Indonesian Journal of Electrical Engineering and Computer Science
Vol. 40, No. 3, December 2025, pp. 1486~1499
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v40.i3.pp1486-1499

QV finder: an accurate Quran verse finder system

Bashar Al-Rfooh[1], Mohammad Abdel-Majeed[1], Shorouq Al-Awawdeh[2], Khalid A. Darabkh[1]
[1]Department of Computer Engineering, The University of Jordan, Amman, Jordan
[2]Applied Science Department, Technical Collage, Middle East University, Amman, Jordan

Article Info

Article history:

Received Dec 17, 2024
Revised Oct 15, 2025
Accepted Nov 16, 2025

Keywords:

Arabic speech recognition
Automatic speech recognition
Speech recognition
String matching algorithms
Whisper

ABSTRACT

A voice-based search is becoming increasingly important for accessing information across various domains. One of the most challenging areas is Quranic verse search, where precise recitation rules (Tajweed), dialectal variations, and background noise affect accuracy. In this work, we present QV finder, an artificial intelligence (AI)-powered system that utilizes a fine-tuned whisper-based automatic speech recognition (ASR) model specifically trained on diverse Quranic recitations for the whole Quran. In this paper, we present a robust pipeline for Quranic verse retrieval that bridges the gap between ASR technology and domain-specific linguistic complexity. The model supports both professional and normal reciters, even under noisy conditions. To enhance the localization of verses from partial recitations, we integrated tokenization and advanced string-matching algorithms such as Levenshtein distance and FuzzyWuzzy. For normal reciters, the proposed model achieves a word error rate (WER) of 10.1% and character error rate (CER) of 3.3%, outperforming Google ASR, which exhibits a WER of 27.04%, and a CER of 7.13%. The model also achieves 100% verse retrieval accuracy with a 2.5% false positive rate. Our best fine-tuned model is uploaded here: https://huggingface.co/basharalrfooh/whisper-small-quran.

Corresponding Author:

Mohammad Abdel-Majeed
Department of Computer Engineering, School of Engineering, University of Jordan
Amman 11942, Jordan
Email: m.abdel-majeed@ju.edu.jo

1. INTRODUCTION

Speech is one of the most natural and effective forms of communication-both between humans and in human-computer interaction. Recent advances in automated speech recognition (ASR) have led to its widespread adoption in real-world applications such as voice search [1], transcription services [2], and virtual assistants [3]. These systems have significantly improved in accuracy, especially for general-purpose, conversational speech. However, ASR performance degrades noticeably when applied to narrow domains with non-standard pronunciation rules or distinctive acoustic patterns [4].

Domains such as singing [5] poetry recitation, and Quranic recitation [6] present unique challenges for ASR systems due to their melodic intonation, rhythm, and adherence to specific linguistic rules. For instance, singing often includes wide pitch variation, elongated phonemes, and background music, all of which interfere with accurate transcription [6]–[9]. Similarly, Quranic recitation exhibits systematic phonetic transformations governed by Tajweed rules and is characterized by melodic prosody [10], varied pacing, and regional influence factors that general-purpose ASR models are not designed to handle [11].

Quranic recitation, in particular, follows the principles of *Tajweed*, a set of phonetic and rhythmic rules that dictate how verses are to be articulated [12]. These rules introduce transformations in pronunciation that differ significantly from modern standard Arabic (MSA) speech [13]. For example, among its key rules

are *Madd* and *Idgham*. *Madd* means elongation the sound of a vowel. *Idgham* means merging letters into a letter that is not written. Additionally, local dialectal influence [14] adds another layer of complexity: while the Quran is recited in Classical Arabic, the phonetic realization by reciters is often affected by their native dialects, leading to pronunciation variations that conventional ASR models struggle to handle. This problem is further compounded by background noise, and differing recording qualities.

Existing ASR systems, including widely-used APIs such as Google ASR [15], are not optimized for these domain-specific and dialectal nuances. Our experiments show that the Google ASR model has 27.04%-word error rate (WER) for normal reciter and it is not functional for professional reciters who apply the Tajweed rule accurately.

Moreover, recent works that attempt to adapt ASR for Quranic recitation often focus on extremely limited portions of the Quran-less than 1.5% in some cases, such as in [6]. Such limitations restrict these models' generalizability and their usefulness for real-world Quranic search applications. In these systems, transcription errors can directly hinder the ability to locate the correct verse or Surah [16].

In this paper, we introduce QV finder, an Artificial intelligence (AI)-powered, open-source ASR system specifically designed for the transcription and retrieval of Quranic verses. Unlike general-purpose ASR models, QV Finder is trained on diverse Quranic recitations and is linguistically informed by Tajweed principles. The system supports both professional and normal (amateur) reciters, with or without strict Tajweed application, and performs robustly under noisy conditions. Moreover, to avoid comprehensive search methods, our trained ASR model output will be fed to post processing steps that include tokenization and the most recent string-matching algorithms to accurately locate the verse given only a small part of it and reduce false positive results. Also, in our model we tested the impact of the two stages search strategy in improving the string-matching accuracy. By leveraging domain-adaptive training, dialect-aware modeling, and advanced acoustic techniques, our model significantly improves transcription fidelity and enables accurate retrieval of verse text, Surah names, and verse numbers from partial or full verse recitations.

The remainder of this paper is structured as follows: section 2 reviews related work in narrow-domain ASR and Quranic recitation processing. Section 3 details the dataset, preprocessing steps, the architecture and training methodology of QV Finder. Section 4 presents experimental results and evaluations. Finally, section 5 concludes the paper and highlights directions for future research.

2. RELATED WORK

Speech recognition has been the focus of researchers for a long time. The proposed solutions have varied from advanced algorithms like mel-frequency cepstral coefficients (MFCC) and hidden markov model (HMM) [17] to the deployment of deep neural architectures like recurrent nural networks (RNNs) [18], long term-short memory (LSTM) [19], Transformers [20], and attention techniques [21].

Siddique *et al.* [22] presented an overview of different models of transformer-based architecture for various speech-processing tasks. These models are pre-trained on a large dataset and have demonstrated high performance. Table 1 summarizes the transformer-based models. Whisper model [23] is a revolutionary multilingual model that has achieved high accuracy on various speech-related tasks including ASR and has proved to work well in noisy environments.

Table 1. Different transformer-based models used for speech processing purposes [22]

Model	Main used	Architecture
Wav2vec	Self-supervised learning for ASR	Convolutional neural network (CNN) front end and transformer encoder
Data2vec	General-supervised learning model	Unified architecture for different modalities like vision, text, and speech
VALL-E	Text-to-speech (TTS) synthesis	Transformer-based model with fixed speech tokens
Tacotron	TTS synthesis	Seq-to-Seq architecture with attention
Whisper	Multilingual ASR and translation	Transformer-based encoder-decoder model
Conformer	ASR	A hybrid of CNNs and transformer
UniSpeech	Unified model for ASR and speech texts	Transformer-based leveraging self-supervised learning
Speechformer	Speech enhancement and recognition	Transformer-based with multi-scale speech modeling
WavLM	Self-supervised pre-trained model for ASR speech enhancement, and recognition	Transformer-based built on wav2vec 2.0

Arabic speech recognition, as one of the most popular languages spoken by more than 382 million people (https://www.ethnologue.com/language/ara/), Arabic has received high attention in the speech recognition domain across various aspects. Developing Arabic ASR models is expected to have a positive impact on many sectors like banking, healthcare, and digital assistants.

Wazir and Chuah [24] applied deep learning approaches to speech recognition. They utilized MFCC to extract features from the dataset for spoken digits. Then the extracted features have been used by LSTM. The LSTM network achieved an accuracy of 94%. In addition, [25] studied the performance of deep neural networks (DNNs) in Arabic phonemes recognition. For feature extraction, they proposed using the Maxout activation function and the MFCC. They suggested a dropout function that experimentally gains high performance compared with both ReLU and Sigmoid activation functions. As a result, the deep architecture utilizing Maxout gave impressive results, where the error rate was lower than other DNNs, including restricted boltzmann machines (RBM), deep belief networks (DBN), CNNs, tensor factorization neural networks (TFNN), and convolutional autoencoders (CAE). Emami and Mangu [26] conducted a comprehensive study on a variety of configurations of neural models, normalization procedures, model size, and other parameters. They utilized two broadcasts such as Arabic news and conversational. The improved NN model showed better improvements than the 4-gram base model with a reduction of up to 0.8% and 3.8% reduction in WER. The changes in these parameters had a small but positive effect on the model performance. Sameer *et al.* [27] presented a transformer-based approach for Arabic ASR. It employed the self-attention mechanisms to accurately convert Arabic speech to text. The model trained on the collected dataset from Mozilla's Common Voice for 112 hours for 110 epochs. It achieved a 3.2% WER, which outperformed the traditional version of ASR such as LSTMs and RNNs. In summary, the transformer will yield good results when using a small amount of data.

The recognition of Arabic speech not only focused on the MSA but also got extended to narrow domains like local dialects, songs, and Quran recitation where the words are pronounced differently. For example, Elmahdy *et al.* [17] developed an Arabic speech recognition system using a creative multilingual method. This method includes different acoustic models based on the HMM. The processes of training and testing included a collection of voice datasets from news broadcasts that contained MSA and Egyptian Arabic dialects. Where their system achieved an accuracy of 99.34%. Nasr *et al.* [28] collected a new dataset with different languages such as Jordanian, Yamani, and Arabic dialects. In addition, they constructed many sequence-to-sequence DNN models for end-to-end recognition, especially for Arabic dialects. Moreover, they trained their collected dataset on Mozilla's DeepSpeech2 model from scratch. The WER and character error rate (CER) of the Yamani speech dataset based on the Bidirectional LSTM (Bi-LSTM) were 59% and 51%, which were good results. For the Jordanian speech dataset using the previous architecture achieved 83% and 70% for WER and CER, respectively. On the different dialectal Yem-Jod-Arab speech, the WER and CER were 53% and 39%. Using the DeepSpeech2 model, the Yamani dataset achieved better results compared with the base model by 31% for WER and 24% for CER. Additionally, the Jordanian dataset achieved 68% and 40 for WER and CER, respectively. As a result, a multi-dialect Arabic outperformed and achieved better results with 30% WER and 20% CER.

There are several works with different targets have been proposed for analyzing the audio files of the Quran recitation using machine learning and deep learning algorithms. For example, the authors in [29], [10] developed a model to detect if the recitation follows the Quran recitation rules. Also, the authors in [30] developed a model to detect and identify the Quran recitation rule itself. In addition, the authors in [31] developed a model to detect the reciter themselves. Nahar *et al.* [32] developed a recognition model to identify the different types of "Qira'ah" (types of recitation) from audio waveforms of the Holy Qur'an. The proposed model extracted the MFCC features and trained the generated data on the SVM model. The model achieved 96% accuracy. Mohammed *et al.* [6] used the transformer-based model to recognize the recited verse from the Quran and convert it to text. They were able to achieve a CER of 1.98% and a WER of 6.16%. However, in their study, they focused on short chapters that cover less than 0.8% of the Quran.

Also, several works have been proposed on locating verses in the written text [33], [34] using different string search and matching techniques. For example, Abokhodair *et al.* [33] suggested an algorithm to find the Quranic verse from tweets. Where it segments a tweet into sentences and finds if each sentence is from the Quranic verses or not *i.e.*, in both forms full and partial verses. In this study, there were no specific techniques for how to find the actual limits of verses, and how to detect the partial verses. In addition, no error detection and correction are mentioned there. El-Beltagy and Rafea [35] proposed a creative system (QDetect) specially designed for detecting and extracting Quranic verses from any written Arabic text. QDetect aims to find an accurate quotation from the Quran, which is essential for citation purposes in academic and other writings. To match the input text accurately with the verse while taking into consideration different spellings, formatting, and wording, the QDetect utilizes a string-matching algorithm. The tool that supports different languages can identify the Quranic verses in different transliterations and languages. However, the proposed matching algorithm is slow and does not apply heuristics to speed up the search and matching process.

3. METHOD

The purpose of this work is to facilitate the search for verses from the Quran using human voice input. To be able to achieve this accurately we followed the procedure shown in Figure 1. As shown, Firstly, we will start with the data collection and preparation as required by the following modules. Secondly, we will discuss the used ASR model and the fine-tuning techniques used to achieve the best accuracy. Thirdly, we will pass the generated text from the ASR model to the search engine that will find the best matching verse with the minimum false-positive cases. Finally, we will evaluate the developed procedure and measure its performance.

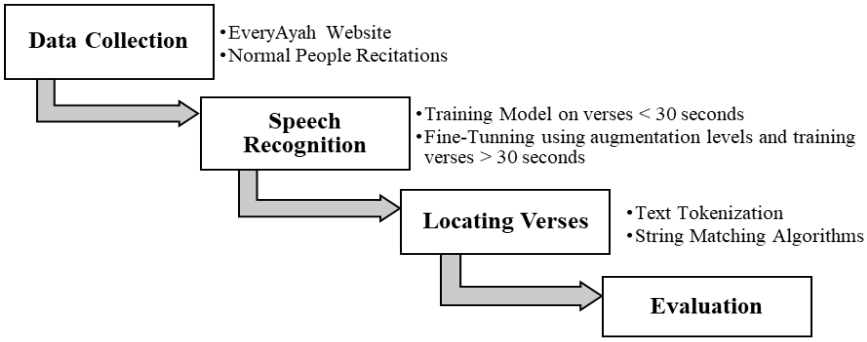

Figure 1. Flow chart of our methodology

3.1. Data collection

3.1.1. Reciters dataset

The dataset used in this study was collected from EveryAyah [36], a website that provides recitations of the Quran. The dataset consists of 64 reciters, with one reciter reserved for validation and another for testing purposes. The data obtained for each reciter includes the recitation for all Quran verses. The website is open-source and includes several types: Quran text, Quran text images (JPG), Quran text high-resolution (PNG), and Quran timing files. Therefore, the user can download full recitations in ZIP format, making it easy to use and access [36]. To download the verses as an XML file and .wav files from the website, we wrote a script code *i.e.*, using the Python programming language, where an XML file allows us to extract information like Sura_id, Sura_name, Ayha_text, and Ayha_id, as shown in Figure 2. After extracting Sura information and making a list of tuples to save verses, the script code splits the .wav files into two categories: shorter than 30 seconds files and longer than 30 seconds, then maps the text with the path to the .wav file as .tsv path, as shown in Table 2. Dividing the file based on the length in seconds is according to the limitations enforced by the ASR model that we are using, as we will discuss later.

```
▼<sura id="1" name="الفاتحة">
   <aya id="0" text="أعوذ بالله من الشيطان الرجي"/>
   <aya id="1" text="بِسْمِ اللهِ الرَّحْمنِ الرَّحِيمِ"/>
   <aya id="2" text="الْحَمْدُ للهِ رَبِّ الْعالَمِينَ"/>
   <aya id="3" text="الرَّحْمنِ الرَّحِيمِ"/>
   <aya id="4" text="مَالِكِ يَوْمِ الدِّينِ"/>
   <aya id="5" text="إِيَّاكَ نَعْبُدُ وإِيَّاكَ نَسْتَعِينُ"/>
   <aya id="6" text="اهدِنَـــا الصِّرَاطَ المُستَقِيمَ"/>
   <aya id="7" text="صِرَاطَ الَّذِينَ أَنعَمتَ عَلَيهِمْ غَيرِ المَغضُوبِ عَلَيهِمْ وَلاَ الضَّالِّينَ"/>
 </sura>
```

Figure 2. A sample of XML file for surah Al-Fatiha

Table 2. A sample of a .tsv file after finalizing collected the dataset from the EveryAyah Website

Verse path	Verse
/data/Ali_Jaber_64kbps_wav/018014.wav	وَرَبَطْنَا عَلَى قُلُوبِهِمْ إِذْ قَامُوا فَقَالُوا رَبُّنَا رَبُّ السَّمَاوَاتِ وَالْأَرْضِ لَن نَّدْعُوَ مِن دُونِهِ إِلَٰهًا لَّقَدْ قُلْنَا إِذًا شَطَطًا
/data/Ali_Jaber_64kbps_wav/007191.wav	أَيُشْرِكُونَ مَا لَا يَخْلُقُ شَيْئًا وَهُمْ يُخْلَقُونَ
/data/Ali_Jaber_64kbps_wav/026042.wav	قَالَ نَعَمْ وَإِنَّكُمْ إِذًا لَّمِنَ الْمُقَرَّبِينَ
/data/Ali_Jaber_64kbps_wav/042001.wav	حم

Figure 3 shows the distribution of the verses based on their words count in the collected dataset. The number of verses in the Quran is 6238, where the maximum and the minimum verses contain 129 words and 1 word, respectively. Therefore, there is no verse with zero length. The average number of words per verse is 12.4687. In total, the dataset from 64 reciters contains 399,232 verses, out of which 56,792 verses are shorter than 30 seconds. The maximum length of a single verse is 936.2 seconds, with an average verse length of 18.2 seconds. The total number of hours of Quran recitation is 2023, out of which 1211 hours are recited for verses shorter than 30 seconds. In comparison, the length of the dataset used in the most recent work [6] is 10 hours and only covers less than 1.5% of the quran verses. Table 3 shows the dataset split for the training, validation, and testing phases depending on the number of hours.

Table 3. Dataset distribution by number of hours

Dataset	Number of hours	Shorter than 30 seconds
Train	1,970	1,173
Validation	28	19
Test	25	19

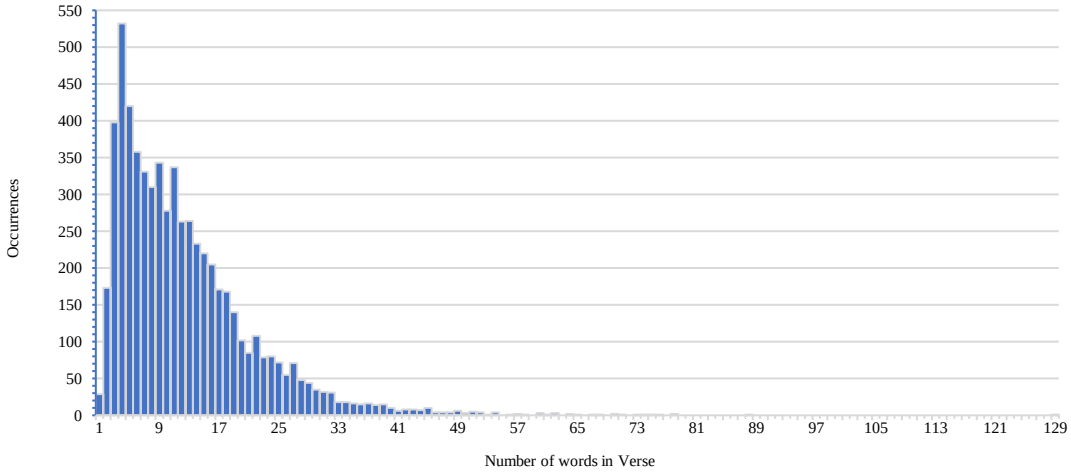

Figure 3. A histogram of statistics of the number of words repeated in each verse

3.1.2. Normal people recitations

Since our specific use case revolves around normal people reciting the Quran, we collected an additional dataset for normal people reciting the Quran. Usually, normal people apply part of the Tajweed rules and their pronounciation style is closer to the normal speech. We have conducted manual annotation on a set of 800 verses. These verses were selected from eight different reciters, with each reciter contributing 100 randomly chosen verses. After cleaning the files manually from human errors such as incorrect Tajweed, word spellings, and grammatical blunders, the total number of annotated audio files was reduced to 771 verses with a total duration of two hours.

3.1.3. Speech recognition model

The conversion of spoken language into written text, known as speech recognition, has become an integral part of various domains, from transcription services to voice-controlled virtual assistants. The Whisper model Voice-to-Text AI, a state-of-the-art deep learning model, stands at the forefront of this field, revolutionizing the accuracy and efficiency of speech recognition systems. In today's fast-paced world, the ability to convert spoken language into written text efficiently and accurately has become increasingly important. The Whisper model Voice-to-Text AI leverages the power of deep learning to bridge this gap, transforming the way we interact with voice-based systems and enabling seamless integration of speech and written text [23].

3.1.4. Model architecture

The Whisper model's architecture is based on a transformer network. The model's encoder processes the input audio features, transforming them into a series of hidden representations. This step is

essential for understanding the complex temporal dependencies in speech. Then the decoder uses these representations to generate text, word-by-word, similar to how a language model might generate text based on context. Whisper employs self-attention and cross-attention layers, which permit the model to concentrate on different parts of the input sequence when making predictions [23].

There are various sizes of the Whisper model with different parameters for each. As shown in Table 4, the number of parameters is increased as the model size is increased *i.e.,* a more complex model needs more advanced hardware requirements and longer inference time.

Table 4. Model size and corresponding parameters [23]

Size	Parameters	Required VRAM	Relative speed
Tiny	39 M	~1 GB	~32x
Base	74 M	~1 GB	~16x
Small	244 M	~2 GB	~6x
Medium	769 M	~5 GB	~2x
Large	1,550 M	~10 GB	1x

3.1.5. Model training and fine tuning

In this sub-section, we will discuss the training strategy that we followed to train the whisper model. The dataset is divided into two categories: verses shorter than 30 seconds and verses longer than 30 seconds. We selected 30 seconds because Whisper was trained using 30-second segments, so sticking close to that length provides the most accurate transcriptions. Hence, the model is trained on the verses with a length of less than 30 seconds. We conducted our training on the first three sizes: tiny, base, and small. We will present the results for each model in detail in the evaluation and results section.

To ensure the robustness and generalization of the model, various techniques are employed, including regularization methods and data augmentation. Regularization techniques, such as dropout and weight decay, prevent overfitting and enhance the model's ability to generalize well to unseen data. Data augmentation techniques, such as adding background noise or altering pitch and speed introduce variability to the training data, enable the model to handle different acoustic conditions and speaking styles to achieve accurate and reliable speech recognition capabilities [37].

Although, the Whisper inference allows for audio files longer than 30 seconds, our model currently struggles to perform well on verses exceeding this length. This is primarily due to the limited vocabulary of our model, which lacks certain phrases explicitly mentioned in longer verses of the Quran. As a result, the model faces difficulties transcribing unfamiliar words that are produced when segmenting the longer verses into verses that are less than 30 seconds. In some cases, the segmentation limit has divided the word into two parts, which means that there is a mismatch with a word in other verses, resulting in reduced model performance. In the results section we will present, how to deal with verses longer than 30 seconds.

3.2. Locating verses

After converting the spoken words to text, the model should find the matching verse/s accurately. To achieve this, we applied tokenization to easily search and access the corrected verse/s, and string matching algorithms such as the Levenshtein metric and FuzzyWuzzy. So, the matching verse will be selected based on the similarity percentage between the true verse and the resulting verse through these techniques. Our goal is to reduce the false positive results for each string-matching algorithm while achieving high accuracy. In this section, we will discuss the procedure we followed to locate verses accurately and quickly.

3.2.1. Text tokenization

Regardless of the algorithm used to identify the target text in the generated output from Whisper's ASR, all string-matching algorithms are designed to handle text of approximately the same length. However, in our case, we face a challenge, as we need to find specific verses that can vary significantly in length. The ASR-generated text may consist of just a few words, while the verses we are searching for can span several tens of words. For example, the spoken words may be is "عَفَا ٱللَّهُ عَنْهَا" and we expect the system at the end to locate it and return the verse which is verse 101 from Surat Alma'idah " يَـٰٓأَيُّهَا ٱلَّذِينَ ءَامَنُواْ لَا تَسْـَٔلُواْ عَنْ أَشْيَآءَ إِن تُبْدَ ل لَكُمْ تَسُؤْكُمْ وَإِن تَسْـَٔلُواْ عَنْهَا حِينَ يُنَزَّلُ ٱلْقُرْءَانُ تُبْدَ لَكُمْ عَفَا ٱللَّهُ عَنْهَا ۗ وَٱللَّهُ غَفُورٌ حَلِيمٌ".

To address this issue, we collected the duplicated verses in a tab-separated file (tsv) format. The first column contains the duplicated verse if there are any duplications, or the unique verse if not. The second column contains the indexes where this verse appeared. This process is known as tokenization the files, where we divide the text into tokens ranging from single words to groups of ten words using the sliding window technique that slides by one for each row of content. Then, we applied this tokenization technique to

both individual words and complete verses. When searching for a specific verse or sub-verse, we calculate the number of words generated by the ASR and load the corresponding tokenized file. This allows us to match approximately similar-length texts, facilitating the identification of the desired verses with reasonable accuracy. Table 5 summarizes the words' occurrences in the token from one word to the full verse.

Table 5. Duplicate distribution based on number of words in the token

Number of words in the token	Duplicates
1	50,802
2	7,285
3	4,202
4	2,407
5	1,406
6	854
7	526
8	346
9	235
10	170
Full verses	48

To enhance the verse localization capability, we will employ tokenized files in conjunction with all string-matching algorithms. This approach allows us to accurately locate verses, even if the transcribed text only contains a small segment of the full verse.

3.2.2. String matching algorithms

The string-matching algorithms are applied in the following manner: First, we obtain the transcribed text generated by the ASR system. This transcribed text is then compared with all the verses present in the Quran using Levenshtein distance and fuzzy string matching. The algorithm helps identify the best match among the verses. By referring to the tokenized files, we can extract the indexes of all the occurrences of this matched verse.

Levenshtein distance: also known as edit distance, is a measure used to quantify the difference between two strings. It measures the minimum number of single-character insertions, deletions, and substitutions required to transform one string into another. The Levenshtein distance is widely used in various fields, including natural language processing (NLP), spell-checking, and DNA sequence analysis [38]. The distance is calculated by considering each character in both strings and determining the operations needed to transform one string into the other. Insertions involve adding a character, deletions involve removing a character, and substitutions replace one character with another. The total Levenshtein distance is the sum of these operations. The Levenshtein distance provides a quantitative measure of string similarity, allowing for effective comparison and analysis. Its simplicity and versatility make it a popular choice for various text-related tasks and algorithms [38].

FuzzyWuzzy: is a Python library widely used in research papers for fuzzy string matching and similarity calculations. It employs various algorithms and methods to measure the similarity between two strings by considering factors such as character substitutions, insertions, deletions, and rearrangements. This algorithm calculates the minimum number of edits required to transform one string into another, which can include substitutions, insertions, and deletions of individual characters. FuzzyWuzzy provides several methods for string matching and similarity scoring. These methods enable researchers to assess the degree of similarity between two strings based on different aspects of the comparison.

The method we used offered by FuzzyWuzzy is the ratio; this method compares two strings and returns a similarity score ranging from 0 to 100. It calculates the Levenshtein distance between the strings and normalizes it based on their length. A higher ratio score indicates a higher degree of similarity between the strings. By employing this method, FuzzyWuzzy provides researchers with the ability to conduct fuzzy string matching and similarity calculations in a flexible and efficient manner. Its rich functionality contributes to a wide range of research applications, including data analysis, text mining, and information retrieval. Overall, the FuzzyWuzzy algorithmic framework, combined with its various similarity-scoring methods, empowers researchers to perform accurate and comprehensive fuzzy string matching [39].

4. RESULTS AND DISCUSSION

4.1. Environmental setup

Our experiments on the Whisper model were implemented on Google Cloud using the A100 GPU accelerator and 80 GB memory with the following parameters: Adam optimizer, 32 batch size. 3,407 seed,

0.0005 learning rate, 0.01 weight decay, 1×10^{-8} adam epsilon, 2 warmup steps, 4 workers, 11 train epochs, and 1 gradient accumulation step.

4.2. ASR results

As a first step, we tested the pre-trained Google ASR model on recitations from normal and professional reciters. For normal reciters, where they do not stick to the Tajweed rules the WER were 27.04% and the CER of 7.13%. On the other hand, for the professional reciters the ASR results were far away from the correct text due to the application of complex Tajweed rules where the spoken letters are different from the written letter for some of the rules as discussed earlier.

Hence, to achieve the best results, we conducted several model training and fine tuning experiments started based on model sizes, augmentation levels, training for longer than 30 seconds, and finally locating verses with several key points. The Whisper model has a constraint on the maximum length of audio files that can be trained, which is limited to 30 seconds. Consequently, we encountered approximately 543 verses that exceeded this duration, depending on the reciter. Out of the total 399,232 verses, we had across all reciters, 56K verses fell into this category. Therefore, it is important to note that the results obtained from training Whisper are primarily applicable to verses shorter than 30 seconds unless otherwise specified. Figure 4 shows the outcomes obtained from training Tiny, Base, and small Whisper models on verses that are shorter than 30 seconds and tested on professional reciter with less than 30 second audios. The results for the three Whisper model sizes are closely aligned, primarily due to overfitting. Our model tends to overfit across the three sizes, resulting in similar outcomes. Subsequently, the models were evaluated on a test dataset consisting of 771 verses recited by normal individuals.

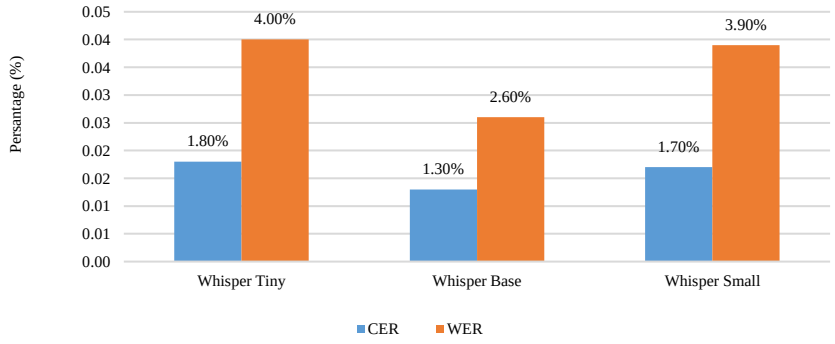

Figure 4. Trained-whisper results for different sizes tested with a professional reciter

Figure 5 illustrates the outcomes obtained from training various Whisper models on three different sizes using verses that are shorter than 30 seconds, which were tested on recitations by normal reciters, all under 30 seconds in length. While the base size model demonstrated the best performance among the three sizes when tested on professional reciter samples, our specific use case revolves around normal people reciting the Quran. Therefore, we have selected the small model as our primary model moving forward.

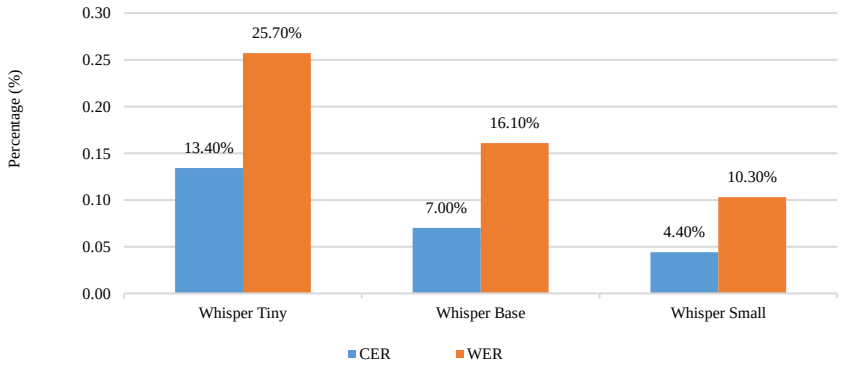

Figure 5. Trained-whisper results for different sizes tested with normal reciters

Since our model is trained on professional recitations, but is expected to perform well on normal people's recitations, we trained the model with and without augmentations. Augmentation techniques were employed to introduce errors and variations found in real-life recitations into the training dataset. This approach helps the model become more robust and better equipped to handle such errors during inference. Table 6 presents the results of testing the model with and without augmentation, considering both professional reciters and normal Quran reciters. As shown in the table, the applied augmentation techniques successfully reduced overfitting, leading to improved CER and WER for professional reciters. Also, it significantly improved the CER and WER for normal reciters. Consequently, we have selected the model with augmentation as our primary model going forward.

Table 6. ASR performance comparison with and without augmentation

	Dataset type	CER	WER
With augmentation (%)	Professional reciter	4.1	9.9
	Normal reciters	3.3	10.1
Without augmentation (%)	Professional reciter	10.3	14.2
	Normal reciters	4.4	10.3

Although the Whisper inference allows for audio files longer than 30 seconds, our model currently struggles to perform well on verses exceeding this length. This is primarily due to the limited vocabulary of our model, which lacks certain phrases explicitly mentioned in longer verses of the Quran. As a result, the model faces difficulty transcribing unfamiliar words. The CER and WER of verses longer than 30 seconds were 12.5% and 24.5%, respectively. To address this issue, we have implemented the following steps as a solution,

- Segmentation of audio files longer than 30 seconds: We divided these files into segments that do not exceed 30 seconds based on periods of silence within the audio files.
- Manual labeling of 5000 segmented verses: To expand our model's vocabulary and enable it to recognize new words, we have manually labeled 5,000 segmented verses. The manual labeling guarantee that the words have not been split in the middle.
- Incorporating newly labeled audio segments: We merged the newly labeled audio segments with the existing dataset of shorter than 30 seconds audio files. This step allowed us to enhance the model's performance by incorporating the additional labeled data into the training set.

Figure 6 shows the result after applying the previous steps and supporting the training set with extra 5,000 samples. As shown, the proposed steps resulted in a notable 40% reduction in CER and a remarkable 21.6% decrease in WER. We can refer this improvement to the extra vocabulary included in the training set. This substantial improvement played a crucial role in accurately locating verses, thereby enhancing the overall effectiveness of our system.

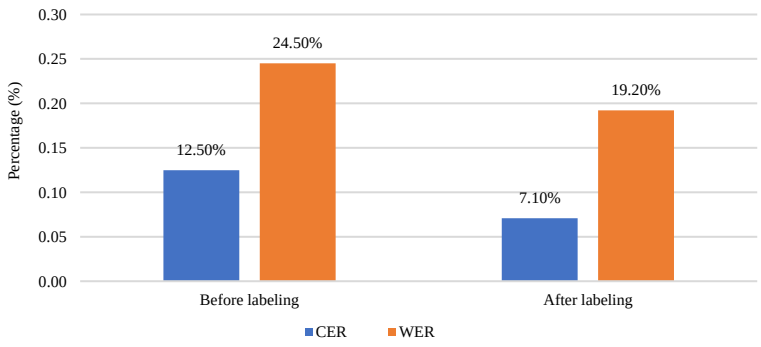

Figure 6. ASR performance for whisper small model for verses longer than 30 seconds before and after segmenting long verses

4.3. Verse localization results

Through the testing phase, we utilized a threshold approach to extract multiple matches that meet a certain similarity threshold. However, it is important to note that this approach may lead to false positive results, where some extracted matches may not correspond to the intended verse. Careful consideration and

analysis are necessary to minimize such occurrences and improve the accuracy of the matching process. The results of locating verses using Levenshtein distance and the FuzzyWuzzy library are shown in Table 7. The results demonstrate the outcomes obtained when extracting the exact best match for a given query. As shown, for verses shorter than 30 seconds the localization accuracy is 97.1% and 95.5% for professional and normal reciters when FuzzyWuzzy is used. On the other hand, the localization accuracy for verses longer than 30 seconds is 92%.

Table 7. Accuracy of verse localization for professional and normal reciters

Metric	Reciter type	Verses > 30 (%)	Verses < 30 (%)
Levenshtein distance	Professional	90.2	95.3
	Normal	-	91.3
FuzzyWuzzy	Professional	92.0	97.1
	Normal	-	95.5

However, it is important to note that this approach may not always yield accurate results due to the presence of errors in our original text. To address this limitation, we introduced a ratio-based method wherein we consider all verses that surpass a specific matching threshold. If the desired verse is found among the selected verses (above the matching threshold), we consider it a correct match. Conversely, any additional extracted verses beyond the desired one are classified as false positives. Setting the threshold ratio in FuzzyWuzzy to 85% gave a remarkable improvement in accuracy across all categories, achieving 100% accuracy with a small false positive rate. Table 8 presents the false positive percentages results obtained in FuzzyWuzzy with a threshold of 85%. The false positive rate within each category is determined by the percentage of extractions using the 85% threshold, minus the percentage obtained when considering only the best match.

Table 8. False positive percentages in verse localization with 85% threshold

Metric	Reciter type	Verses > 30 (%)	Verses < 30 (%)
Levenshtein distance	Professional	4.6	2.5
	Normal	-	4.1
FuzzyWuzzy	Professional	8.0	2.9
	Normal	-	4.5

The selection of the FuzzyWuzzy library was based on its superior performance in comparison to the Levenshtein distance algorithm. FuzzyWuzzy consistently produced better results and demonstrated a lower rate of false positive answers. This made it the preferred choice for our application, providing more accurate and reliable outcomes.

To mitigate the issue of false positive results generated by the search with a threshold approach, we implemented a two-stage search mechanism *i.e.,* using the FuzzyWuzzy library. In this approach, we employed a fast search algorithm in the first stage to identify potential candidate verses. These candidates were then subjected to a slower but more accurate search algorithm in the second stage, resulting in the selection of a single verse with significantly reduced false positive occurrences. As shown in Figure 7, this two-stage search strategy effectively enhances the precision and reliability of our verse localization system.

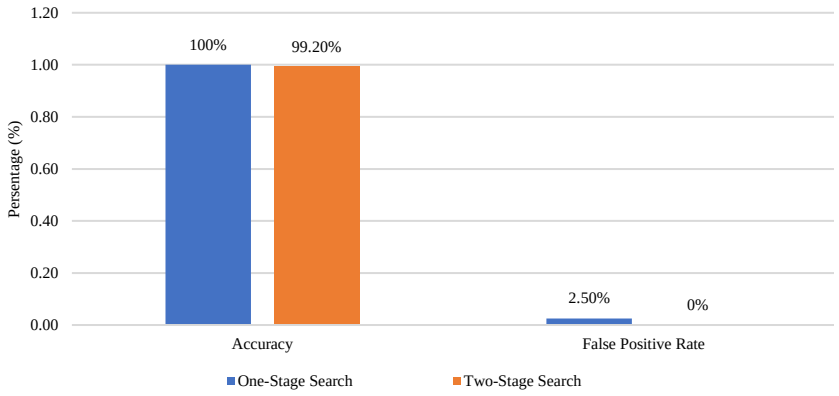

Figure 7. Verse localization performance comparison between one-stage and two-stage search mechanisms

5. CONCLUSION

In this work, we introduced QV finder, a novel AI-based Quran verse retrieval system that effectively bridges the gap between traditional speech recognition techniques and the unique linguistic characteristics of Quranic recitation. By leveraging the Whisper architecture and training it on a large and diverse dataset of Quranic audio from both professional and normal reciters, we successfully enhanced the system's ability to transcribe Quranic speech accurately under a variety of acoustic conditions. Our model demonstrated superior performance over general-purpose systems such as Google ASR, achieving significant reductions in CER and WER. Beyond transcription, we developed a robust pipeline for verse localization by integrating tokenization and powerful string-matching algorithms, namely Levenshtein distance and FuzzyWuzzy. These methods, particularly when enhanced with a two-stage search strategy, yielded highly accurate verse matching with minimal false positives. Notably, our approach achieved 100% verse retrieval accuracy under a threshold-based search with a low false positive rate of just 2.5%. The proposed methodology opens the path for using the same procedure to narrow ASR domains like poetry and singing and local dialectics.

As a future work, we can expand the functionality of the system to provide real-time feedback on Tajweed compliance and support searching the verses not by their pronunciation but by their context and meaning and broader semantic understanding.

ACKNOWLEDGEMENTS

We would like to express our sincere gratitude to Maqsam for providing us with the Google Cloud credits to train our machine learning model.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Bashar Al-Rfooh	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	
Mohammad Abdel-Majeed	✓	✓		✓	✓	✓				✓	✓	✓	✓	
Shorouq Al-Awawdeh		✓			✓					✓	✓		✓	
Khaled A. Drabkeh		✓			✓					✓				

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

INFORMED CONSENT

We have obtained informed consent from all individuals included in this study.

DATA AVAILABILITY

- The data that support the findings of this study are openly available in Hugging Face at `https://huggingface.co/basharalrfooh/whisper-small-quran`.

REFERENCES

[1] D. Yu and L. Deng, *Automatic speech recognition*. Berlin: Springer, 2016.

[2] J. A. Thuestad and Ø. Grutle, "Speech-to-text models to transcribe emergency calls," Univ. of Bergen, 2023.

[3] S. Alharbi *et al.*, "Automatic speech recognition: systematic literature review," *IEEE Access*, vol. 9, pp. 131858–131876, 2021, doi: 10.1109/ACCESS.2021.3112535.

[4] S. Haboussi, N. Oukas, T. Zerrouki, and H. Djettou, "Arabic speech recognition using neural networks: concepts, literature review and challenges," *Journal of Umm Al-Qura University for Applied Sciences*, 2025, doi: 10.1007/s43994-025-00213-w.

[5] A. Kruspe, "More than words: Advancements and challenges in speech recognition for singing," 2024, [Online]. Available: http://arxiv.org/abs/2403.09298.

[6] M. Hadwan, H. A. Alsayadi, and S. AL-Hagree, "An end-to-end transformer-based automatic speech recognition for Qur'an reciters," *Computers, Materials and Continua*, vol. 74, no. 2, pp. 3471–3487, 2023, doi: 10.32604/cmc.2023.033457.

[7] R. U. Khan, A. M. Qamar, and M. Hadwan, "Quranic reciter recognition: a machine learning approach," *Advances in Science, Technology and Engineering Systems Journal*, vol. 4, no. 6, pp. 173–176, 2019, doi: 10.25046/aj040621.

[8] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: a neural network for large vocabulary conversational speech recognition," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2016, vol. 2016-May, pp. 4960–4964, doi: 10.1109/ICASSP.2016.7472621.

[9] A. Rahman, M. M. Kabir, M. F. Mridha, M. Alatiyyah, H. F. Alhasson, and S. S. Alharbi, "Arabic speech recognition: advancement and challenges," *IEEE Access*, vol. 12, pp. 39689–39716, 2024, doi: 10.1109/ACCESS.2024.3376237.

[10] A. M. Alagrami and M. M. Eljazzar, "SMARTAJWEED automatic recognition of Arabic quranic recitation rules," pp. 145–152, 2020, doi: 10.5121/csit.2020.101812.

[11] F. Morbini *et al.*, "Which ASR should i choose for my dialogue system?," in *SIGDIAL 2013 - 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue, Proceedings of the Conference*, 2013, pp. 394–403, [Online]. Available: https://aclanthology.org/W13-4064/.

[12] A. H. Ishaq and R. Nawawi, "The science of Tajweed and its implications for the science of Qira'ah (Ilmu Tajwid Dan Implikasinya Terhadap Ilmu Qira'Ah)," *Qof*, vol. 1, no. 1, pp. 15–24, 2017, doi: 10.30762/qof.v1i1.926.

[13] B. H. A. Ahmed and A. S. Ghabayen, "Arabic automatic speech recognition enhancement," *Proceedings - 2017 Palestinian International Conference on Information and Communication Technology, PICICT 2017*, pp. 98–102, 2017, doi: 10.1109/PICICT.2017.12.

[14] H. A. Alsayadi, A. A. Abdelhamid, I. Hegazy, B. Alotaibi, and Z. T. Fayed, "Deep investigation of the recent advances in dialectal Arabic speech recognition," *IEEE Access*, vol. 10, pp. 57063–57079, 2022, doi: 10.1109/ACCESS.2022.3177191.

[15] R. Matarneh, S. Maksymova, V. V. Lyashenko, and N. V. Belova, "Speech recognition systems: a comparative review," *Journal of Computer Engineering (IOSR-JCE)*, vol. 19, no. 5, pp. 71–79, 2017, [Online]. Available: www.iosrjournals.orgwww.iosrjournals.org.

[16] A. A. H. Qatanany, S. N. Abdullah Samarh, M. H. Saleh, and A. K. Toure, "The reasons of errors that change the meaning in the subject of Holy Quran memorization from The University Science Islam Malaysia students' perception," *Journal of Quran Sunnah Education & Special Needs*, vol. 7, no. 2, pp. 163–174, 2023, doi: 10.33102/jqss.vol7no2.200.

[17] M. Elmahdy, R. Gruhn, W. Minker, and S. Abdennadher, "Modern standard Arabic based multilingual approach for dialectal Arabic speech recognition," in *2009 8th International Symposium on Natural Language Processing, SNLP '09*, 2009, pp. 169–174, doi: 10.1109/SNLP.2009.5340923.

[18] A. Graves, A. R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 2013, pp. 6645–6649, doi: 10.1109/ICASSP.2013.6638947.

[19] A. Graves, "Sequence transduction with recurrent neural networks," 2012, [Online]. Available: http://arxiv.org/abs/1211.3711.

[20] A. Vaswani *et al.*, "Attention Is all you need," Aug. 2023, [Online]. Available: http://arxiv.org/abs/1706.03762.

[21] J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," *Advances in Neural Information Processing Systems*, vol. 2015-January, pp. 577–585, 2015.

[22] S. Latif *et al.*, "Transformers in speech processing: a survey," Jun. 2025, [Online]. Available: http://arxiv.org/abs/1211.3711.

[23] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," *Proceedings of Machine Learning Research*, vol. 202, pp. 28492–28518, 2023.

[24] M. B. Wazir, A. Saleh, and J. Huang Chuah, "Spoken Arabic digits recognition using deep learning," in *2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, Jun. 2019, pp. 339–344, doi: 10.1109/I2CACIS.2019.8825004.

[25] A. AbdAlmisreb, A. F. Abidin, and N. M. Tahir, "Maxout based deep neural networks for Arabic phonemes recognition," in *2015 IEEE 11th International Colloquium on Signal Processing & Its Applications (CSPA)*, Mar. 2015, pp. 192–197, doi: 10.1109/CSPA.2015.7225644.

[26] A. Emami and L. Mangu, "Empirical study of neural network language models for Arabic speech recognition," in *2007 IEEE Workshop on Automatic Speech Recognition and Understanding, ASRU 2007, Proceedings*, 2007, pp. 147–152, doi: 10.1109/asru.2007.4430100.

[27] M. Sameer, A. Talib, and A. Hussein, "Arabic speech recognition based on encoder-decoder architecture of transformer," *Journal of Techniques*, vol. 5, no. 1, pp. 1–8, 2023, doi: 10.51173/jt.v5i1.749.

[28] S. Nasr, R. Duwairi, and M. Quwaider, "End-to-end speech recognition for Arabic dialects," *Arabian Journal for Science and Engineering*, vol. 48, no. 8, pp. 10617–10633, 2023, doi: 10.1007/s13369-023-07670-7.

[29] M. Lataifeh, A. Elnagar, I. Shahin, and A. B. Nassif, "Arabic audio clips: Identification and discrimination of authentic Cantillations from imitations," *Neurocomputing*, vol. 418, pp. 162–177, 2020, doi: 10.1016/j.neucom.2020.07.099.

[30] A. Mohammed, M. S. Bin Sunar, and M. S. H. Salam, "Recognition of Holy Quran recitation rules using phoneme duration," in *Lecture Notes on Data Engineering and Communications Technologies*, 2018, vol. 5, pp. 343–352, doi: 10.1007/978-3-319-59427-9_37.

[31] T. S. Gunawan, N. A. M. Saleh, and M. Kartiwi, "Development of quranic reciter identification system using MFCC and GMM classifier," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 1, pp. 372–378, 2018, doi: 10.11591/ijece.v8i1.pp372-378.

[32] K. M. O. Nahar, R. M. Al-Khatib, M. A. Al-Shannaq, and M. M. Barhoush, "An efficient holy quran recitation recognizer based on SVM learning model," *Jordanian Journal of Computers and Information Technology*, vol. 6, no. 4, pp. 392–414, 2020, doi: 10.5455/jjcit.71-1593380662.
[33] N. Abokhodair, A. Elmadany, and W. Magdy, "Holy tweets: exploring the sharing of Quran on Twitter," in *Proceedings of the ACM on Human-Computer Interaction*, 2020, vol. 4, no. CSCW2, doi: 10.1145/3415230.
[34] M. Shahmohammadi, T. Alizadeh, M. H. Bijani, and B. Minaei, "A framework for detecting Holy Quran inside Arabic and Persian texts," in *Workshop Organizers*, pp. 71–76.
[35] S. R. El-Beltagy and A. Rafea, "QDetect: an intelligent tool for detecting Quranic verses in any text," *Procedia CIRP*, vol. 189, pp. 374–384, 2021, doi: 10.1016/j.procs.2021.05.107.
[36] E. Ayah, "Everyayah Quran Files." http://www.everyayah.com.
[37] L. Nanni, G. Maguolo, and M. Paci, "Data augmentation approaches for improving animal audio classification," *Ecological Informatics*, vol. 57, 2020, doi: 10.1016/j.ecoinf.2020.101084.
[38] S. Zhang, Y. Hu, and G. Bian, "Research on string similarity algorithm based on Levenshtein Distance," in *Proceedings of 2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference, IAEAC 2017*, 2017, pp. 2247–2251, doi: 10.1109/IAEAC.2017.8054419.
[39] P. K. K. Gandla, R. K. Verma, C. R. Panigrahi, and B. Pati, "Ticket classification using machine learning," *Lecture Notes in Networks and Systems*, vol. 1, pp. 487–501, 2024, doi: 10.1007/978-981-99-5015-7_41.

BIOGRAPHIES OF AUTHORS

Bashar Al-Rfooh is a computer engineer and data scientist passionate about NLP, Generative AI, and ASR systems, with published research on topics including diacritization, ASR systems, and retrieval-augmented generation (RAG). He can be contacted at email: bashar@alrfou.com.

Mohammad Abdel-Majeed is an associate professor at the Computer Engineering Department at University of Jordan (UJ) since June 2016. He earned his Ph.D. from University of Southern California (USC) in 2016. Also, he earned his MS from USC in 2010 and his BS from University of Jordan in 2007. His research focuses on the design of Artificial Intelligence based systems in the healthcare and smart city domains. He has been serving as the head of the computer engineering department at UJ in the time period between December-2021 and September 2023. He has been on the organizing committee/program committee and/or external reviewer for several local and international IEEE conferences like IPDPS 2019 and JEEIT 2019. Also, he is an external reviewer for several outstanding IEEE journals like TLVLSI, TPDS, TCAS and IEEE transactions on reliability. In addition, he participated in judging several competitions for undergraduate students and elementary school students. He also has been part of multiple projects with European partners like DECAIR, Smart.Eco and Data-Literacy projects). He can be contacted at email: m.abdel-majeed@ju.edu.jo.

Shorouq Al-Awawdeh received her B.Sc. degree in computer engineering from the University of Jordan (UJ), Jordan, in 2016, and a MSc. degree in networks and computer engineering from the University of Jordan (UJ), Jordan, in 2021. She worked as a research assistant in the AI field, and as a part-time lecturer and teaching assistant at the University of Jordan from 2021 until January 2025. Currently, she works as a lecturer at the Technical Collage at the Middle East University (MEU). Her research interests include machine learning, computer vision, expert systems, natural language processing (NLP), and cyber security. She can be contacted at email: s.awawdeh@meu.edu.jo.

Khalid A. Darabkh received the Ph.D. degree in Computer Engineering from the University of Alabama in Huntsville (UAH), USA, in 2007 with honors. He was selected for inclusion in the Who's Who Among Students in American Universities and Colleges. He has joined the Computer Engineering Department at the University of Jordan as an Assistant Professor since 2007 and promoted exceptionally for professorship in 2016. He authored and co-authored of at least two hundred and twenty highly esteemed research articles. He is among World's Top 2% Scientists List, for various disciplines, compiled by Stanford University in 2020, 2021, 2022, 2023, and 2024. Prof. Darabkh is the recipient of the Distinguished Researcher Award of the scientific faculties/schools at the University of Jordan for his distinctive research during the period 2018-2022. Prof. Darabkh is the recipient of 2020 Federation of Arab Scientific Research Councils Reward – Theme of Invention and Innovation. Prof. Darabkh is the recipient of Ali Mango Distinguished Researcher Award for Scientific Colleges and Research Centers in Jordan for his distinctive research during the period 2012-2016. He is further the recipient of the Most Cited Researchers Award at the University of Jordan at Scopus during 2017-2021. He is among Top 1% JU Researcher's List who published the highest number of quality manuscripts in Scopus. Prof. Darabkh serves on the Editorial Board of Telecommunication Systems, published by Springer, Computer Applications in Engineering Education, published by John Wiley & Sons, and Journal of High-Speed Networks, published by IOS Press. Moreover, he is a member of many professional and honorary societies, including Eta Kappa Nu, Tau Beta Pi, Phi Kappa Phi, and Sigma XI. He was listed among those of IBC Leading Scientists of the World, Cambridge, England as well as IBC 2000 Outstanding Scientists, Cambridge, England. He was selected for inclusion in the Marquis Who's Who in Science and Engineering and Marquis Who's Who in the World. He was further selected for inclusion in the Who's Who in Engineering Higher Education. Additionally, he serves as a TPC member of highly reputable IEEE conferences such as GLOBECOM, ICC, LCN, VTC-Fall, PIMRC, ISWCS, ATC, ICT, and IAEAC. He supervised/co-supervised and examined at least 120 master's theses in most of Jordanian universities besides Ph.D. dissertations in foreign countries. In the end, he is engaged in research mainly on the Internet of things, industrial smart cities software-defined networks, vehicular networks, flying ad-hoc networks, Fog networking, full duplex cognitive radio networks, queuing systems and networks, multimedia transmission, channel coding, steganography, and watermarking, as well as innovative and interactive learning environments. He can be contacted at email: k.darabkeh@ju.edu.jo.