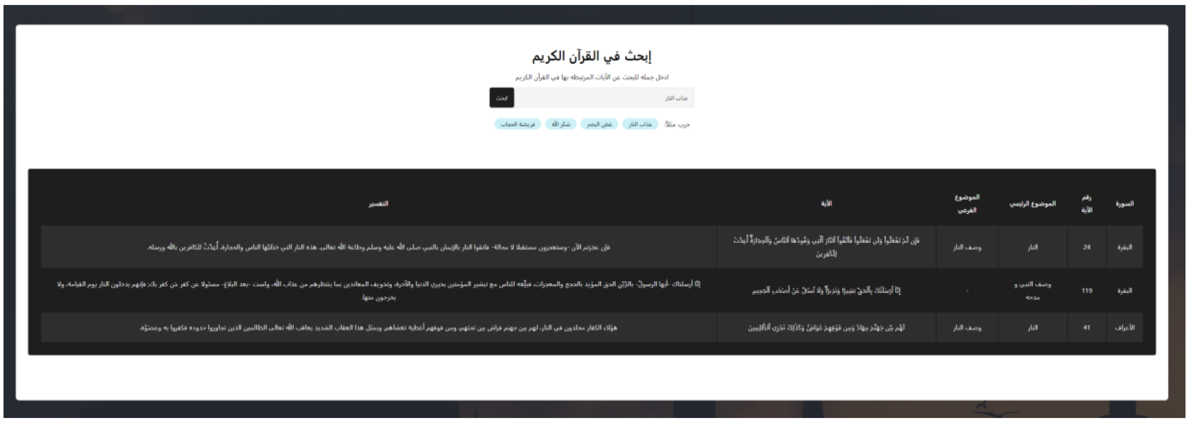
إبحث في القرآن الكريم

ادخل جمله للبحث عن الآيات المرتبطه بها في القرآن الكريم

السورة	رقم الآية	الموضوع الرئيسي	الموضوع الفرعي	الآية	التفسير
لا توجد نتائج بحث					

Figure 20. QSE web app.

Below are the final results after the search (see Figure 21).



إبحث في القرآن الكريم

ادخل جمله للبحث عن الآيات المرتبطه بها في القرآن الكريم

السورة	رقم الآية	الموضوع الرئيسي	الموضوع الفرعي	الآية	التفسير
البقرة	24	النار	وصف النار	فَإِن لَّمْ تَفْعَلُوا وَلَن تَفْعَلُوا فَاتَّقُوا النَّارَ الَّتِي وَقُودُهَا النَّاسُ وَالْحِجَارَةُ ۖ أُعِدَّتْ لِلْكَافِرِينَ	فإن عجزتم الآن -وستعجزون مستقبلا لا محالة- فاتقوا النار بالإيمان بالنبي صلى الله عليه وسلم وطاعة الله تعالى. هذه النار التي حطبها الناس والحجارة، أُعِدَّتْ للكافرين بالله ورسله.
البقرة	119	وصف النبي و مدحه	-	إِنَّا أَرْسَلْنَاكَ بِالْحَقِّ بَشِيرًا وَنَذِيرًا ۖ وَلَا تُسْأَلُ عَنْ أَصْحَابِ الْجَحِيمِ	إنا أرسلناك -أيها الرسول- بالدِّين الحق المؤيد بالحجج والمعجزات، فبلِّغه للناس مع تبشير المؤمنين بخيري الدنيا والآخرة، وتخويف المعاندين بما ينتظرهم من عذاب الله، ولست -بعد البلاغ- مسئولا عن كفر مَن كفر بك، فإنهم يدخلون النار يوم القيامة، ولا يخرجون منها.
الأعراف	41	النار	وصف النار	لَهُم مِّن جَهَنَّمَ مِهَادٌ وَمِن فَوْقِهِمْ غَوَاشٍ ۚ وَكَذَٰلِكَ نَجْزِي الظَّالِمِينَ	هؤلاء الكفار مخلدون في النار لهم من جهنم فراش من تحتهم ومن فوقهم أغطية تغشاهم، وبمثل هذا العقاب الشديد يعاقب الله تعالى الظالمين الذين تجاوزوا حدوده فكفروا به وعصَوْه.

Figure 21. QSE web app results.

5 Conclusions

The real challenge started with collecting and annotating the data, it required a lot of searches to find multiple sources then checking those sources one by one evaluating them in a table then choosing the best of them which would fit this task.

After this step, it was necessary to remove Arabic diacritics to make it easy to remove some complexity and increase the readability. After this, the research went through 3 different experiments.

Experiment 1 relied on the concept of the pre-trained model with cosine similarity, since the project is targeting the Arabic language then there was some lack of resources. 2 different pre-trained models from Hugging Face were tried, the first one was Arabic BERT, and the second one was Quranic large model. However, experiment 1 was the most promising theoretically, but it ended up with very low results.

The evaluation for experiment 1 relied on the cosine similarity scores, however, it was good “around 6.5” but the results for native speakers shouldn’t be satisfying so it ended up with a decision that these embeddings may not be the best.

Experiment 2 was about topic modeling and how to extract topics from the data and then use those topics to build on top of that such as checking similarity distance. Sadly, the LDA technique failed to extract topics then the step on top of that is not achievable.

Experiment 3 was about using cosine similarity with word2vec which had better embeddings for this task than the transformers BERT model, the results were very good and illustrated by example in the experiment 3 section.

Experiment 4 was about training our embeddings and using this model with a combination of cosine similarity, but the results were not good due to lack of training data.

Table 10. Experiments comparison for 1 person.

	Approach	Evaluation score
Experiment 1	BERT pre-trained model	1/9
Experiment 2	LDA	1/9
Experiment 3	Word2vec	8/9
Experiment 4	Trained word2vec	0/9

Table 11. Experiments comparison against surveying 5 different persons.

	Approach	Evaluation score
Experiment 1	BERT pre-trained model	1/30
Experiment 2	LDA	4/30
Experiment 3	Word2vec	20/30

So, the final choice was experiment 3.

In the end, a web app with React and Flask was built to use and visualize experiment 3 “screenshots at the end of chapter 4” to help users enter a query using the natural language and then get a table of results “Surah name, verse number, verse, interpretation” and that was the target from the beginning.

What next? The code of the last experiment was published to GitHub and the code for the previous experiments was added to this research in experiments 1 and 2. The next would be to check if experiment 3 was really the best for adding tricky parts in the previous experiments would be better. Another idea also is to check a multi-language approach to serve multiple languages not only Arabic.

6 Summary

Searching in the Holy Quran is a sensitive topic as it requires a lot of research not only for the scientific field but also for the religious field. The notebooks and the whole code for this idea should be published to give the chance for experts to be inspired and build more advanced tools to make the world a better place.

Semantic similarity topic is not a new topic but building on top of that was the purpose of this research. As illustrated in the research, how to start from the theory of which field or sector needs to have some AI to make things easier.

Through this research there were a lot of challenges and how I managed to overcome until the target, even if the solution may not be the best it came from zero experience in this field and it was a great journey to learn something new and develop a new topic that should help my nation and the Islamic society.

The starting point is always the most difficult and below is a summary of the whole journey.

- In Chapter 1 the idea was to set motivation for readers and to illustrate the initial plan
- Chapter 2 was around the current implementations, challenges and opportunities.
- Chapter 3 was about the methodology, the whole technical plan from data collection until the end of the project but in a theory mode.
- Chapter 4 was the real implementation where I showed all my tries and failings until finding the optimal solution.

References

1 How many verses are in the holy Quran? <https://islamqa.org/hanafi/daruliftaa-birmingham/88370/how-many-verses-are-there-in-the-holy-quran/>. Accessed 25 Apr 2024.

2 What is semantic search? <https://www.elastic.co/what-is/semantic-search/>. Accessed 25 Apr 2024.

3 Quran corpus <https://corpus.quran.com/>. Accessed 25 Apr 2024.

4 Search the way you think how personalized semantic search is disrupting traditional search <https://www.shaped.ai/blog/search-the-way-you-think-the-personalized-semantic-search-revolution-disrupting-traditional-keyword-based-search-with-ai/>. Accessed 25 Apr 2024.

5 Semantic Search Challenges <https://www.linkedin.com/pulse/pitfalls-semantic-search-reza-bonyadi/>. Accessed 25 Apr 2024.

6 Quran Search Engine <https://holyquran.net/search/sindex.php/>. Accessed 25 Apr 2024.

7 Quran Search Engine <https://surahquran.com/quran-search/>. Accessed 25 Apr 2024.

8 Word Embeddings <https://www.ibm.com/topics/word-embeddings/>. Accessed 25 Apr 2024.

9 Matrix factorization vs SVD <https://www.freecodecamp.org/news/singular-value-decomposition-vs-matrix-factorization-in-recommender-systems-b1e99bc73599/>. Accessed 25 Apr 2024.

10 Quran Interpretation. <https://quranenc.com/ar/browse/arabic_moyassar/>. Accessed 20 Apr 2024.

11 Quranic Topics. <https://surahquran.com/quran-search/Topic-page-1/>. Accessed 20 Apr 2024.

12 Arabic BERT Embeddings <https://sparknlp.org/2022/04/11/bert_embeddings_bert_base_arabert_ar_3_0.html/>. Accessed 25 Apr 2024.

13 Transformers. <https://blogs.nvidia.com/blog/what-is-a-transformer-model/>. Accessed 21 Apr 2024.

14 asafaya/bert-base-arabic. <https://huggingface.co/asafaya/bert-base-arabic>. Accessed 21 Apr 2024.

15 Understanding Cosine Similarity: A key concept in data science. <https://medium.com/@arjunprakash027/understanding-cosine-similarity-a-key-concept-in-data-science-72a0fcc57599>. Accessed 21 Apr 2024.

16 What is Tokenization? <https://www.datacamp.com/blog/what-is-tokenization>. Accessed 21 Apr 2024.

17 Tokenization in Machine Learning Explained. <https://vaclavkosar.com/ml/Tokenization-in-Machine-Learning-Explained>. Accessed 21 Apr 2024.

18 omarelsayeed/arabert_quran_large. <https://huggingface.co/omarelsayeed/arabert_quran_large>. Accessed 21 Apr 2024.

19 Topic Modelling with LSA, PLSA, LDA & lda2Vec. <https://medium.com/nanonets/topic-modeling-with-lsa-psla-lda-and-lda2vec-555ff65b0b05>. Accessed 21 Apr 2024.

20 Latent Dirichlet Allocation. <https://medium.com/@corymaklin/latent-dirichlet-allocation-dfcea0b1fddc>. Accessed 29 Apr 2024.

21 A Beginner’s Guide to Latent Dirichlet Allocation(LDA). <https://towardsdatascience.com/latent-dirichlet-allocation-lda-9d1cd064ffa2>. Accessed 29 Apr 2024.

22 word2vec. <https://www.tensorflow.org/text/tutorials/word2vec>. Accessed 21 Apr 2024.

23 NLP word embeddings repository. <http://vectors.nlpl.eu/repository>. Accessed 20 Apr 2024.

24 How To Create a React + Flask Project. <https://blog.miguelgrinberg.com/post/how-to-create-a-react--flask-project>. Accessed 21 Apr 2024.

25 QSE repository. <https://github.com/OmarioHassan/quran-search-engine>. Accessed 21 Apr 2024.