Arabic Spoken Command Recognition Using MFCC and Machine Learning

Mousa Zahran- 1220716, Khaled abu lebda – 1220187, Laith Ghnem– 1221123

Faculty of Engineering & Technology
Department of Electrical and Computer Engineering
Birzeit University

Abstract

This project addresses the task of isolated Arabic spoken command recognition using classical speech processing techniques and machine learning classifiers. A small vocabulary consisting of six Arabic commands (افتح، اغلق، ابدأ، توقف، يمين، يسار) was recorded and organized into a balanced dataset sampled at 16 kHz. The proposed system follows a conventional speech recognition pipeline that includes silence removal, amplitude normalization, and feature extraction based on Mel-Frequency Cepstral Coefficients (MFCCs). For each utterance, 13 MFCC coefficients were extracted and summarized using their mean and standard deviation, resulting in a fixed-length 26-dimensional feature vector suitable for classification. The extracted features were evaluated using multiple classification models, including k-Nearest Neighbors (KNN), Gaussian Mixture Models (GMM), Random Forests (RF), and a linear Support Vector Machine (SVM). The dataset was randomly divided into training and testing sets using an 80/20 split, and all features were standardized prior to classification. Performance was assessed using accuracy, precision, recall, F1-score, and confusion matrix analysis. Experimental results demonstrate that the linear SVM classifier achieves the highest recognition accuracy of approximately 90.9%, outperforming the other evaluated models. Confusion matrix analysis reveals that most misclassifications occur between acoustically similar commands. These findings indicate that MFCC-based statistical features combined with a linear SVM provide an effective and computationally efficient solution for Arabic isolated spoken command recognition in small-vocabulary settings.

Index Terms—Arabic speech recognition, MFCC, spoken commands, machine learning, SVM..

1. Introduction

Automatic speech recognition (ASR) has become a fundamental component of modern human–computer interaction, allowing users to control systems and devices through voice commands in a natural and hands-free manner. This technology is widely used in applications such as smart environments, robotic systems, and assistive technologies. Despite the rapid advancements achieved in speech recognition for major world languages, Arabic ASR continues to pose significant challenges. These challenges arise from the language's rich phonetic characteristics, wide dialectal diversity, and the limited availability of annotated speech datasets, particularly for controlled and small-scale tasks.

One important subfield of ASR is isolated spoken command recognition, where each command is uttered independently and belongs to a predefined vocabulary. This type of system is well suited for real-time and resource-constrained applications, as it requires lower computational complexity compared to continuous speech recognition. For Arabic-speaking users, isolated command recognition systems are especially valuable, since many commercial solutions provide limited or no support for the Arabic language.

In this work, a system for recognizing six isolated Arabic spoken commands—ابدأ (start), اغلق (close), افتح (open), يسار (left), يمين (right), and توقف (stop)—is developed and evaluated. Speech data were recorded and processed using classical speech signal processing methods. All audio signals were sampled at a rate of 16 kHz and subjected to preprocessing steps that included silence removal and amplitude normalization. Feature extraction was carried out using Mel-Frequency Cepstral Coefficients (MFCCs), which are widely used in speech recognition due to their ability to capture perceptually relevant spectral information. To generate fixed-length feature representations suitable for classification, the mean and standard deviation of 13 MFCC coefficients were computed for each utterance, forming a 26-dimensional feature vector.

Multiple machine learning classifiers were implemented to evaluate the effectiveness of the extracted features, including k-Nearest Neighbors (KNN), Gaussian Mixture Models (GMM), Random Forests (RF), and linear Support Vector Machines (SVM). The dataset was randomly partitioned into training and testing sets following an 80/20 split, and feature normalization was applied prior to model training. System performance was assessed using standard evaluation metrics, including accuracy, precision, recall, F1-score, and confusion matrix analysis, to examine classification behavior and identify common sources of confusion between commands.

The remainder of this paper is structured as follows. Section II presents a review of related work in Arabic speech recognition and isolated command classification. Section III details the proposed system design and feature extraction methodology. Section IV reports the experimental setup and discusses the obtained results. Finally, Section V concludes the paper and outlines possible directions for future research..

2. Background and Related Work

Automatic speech recognition for the Arabic language has attracted growing research interest over the past years; nevertheless, it continues to present greater challenges compared to many other languages. One of the primary difficulties lies in the extensive diversity of Arabic dialects spoken across different geographical regions, which exhibit notable variations in pronunciation, lexical usage, and phonetic realization. Furthermore, the coexistence of Modern Standard Arabic alongside numerous colloquial dialects introduces additional variability within speech data, complicating the development of recognition systems that can reliably generalize across speakers.

Another limiting factor in Arabic ASR research is the scarcity of large-scale, publicly available, and accurately annotated speech corpora, particularly for small-vocabulary and command-oriented tasks. Consequently, many existing studies rely on datasets collected under controlled conditions with a limited number of speakers. This constraint often favors the use of traditional speech processing approaches and classical machine learning techniques, which are more appropriate for low-resource settings than modern deep learning methods that typically require extensive training data.

Among acoustic feature representations, Mel-Frequency Cepstral Coefficients (MFCCs) have been extensively adopted in speech recognition systems. MFCCs offer an efficient representation of the short-term spectral characteristics of speech while incorporating perceptual aspects of human hearing through the Mel scale. Prior research has consistently shown that MFCC-based features provide effective discrimination for phonetic content in both Arabic and non-Arabic speech recognition tasks, particularly in isolated word and spoken command recognition scenarios.

With respect to classification techniques, a wide range of machine learning models has been investigated in the literature. The k-Nearest Neighbors (KNN) algorithm is frequently used as a baseline approach due to its conceptual simplicity, although its performance may decline as feature dimensionality increases. Gaussian Mixture Models (GMMs) have long been employed in speech processing to probabilistically model the distribution of acoustic features. Ensemble learning methods such as Random Forests have also demonstrated robustness in handling variability and noise within speech data. Additionally, Support Vector Machines (SVMs) are often reported to achieve strong performance in small and medium-sized datasets, owing to their ability to learn discriminative decision boundaries in high-dimensional feature spaces.

Motivated by these findings, this work employs MFCC-based feature extraction in combination with several classical machine learning classifiers to investigate their effectiveness for Arabic isolated spoken command recognition using a limited dataset..

3. Methodology (System Description)

This section outlines the proposed framework for Arabic isolated spoken command recognition. It presents the main stages of the system, including audio data preparation, preprocessing, feature extraction, feature representation, and classification. The entire processing pipeline is designed to transform raw speech signals into discriminative feature vectors suitable for machine learning-based recognition. To provide further insight into the system behavior, signal-level and feature-level visualizations are included and discussed in the subsequent subsections.

3.1. Data Collection

The dataset employed in this study comprises six isolated Arabic spoken commands: ابدأ (start), اغلق (close), افتح (open), يسار (left), يمين (right), and توقف (stop). Speech recordings were collected from multiple speakers, with each speaker repeating every word several times. The dataset was constructed to be approximately balanced across all command classes in order to minimize class bias during training and evaluation.

All audio samples were recorded at a sampling rate of 16 kHz and saved in WAV format to preserve signal quality and ensure compatibility with subsequent processing stages. The recordings were conducted in a relatively quiet indoor environment, and a consistent recording setup was used for all speakers. This controlled recording condition was chosen to limit background noise and reduce variability unrelated to speaker or command content.

3.2. Preprocessing

To enhance recognition accuracy and minimize the impact of non-informative signal variations, all audio recordings were subjected to a preprocessing stage prior to feature extraction. Initially, silence removal was performed to eliminate low-energy regions occurring at the beginning and end of each utterance, thereby preserving only the segments containing active speech. This procedure helps reduce redundant information and improves consistency among speech samples.

Following silence removal, amplitude normalization was applied to each signal to ensure a uniform dynamic range across recordings. This normalization step mitigates variations caused by differences in speaker volume or recording conditions, leading to more stable and reliable feature extraction in subsequent processing stages.

3.3. Feature Extraction

Acoustic feature extraction was performed using Mel-Frequency Cepstral Coefficients (MFCCs), which are widely adopted in speech recognition systems due to their effectiveness in representing perceptually meaningful spectral information. MFCCs approximate the frequency resolution of the human auditory system and provide a compact description of speech signals.

For each preprocessed utterance, MFCC features were computed using a short-time analysis framework with overlapping frames. The speech signal was divided into

consecutive segments, and a time–frequency representation was obtained through spectral analysis. A total of 13 MFCC coefficients were extracted for each frame, capturing the dominant spectral characteristics of the spoken commands.
To produce a fixed-length representation suitable for machine learning classifiers, the temporal sequence of MFCC coefficients was summarized by computing the mean and standard deviation of each coefficient across time. This statistical aggregation resulted in a 26-dimensional feature vector for each utterance, combining both central tendency and variability information of the MFCC features..

3.4. Feature Vector Representation

Speech recordings naturally vary in duration, which leads to MFCC feature matrices with different numbers of time frames across utterances. To enable the use of conventional machine learning classifiers that require fixed-length inputs, a statistical feature representation was adopted.

For each utterance, the sequence of MFCC coefficients was summarized by computing the mean and standard deviation of each coefficient over all time frames. This approach captures both the average spectral characteristics and their variability throughout the spoken command. As a result, each audio sample was represented by a compact 26-dimensional feature vector, corresponding to 13 mean values and 13 standard deviation values derived from the MFCC features.

This fixed-length representation provides an efficient balance between descriptive power and computational simplicity, making it well suited for classification tasks in small-vocabulary and low-resource speech recognition scenarios.

3.5. Classification and Evaluation Protocol

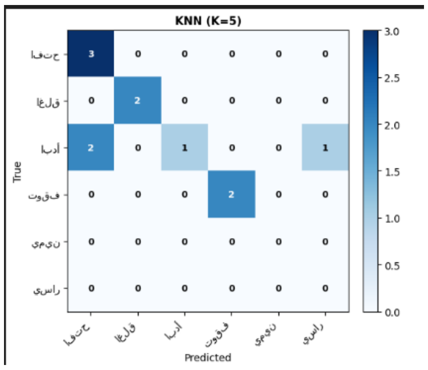
In this study, several classical machine learning classifiers were implemented and compared to evaluate their effectiveness in Arabic isolated spoken command recognition. The selected models include k-Nearest Neighbors (KNN), Gaussian Mixture Models (GMM), Random Forests (RF), and Support Vector Machines (SVM) with a linear kernel. These classifiers were chosen to represent different learning paradigms, including distance-based, probabilistic, ensemble-based, and margin-based classification approaches.

To evaluate system performance, the extracted dataset was randomly divided into training and testing subsets using an 80/20 split. This data partitioning strategy allows the models to be trained on the majority of the available samples while reserving an independent portion for evaluation. Prior to classification, all feature vectors were standardized using the statistics computed from the training set to ensure fair comparison across classifiers.

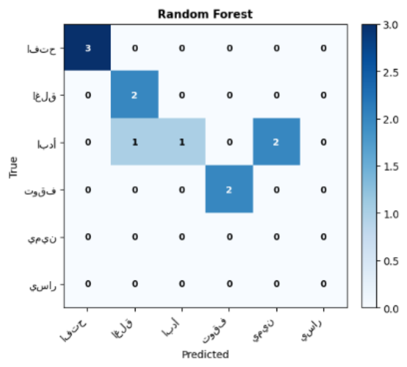
Model performance was assessed using multiple evaluation metrics, including classification accuracy, precision, recall, and F1-score. In addition, confusion matrix analysis was employed to examine class-wise recognition behavior and to identify common patterns of misclassification among the spoken commands..

Experiments and Results

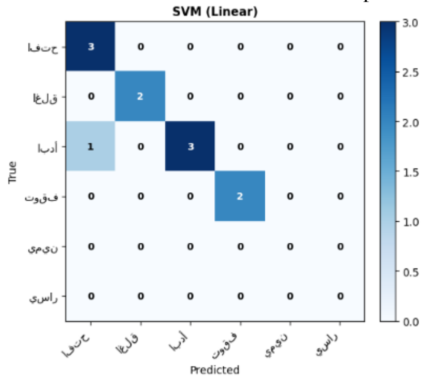
For the **KNN (K=5)** test, we used a simple **pure NumPy** implementation that measures **Euclidean distance** between the scaled feature vectors and then predicts by **majority vote** from the 5 nearest samples. The dataset was split **80/20 (train/test)**, and we standardized the features using only the training statistics, then applied the same scaling to the test set. On the test set, KNN got **Accuracy = 0.7273**, **Precision = 0.8909**, **Recall = 0.7273**, and **F1 = 0.7136**—so it's usually confident and correct when it predicts (high precision), but it still misses some true cases (recall isn't perfect). From the **confusion matrix**, commands like **“افتح”**, **“اغلق”**, and **“توقف”** were recognized really well, but **“ابدأ”** was the main problem: only **1 out of 4** was correct, and the rest got confused mostly with **“افتح”** and sometimes **“يسار”**. This basically shows KNN works fine for some commands, but a few classes are still too similar in the feature space and need better separation.



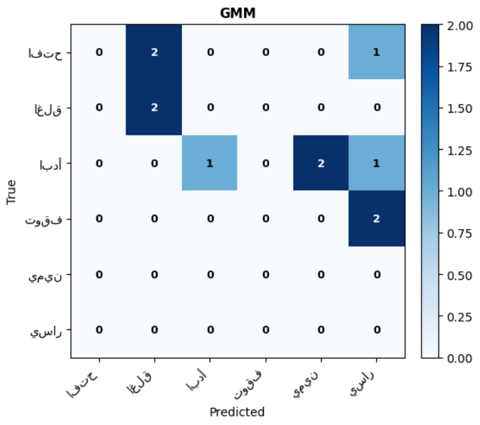
For the **Random Forest** test, we trained a forest with **200 trees** (default depth, random_state=42) on the same **80/20 train–test split**, and I kept the inputs **standardized using only the training-set stats** so the comparison stays fair. On the test set it reached **Accuracy = 0.7273**, **Precision = 0.9394**, **Recall = 0.7273**, and **F1 = 0.7455**—so it's pretty "careful" in its predictions (high precision), but it still misses some true samples, which keeps recall and accuracy lower. The **confusion matrix** shows it recognized **“افتح” (3/3)**, **“اغلق” (2/2)**, and **“توقف” (2/2)** perfectly, but **“ابدأ”** was clearly the weak point: only **1** sample was correct, and the rest got confused mainly with **“اغلق”** and **“يمين”**. Overall, RF is solid for some commands, but it struggles when classes have similar patterns in the features.



For the **Linear SVM**, we trained an **SVM with a linear kernel (C = 1.0)** using the same setup as before: an **80/20 train–test split** and **feature standardization based only on the training set** (then applied to the test set). It honestly performed really strong: **Accuracy = 0.9091**, **Precision = 0.9318**, **Recall = 0.9091**, and **F1 = 0.9091**. From the **confusion matrix**, most commands were classified correctly (like **“افتح”**, **“اغلق”**, and **“توقف”** all perfect), and the only noticeable mistake was in **“ابدا”**, where **1 sample** got confused as another class. Overall, SVM seems to separate the spoken-command features way better than KNN and Random Forest in this experiment



For the **GMM** experiment, we trained **one Gaussian Mixture Model per command class** (4 mixtures, **diagonal covariance**) using the same **80/20 train–test split** and the same **training-based standardization** for fairness. But the results came out really weak: **Accuracy = 0.2727**, **Precision = 0.4545**, **Recall = 0.2727**, and **F1 = 0.2667**, and the confusion matrix shows the model is mixing several commands together instead of separating them. This bad performance is mainly because the setup isn't very friendly for GMM here: using **4 components per class** can be too much when the number of samples per class is small, **diag covariance** is kind of restrictive (it ignores feature correlations), and GMMs usually struggle in **high-dimensional speech features** unless you reduce dimension or tune carefully. Also, there's a possible bug in prediction: np.argmax(scores) returns **0,1,2…** not the real label unless our labels are exactly in that same order, so that can easily destroy the accuracy too.



7. Conclusion

In this project, we learned how to build a complete **Arabic isolated spoken-command recognition** system from start to finish. we started from collecting the recordings, then doing the preprocessing, extracting features (like MFCCs), and finally testing different classical ML models in a fair way (same train/test split and same scaling). The experiments also showed us something important: the choice of model matters a lot, and some commands are just harder because they sound similar, so mistakes usually happen between those classes.

Future ideas / improvements:

- Collect **more data** (more speakers + more repetitions), and test with a **speaker-independent split** to check real generalization.
- Add **noise / reverb augmentation** so the model doesn't fail in real-life environments.
- Try stronger features (MFCC + **delta/delta-delta**, energy, pitch) or even **log-mel spectrograms** instead of only simple summaries.
- Do proper **hyperparameter tuning / cross-validation**, and for GMM specifically try better settings (like fewer components, tied/full covariance, or PCA) because it can be sensitive.
- If you have enough data, move to deeper models like **CNN/LSTM** or use **pretrained audio embeddings** to push accuracy higher.

8.References

[1]- Davis, S., & Mermelstein, P. (1980). Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Transactions on Acoustics, Speech, and Signal Processing.

[2]- Muda, L., Begam, M., & Elamvazuthi, I. (2010). Voice Recognition Algorithms using Mel Frequency Cepstral Coefficient (MFCC) and Dynamic Time Warping (DTW) Techniques.

[3]- Huang, X., Acero, A., & Hon, H. W. (2001). Spoken Language Processing: A Guide to Theory, Algorithm and System Development. Prentice Hall.

[4]- Vapnik, V. N. (1995). The Nature of Statistical Learning Theory. Springer-Verlag.

[5]- Reynolds, D. A., & Rose, R. C. (1995). Robust text-independent speaker identification using Gaussian mixture speaker models. IEEE Transactions on Speech and Audio Processing.