JOURNAL OF QURANIC SCIENCES AND RESEARCH
e-ISSN: 2773-5532

JQSR

Vol. 5 No. 1 (2024) 46-60
https://publisher.uthm.edu.my/ojs/index.php/jqsr

A Tapestry of Tongues: A Novel Provenance Approach for Arabic Linguistic Styles and Lineage Tracing

Adel Sabour[1], Abdeltawab Hendawi[2], and Mohamed Ali[1]

[1] *University of Washington, Computer Science and Systems Tacoma, USA*
[2] *University of Rhode Island, Computer Science and Statistics, Rhode Island, USA*

*Corresponding Author: *sabour@uw.edu*

DOI: https://doi.org/10.30880/jqsr.2024.05.01.005

Article Info

Received: 7th December 2023
Accepted: 9th May 2024
Available online: 30th June 2024

Keywords

Quranic linguistic styles, arabic diacritization, provenance tracking, arabic dialect, textual data processing, Quran

Abstract

This research introduces a novel framework for analyzing Arabic linguistic styles, focusing on the Quran as a case study. Unlike previous studies that often focus on a single style or regional variations, ours examines the unique features of multiple styles within the single text. Our approach addresses inconsistencies in Arabic character representation. This creates a standardized format for analyzing the text, ensuring consistent results. This research utilizes a Provenance Tracker to connect different linguistic styles and document their lineage through narrators By analyzing a single text that showcases multiple styles, this study paves the way for identifying the unique characteristics of each linguistic style. Additionally, the research develops algorithms to map individual letters, words, and diacritics between each style. Moreover, this allows for in-depth linguistic analysis, revealing the unique characteristics that define each style. This opens new avenues for research into the Arabic language's rich stylistic diversity.

1. Introduction

Arabic, spoken as a primary language in twenty-two countries and a second language in many other countries (Habash, 2022), holds significant historical and cultural importance. It is ranked as the fourth most used language on the Internet (Boudad et al.,2017) and is spoken by over 400 million people worldwide (Abu Dalhoum and Al-Taani, 2017). These numbers reflect the widespread use and influence of the Arabic language. Arab scholars played a pivotal role in advancing various scientific disciplines and expanding the boundaries of knowledge. Their contributions to astronomy, mathematics, medicine, and chemistry remain a testament to their impact on human history (Ofek, 2011). Additionally, Arabic heritage books serve as valuable repositories of knowledge from ancient civilizations. Although these books were meticulously handwritten, their digital transfer often received inadequate attention (Amazouz et al.,2020).

The styles of Arabic language are usually classified into three main categories: 1) Classical Arabic (CA), which is used for the heritage books like the Quran (the Muslim holy book) and other literary texts, 2) the Modern Standard Arabic (MSA), which is the language of formal writing and conversation and is derived from the CA, and 3) the Dialectal Arabic (DA), which is the regionally varying informal colloquial language (Habash, 2010); (Farghaly and Shaalan, 2009); (Harrat et al., 2017). This research focuses on CA, as it represents the most detailed linguistic level from which all other levels are derived (Yousef et al.,2023). Moreover, the wealth of Arabic heritage literature is preserved in books that use the CA style.

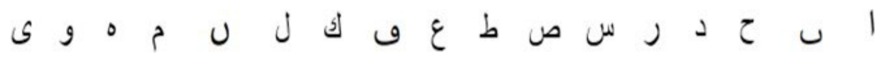
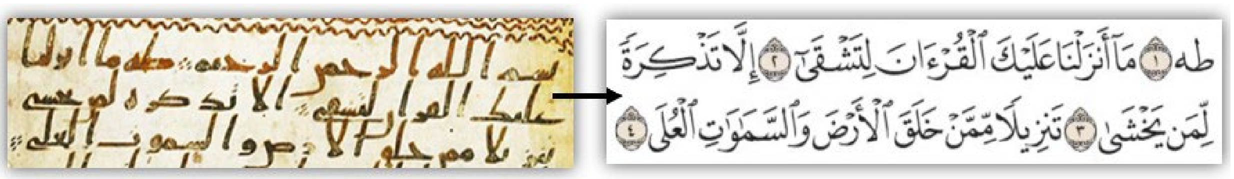


Fig. 1 *The first drawing of writing FDW for Arabic letters*

Initially, Arabic writing consisted of 19 letters without any dots or diacritics (See figure 1), yet the speakers' eloquence reached its peak. However, as time passed, the eloquence of speakers declined, and linguistic errors became more prevalent. Linguists introduced dots and diacritics to the script to eliminate errors while preserving the language's details. By utilizing diacritics and special signs, these experts were able to accurately represent the diverse sounds and dialects present in the Arabic language during that era. The evolution of Arabic writing (See figure 2) can be classified into five levels of details or annotations, as illustrated in Table 1. Level 5 features the highest number of annotations and lower levels show a decreasing level of annotations.



طه ﴿١﴾ مَآ أَنزَلۡنَا عَلَيۡكَ ٱلۡقُرۡءَانَ لِتَشۡقَىٰٓ ﴿٢﴾ إِلَّا تَذۡكِرَةٗ لِّمَن يَخۡشَىٰ ﴿٣﴾ تَنزِيلٗا مِّمَّنۡ خَلَقَ ٱلۡأَرۡضَ وَٱلسَّمَٰوَٰتِ ٱلۡعُلَى ﴿٤﴾

This Arabic manuscript dates back to over 1440 years ago.

the content of the same manuscript in its current form of writing

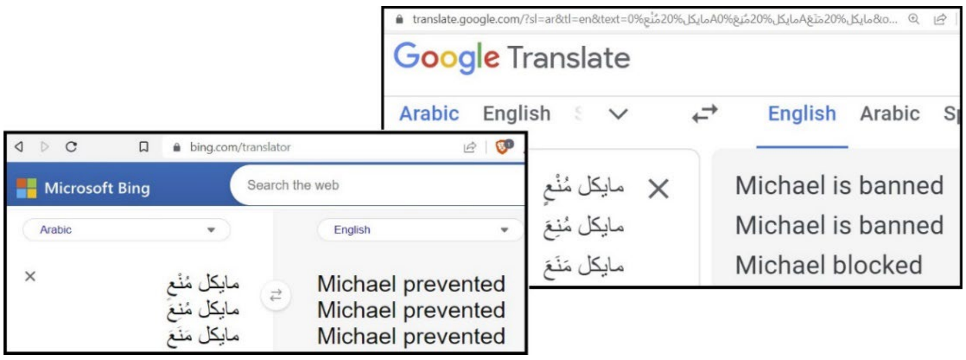
Fig. 2 *A comparison between ancient and modern Arabic writing styles*

The first level, known the First Drawing of Writing (FDW), is the simplest form and consists of 17 letter shapes without dots. The second level adds dots to letter shapes to differentiate between letters that have the same shape and, consequently, end up with an alphabet of 28 letters. Most current Arabic NLP techniques focus on the second level, which is the most commonly used. The third level includes diacritic marks (*Tashkeel*) to clarify meaning and pronunciation according to grammar rules. The fourth level introduces pronunciation marks that indicate variations in pronunciation, often associated with dialects. The fifth level incorporates Tajwid marks found in the Quran (the holy book of Muslims). This Tajwid marks indicate specific rules that the reader must follow to accurately pronounce the Arabic letters.

Table 1 *Levels of details and annotations in the Arabic writing styles*

واعدالحطالعرى	قواعدالخطالعربى	قَوَاعِدُالخَطِّ العَرَبِيّ	قَوَاعِدُالخَطِّ العَرَبِيّ	∴ ٠ ● ۬ ۚ
a) First drawing of writing.	b) Dotted letters.	c) Diacritic letters (tashkeel).	d) Pronunciation marks.	e) Tajwid marks

There have been many advancements in NLP technologies in languages that use the Latin alphabet like English. However, the Arabic language is very special in the way it is written or even spoken. The technology transfer of NLP techniques from English to Arabic is not straightforward. For example, Arabic letters have multiple shapes, even in the same position. Each letter can have multiple diacritics, making Arabic distinct.



translate.google.com/?sl=ar&tl=en&text=...

Google Translate

Arabic English → English Arabic

مايكل مُنِع	Michael is banned
مايكل مُنِع	Michael is banned
مايكل مَنَع	Michael blocked

bing.com/translator

Microsoft Bing

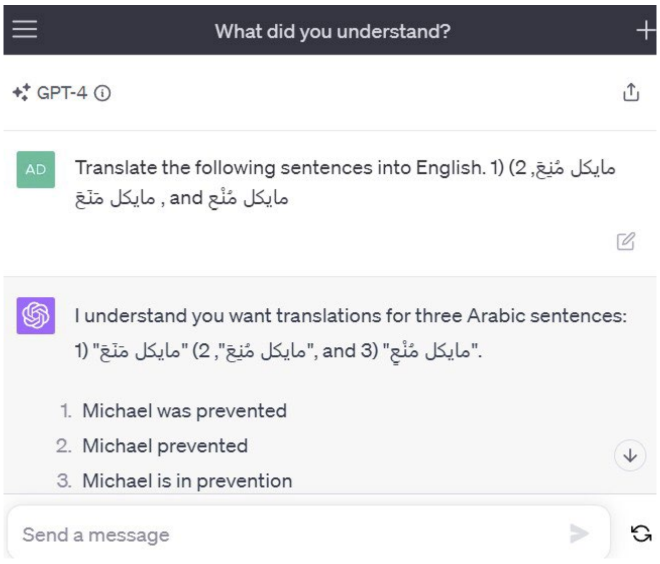
Arabic → English

مايكل منع	Michael prevented
مايكل مُنِع	Michael prevented
مايكل مَنَع	Michael prevented

Fig. 3 *Shows incorrect translation results from Microsoft Bing and Google translation services*

The same word can have different meanings or different derivatives based on the diacritic marks added to it. For example, **مايكل منع** written at the second level can have antonyms. With diacritics, **مايكل مُنِعَ** means Michael is *forbidden*. However, with diacritics **مايكل مَنَعَ** means Michael *made something not allowed*, and **مايكل مُنْعٍ** means Michael *obituary* (notice of death), someone. Figure 3: Wrong translation due to not recognizing the diacritical characters: Comparison between Google Translator and Bing Translator.

Figure 4 shows the same issue of the inaccurate translation from chat GPT (Generative Pre-Trained Transformer) (OpenAI, 2023). The translation services and chat GPT did not recognize the diacritical marks when determining the meaning of words. The figures (3 and 4) present a simple example of a challenge instilled by the nature of the Arabic language. Additional research is needed to support the various styles and levels of annotations in the Arabic language. This will enable the transfer of state-of-the-art research in NLP to the Arabic world.



Translate the following sentences into English. 1) مايكل مُنِعَ, 2) مايكل مُنْعِ, and مايكل مَنَعَ

I understand you want translations for three Arabic sentences: 1) "مايكل مَنَعَ", 2) "مايكل مُنِعَ", and 3) "مايكل مُنْعِ".

1. Michael was prevented
2. Michael prevented
3. Michael is in prevention

Fig. 4 *Chat GPT fails to understand some Arabic words with diacritics letters*

The linguistic style encompasses any dialect that is linguistically correct, can be traced back to its origin, is associated with known proficient speakers, and can be represented in written form. As a case study, the text of the Qur'an is chosen due to its representation of the highest level of the Arabic language (refer to Table 1) and their inclusion of various linguistic styles. In Islamic studies," *linguistic style*" is known as "*Riwayah*" or "*Qira'ah*". Linguistic styles are named after renowned narrators who preserved and passed them down. During the Qur'an's revelation, the Prophet Muhammad (*peace be upon him - PBUH*) allowed for specific words and dialects within established principles. Narrators had the freedom to choose from these options, selecting styles closer to their dialect backgrounds.

Table 2 *Example of the orthographic and phonetic diversity for one word across different Arabic linguistic styles*

The word:الصراط		
Linguistic Styles	Word	Pronunciation
Qonb and Ruwais	ٱلصِّـرَٰطَ	as-ziraatta
Khalf and Khallaad	ٱلصِّرَٰطَ	as-siiraatta
The other styles and Khallaad	ٱلصِّرَٰطَ	as-saraatta

Table 2 examines the Arabic word "**الصراط**" across different linguistic styles. This examination reveals variations in the representation and pronunciation of the word across different Arabic linguistic styles. The table is organized into three columns: The first column lists the names of the linguistic styles. The second column shows how the word is written in each style. The third column details the pronunciation in each respective style. While the basic shape of the word remains consistent, variations in diacritics alter its pronunciation. These variations might add meaning or reflect regional pronunciations. The table reveals that some linguistic styles share identical forms, while others exhibit distinct variations. Notably, the Khallad style stands out, as it allows for two correct pronunciations of the same word. That demonstrates the complexity and nuance of Arabic linguistic styles.

This study introduces a dedicated system for analyzing the diverse linguistic styles present within the Quran. A major challenge was the lack of existing open-source databases encompassing this variety. To address this gap, our system focuses on the collection, normalization, and analysis of Quranic texts across various styles. The database design captures information about narrators, lineage paths, and the Quranic linguistic styles. This required a custom approach due to the diverse structures and formats encountered in the data.

This paper is organized as follows: After this introduction, we will discuss the related work in the field of Arabic NLP and Quranic database in Section 2: Related Work. This will be followed by Section 3: The Proposed Process-Oriented Provenance Model for Quranic Linguistic Styles, where we will present our proposed model. Next, we will describe the System Modules for Building a Database for Quranic Linguistic Styles in Section 4. Finally, we will conclude with some insights and future research directions in Section 5: Concluding Insights and Future Research, and summarize our work in Section 6: Conclusion.

2. Related Work

(Guellil et al., 2021) conducted a comprehensive survey of Arabic Natural Language Processing (ANLP) research. They analyzed 90 papers, highlighting the complexity of the Arabic language and its challenges for NLP tasks. The study revealed a focus on Modern Standard Arabic (MSA) and Dialectal Arabic (DA), with less attention to Classical Arabic (CA). This gap in research motivated our work to contribute to CA advancement. Recognizing the importance of resource development in ANLP, we built systems to support Arabic NLP. Our research aligns with their findings, emphasizing the significance of resource construction for tasks like semantic analysis, speech recognition, and text processing.

(Kadir et al., 2020) conducted a study on the manuscript of the Turkish Quran. It aims to store the manuscript in a Database Management System (DBMS). The preservation of Qur'anic manuscripts holds importance. These manuscripts are not only religious texts but also valuable cultural artifacts. The history of some of these manuscripts spans over 1,400 years. Their research focuses on the storage and preservation of a specific historical copy of the Quran. In contrast, our research aims to create a comprehensive database. The database is capable of storing both historical and modern versions of the Quran. Additionally, it facilitates comparisons between these versions. By expanding the scope, we aim to enhance the accessibility of the Quranic texts and their interconnected components. Furthermore, our research seeks to improve the analysis and preservation of these texts.

In a study conducted by (Oktaviani et al., 2019), the authors designed a Quran database. Their database was an on-device database for mobile applications. In contrast, our research focuses on developing a server-side DB. Our database is accessible through an API, enabling its utilization by various types of applications. It also overcomes storage limitations on devices. While the researchers acknowledged the presence of incomplete data and expressed their pursuit of reliable data. In our research, we have developed a web scrapping module. This module extracts data from different websites. It includes complete copies of the Quran in various linguistic styles. Furthermore, we have implemented algorithms to compare different versions, detect errors, and validate the data.

The research conducted by (ElSayed, 2015) focuses on designing an Arabic natural language interface system for a database of the Quran. The author's database system is designed specifically for the linguistic style of Hafs. The database is obtained from a single website without data verification or data lineage tracking. In contrast, our research is different from ElSayed's in several aspects. Firstly, we download data from multiple websites. Secondly, we implement algorithms for copy comparison, error detection, and data validation. Moreover, our database encompasses various linguistic styles of the Quran. However, our research goes a step further by providing statistics based on multiple linguistic styles of the Quran.

Previous research has explored Arabic NLP, Quranic databases. However, analyzing the Qur'an's diverse linguistic styles and data provenance remains an unaddressed challenge. Our work takes a unique approach to address this challenge:

- We develop a framework to analyze linguistic styles within the Qur'an.
- Our "Provenance Tracker" traces the lineage of each style, providing historical context and insights into their evolution.
- We implement a custom character normalization algorithm to ensure consistent text representation across styles.
- Our database integrates data on narrators, styles, and variations, enabling in-depth research.

3. The Proposed Process-Oriented Provenance Model for Quranic Linguistic Styles

We acknowledge the existence of various dialects during the Quran's revelation. The Prophet Muhammad (PBUH) allowed the use of certain words with specific dialects, following established guidelines. Narrators, within these guidelines, exercised some choice in style selection. Narrators refer to individuals who have memorized the entire Qur'an with one or more linguistic style and are qualified to teach others. This flexibility could involve choosing a dialect, method, or style most familiar to them. Over time, this practice resulted in the emergence of the twenty known and defined linguistic styles. It is important to emphasize that narrators did not modify the Quranic text. Their choices involved selecting from the options permitted by the Prophet (PBUH). A narrator's style preference could be transmitted to their students, potentially leading to a lineage of narrators using the same style.

This model sheds light on the dynamic process of Quranic transmission among the narrators and the development of diverse linguistic styles. It traces how Quranic linguistic styles evolved and were passed down through narrators. It also examines the choices narrators made when transmitting these styles, providing valuable insights. This research follows the development and transmission of Quranic linguistic styles through narrators, shedding light on their transmission choices.

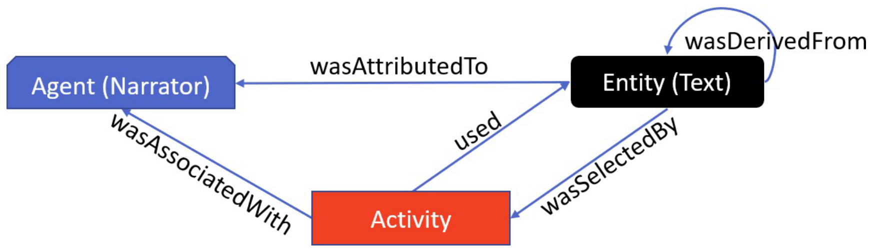


Fig. 5 *Diagram of the process-oriented provenance model tracing quranic style evolution through narrators*

1. **Activity:** This component represents the narrator's choice, which shapes the linguistic style.
   - An arrow labeled 'wasAssociatedWith' connects the Activity to the Agent (Narrator), indicating the narrator's involvement in the activity.
   - An arrow labeled 'used' connects the Activity to the Entity (Text), specifying the text used in the activity.
   - An arrow labeled 'wasSelectedBy' leads from Entity (Text) to Activity, denoting the text selections during the activity.
2. **Agent (Narrator):** This represents individuals (narrators) who have transmitted the Quranic text.
   - An arrow labeled 'wasAttributedTo' connects Entity (Text) to Agent, attributing the text selections to the narrator.
3. **Entity (Text):** This pertains to the Quranic text or the linguistic style.
   - An arrow labeled 'wasDerivedFrom' loops from Entity to itself, reflecting the derivation of a specific linguistic style from the text.

This provenance model is the foundational framework we followed to create our database. It effectively captures the interconnections between Quranic narrators and linguistic styles. This framework is crucial for understanding the historical and stylistic evolution of the Quranic text. It confirms the systematic approach we used to document and analyze the development of linguistic styles within the Quran.

4. System Modules for Building a Database for Quranic Linguistic Styles

This section outlines the core functionalities of the system. The system follows a modular structure outlined in (Siddiqui et al.,2022). Our system for analyzing Quranic Linguistic Styles begins with data collection. We extract text data from various sources and transform it into a unified format. This data is then stored in the database. This database provides us with datasets for training and evaluation of the AI models employed in the system. To handle the challenges posed by different Unicode representations of Arabic characters (Shaalan et al.,2019), the Text Digitizer and Analyzer Module is employed. This module maps various representations to a unified format, facilitating easier comparison and analysis of the text. The Data Validation and Integration module ensures the quality and consistency of the integrated data by validating and contrasting multiple copies of the same text. Additionally, this module ensures specific information is available for each word, letter, and diacritical mark across the different Linguistic Styles. This allows us to track not only the stylistic variations but also the choices made by different narrators in their transmissions. The following subsections will delve deeper into each of these modules, providing a detailed explanation of their functionalities.

4.1 The Data Collector

In this module, we utilize Extract, transform, and load (ETL) techniques to extract, transform, and load data into a suitable format for processing within the system. Due to the lack of open-source databases for the Quran with more than linguistic style, the development of web scraping mechanisms became essential for data extraction. The system incorporates web scraping algorithms that automatically extract Quran text from various online sources Diouf et al.,2018).

While the system also encompasses algorithms for extracting audio data and images, this particular part of the study concentrated solely on textual data. The text data is scraped from a range of websites, documents, and databases, each with its unique structure. We exclusively utilized publicly available, non-proprietary Quranic texts, ensuring that our data collection process does not involve any privacy-sensitive information. As a consequence, dedicated algorithms were implemented to handle the extraction process for each specific data source. The system records the data from each source and the corresponding acquisition timestamps, ensuring that only new data is collected during subsequent extraction runs. Once the data is obtained, it undergoes a transformation process to unify its format. After the data acquisition, a transformation process is applied to convert this heterogeneous data into a unified format. This step incorporates established data science practices (Ilyas, et al., 2015) like the standardization of data cleansing. We use deduplication algorithms to eliminate duplicates. The organization is based on database normalization principles (Khodorovskii, 2002). These practices facilitate rigorous analysis and validation.

Table 3 *Data of the Quran and its associated linguistic styles*

Place	City	Main Narrator	Lineage	Counting scheme	Narrator	Lineage	Typed	Scanned pages
Hejaz	Almadina	Nafie	4	Madani Awal	Qalun	5	Yes	559
			4	Madani Akhir	Warsh	5	Yes	604
		AboJaafr	3	Madani Awal	Abn-Wardan	4	No	559
			3		Abn-Jammaz	4	No	604
	Makkah	Ibn-Kathir	3	Makki	Al-Bazi	6	Yes	604
			3		Qonbl	7	Yes	604
Iraq	Basra	AbuAmr	4	Basri	Al-Dawri	6	Yes	604
			4		Al-Suwsi	6	Yes	521
		Ya’qub	4		Ruwais	5	No	604
			4		Rauoh	5	No	604
	Kufa	Hamza	4	Kufi	Khalf	6	No	604
			4		Khallaad	6	No	604
		AlKisayiy	5		Abul-Harith	6	No	604
			5		AlDawri	6	No	604
		Assem	3		Hafs	4	Yes	604
			3		Shueba	4	Yes	604
		Khalaf	6		Ishaq	7	No	604
			6		Iddris	7	No	604
Sham	Damascus	IbnAmer	2	Damascus	Hisham	5	No	604
					Ibn-Thakwan	5	No	604

Furthermore, advanced tools are developed to partition and distribute the transformed data efficiently within our database structure. By implementing this module, the system minimizes manual effort and reduces errors. It achieves automation in data collection, allowing for consistent and up-to-date data gathering from diverse sources.

4.2 Linguistic Styles’ Provenance Tracker

The data includes multiple linguistic styles with potential variations from different sources. To differentiate version errors from style differences, we developed the Linguistic Styles’ Provenance Tracker (Puri et al., 2012). The tracker aids data lineage analysis that leads to understanding the interdependencies among linguistic styles. Lineage metadata (Simmhan et al., 2005) provides insights into the individuals involved in moving the linguistic style data from its original source to its final state.

Each linguistic style has a group of narrators who have preserved and passed it down through the generations. The linguistic style is named after the most renowned individual who mastered it, as well as his teacher’s name.

In Table 3, the teacher's name is referred to as the "Main Narrator". The most famous expert in the linguistic style is referred to as the "Narrator". Usually, dialects appear in one region and then spread to neighboring regions. We determined the region where the linguistic style appears according to the location of the "main narrator". Region data is shown by the "Place" column that defines the main region, and the "City" column.

The origin of the Quranic linguistic style can be traced back to the time of Prophet Muhammad PBUH, with individuals directly taught by him being assigned a lineage level of 1.

Those who learned from anyone from level 1 are assigned a lineage level of 2, and so on. This lineage level is denoted as "Lineage" in the table. The "Typed" column indicates whether or not we have access to an electronically typed version. The "Scanned pages" column represents the number of pages scanned from the respective Quran book. Quran book is known as Mushaf. The column "Counting Scheme" specifies the name of the Scheme used to determine the start and end of verses.

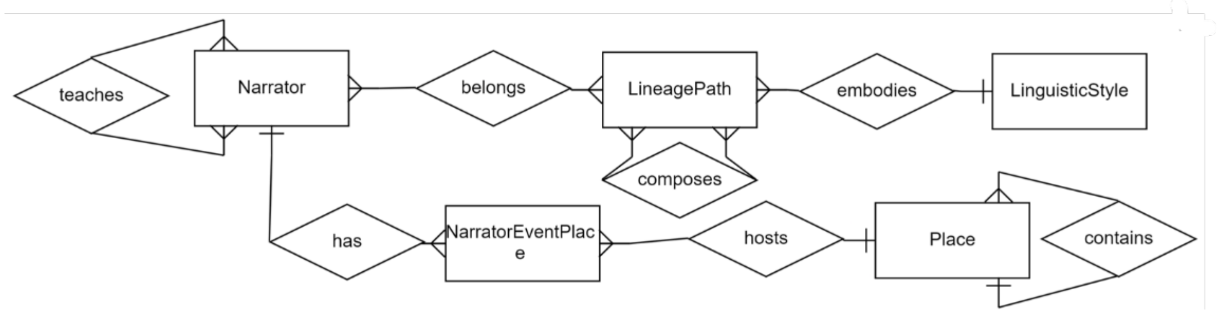


Fig. 6 *ER diagram for narrators, linguistic styles, and data lineage in the Quranic context*

Figure 6 illustrates a part of database design that captures information about narrators' data and linguistic styles of the Quran.

- Each narrator learns from one or more other narrators. Basic data about each narrator is recorded, along with some data related to events that occurred for each narrator.
- Event data related to each narrator, including their birth, transition, death, etc., is recorded in the entity 'NarratorEventPlace.' The 'Place' entity stores data about the location of each event. The temporal and spatial data of the narrators are recorded to verify their presence at the same place and time as the narrator who taught them, ensuring the authenticity of the linguistic style.
- Narrators transfer linguistic styles through 'lineage paths,' which start with the origin (the first narrator) and end with a specific narrator. Each lineage path is assigned a 'lineage level.' For example, a narrator learning directly from the origin is at level 1, while a narrator learning from a level 1 narrator is at level 2, and so on. Lineage paths can also include multiple other lineage paths
- Each narrator can be associated with multiple lineage paths, and vice versa. A linguistic style is specifically associated with one lineage path. The last narrator in that lineage path is considered the most renowned master of the particular linguistic style.

This part of the database can be regarded as a data-oriented provenance model. This model captures information about narrators, lineage paths, and linguistic styles within the Quranic narrators' network. By providing a provenance trail, offers insights into the lineage, authenticity, and evolution of linguistic styles. To visualize the tracking process, a directed acyclic graph (DAG) can be employed (Syalim et al., 2010). In a DAG, each edge has a specific direction, indicating a unidirectional flow. Importantly, a DAG is acyclic, meaning it does not contain any loops. Graphically, vertices are used to represent narrators, and edges symbolize the flow between them.

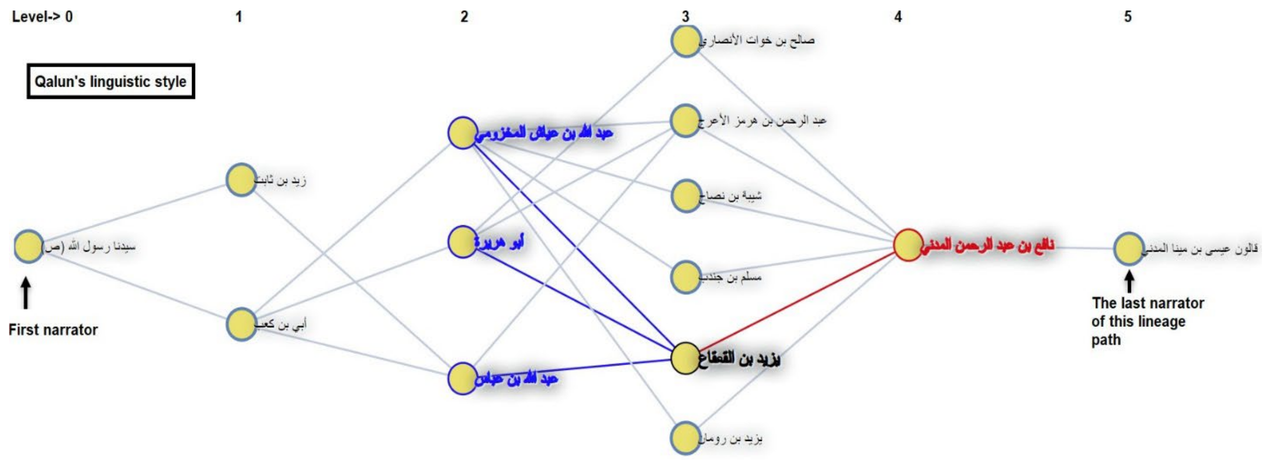


Fig. 7 *The lineage path of a linguistic style (Qalun) in the Quranic narrators' network*

Figure 7 illustrates one lineage path associated with a linguistic style. The origin (first narrator) is depicted as the initial point on the left side. Vertices connected to the right of the origin represent the subsequent levels of students. When a narrator is selected, the corresponding vertex is displayed in black. Narrators who taught the selected narrator are shown on the left in blue, while narrators who are students of the selected narrator are displayed on the right in red. A more comprehensive visualization of the interconnected network of narrators can be found on our website associated with this research project (Sabour and Ali, 2023).

4.3 Text Digitizer and Analyzer Module

The Text Digitizer and Analyzer Module addresses the challenge posed by multiple Unicode representations for Arabic characters, resulting in variations in the encoded text. In Arabic, certain characters can have different Unicode representations, resulting in variations in the encoding of the same text 8. These variations can be attributed to different font styles, regional preferences, or historical conventions.

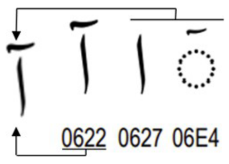


Fig. 8 *Illustration of diverse unicode representations for an Arabic character*

For example, the letter آ (*alef with madda*) can be represented by Unicode code point 0622 or by combining Unicode code points 0627 (*alef*) and 06E4 (*madda above*). Given the diverse origins and sources of the data, achieving a unified format becomes essential for meaningful comparison and analysis of the datasets. The Linguistic Styles' Provenance Tracker enables this module to incorporate linguistic style-specific rules and considerations, to apply appropriate normalization techniques. To tackle this, we have developed an algorithm known as the Arabic Character-based Normalization Algorithm: Unified Unicode Representation 1. Character-based normalization is a process of transforming characters in the text to a standardized or unified form. This algorithm 1 is an integral part of the Text Digitizer and Analyzer Module and plays a pivotal role in achieving a standardized representation of Arabic text within our system. This module 1) improves the search in the Arabic and Quranic text, 2) discovers differences between copies of the same linguistic style,2) increases the accuracy of discovering differences between different linguistic styles, and 4) creates consistent statistical results.

Algorithm 1 addresses the challenge posed by variations in Unicode encodings for Arabic characters, which can lead to inconsistencies and hinder the analysis and processing of the Arabic text. The algorithm utilizes several key data structures to facilitate the normalization process. Firstly, the "ArabicCharacterNormalizationMapping" data structure contains all Arabic characters. For each character, the structure includes its CharacterId, CharacterDescription, and the target Unicode representation based on each linguistic style referred to as "TargetUnicodeEncoding." The Unicode encoding can be a composite of multiple Unicode codes required to represent a character.

Moreover, the "CharacterEncodingAlternatives" data structure records all possible Unicode representations for each character based on the linguistic style except TargetUnicodeEncoding. Since a character can have multiple valid Unicode representations, this data structure accounts for the various possibilities.

The normalization process begins by parsing the input Arabic text. The text undergoes an evaluation within the NonStandardUnicodeEncodingDetection function. This function checks for the presence of any PossibleUnicodeEncoding within the CharacterEncodingAlternatives data structure based on the linguistic style. Whenever a PossibleUnicodeEncoding is found, the algorithm records its position in the text, creating a list known as "CharacterConversionPositionsList." This list accumulates the CharacterId and the respective position of each identified PossibleUnicodeEncoding within the text.

Following the detection of non-standard Unicode encodings, the EncodingConversionMapper function comes into play. This function maps the PossibleUnicodeEncoding found in the input text to their corresponding TargetUnicodeEncoding based on the associated CharacterId and the linguistic style. By performing this transposition, the algorithm ensures that the text adopts the unified Unicode representation specified in the ArabicCharacterNormalizationMapping data structure. The resulting transformed text is then returned as the output of the function.

Algorithm 1: *Arabic Character-based Normalization Algorithm: Unified Unicode Representation*

```
Input:   ArText: Arabic text in any Unicode representation.
         LingStyle: The type of linguistic style used in the ArText.
Output: UnifiedArText: The unified Arabic text
1  Initialize ArabicCharacterNormalizationMapping data structure;
2  Initialize CharacterEncodingAlternatives data structure;
3  function NonStandardUnicodeEncodingDetection(ArText,LingStyle);
4  begin
5      Initialize CharacterConversionPositionsList as an empty list;
6      forall PossibleUnicodeEncoding in CharacterEncodingAlternatives(LingStyle) do
7          if PossibleUnicodeEncoding is found in ArText then
8              Push characterId, position to CharacterConversionPositionsList;
9      Return CharacterConversionPositionsList;
10 function EncodingConversionMapper(ArText, LingStyle, CharacterConversionPositionsList);
11 begin
12     forall CharacterId and position in CharacterConversionPositionsList do
13         Retrieve TargetUnicodeEncoding based on CharacterId, LingStyle from
             ArabicCharacterNormalizationMapping;
14         Transpose PossibleUnicodeEncoding to TargetUnicodeEncoding in ArText;
15     Return ArText;
16 Main:
17 Read ArText in any Unicode representation;
18 Read LingStyle that used in ArText;
19 Call NonStandardUnicodeEncodingDetection with ArText and LingStyle;
20 Obtain CharacterConversionPositionsList;
21 Call EncodingConversionMapper with ArText, LingStyle, and CharacterConversionPositionsList;
22 Obtain the UnifiedArText after encoding conversion;
```

By applying the Arabic Character-based Normalization Algorithm, the Text Digitizer and Analyzer Module achieves a standardized and digitized representation of Arabic text. This unified format enables the comparison and analysis of diverse datasets, regardless of the variations in Unicode encodings.

4.4 The Data Validation and Integration Module

This module accepts the unified format of the data, as given by the Text Digitizer and Analyzer Module. This module takes advantage of Linguistic Styles' Provenance Tracker to differentiate between different styles and between multiple versions of the same style. The module brings together data from multiple linguistic styles, creating a comprehensive dataset. It then identifies and matches equivalent words, letters, and diacritical marks across styles, enabling the detection of variations and potential errors. The Data Validation and Integration Module is designed to enhance the trustworthiness (Bertino et al., 2009) and adoption of the system by ensuring the quality and consistency of the input data.

One of the algorithms within this module is the Stylistic Conflation and Text Validation Algorithm 2. Its main objective is to facilitate the mapping of words among different books that contain the same text but may differ linguistic style, or contain typos. This algorithm plays a critical role in identifying variations in word representation across different linguistic styles used in the books and detecting misspellings, missing words, or extra words in the text.

The proposed algorithm aims to facilitate the mapping of words. The primary objectives of this mapping process are: To identify variations in the representation of each word across different linguistic styles used in the books. To detect misspellings, missing words, or extra words in the text. The algorithm operates as follows:

Algorithm 2: Stylistic Conflation and Text Validation Algorithm

```
Input:  ConflatedWords, WordsToConflation
Output: ConflatedWords
while WordsToConflation.hasUnConflatedBooks do
    WordsToConflation.InitializeNextBook();
    while true do
        BestMappingCase ← WordConflationMapper(ConflatedWords, WordsToConflation);
        if BestMappingCase.AreSimilar then
            InConflationMapping.add(ConflatedWords, WordsToConflation)
        else
            if ConflatedWords.hasWord and WordsToConflation.hasWord then
                InConflationMapping.add(ConflatedWords, null);
                InConflationMapping.add(null, WordsToConflation);
            else if not ConflatedWords.hasWord and WordsToConflation.hasWord then
                InConflationMapping.addAll(null, WordsToConflation);
            else if ConflatedWords.hasWord and not WordsToConflation.hasWord then
                InConflationMapping.addAll(ConflatedWords, null);
            else if not ConflatedWords.hasWord and not WordsToConflation.hasWord then
                break;
            else
                SaveUnexpectedCase (ConflatedWords, WordsToConflation);
        ConflatedWords ← InConflationMapping;
    return ConflatedWords;
```

Initially, the words from the first book are loaded into the ConflatedWords, which serves as the base mapping for all subsequent comparisons. For each additional book, the words are loaded into WordsToConflation, representing the words to be mapped. The algorithm iterates through the process until all the books have been compared. The wordConflationMapper function is called, which returns the BestMappingCase representing the similarity between WordsToConflation and ConflatedWords. If BestMappingCase indicates that the words are similar, the similarity information between WordsToConflation and ConflatedWords is stored in InConflationMapping. If BestMappingCase does not indicate similarity, the algorithm evaluates five conditions to determine the case and perform the appropriate mapping:

- If there are words in both WordsToConflation and ConflatedWords, InConflationMapping adds the data from WordConflationMapping and ConflatedWords. The process then moves on to the next words for comparison.
- If ConflatedWords has no more words to map and there are still words in WordsToConflation, InConflationMapping adds the remaining data from WordsToConflation without corresponding words.
- If WordsToConflation has no more words and there are still words in ConflatedWords, InConflationMapping adds the remaining data from ConflatedWords without corresponding words.
- If both WordsToConflation and ConflatedWords have no more words to map, the mapping process is completed, and the algorithm exits.
- If none of the expected cases occur, the algorithm adds the situation as an unexpected case.

The “hasWord” function returns true if one or more words exist in a given set, while the “hasWords” function returns true if more than one word exists in the set. Finally, the data in InConflationMapping, representing the updated mapping between the words, is saved as the new ConflatedWords, ready to be used for subsequent comparisons.

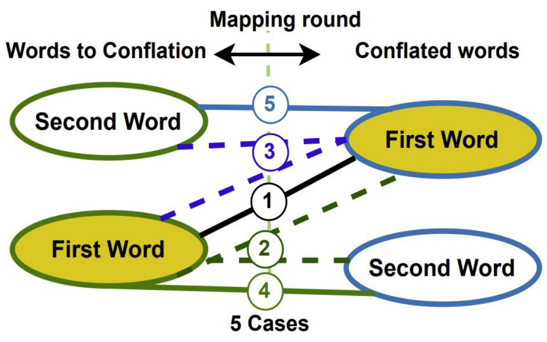


Fig. 9 *Word mapping cases for word alignment*

The Word Mapping Algorithm 3 is a part of the Stylistic Conflation and Text Validation Algorithm. Algorithm 3 is specifically designed to map two sets of words, namely ConflatedWords and WordsToConflation. The algorithm leverages the concept of Levenshtein similarity, which quantifies the degree of similarity between words based on the minimum number of operations required to transform one word into another. The Word Mapping Algorithm aims to determine the best mapping case between the words. The Word Mapping Algorithm 3 proceeds by evaluating five distinct cases of word collocations. These cases dictate how the words from WordsToConflation correspond to those from ConflatedWords, aiming to establish associations and alignments between the two sets of words. The five cases are as follows and illustrated in figure 9:

Algorithm 3: *The Word Mapping Algorithm*

```
Input:  ConflatedWords, WordsToConflation
Output: BestMappingCase
1  Function WordConflationMapper(ConflatedWords, WordsToConflation):
2      DecisionList ← EmptyList;
3      if WordsToConflation.hasWord and ConflatedWords.hasWord then
           // Case 1
4          SimilarityResult ← LevensteinSimilarity(WordsToConflation.nextWord,
             ConflatedWords.nextWord);
5          DecisionList.push(Case1, SimilarityResult);
6      if WordsToConflation.hasWord and ConflatedWords.hasWords then
           // Case 2
7          SimilarityResult ← LevensteinSimilarity(WordsToConflation.nextWord,
             ConflatedWords.nextTwoWords);
8          DecisionList.push(Case2, SimilarityResult);
           // Case 4
9          SimilarityResult ← LevensteinSimilarity(WordsToConflation.nextWord,
             ConflatedWords.secondNextWord);
10         DecisionList.push(Case4, SimilarityResult);
11     if WordsToConflation.hasWords and ConflatedWords.hasWord then
           // Case 3
12         SimilarityResult ← LevensteinSimilarity(WordsToConflation.nextTwoWords,
             ConflatedWords.nextWord);
13         DecisionList.push(Case3, SimilarityResult);
           // Case 5
14         SimilarityResult ← LevensteinSimilarity(WordsToConflation.secondNextWord,
             ConflatedWords.nextWord);
15         DecisionList.push(Case5, SimilarityResult);
16     Return DecisionList.BestDecision();
```

- **Case 1:** The algorithm examines whether the next first word in WordsToConflation corresponds to the next first word in ConflatedWords.
- **Case 2:** This case considers situations where the next first word in WordsToConflation corresponds to the next two words in ConflatedWords.
- **Case 3:** In this case, the algorithm examines whether the next first and second words of WordsToConflation correspond to the next first word in ConflatedWords.

- **Case 4:** The algorithm evaluates scenarios where the next first word of WordsToConflation corresponds to the next second word in ConflatedWords.
- **Case 5:** This case addresses situations where the next second word of WordsToConflation corresponds to the next first word in ConflatedWords.

The algorithm aims to determine the highest similarity, prioritizing the case order. Ultimately, the algorithm returns the best mapping case, encompassing information such as the case number, and the similarity.

5. Concluding Insights and Future Research

This section synthesizes the core outcomes of our research, highlighting contributions. We underscore the key findings and outline future avenues for exploration. This analysis reflects on our achievements and sets the stage for further innovations in the study of Arabic linguistic styles.

5.1 A Database for Quranic Linguistic Styles

This section describes the design and functionalities of a database built to accommodate Quranic linguistic styles.

- **Data Collection:** The system gathers text data from public online sources. This data may contain errors or multiple forms of representing letters, which we must consider.
- **Unicode Normalization:** A custom algorithm addresses Unicode representation inconsistencies, ensuring consistent text encoding into a unified format.
- **Linguistic Style Tracking:** A lineage tracking system differentiates between stylistic variations and version errors.
- **Data Validation and Integration:** integrates data from different linguistic styles, aligning corresponding words, letters, and diacritical marks to identify variations and potential errors.

The database constructed through this process provides a foundation for in-depth analysis of Quranic linguistic styles. This resource enables researchers to explore unique characteristics and offer insights into the Arabic language's evolution. Table 4 provides an overview of the database, including statistics on linguistic styles, tokens/words, characters, and diacritics.

Table 4 *Database statistics*

Number of Linguistic Styles	7
Number of Tokens/Words	542,020
Number of Characters without Diacritics	2,283,177
Number of Diacritics	2,203,148
Number of Characters in the dataset	4,486,325

5.2 The Contributions

This research makes contributions to the Natural Language Understanding (NLU) field. The contributions of this study are:

- **Process-Oriented Provenance Model:** A novel model to trace the evolution of Quranic linguistic styles.
- **Provenance Tracking of Linguistic Styles:** A system to track the provenance of Quranic styles through narrators and their lines.
- **Arabic Character Normalization Algorithm:** An algorithm ensuring consistent representation of Arabic characters across texts.
- **Data Management System for Quranic Styles:** A system for managing and integrating data regarding Quranic narrators and styles.
- **Interactive Narrators' Data Visualization Tools:** Interactive tools visualize the narrators' series.
- **Accessibility of Linguistic Research:** We have made these resources available to a wider research community through our website (Sabour and Ali, 2023).

5.3 Future Work

Our study serves as a cornerstone for future research, providing a solid base for upcoming investigations. It opens up new avenues for exploration, enabling future scholars to build upon our findings, and paving the way for new opportunities:

- **Comprehensive Linguistic Styles Database:** Expand the database to include all 20 Quranic linguistic styles.

- **Linguistic Style Detection Model:** Develop a model to automatically identify and categorize linguistic styles in Arabic texts.
- **Integration with Multilingual Data:** Incorporate language transfer learning techniques to handle multi-lingual and mixed-language texts.
- **Advanced NLP Integration:** A stylistic diacritization system with other NLP tasks like sentiment analysis and named entity recognition.
- **Diachronic Linguistic Analysis:** Perform historical analyses to trace the development of dialects linked to known linguistic styles.
- **Customization of Diacritization Models:** Implement model fine-tuning to better adapt to specific linguistic styles.
- **Integration of Audio Data:** Combine text analysis with audio recordings to enrich linguistic style analysis (Sabour et al., 2023).
- **Automated Style Classification:** Apply machine learning to automate the identification and classification of linguistic styles.

Addressing future work directions can improve diacritization systems for Arabic text, especially with diverse linguistic styles.

6. Conclusion

This research presents a novel framework for analyzing Arabic linguistic styles. The Quran, with its inherent stylistic diversity of the same text, serves as a perfect case study. The research breaks new ground by analyzing multiple linguistic styles within a single text. The framework addresses challenges like character representation inconsistencies and style lineage tracking. It achieves this through a unified text format and style-specific character mapping algorithms. Our approach, using a Provenance Tracker, enabled us to analyze and document the evolution of styles and their transmission over time. This resulted in a valuable database for further research, providing a detailed view of Arabic stylistic nuances. This research impact extends beyond Quranic studies, providing a valuable methodology for analyzing Arabic texts with stylistic variations. This framework empowers scholars to explore the evolution and transmission of these texts and develop tools for automatic style classification and variant identification in Arabic texts. Ultimately, our research enhances the understanding and appreciation of Arabic language and literature.

Acknowledgement

The authors would like to thank Computer Science and Systems, University of Washington, for their invaluable assistance and support in this study.

Conflict of Interest

Authors declare that there is no conflict of interests regarding the publication of the paper.

Author Contribution

The authors confirm contribution to the paper as follows: ***study conception and design:*** *Adel Sabour;* ***data collection:*** *Adel Sabour;* ***analysis and interpretation of results:*** *Adel Sabour, Abdeltawab Hendawi , and Mohamed Ali;* ***draft manuscript preparation:*** *Adel Sabour, Abdeltawab Hendawi, and Mohamed Ali. All authors reviewed the results and approved the final version of the manuscript.*

References

Adel Sabour and Mohamed Ali. Quran research, 2023. URL https://quranresearch.org/. Accessed: December 12, 2023.

Adel Sabour, Abdeltawab Hendawi, and Mohamed Ali. Diacritic-aware alignment and classification in arabic speech: A fusion of fuztpi and ml models. JISTech (Journal of Islamic Science and Technology), 8(2):169–191, 2023.

Ahmad Abu Dalhoum and Ahmad Al-Taani. Data analytics of arabic language: A comprehensive study. International Journal of Computer Science and Information Security (IJCSIS), 15(10):32–39, 2017.

Ali Farghaly and Khaled Shaalan. Arabic natural language processing: Challenges and solutions. ACM Transactions on Asian Language Information Processing (TALIP), 8(4):1–22, 2009.

Amril Syalim, Takashi Nishide, and Kouichi Sakurai. Preserving integrity and confidentiality of a directed acyclic graph model of provenance. In Data and Applications Security and Privacy XXIV: 24th Annual IFIP WG 11.3 Working Conference, Rome, Italy, June 21-23, 2010. Proceedings 24, pages 311–318. Springer, 2010.

Chouki Amazouz, Mohamed Khalidi Idrissi, Karim Afdel, and Mohammed Hafidi. Digital preservation of arabic heritage books: A review of the state of the art. Journal of King Saud University-Computer and Information Sciences, 32(9):1111–1119, 2020.

Colin Puri, Doo Soon Kim, Peter Z Yeh, and Kunal Verma. Implementing a data lineage tracker. In Data Warehousing and Knowledge Discovery: 14th International Conference, DaWaK 2012, Vienna, Austria, September 3-6, 2012. Proceedings 14, pages 390–403. Springer, 2012.

Devi Oktaviani, Moch Arif Bijaksana, and Ibnu Asror. Building a database of recurring text in the quran and its translation. Procedia Computer Science, 157:125–133, 2019.

Elisa Bertino, Chenyun Dai, and Murat Kantarcioglu. The challenge of assuring data trustworthiness. In Database Systems for Advanced Applications: 14th International Conference, DASFAA 2009, Brisbane, Australia, April 21-23, 2009. Proceedings 14, pages 22–33. Springer, 2009.

Hillel Ofek. Why the arabic world turned away from science. The New Atlantis, pages 3–23, 2011.

Ihab F Ilyas, Xu Chu, et al. Trends in cleaning relational data: Consistency and deduplication. Foundations and Trends® in Databases, 5(4):281–393, 2015.

Imane Guellil, Houda Saˆadane, Faical Azouaou, Billel Gueni, and Damien Nouvel. Arabic natural lan- guage processing: An overview. Journal of King Saud University-Computer and Information Sciences, 33(5):497–507, 2021.

Khaled Nasser ElSayed. An arabic natural language interface system for a database of the holy quran. International Journal of Advanced Research in Artificial Intelligence, 4(7):9–14, 2015.

Khaled Shaalan, Sanjeera Siddiqui, Manar Alkhatib, and Azza Abdel Monem. Challenges in arabic

M.N. Kadir, R. Azmi, M.F.M. Saad, N.Z.N. Zainol, and M.A.M. Yunus. A review on the manuscript of turkish quran ywj68197msi. 4 in database management system (dbms). Journal Of Quranic Sciences And Research, 1(1):34–41, 2020.

Naaima Boudad, Rdouan Faizi, Rachid Oulad Haj Thami, and Raddouane Chiheb. Sentiment classification of arabic tweets: a supervised approach. Journal of Mobile Multimedia, pages 233–243, 2017.

natural language processing. In Computational linguistics, speech and image processing for arabic language, pages 59–83. World Scientific, 2019.

Nizar Y Habash. Introduction to arabic natural language processing. Synthesis lectures on human language technologies, 3(1):1–187, 2010.

Nizar Y Habash. Introduction to Arabic natural language processing. Springer Nature, 2022.

OpenAI. GPT-4 Technical Report. Technical report, Mar 2023.

Papa Senghane Diouf, Aliou Boly, and Samba Ndiaye. Variety of data in the etl processes in the cloud: State of the art. In 2018 IEEE International Conference on Innovative Research and Development (ICIRD), pages 1–5. IEEE, 2018.

Salima Harrat, Karima Meftouh, and Kamel Smaili. Creating parallel arabic dialect corpus: pitfalls to avoid. In 18th International Conference on Computational Linguistics and Intelligent Text Processing (CICLING), 2017.

Tariq Yousef, Lisa Mischer, Hamid Reza Hakimi, and Maxim Romanov. Enhancing state-of-the-art nlp models for classical arabic. In Proceedings of the Ancient Language Processing Workshop, pages 160–169, 2023.

Umar Siddiqui, Habiba Youssef, Adel Sabour, and Mohamed Ali. Scalability, availability, reproducibility and extensibility in islamic database systems. IJASAT (International Journal on Islamic Applications in Computer Science and Technology), 10(1):12–21, 2022.

VV Khodorovskii. On normalization of relations in relational databases. Programming and Computer Software, 28:41–52, 2002.

Yogesh L Simmhan, Beth Plale, Dennis Gannon, et al. A survey of data provenance techniques. Computer Science Department, Indiana University, Bloomington IN, 47405:69, 2005.