People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research
University of 20 aout 1955 -SKIKDA
Faculty of Science
Department of Computer science

A project to obtain a master's degree within the framework of the ministerial decision 1275

THEME:

DziriTalk: An algerian desktop voice assistant



Directed by :
Lina BENKREA

Supervised by:
Dr. Samira HAZMOUNE

Academic Year: 2025/2026

Acknowledgements

In the preamble to this thesis, I thank God who gave me the patience, courage, and perseverance to complete these long years of study.

I wish to thank my very dear parents who sacrifice every day so that I may succeed in my life, and whose unconditional love has been my greatest source of strength.

I express my deep gratitude to my research supervisor, Dr. Hazmoune Samira, for her wise advice, constant availability, and encouragement throughout every stage of this project.

My sincere thanks go to the jury members who have kindly agreed to review and evaluate my work.

Finally, I extend my most heartfelt thanks to all my family, loved ones, friends, and classmates for their support, understanding, and constant encouragement.

Thank yo

Dedication

I dedicate this Master's dissertation to my loving family, whose unwavering support, encouragement, and sacrifices have been the foundation of my journey. To my parents, your belief in me has fueled my determination to succeed. To my brother, your unwavering support and understanding have been my pillars of strength. This accomplishment is as much yours as it is mine.

I also dedicate this dissertation to my dear friends, for their continuous encouragement, help, laughter, and companionship throughout this endeavor. Your belief in my abilities has been a constant source of motivation

I wouldn’t have done it without you all.

Abstract

Voice User Interfaces (VUIs) are pivotal in modern Human-Computer Interaction (HCI), yet mainstream assistants consistently fail to process regional, non-standardized dialects like Algerian Darija due to their reliance on Modern Standard Arabic or foreign datasets. This creates a stark digital accessibility barrier for over 40 million speakers.

This thesis presents DziriTalk, an offline, privacy-focused desktop voice assistant engineered explicitly for the Algerian dialect. To address low-resource challenges and code-switching, we implement a dual-model pipeline: a BiGRU text classifier and a CNN-BiLSTM audio fallback, paired with Whisper-based ASR and a three-tier TTS. The NLU module maps to a system automation layer, enabling desktop control via spoken dialect.

Evaluation on a collected Darija dataset (110 speakers, 400+ recordings) yields 94% text and 98% audio accuracy, with under 3 seconds latency on CPU-only hardware. Notably, an SVM baseline drops from 96.93% on clean text to 63% in the vocal pipeline, confirming the critical value of end-to-end evaluation. Operating fully offline, DziriTalk bridges the digital divide, allowing users to interact naturally without frustration or internet dependency.

Keywords: Algerian Darija, ASR, BiGRU, CNN-BiLSTM, Code-Switching, Offline Voice Assistant, Low-Resource Language.

Résumé

Les interfaces vocales (VUI) sont essentielles dans l'interaction homme-machine (IHM) moderne, mais les assistants classiques échouent à traiter les dialectes non standardisés comme le darija algérien, car ils sont entraînés sur l'arabe standard ou des données étrangères. Cela crée une barrière d'accessibilité numérique pour plus de 40 millions de locuteurs. Cette thèse présente DziriTalk, un assistant vocal de bureau hors ligne et respectueux de la vie privée, conçu spécifiquement pour le dialecte algérien. Pour relever les défis des ressources limitées et de l'alternance codique, nous mettons en œuvre un pipeline à double modèle : un classificateur textuel BiGRU et un modèle audio de secours CNN-BiLSTM, couplés à un système de reconnaissance vocale basé sur Whisper et une synthèse vocale à trois niveaux. Le module de compréhension est relié à une couche d'automatization système, permettant le contrôle du bureau par la parole. L'évaluation sur un ensemble de données collecté (110 locuteurs, plus de 400 enregistrements) atteint 94 % de précision pour le texte et 98 % pour l'audio, avec une latence inférieure à 3 secondes sur processeur seul. Une SVM de référence chute de 96,93 % sur texte propre à 63 % en situation vocale réelle, confirmant l'importance de l'évaluation de bout en bout. Fonctionnant entièrement hors ligne, DziriTalk réduit la fracture numérique, permettant une interaction naturelle sans frustration ni dépendance à Internet.

Mots-clés : Darija algérien, reconnaissance vocale, BiGRU, CNN-BiLSTM, alternance codique, assistant vocal hors ligne, langue peu dotée.

الملخص

تُعد واجهات المستخدم الصوتية (VUIs)محورية في التفاعل بين الإنسان والحاسوب (HCI)الحديث، ومع ذلك تفشل المساعدات الصوتية التقليدية في معالجة اللهجات غير الموحدة مثل الدارجة الجزائرية، لأنها تعتمد على الفصحى أو مجموعات بيانات أجنبية . وهذا يخلق حاجزًا رقميًا لأكثر من 40مليون متحدث .يقدم هذا المشروع مساعدًا صوتيًا لسطح المكتب يعمل دون اتصال ويراعي الخصوصية، صُمم خصيصًا للدارجة الجزائرية، واسمه ."DziriTalk"لمواجهة تحديات الموارد المحدودة والتبديل اللغوي، نطبق نظامًا مزدوجًا :مصنف نصي يعتمد على BiGRUومصنف صوتي احتياطي يعتمد على CNN-BiLSTM، إلى جانب تقنية Whisperللتعرف على الكلام ونظام تركيب صوتي ثلاثي المستويات .ترتبط وحدة الفهم اللغوي بطبقة أتمتة النظام، مما يسمح بالتحكم في سطح المكتب عن طريق اللهجة .أظهر التقييم على مجموعة بيانات تم جمعها 110)متحدثًا، أكثر من 400تسجيل (دقة 94%للنص و 98%للصوت، مع زمن استجابة أقل من 3ثوانٍ على معالج عادي .انخفضت دقة نموذج SVMالمرجعي من 96.93%على النصوص النظيفة إلى 63%في التطبيق الصوتي الفعلي، مما يؤكد أهمية التقييم الشامل للنظام .من خلال العمل دون اتصال بالإنترنت، يسد هذا المساعد الفجوة الرقمية، مما يتيح للمستخدمين التفاعل بلغتهم الطبيعية دون إحباط أو اعتماد على الإنترنت.

الكلمات المفتاحية :الدارجة الجزائرية، التعرف على الكلام، التبديل اللغوي، مساعد صوتي دون اتصال، لغة محدودة الموارد.

Table of content

List of abbreviations XI

List of figures XII

Liste of tables…….. XIII

Introduction general……1

Part 1 Theoretical and technical aspect…… 4

Chapter 1: Linguistic Background and State of the Art 5

1.1Introduction 6

1.2 Overview of voice assistants 6

1.3Algerian dialect (Darija).....8

1.3.1 The Dual-Script System: Arabic Script and Arabizi 9

1.3.2 Diglossia in the algerian context 9

1.4 Challenges of low-resource languages 10

1.5 Existing solutions and limitations 11

1.6 Arabic and dialectal ASR benchmarks.....12

1.7 Algerian and maghrebi arabic datasets.....14

1.8 Dialect NLP and code-switching.....16

1.9 Key gaps identified.....18

1.10 Conclusion.....19

Chapter 2: Core technologies and techniques for voice-based interaction systemes.....20

2.1 Introduction 21

2.2 Machine learning and deep learning 21

2.2.1 Machine learning 21

2.2.2 Deep learning.....22

2.3 Speech recognition (ASR).....24

2.3.1 Definition.....24

2.3.2 Fundamental concepts of speech recognition 25

2.3.3 Architecture of speech recognition systems 266

2.3.4 Applications and use cases of speech recognition 27
2.3.5 From hidden markov models to end-to-end transformers 28
2.3.6 Wav2Vec 2.0: self-supervised pre-training for low-resource ASR 28
2.3.7 Whisper: multilingual weak supervision at scale 29
2.4 Natural language processing (NLP) 29
2.4.1 Definition 29
2.4.2 How NLP works 29
2.4.3 Natural language processing techniques 31
2.4.4 Opportunities and challenges of NLP 33
2.4.5 The intent detection and slot filling problem 34
2.4.6 BERT and transformer-based NLU 34
2.4.7 Text normalization for darija 35
2.5 Text-to-Speech (TTS) 35
2.6 Tools and frameworks 35
2.7 Conclusion 36
Chapter 3: System design and performance evaluation 37
3.1Introduction……..……...38
3.2 Data Collection and Dataset Preparation……..38
3.2.1 Data Collection Methodology……38
3.2.1.1 Crowdsourced Collection (Google Forms) ……38
3.2.1.2 Manual Street Recordings……41
3.2.2 Data Labeling and Organization……...41
3.2.3 Dataset Processing Pipeline……...41
3.2.3.1 Audio Conversion and Standardization……42
3.2.3.2 Speaker-Level Splitting……42
3.2.3.3 Data Augmentation……..42
3.2.4 Dataset Statistics……43
3.3 Overall Pipeline……43
3.4 Wake Word Detection and Audio Capture……44
3.4.1 Wake Word Detection……44

3.4.2 Audio Capture……44
3.5 Automatic Speech Recognition (ASR) ……45
3.6 Text Normalization……...45
3.7 Intent Classification……..47
3.7.1 Text Classifier (BiGRU) ……48
3.7.2 Audio Classifier (CNN-BiLSTM) ……48
3.7.3 Fusion Logic……..49
3.7.4 Intent Categories……49
3.7.5 Evaluation Methodology……50
3.7.6 Deep Learning Classifier Performance……..50
3.7.6.1 Text Classifier (BiGRU) – Evaluation……50
3.7.6.2 Audio Classifier (CNN BiLSTM) – Vocal Evaluation……51
3.7.7 Classical Machine Learning Baseline……51
3.7.8 Comparative Discussion: Classical Machine Learning vs.Deep Learning……...52
3.7.9 Justification of the Deep Learning Approach……54
3.7.10 Fusion System Performance……55
3.8 Action Execution……..56
3.9 Text to Speech (TTS) Output……...…57
3.10 Conclusion ……..……...……58
Chapter 4:Development Environment and Implementation Tool……60
4.1 Introduction.....61
4.2 Hardware Environment.....61
4.3 Software Framework and Libraries 62
4.3.1 Development Tools……...…64
4.3.3 Speech Recognition and Wake-Word Detection 64
4.3.4 Audio Capture and Feature Extraction.....64
4.3.5 Natural Language Understanding (NLU) 65
4.3.6 Text-to-Speech (TTS) and Audio Playback 66
4.3.7 Graphical User Interface (GUI) 66
4.3.8 Serialization and Utilities.....67

4.3.9 Implementation tools summary……....67

4.4 Graphical User Interface Implementation.....68

4.5 Modular Code Organization 71

4.6 Conclusion 72

General Conclusion.....74

List of abbreviations

AI: Artificial Intelligence

BiGRU: Bidirectional Gated Recurrent Unit

BiLSTM: Bidirectional Long Short-Term Memory

ASR: Automatic Speech Recognition

CPU: Central Processing Unit

DNN: Deep Neural Network

GPU: Graphics Processing Unit

GRU: Gated Recurrent Unit

INT8: 8-bit Integer Quantization

MFCC: Mel-Frequency Cepstral Coefficients

ML: Machine Learning

MSA: Modern Standard Arabic

NLU: Natural Language Understanding

NLP: Natural Language Processing

PCM: Pulse-Code Modulation

TTS: Text-to-Speech

UI: User Interface

WER: Word Error Rate

List of figures

Figure 2.1 – NLP preprocessing pipeline from raw data to evaluation metrics.....30
Figure 2.2 – Tokenization example: splitting a sentence into semantic units.....31
Figure 2.3 – Stop word removal: only content words remain.....31
Figure 2.4 – Bidirectional RNN (BiRNN) architecture for sequence classification.....33
Figure 3.5 – Text normalization pipeline from raw ASR output to canonical form 46
Figure 3.6 – Dual-model intent classification and fusion pipeline 47
Figure 3.7 – Action execution flow from intent to system response 57
Figure 3.8 – Three-tier Text-to-Speech (TTS) synthesis pipeline 58
Figure 3.9 – Graphical User Interface (GUI) interaction flow 57
Figure 3.10 – Data collection and model training pipeline.....58
Figure 4.1 – Python logo 63
Figure 4.2 – Google Form introduction page 39
Figure 4.3 – Google Form metadata fields 39
Figure 4.4 – Google Form command list and recording instructions 40
Figure 4.5 – Main chat window 69
Figure 4.6 – Settings panel 69
Figure 4.7 – Help window 70
Figure 4.8 – Wake word setup dialog 70

.

List of tables

Table 1.1: Comparative Performance of ASR Systems on Arabic Dialects..... 14
Table 2.1 : Machine Learning Techniques in Voice & Language Technologies..... 22
Table 2.2 : Deep Learning Techniques in Voice & Language Technologies.....23
Table 3.1 : Dataset Statistics.....43
Table 3.2 : Examples of Text Normalization Rules.....46
Table 3.3 : Supported Intents and Example Commands.....49
Table 3.4 : BiGRU Text Classifier Performance (Vocal Evaluation) 50
Table 3.5 : CNN-BiLSTM Audio Classifier Performance (Vocal Evaluation) 52
Table 3.6 : SVM-TF-IDF Baseline : Text-Only vs. Vocal Evaluation.....53
Table 3.7 : Comparative Summary : SVM Baseline vs. DziriTalk Deep Learning Architecture.....54
Table 3.8 : Justification for Adopting the Deep Learning Architecture over the Classical SVMBaseline..... 56
Table 4.1 : Hardware Environment Specifications.....62
Table 4.2 : Speech Recognition Tools..... 64
Table 4.3 : Audio Capture and Processing Tools.....65
Table 4.4 : Natural Language Understanding (NLU) Tools.....65
Table 4.5 : Text-to-Speech (TTS) Tools.....66
Table 4.6 : Graphical User Interface (GUI) Tools.....66
Table 4.7 : Serialization and Utilities.....67
Table 4.8 : Tool Implementation Summary for the Intent Classification Pipeline…… 67
Table 4.9 : System Modules and Their Responsibilities.....72

General Introduction

Artificial intelligence has profoundly transformed the relationship between humans and computers, particularly through the emergence of voice-based interaction systems. Modern voice assistants enable users to communicate with machines in a natural and intuitive way using spoken language rather than traditional keyboard or touch interfaces. Over the past decade, systems such as Amazon Alexa, Apple Siri, and Google Assistant have become widely integrated into smartphones, smart speakers, and desktop operating systems, demonstrating the growing importance of conversational artificial intelligence in everyday life. These technologies have significantly improved accessibility, productivity, and user experience by enabling hands-free interaction with digital systems.

Despite these advancements, the benefits of voice technology remain unevenly distributed across languages and dialects. Most commercial speech recognition systems are primarily trained on high-resource languages such as English, French, and Modern Standard Arabic (MSA). Consequently, speakers of regional dialectoften experience poor recognition accuracy and limited usability. This issue is especially evident in Algeria, where the majority of the population communicates daily using Algerian Arabic, commonly known as Darija. Darija is a complex and dynamic dialect characterized by extensive code-switching between Arabic, French, and Tamazight, with additional influences from Turkish and Spanish. Its phonetic diversity, informal structure, and lack of linguistic standardization make it particularly challenging for conventional Automatic Speech Recognition (ASR) systems.

Although Algeria represents a large and technologically active market, existing voice assistants consistently fail to provide reliable support for Darija speakers. Commercial Arabic ASR models struggle to recognize the dialect’s pronunciation patterns and mixed-language vocabulary, leading to inaccurate transcriptions and ineffective interaction. Furthermore, the development of speech technologies for Darija is hindered by the scarcity of publicly available annotated datasets and research resources. Unlike major international languages that benefit from massive

speech corpora and extensive NLP research, Darija remains a low-resource dialect with limited digital representation.

In response to these challenges, this project proposes the design and implementation of an offline desktop voice assistant specifically tailored for Algerian Darija. The system is built entirely using open-source technologies and is designed to operate efficiently on standard Windows computers without requiring constant internet connectivity or dedicated GPU hardware. The assistant continuously listens for a user-defined wake word, transcribes spoken commands in Darija, identifies the user's intent, and executes the corresponding action in real time. By functioning locally on the user's device, the system enhances privacy, reduces dependency on cloud services, and ensures accessibility in environments with unstable or limited internet access.

The proposed assistant integrates several advanced components, including wake-word detection, speech-to-text transcription, intent classification, text-to-speech synthesis, and desktop action automation. Particular emphasis is placed on improving the recognition of code-switched Darija speech through the use of multilingual Whisper-based transcription models and intent classification using a Bidirectional GRU network, with an auxiliary CNN-BiLSTM audio classifier that processes acoustic features to resolve ambiguous transcriptions. In addition to its technical objectives, the project aims to contribute to the growing field of low-resource language processing by documenting methodologies, challenges, and solutions that may support future research on Algerian dialect technologies.

This thesis is organized into two main parts, reflecting both the scientific research and the entrepreneurial dimensions required by Ministerial Decision No. 1275.

The first part, *Theoretical and Technical Aspects*, is dedicated to the scientific and technological foundation of the work. It comprises four chapters. Chapter 1 establishes the linguistic background of Algerian Darija and reviews the state of the art in voice assistants, Arabic ASR, and low-resource language processing. Chapter 2 presents the core technologies and techniques underpinning voice-based interaction systems, including Automatic Speech Recognition, Natural Language Processing, Text-to-Speech, and the machine learning and deep learning

frameworks selected for the project. Chapter 3 details the overall system design, data collection methodology, dataset preparation, and the performance evaluation of the intent classification pipeline. Chapter 4 describes the hardware and software development environment, the implementation tools, and the graphical user interface of the DziriTalk prototype.

The second part, *Startup Project Guide*, complies with the requirements of the ministerial guide for the Diplôme-Startup track. It is structured around six thematic pillars. Pillar 1 presents the project idea, the team, and the implementation schedule. Pillar 2 describes the innovative aspects of the solution. Pillar 3 provides a strategic market analysis, including target segments, competition, and the marketing strategy. Pillar 4 outlines the production and organization plan. Pillar 5 presents the projected financial plan, revenue models, and cash flow analysis. Pillar 6 validates the experimental prototype. This part is supplemented by financial appendices, including a simplified projected balance sheet and a Business Model Canvas.

The thesis concludes with a general conclusion that synthesizes the key findings from both the technical implementation and the entrepreneurial feasibility analysis.

Part 1
Theoretical and technical aspect

Chapter 1: Linguistic Background and State of the Art

1.1 Introduction

This chapter establishes the linguistic, technological, and competitive context for the Darija Voice Assistant. It begins by surveying the landscape of modern voice assistant systems and the core technologies that enable them. A detailed characterization of Algerian Darija is then provided as a linguistic phenomenon, examining the features that make it both rich and challenging for computational processing. Finally, the chapter reviews the current state of the art in Arabic and dialectal ASR, identifying key datasets, benchmarks, and gaps that motivate the present work.

1.2 Overview of voice assistants

Voice assistants are software agents that can understand spoken natural language, process it, and respond with meaningful, contextually appropriate output.

At their core, every modern voice assistant operates as a pipeline of three tightly coupled subsystems: Automatic Speech Recognition (ASR), Natural Language Processing (NLP), and Text-to-Speech synthesis (TTS). Understanding how each layer works and where each can fail is essential to appreciating the challenges of building a dialect-aware assistant.

The ASR layer converts an audio waveform into a textual transcription. Early ASR systems, developed in the 1980s and 1990s, relied on Hidden Markov Models (HMMs) combined with Gaussian Mixture Models (GMMs) to model acoustic-phonetic relationships. While these approaches proved effective for controlled, single-speaker environments, they struggled with spontaneous conversational speech, heavy accents, and noisy conditions. The transition to deep learning in the 2010s particularly the adoption of Recurrent Neural Networks (RNNs) and Connectionist Temporal Classification (CTC) brought significant gains in robustness and accuracy. More recently, fully end-to-end transformer-based models have redefined what is possible in speech recognition, achieving near-human performance on major English benchmarks [1][2].

The NLP layer receives the transcribed text and interprets the user's intent, extracting the action to be performed (intent detection) and any relevant parameters (named entity or slot extraction). Classical NLP pipelines used rule-based grammars and statistical classifiers; modern pipelines instead rely on large pre-trained language models that capture deep contextual semantics across entire utterances.

The TTS layer closes the loop by converting the system's text response back into audible speech. Early concatenative TTS systems stitched together prerecorded phoneme units,

resulting in intelligible but robotic-sounding output. Neural TTS, which learns to map text directly to spectral features and then to waveforms, has largely replaced older approaches, enabling natural prosody and expressive intonation [3][4].

The commercial landscape for voice assistants is dominated by a small number of major platforms: Amazon Alexa, Google Assistant, Apple Siri, and Microsoft Copilot. These systems support dozens of languages, but their support for non-standard, low-resource, or dialectal varieties of those languages remains shallow [5]. Algerian Arabic (Darija) falls squarely into this gap, motivating the work described in this thesis.

Brief History of Arabic Speech Processing :

The evolution of Arabic Automatic Speech Recognition (ASR) is shaped by the unique linguistic characteristics of the Arabic language. Early Arabic ASR systems, developed in the 1980s and 1990s, relied on Hidden Markov Models (HMMs) combined with Gaussian Mixture Models (GMMs). A crucial advantage was the nearly one to one letter to phone correspondence, which simplified pronunciation lexicon construction. However, three persistent challenges remained: non-diacritized text, dialectal variety, and morphological complexity [6].

The transition to deep learning in the 2010s brought significant gains through Recurrent Neural Networks (RNNs) and Connectionist Temporal Classification (CTC)[7]. However, progress has been gradual, with early neural approaches achieving recognition rates comparable to HMM based systems rather than surpassing them decisively. The introduction of end to end architectures using Transformers (Vaswani et al., 2017)[1] has since enabled models to learn representations directly from data.

The progression from HMM/GMM to end to end Transformers represents a paradigm shift in how Arabic ASR systems are built. Self-supervised approaches like wav2vec 2.0 have demonstrated that powerful speech representations can be learned from unlabeled audio, dramatically reducing labeled data requirements (Baevski et al., 2020)[7].

Despite these advances, dialectal Arabic remains severely underrepresented. As Besdouri et al. (2024)[6] observe in their comprehensive review, "the scarcity of large annotated datasets is a key challenge in Arabic ASR, particularly for dialectal varieties." The authors further emphasize that "dialectal variations and non-standard orthography complicate Arabic speech recognition."

Algerian Darija exemplifies what researchers classify as a low-resource language with tens of

millions of speakers but limited digital resources. Three features make Darija particularly challenging for speech processing: pervasive code-switching with French, the use of Arabizi numerals in Romanized writing, and the complete lack of orthographic standardization. This combination of factors (large speaker population, complex linguistic features, and limited digital resources) makes Darija an ideal test case for developing methodologies applicable to other underserved languages.

1.3 Algerian dialect (Darija)

Algerian Arabic, commonly known as Darija or simply "Algerian dialect," is the mother tongue of the overwhelming majority of Algeria's approximately 45 million inhabitants.

It occupies a unique position in the country's sociolinguistic landscape: it is the dominant language of everyday spoken communication in markets, homes, social media, and informal professional settings yet it has no official status and has historically been excluded from formal education, government communication, and media.

Linguistically, Darija is a Maghrebi Arabic dialect that developed from Classical Arabic through centuries of contact with Berber (Tamazight), Ottoman Turkish, Spanish, Italian, and most consequentially French. The result is a language with a distinctly Arabic morphosyntactic core overlaid with a rich layer of borrowed lexicon and adapted phonology. A speaker of Egyptian Arabic or Levantine Arabic will recognize many individual words in Darija but will typically find natural conversation difficult to follow.

Three features of Darija are particularly relevant to automatic speech and language processing:

Code-switching with French: Algerian speakers routinely mix Arabic and French within a single utterance, sometimes within a single sentence. This phenomenon known academically as code-switching or translanguaging is not random but governed by social and pragmatic conventions. Technical vocabulary (computing, medicine, administration) is overwhelmingly French-origin; emotional or informal expression tends toward Arabic. A typical utterance might be: "7el l-application t3a-ak" ("Open your application"), which blends Arabized numerals, Arabic morphology, and French lexical items. Code-switching is a pervasive feature of Algerian speech, as documented in the CAFE dataset, which captures spontaneous multilingual conversations involving Algerian Arabic, French, and English [9].

Arabized numerals in Romanized writing: When Darija is written using the Latin alphabet as is standard in online communication, SMS, and social media speakers use a system of digit-

letter substitutions to represent sounds absent from the Latin alphabet. The digit 7 represents the Arabic pharyngeal fricative 'ha', and the digit 3 represents the emphatic pharyngeal 'ayn'. This convention, while consistent within communities, produces transcriptions that are opaque to standard NLP tokenizers and spell-checkers trained on formal Arabic or French.

Lack of orthographic standardization: Unlike Modern Standard Arabic (MSA), which has a well-defined and universally taught orthography, Darija has no official written standard. The same word may be spelled in three or four different ways by different writers, and ASR models trained on MSA will typically produce inconsistent or incorrect transliterations when exposed to Darija speech. As noted in the DziriBOT paper, Darija is characterized by non-standardized orthography, extensive code-switching with French, and the simultaneous use of Arabic and Latin (Arabizi) scripts" [10][6].

These characteristics collectively mean that existing NLP and ASR pipelines ven those that nominally support Arabic perform poorly on Darija.

1.3.1 The Dual-Script System: Arabic Script and Arabizi

In addition to the Arabic script, Algerian Darija is frequently written using the Latin alphabet in a system known as *Arabizi*. This practice is common in informal written communication such as SMS, social media, and online chats. Speakers switch fluidly between Arabic characters and Latin-based Arabizi, often within the same sentence or conversation. This dual-script system, combined with non-standardized orthography, creates significant obstacles for NLP pipelines that expect clean, standardized text input. Off-the-shelf tools trained on formal Arabic or French typically break when processing Arabizi, as they cannot recognize the digit-letter substitutions or the mixed-script patterns that characterize this writing system. The DziriBERT model was specifically designed to address this challenge, handling Algerian text contents written using both Arabic and Latin characters [10][6]. Similarly, the ELNER-DZ dataset includes annotations in both Arabic script and Arabizi, reflecting the reality of written Darija communication [11][6].

1.3.2 Diglossia in the algerian context

Arabic is characterized by a diglossic situation, where Modern Standard Arabic (MSA) is the formal, written language used in education, media, and official communication, while regional dialects such as Algerian Darija are the spoken languages of everyday life. This diglossia creates a fundamental challenge for speech and language technologies: systems trained on

MSA perform poorly on dialectal speech, and vice versa. In Algeria, this gap is particularly acute, as Darija diverges significantly from MSA in phonology, morphology, and lexicon. Speakers often perceive MSA as a prestigious variety while attaching a stigma to the use of dialect in formal contexts, further complicating the development of dialect-aware technologies. The limited availability of suitable datasets for dialectal Arabic is a persistent bottleneck; as Besdouri et al. (2024) observe, "the scarcity of large annotated datasets" is a key challenge in Arabic ASR, particularly for dialectal varieties [6].

1.4 Challenges of low-resource languages

A "low-resource" language, in the NLP and speech communities, refers to any language or variety for which comparatively little labeled training data is publicly available. The threshold is relative: a language spoken by millions of people can still be low-resource if those speakers have not been the subjects of systematic data collection efforts.

Darija is low-resource in precisely this sense. The training of state-of-the-art ASR and NLP models typically requires hundreds or thousands of hours of transcribed speech; commercial systems like Google Speech-to-Text were trained on millions of hours. For Darija, public resources remain in the range of a few dozen hours at best, and those resources are often limited in geographic diversity, age range, and recording quality. The consequences are predictable: models trained on MSA or Standard French exhibit high word error rates when tested on Darija speech, with errors clustering around dialect-specific vocabulary, code-switched segments, and non-standard phonology.

Several additional challenges compound the data scarcity problem:

Dialect variability: Algeria is a large country with significant regional variation in spoken Darija. The dialect of Algiers differs perceptibly from that of Oran, Constantine, or Tamanrasset. The Arabic dialect spoken in Oran, for example, exhibits distinct phonetic, morphosyntactic, and lexical features that differentiate it from other urban dialects such as that of Algiers or Constantine. It belongs to the Bedouin-based urban dialects, with influences from Spanish and French that are not equally present in other Algerian cities [12][6]. A model trained primarily on speech from one region may generalize poorly to others.

Absence of standardized orthography: Without a writing standard, even the task of transcribing speech for training data is complicated – different annotators may produce different transcriptions of the same utterance, introducing label noise.

Limited pre-trained models: The dominant paradigm in modern NLP – pre-training a large transformer model on a massive text corpus and then fine-tuning for specific tasks – requires a large and representative text corpus. For Darija, no such corpus approaches the scale of those available for English, French, or even MSA. While models like DziriBERT represent a significant step forward, they are trained on relatively modest datasets (e.g., one million tweets) compared to the billions of tokens used for major languages [10][13].

Evaluation infrastructure: Meaningful progress in low-resource NLP requires benchmark datasets with standardized splits and evaluation metrics. For Darija, these are only beginning to emerge, making it difficult to compare systems or track progress over time.

1.5 Existing solutions and limitations

Commercial voice assistants have made uneven progress in supporting Arabic. Google Assistant, Apple Siri, and Amazon Alexa offer varying levels of support for Modern Standard Arabic (MSA) and a handful of regional dialects, notably Egyptian, Saudi, and Levantine Arabic. However, Algerian Darija is conspicuously absent from the officially supported dialect list of every major commercial platform as of 2025 [5][14]. Even when these systems claim "Arabic" support, they are overwhelmingly optimised for MSA; dialectal speech is either misinterpreted, rejected, or transcribed with very high Word Error Rates (WER). Empirical evaluations confirm this: on the NADI 2025 shared task, zero-shot Whisper achieved an average WER of 93.89% across eight Arabic dialects, with Algerian and Moroccan dialects exceeding 100% WER [14]. Users who attempt to interact with these systems in Darija therefore receive at best a degraded partial response, and at worst a response in MSA that is pragmatically and socially inappropriate further reinforcing the digital exclusion of dialect speakers.

On the research side, several academic systems have targeted Maghrebi or Algerian Arabic specifically. Early approaches relied on rule-based NLU with hand-crafted grammars, limiting scalability and robustness to spontaneous speech. More recent neural systems have leveraged multilingual BERT, AraBERT, or CAMeLBERT for dialectal text understanding [15][16]. Notably, DziriBOT demonstrated the feasibility of building retrieval-augmented conversational agents for Algerian Darija, integrating specialised NLU with retrieval-augmented generation (RAG) to handle both structured service flows and knowledge-intensive responses [10]. Similarly, DarijaBanking provided a benchmark for

intent detection in Moroccan Darija within the banking domain, fine-tuning BERT-based models to achieve competitive results on clean text [17].

Despite these contributions, existing research prototypes share critical limitations that prevent their deployment as practical voice assistants. First, they operate on small, domain-specific vocabularies often fewer than 50 intents and are evaluated on text-only inputs, bypassing the full speech-to-action pipeline required for a usable voice assistant [10][17]. Second, most lack an integrated Text-to-Speech (TTS) component for Darija, and those that incorporate speech synthesis rely on external cloud APIs, compromising offline usability and user privacy [10]. Third, models fine-tuned on limited dialectal corpora tend to overfit to specific regions and topics, with generalisation to unseen speakers or acoustic conditions remaining an open challenge [18][19].

The path to building a full end-to-end system is further obstructed by the complexity of adapting open-source ASR toolkits such as Kaldi and ESPnet to a new dialect. As Besdouri et al. observe in their comprehensive review, "the scarcity of large annotated datasets is a key challenge in Arabic ASR, particularly for dialectal varieties," and the configuration of acoustic models, pronunciation lexicons, and language models demands specialised speech engineering expertise [6]. This explains why most academic projects stop at text-based NLU rather than tackling the complete audio-to-action pipeline.

Conversely, multilingual end-to-end models such as Whisper offer promising zero-shot capabilities that could, in principle, be applied to Darija without dialect-specific training [2]. However, zero-shot performance on North African dialects remains extremely poor (WER > 100% in some evaluations) unless the model is guided by prompt engineering or fine-tuned on in-domain data [20][2]. Recent work has shown that fine-tuning Whisper on Algerian or Moroccan Darija can reduce WER to approximately 23% and 10.6% respectively [18][21], indicating that adaptation is both necessary and effective—but it requires curated dialectal speech datasets, which are not yet widely available.

In summary, the gap in the existing landscape is not a lack of component technologies, but a lack of integration: no publicly available, end-to-end, dialect-aware voice assistant exists for Algerian Arabic that covers the full ASR → NLU → TTS pipeline with a conversational interaction model, operates entirely offline, and runs on consumer-grade hardware. This integration gap is precisely the problem that DziriTalk addresses.

1.6 Arabic and dialectal ASR benchmarks

Research on Arabic ASR has historically focused on Modern Standard Arabic and broadcast speech, where relatively large transcribed corpora are available. The Multi-Genre Broadcast (MGB) challenge series has been instrumental in pushing the field toward more naturalistic data. MGB-3, held in 2017, was the first edition to target dialectal Arabic in the wild, using content harvested from YouTube across multiple Arabic-speaking regions. Results from MGB-3 revealed a substantial performance gap between MSA-trained systems and those exposed to dialectal speech, quantifying the practical cost of ignoring dialect variability in ASR development [22].

More recently, the CommonVoice project by Mozilla has expanded its Arabic language support to include contributions from Algerian speakers, though the dataset remains comparatively small and dominated by read speech rather than spontaneous conversation.

Comparative Performance of ASR Systems on Arabic Dialects

To contextualise the performance of DziriTalk, Table 1.1 presents a comparative overview of Automatic Speech Recognition (ASR) performance across various Arabic dialects. The Word Error Rate (WER) is used as the primary metric, as it is the standard measure for evaluating ASR systems.

The table highlights several important observations. First, zero-shot performance on dialectal speech remains extremely challenging for multilingual models. Recent evaluations on the NADI 2025 test set show that Whisper in zero-shot mode achieves an average WER of 93.89% across eight Arabic dialects, with Algerian and Moroccan dialects exceeding 100% WER [14]. This confirms the difficulty of recognising North African dialects without adaptation.

Second, fine-tuning significantly improves performance. A Whisper-tiny model fine-tuned specifically for the Algerian dialect achieves a WER of approximately 23% [21]. Similarly, fine-tuning Wav2Vec2-XLSR-53 on Moroccan Darija achieves a WER of 10.64% in clean conditions [18].

Third, our approach using prompt engineering with Whisper-base achieves a WER of 22% on Algerian Darija, comparable to fine-tuned models but without the need for additional training data or computational resources. This demonstrates that prompt engineering offers a lightweight and effective alternative to fine-tuning for low-resource dialects.

Table 1.1: Comparative Performance of ASR Systems on Arabic Dialects

Dialect	System	WER (%)	Reference
Algerian Darija	Whisper-base (zero-shot, no prompt)	48%	This work
Algerian Darija	Whisper-base + Darija prompt	22%	This work
Algerian Darija	DziriTalk (full pipeline)	22%	This work
Algerian Darija	Whisper-tiny (fine-tuned)	~23%	Nasri, 2024 [21]
Moroccan Darija	Wav2Vec2-XLSR-53 (fine-tuned, clean)	10.64%	Agoujil and Yousfi, 2025 [18]
Moroccan Darija	Whisper (fine-tuned, noisy)	More accurate than Wav2Vec2	Agoujil and Yousfi, 2025 [18]
Egyptian Arabic	Whisper-large-v2	55.1%	Waheed et al., 2024 [20]
Egyptian Arabic	Best-distilled model	45.0%	Waheed et al., 2024 [20]
Gulf Arabic	CNN-LSTM	3.63%	2024 [20]
Multiple dialects (avg)	Whisper (zero-shot)	93.89%	NADI 2025 Shared Task [14]
Multiple dialects (avg)	BYZÖ-ctc + SFT + LM	44.40%	NADI 2025 Shared Task [14]

Note: WER values are based on different test sets and evaluation conditions. Lower WER values indicate better performance. The results demonstrate that prompt engineering can achieve performance comparable to fine-tuning for low-resource dialects without additional training data.

1.7 Algerian and maghrebi arabic datasets

A growing number of specialized datasets have begun to address the data scarcity problem for Algerian and Maghrebi Arabic:

- **CAFE (2025):** A spontaneous Algerian dialect code-switching speech dataset comprising approximately 37 hours of real human-to-human conversations collected across multiple Algerian regions. Designed to support both ASR and multilingual NLP research, CAFE is notable for capturing the natural code-switching patterns that characterise everyday Darija speech, making it more representative of real-world use conditions than earlier read-speech corpora. The dataset is released in two tiers: CAFE-small (2h36m) with full human annotations, and CAFE-large (~34h35m) with automatic labels [9].
- **MADAR:** A parallel corpus developed by the NYU Abu Dhabi CAMeL Lab covering dialects of 25 Arabic-speaking cities, plus MSA, English, and French. While not Algerian-specific, MADAR provides a useful benchmark for dialect identification and cross-dialect transfer, and its coverage of Maghrebi dialects offers partial applicability to Darija NLP tasks [23].
- **Casablanca (EMNLP 2024):** A large supervized multi-dialect Arabic speech dataset presenting baseline ASR models across a broad range of Arabic dialects, including Maghrebi varieties. The dataset represents a community effort to develop reproducible baselines and to quantify the state of the art for under-resourced Arabic dialects [24].
- **Masader:** A community-maintained catalog of Arabic NLP and speech datasets. As of 2025, Masader indexes over 200 datasets spanning text, audio, and multimodal resources, and serves as a useful starting point for researchers seeking Algerian-relevant training data [25].

The broader resource gap :

Beyond the datasets discussed above, the broader landscape of Arabic speech corpora reveals a significant disparity between resources for Modern Standard Arabic and those for dialectal varieties. A systematic review of Arabic dialect speech corpora identified 30 publicly accessible databases and an additional 39 self-databases, highlighting the growing but still insufficient availability of dialectal speech resources [13][25]. Most studies focus on MSA, while dialectal Arabic particularly North African varieties remains severely underrepresented in large-scale, publicly available speech corpora [6][25]. The scarcity of annotated dialectal speech data is a critical bottleneck for the development of robust ASR and NLP systems for the Maghrebi region.

1.8 Dialect NLP and code-switching

Intent detection and slot filling in dialectal Arabic present challenges that extend well beyond vocabulary coverage. Algerian Darija exhibits three features that are particularly problematic for standard NLU pipelines: (1) informal and inconsistent orthography when written in Romanized form; (2) heavy code-switching with French, which requires a model to handle bilingual or even trilingual input; and (3) strong regional phonetic variation that introduces surface-level diversity into both audio and text input.

State-of-the-art NLU approaches for Arabic typically build on transformer-based language models, with AraBERT (a BERT model pre-trained on a large Arabic text corpus) and CAMeLBERT being among the most cited. These models achieve competitive intent accuracy on formal Arabic NLU benchmarks, but their performance degrades significantly on Darija input, particularly when code-switching introduces French-origin tokens that are not represented in the Arabic pre-training vocabulary.

Hybrid tokenization strategies treating code-switched text as a mixed-language sequence and multilingual models such as mBERT offer partial remedies, but no dedicated, publicly released NLU model for Darija intent classification currently exists.

Despite this gap, recent years have seen the emergence of several NLP resources for Algerian Darija:

- **ELNER-DZ** is the first large-scale dataset for Named Entity Recognition (NER) and Entity Linking in Algerian Arabic, containing over 2 million dialectal sentences labeled with more than 1.9 million named entities linked to Wikidata. The dataset includes annotations in both Arabic script and Arabizi, making it a valuable resource for dialectal NLP [11][9].
- **DziriBERT** is the first pre-trained language model specifically for the Algerian dialect, pre-trained on one million Algerian tweets. It has demonstrated state-of-the-art performance on sentiment, emotion, and topic classification tasks, and handles text written in both Arabic and Latin characters [10].
- **Darija Speech To Text Dataset** provides 8.23 GB of audio data with 13,178 rows of transcribed speech, covering Algerian and Moroccan Darija, supporting speech recognition tasks for the dialect [26].

These resources, while not yet covering all NLP tasks, represent important steps toward

addressing the resource gap for Algerian Darija.

Critical Evaluation of Existing Darija NLP Resources :

While recent years have seen the emergence of several NLP resources for Algerian Darija, a critical evaluation reveals both significant progress and persistent gaps that directly motivate the present work.

- **DziriBERT** (Bechiri and Lanasri, 2026) [10]
  DziriBERT represents a significant milestone as the first pre-trained language model specifically for the Algerian dialect, pre-trained on one million Algerian tweets. It has demonstrated state-of-the-art performance on sentiment, emotion, and topic classification tasks, and handles text written in both Arabic and Latin characters. However, its pre-training corpus, primarily social media data, introduces biases toward informal, short-form text that may not generalise to longer or more formal contexts. Furthermore, the model's performance on code-switched text with French, while better than MSA-based models, still shows degradation compared to monolingual Arabic text. Critically, DziriBERT is designed for text input only and cannot directly process speech, leaving a gap for voice-based applications.
- **ELNER-DZ** (Bouguettoucha and Djouablia, 2025) [11]
  ELNER-DZ provides the first large-scale dataset for Named Entity Recognition (NER) in Algerian Arabic, containing over 2 million dialectal sentences labeled with more than 1.9 million named entities linked to Wikidata. The inclusion of annotations in both Arabic script and Arabizi is a notable strength, reflecting the reality of written Darija communication. However, the dataset's coverage of entity types is limited to pre-defined categories, and the annotation quality relies on crowd-sourced workers whose consistency may vary. Furthermore, the dataset focuses on named entities rather than the command-style utterances typical of voice assistant interactions, limiting its direct applicability to the present work.
- **CAFE Dataset** (Lachemat et al., 2025) [9]
  CAFE offers the first spontaneous Algerian dialect code-switching speech dataset, comprising approximately 37 hours of real human-to-human conversations. This represents a significant advance over read-speech corpora, capturing natural code-switching patterns. However, the dataset is released in two tiers: CAFE-small (2h36m) with full human annotations, and CAFE-large (~34h35m) with automatic labels. The

reliance on automatic labeling for the larger tier introduces potential label noise that may affect model training. Additionally, the conversational nature of the data differs from the command-style utterances needed for voice assistant development, limiting its direct applicability.

- **Darija Speech To Text Dataset** (Kirouane, 2024) [26]
  This comprehensive collection supports speech recognition tasks for the Darija dialect, including audio data totaling 8.23 GB with 13,178 rows of transcribed speech. It covers Algerian and Moroccan Darija, providing a valuable resource for ASR development. However, the dataset is primarily designed for speech recognition rather than intent classification, and it does not include labels for command-style interactions.

Gaps Addressed by This Work :

Despite these valuable resources, several critical gaps remain that this work directly addresses:

1. **Intent classification dataset**: While pre-training, NER, and speech resources exist, there is no dedicated NLU dataset for intent classification and slot filling, which are the core components of voice assistant interaction. Our collected dataset fills this gap with over 400 recordings across 6-8 intent classes.

2. **Speech-text alignment with intents**: The CAFE dataset provides conversational speech, but there is no public corpus of Darija speech aligned with intent labels suitable for command-based interactions. Our dataset addresses this with speaker-level splitting and manual labeling.

3. **End-to-end evaluation framework**: No existing work has evaluated the performance gap between oracle (clean text) and vocal (ASR-transcribed) evaluation for Darija. Our empirical finding provides a methodological contribution that extends beyond Darija to all low-resource ASR evaluation.

4. **Open-source voice assistant pipeline**: While individual components exist, no integrated, end-to-end voice assistant pipeline exists for Algerian Darija. DziriTalk fills this gap with a complete implementation from wake-word detection to TTS output.

1.9 Key gaps identified

Based on the review above, the primary gaps motivating this project can be summarized as follows:

- No end-to-end voice assistant pipeline for Algerian Darija that integrates ASR, NLU, and TTS with a fully conversational interaction model.
- Limited labeled audio data for Darija, particularly for command-type utterances (as opposed to conversational or read speech).
- Absence of a standardized intent/entity dataset for Darija voice commands, making it impossible to benchmark NLU performance with established metrics.
- No publicly available TTS voice specifically optimized for Algerian Arabic phonology, forcing current projects to use MSA or pan-Arabic neural voices that produce mispronunciations on Darija lexical items.

1.10 Conclusion

This chapter has established the linguistic complexity of Algerian Darija and reviewed the current state of the art in Arabic and dialectal ASR. The combination of code-switching, regional phonological variation, orthographic instability, and the dual-script system (Arabic and Arabizi) makes Darija a genuinely challenging target for voice AI. Furthermore, the review of existing datasets CAFE, MADAR, Casablanca, and Masader alongside the broader resource gap identified in systematic reviews, highlights the severe under-representation of Algerian Darija despite recent initiatives. Recent advances such as DziriBERT and ELNER-DZ demonstrate progress in dialectal NLP, but no integrated, end-to-end voice assistant exists for Algerian Darija. The lack of large-scale, spontaneous speech corpora and dedicated NLU models directly justifies the need for a practical, end-to-end voice assistant built with open-source, offline-capable technologies.

Chapter 2: Core technologies and techniques for voice-based interaction systemes

2.1 Introduction

This chapter presents the fundamental technologies that enable voice-based interaction systems. The core components are presented in pipeline order: Automatic Speech Recognition (ASR), Natural Language Processing (NLP), and Text-to-Speech (TTS). For each technology, the underlying principles and common implementation approaches are introduced.

2.2 Machine learning and deep learning

2.2.1 Machine learning

Machine Learning (ML) is a subfield of Artificial Intelligence that focuses on developing algorithms that enable computers to learn from and make decisions based on data, without being explicitly programmed for every possible outcome. As Tom Mitchell provided the most widely cited formal definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" (Mitchell, 1997)[27]. This means that instead of writing a program that directly performs a task, one writes a program that allows the computer to learn how to perform the task by examining examples or experiences, with the learning process itself producing the program.

ML techniques are broadly categorized into three main types:

- **Supervised learning:** The algorithm learns from labeled data, where each training example consists of an input paired with a known output. The goal is to learn a general rule that maps inputs to outputs, enabling predictions on new, unseen data. Common tasks include classification and regression.
- **Unsupervised learning:** The algorithm learns from unlabeled data, without predefined outputs. It aims to discover hidden structures, patterns, or relationships within the dataset.
- **Reinforcement learning:** The algorithm learns through interaction with an environment, receiving feedback in the form of rewards or penalties. The agent's objective is to learn a policy that maximizes cumulative reward.

To illustrate the practical application of machine learning techniques in the context of voice assistants and dialectal speech processing, Table 2.1 provides examples of existing technologies, the ML methods they employ, and the dialects they support.

Table 2.1 – Some machine learning techniques in voice & language technologies

Technology / Application	ML technique used	Reference	Dialect/Language supported
Classical ASR Systems	Hidden Markov Models (HMMs) with Gaussian Mixture Models (GMMs)	Khelifa et al., 2017 **[28]**	Modern Standard Arabic (MSA)
Darija Banking Intent Detection	Fine-tuning of BERT-based models (BERTouch)	Skiredj et al., 2024 **[17]**	Moroccan Darija, MSA
Arabic Dialect Identification	Statistical Classification (SVM, Logistic Regression)	Bouamor et al., 2018 **[23]**	25 Arabic dialects (including Maghrebi)

Table 2.1 presents an overview of machine learning techniques currently used in voice and language technologies. Classical approaches such as Hidden Markov Models (HMMs) with Gaussian Mixture Models (GMMs) have historically been the foundation of ASR systems, primarily supporting Modern Standard Arabic. More recent works have explored fine-tuning BERT-based models like BERTouch for Moroccan Darija intent detection, while classical classifiers such as Logistic Regression and Support Vector Machines remain widely used in NLP tasks. The table highlights that while ML techniques have been applied to various Arabic dialects, Algerian Darija remains largely unexplored.

2.2.2 Deep learning

Deep Learning (DL) is a specialized and more advanced branch of machine learning that employs artificial neural networks with multiple hidden layers to learn hierarchical representations of data. As defined by Goodfellow, Bengio, and Courville (2016) in their seminal textbook: "Deep learning is a form of machine learning that enables computers to learn from experience and understand the world in terms of a hierarchy of concepts. Because the computer gathers knowledge from experience, there is no need for a human computer operator to formally specify all the knowledge that the computer needs."

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton (2015)[30] further elaborate that deep learning "allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction." This hierarchical structure allows the computer to learn complicated concepts by building them out of simpler ones, with the depth of these hierarchies enabling the model to capture increasingly abstract and complex features.

Unlike traditional machine learning approaches, which often rely on manual feature engineering, deep learning models automatically discover the representations needed for detection or classification directly from raw data. This capability has led to dramatic improvements across numerous domains, including speech recognition, visual object recognition, object detection, natural language processing, and many other fields.

The effectiveness of machine learning and deep learning models depends on the quality of data, algorithm design, and computational power (Goodfellow et al., 2016)[29]. Historically, the evolution from rule-based "IF-THEN" systems to statistical machine learning, and subsequently to deep learning, has been driven by increases in computational power, the availability of large datasets, and advances in algorithmic design (Schmidhuber, 2015)[31].

To demonstrate the application of deep learning in voice and language technologies, Table 2.2 presents examples of existing systems, the DL architectures they utilize, and the dialects they address.

Table 2.2 – Deep Learning Techniques in Voice & Language Technologies

Technology/ Application	**DL technique used**	**Reference**	**Dialect/Language supported**
Whisper (Fine-tuned for Darija)	Transformer-based Sequence-to-Sequence Model	Enhancing ASR for Moroccan Darija, 2025 **[18]** ; Radford et al., 2023 **[2]**	Moroccan Darija
Wav2Vec2-XLSR-53 (Fine-tuned for Darija)	Self-Supervised Transformer (wav2vec 2.0)	Enhancing ASR for Moroccan Darija, 2025 **[18]**	Moroccan Darija
Octopus (Arabic	Whisper-V3 + LLMs	Octopus, 2025 **[32]**	Arabic-English,

Speech LLM)	(ALLaM, LLaMA, DeepSeek) + projection layer		Multiple Arabic dialects
Qwen3-ASR (Fine-tuned on MoulSot corpus)	Transformer-based ASR (1.7B params)	MoulSot Project, 2026 **[33]**	Moroccan Darija
CNN-based ASR for Drone Control	Convolutional Neural Networks (CNN)	Moroccan Darija ASR using CNN, 2025 **[34]**	Moroccan Darija
Multi-Dialect Arabic ASR	Transformer-based architectures with Transfer Learning	Talafha et al., 2024 **[24}**	Multiple Arabic dialects (including Maghrebi)

Table 2.2 provides an overview of deep learning techniques employed in modern voice and language technologies. Transformer-based models such as Whisper, Wav2Vec2-XLSR-53, and Qwen3-ASR have been fine-tuned for Moroccan Darija, demonstrating the feasibility of adapting large-scale pre-trained models to low-resource dialects. Multi-dialect Arabic ASR systems have also emerged, covering Maghrebi varieties. However, no dedicated end-to-end voice assistant exists for Algerian Darija. Most available models are either focused on Moroccan Darija, trained on Modern Standard Arabic, or designed for other dialect groups. This gap directly motivates the development of DziriTalk, a voice assistant specifically tailored for Algerian Darija.

2.3 Speech recognition (ASR)

2.3.1 Definition

Speech recognition refers to the ability of machines or computers to identify and interpret spoken language, converting it into a textual or command format, thereby enabling humans to interact with devices or systems using voice commands for hands-free control and efficient communication [1][35]. This technology has evolved significantly over the years, moving from basic voice recognition to advanced systems capable of understanding continuous speech and handling various accents and languages [2]. Speech recognition systems are designed to process audio signals, analyze them, and generate textual transcriptions or execute specific

commands based on the input, relying on probabilistic models such as acoustic models and language models to create robust representations of human speech and language structures [7].

2.3.2 Fundamental concepts of speech recognition

The fundamental concepts of speech recognition involve converting speech signals into text or commands through identification and understanding processes [2]. This technology typically consists of several modules, including front-end processing, recognition, and post-processing [36].

- **Speech signal and preprocessing:** The speech signal is a complex audio signal that includes the spoken words, pauses, and background noise. Preprocessing is a critical step in speech recognition, as it reduces the amount of processing required in later stages. This involves blocking the speech samples into frames (e.g., 25 ms frames with a 10 ms shift) and extracting unique patterns from each frame. Preprocessing helps in enhancing the quality of the speech signal and preparing it for further analysis [7].
- **Acoustic model:** The acoustic model is a probabilistic model that represents the relationship between speech sounds and their acoustic features. It is trained on a large dataset of speech samples and their corresponding transcriptions. The acoustic model is responsible for identifying the sequence of sounds (phonemes) in the speech signal [7]. Modern implementations often use deep neural networks (DNNs) or end-to-end architectures such as Transformers [1].
- **Language model:** The language model is another probabilistic model that predicts the likelihood of a sequence of words occurring in a language. It is trained on large text corpora and is used to constrain the possible interpretations of the speech signal, ensuring that the recognized text is grammatically correct and contextually appropriate [36].
- **Lexical model:** The lexical model, also known as the pronunciation model, contains information about the pronunciation of words in a language. It maps words to their phonemic representations, which are then used by the acoustic model to match the speech signal [36].
- **Decoder:** The decoder is the component responsible for combining the outputs of the acoustic and language models to generate the final transcription. It searches for the most likely sequence of words that match the speech signal,

considering both acoustic and linguistic constraints [7]. In practical systems, decoders often use weighted finite-state transducers (WFSTs) with beam search [28].

2.3.3 Architecture of speech recognition systems

The architecture of speech recognition systems typically consists of several key components, each performing a specific function in the recognition process. These components work together to convert the speech signal into a textual representation or execute commands based on the input [2].

- **Speech signal acquisition:** The first step in the architecture is the acquisition of the speech signal. This is typically done using a microphone, which captures the audio input from the user.
- **Preprocessing:** The acquired speech signal is then preprocessed to enhance its quality and prepare it for further analysis. This step may include noise reduction, normalization, and segmentation of audio into frames. Feature extraction techniques like Fourier Transform or Mel-Frequency Cepstral Coefficients (MFCCs) extract spectral features that represent speech characteristics [28].
- **Feature extraction:** Feature extraction is the process of converting the speech signal into a set of acoustic features that can be analyzed by the acoustic model. Common features include Mel-Frequency Cepstral Coefficients (MFCCs) and spectrograms [28][7].
- **Acoustic model:** The acoustic model processes the extracted features to identify the sequence of phonemes in the speech signal. This model is typically implemented using techniques such as Hidden Markov Models (HMMs) with Gaussian Mixture Models (GMMs) or deep neural networks (DNNs), Transformers like Whisper [28][2].
- **Language model:** The language model processes the sequence of phonemes generated by the acoustic model to predict the most likely sequence of words. This model is typically implemented using n-gram models or recurrent neural networks (RNNs), Transformer-based models (e.g., BERT for contextual understanding) [39][1].
- **Decoder:** The decoder combines the outputs of the acoustic and language models to generate the final transcription. It searches for the most likely

sequence of words that match the speech signal, considering both acoustic and linguistic constraints [7].

- **Postprocessing:** The final step in the architecture is postprocessing, which may include spell-checking, grammar-checking, and formatting the output to ensure it is accurate and readable [35].

2.3.4 Applications and use cases of speech recognition

Speech recognition technology has a wide range of applications across various industries, from consumer electronics to healthcare and education [41]. Some of the most common applications and use cases include:**Voice assistants:** Voice assistants, such as Siri, Alexa, and Google Assistant, use speech recognition to understand and execute voice commands. These assistants can perform tasks such as setting reminders, playing music, and answering questions [5].

- **Speech-to-text systems:** Speech-to-text systems convert spoken words into written text, allowing users to dictate documents, emails, and messages. This is particularly useful for individuals with disabilities or those who prefer hands-free typing [35].
- **Voice-controlled devices:** Speech recognition is used in voice-controlled devices such as smart speakers, televisions, and home automation systems. These devices allow users to control their surroundings using voice commands [5].
- **Transcription services:** Speech recognition is used in transcription services to convert audio and video recordings into written text. This is commonly used in industries such as media, education, and law enforcement [35].
- **Language learning:** Speech recognition technology is used in language learning applications to help users improve their pronunciation and speaking skills. These applications can provide real-time feedback on pronunciation and intonation [35].
- **Medical applications:** In the healthcare industry, speech recognition is used for medical transcription, voice-controlled medical devices, and patient-doctor communication. It allows healthcare professionals to dictate notes and commands without the need for manual input [35].
- **Customer service:** Speech recognition is used in customer service applications

such as interactive voice response (IVR) systems. These systems allow customers to interact with automated agents using voice commands, reducing the need for human intervention [41].

2.3.5 From hidden markov models to end-to-end transformers

Automatic speech recognition is the task of mapping a raw audio waveform to its corresponding text transcription. Historically, ASR systems decomposed this problem into a sequence of specialized components: an acoustic model (mapping acoustic frames to phonemes), a pronunciation lexicon (mapping phonemes to words), and a language model (assigning probabilities to word sequences).

Hidden Markov Models (HMMs) trained with the Expectation-Maximization algorithm dominated this architecture for nearly three decades, with Gaussian Mixture Models used to model the observation distributions of each HMM state. The deep learning revolution of the early 2010s brought recurrent neural networks into the ASR acoustic model, replacing hand-crafted features like Mel Frequency Cepstral Coefficients (MFCCs) with learned representations. The introduction of the Connectionist Temporal Classification (CTC) objective by Graves et al. (2006) [8] enabled end-to-end training of acoustic models without requiring frame-level alignment labels. Subsequent work on attention-based encoder-decoder models, culminating in the transformer architecture, enabled fully end-to-end ASR systems that jointly learned acoustic, lexical, and language model components from transcribed audio.

2.3.6 Wav2Vec 2.0: self-supervised pre-training for low-resource ASR

A significant breakthrough for low-resource ASR came with the introduction of wav2vec 2.0 by Baevski et al. (2020) [7] at Facebook AI Research. Wav2vec 2.0 demonstrated that powerful speech representations could be learned from large quantities of unlabeled audio through self-supervised pre-training, using a contrastive learning objective operating on quantized latent representations. The resulting representations could then be fine-tuned on very small amounts of labeled data: the model achieved competitive word error rates on standard English benchmarks using as little as ten minutes of transcribed speech. This paradigm is directly relevant to low-resource settings like Darija, where collecting labeled speech is expensive and time-consuming.

The wav2vec 2.0 architecture consists of three components: a multi-layer convolutional feature encoder that processes raw waveforms into latent representations, a transformer context network that captures long-range dependencies across the sequence, and a quantization

module that produces discrete speech units used as targets during pre-training. After pre-training on unlabeled audio, the context network is fine-tuned end-to-end with a CTC loss on labeled transcriptions.

2.3.7 Whisper: multilingual weak supervision at scale

OpenAI's Whisper (Radford et al., 2023) [2] is a multilingual ASR model trained end-to-end on 680,000 hours of audio-transcript pairs from the internet, spanning 99 languages. Its architecture is a standard transformer sequence-to-sequence model. Log-Mel spectrogram features are extracted from input audio and processed by a convolutional encoder; the transformer decoder then autoregressively generates the output transcription, conditioned on encoder representations and special task-specifying tokens. This design allows a single model to perform multiple tasks (transcription, translation, voice activity detection) without architectural modifications. Whisper has shown strong zero-shot performance on diverse speech domains, including Arabic. For low-resource dialects like Darija, Whisper offers a usable baseline without requiring language-specific training. Inference can be guided using an initial prompt – a short textual hint – to bias transcriptions toward expected vocabulary.

2.4 Natural language processing (NLP)

2.4.1 Definition

Natural language processing is a subfield of Artificial Intelligence (AI) that assists computers in understanding, interpreting, and manipulating human language to automate tasks between machines and humans [1][35]. It evolved from computational linguistics, which uses computer science to understand the principles of language, but rather than developing theoretical frameworks, NLP is an engineering discipline that seeks to build technology to accomplish useful tasks [1]. Specifically, NLP can be divided into two overlapping subfields: natural language understanding (NLU), which focuses on semantic analysis or determining the intended meaning of text, and natural language generation (NLG), which focuses on text generation by a machine [1][35].

2.4.2 How NLP works

NLP models work by finding relationships between the constituent parts of language—such as letters, words, and sentences found in a text dataset—and architectures employ various methods for data preprocessing, feature extraction, and modeling to achieve this understanding [1][37]. The typical sequence of preprocessing steps in an NLP pipeline is visualized in the

following Figure.

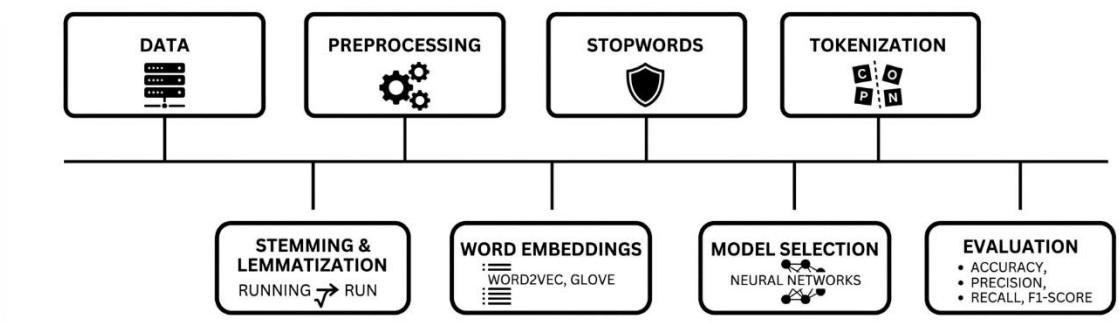


Figure 2.1 – NLP preprocessing pipeline from raw data to evaluation metrics.

Some of these processes are:

Data preprocessing: It is frequently necessary to preprocess text before a model processes it for a particular job in order to enhance model performance or to convert words and characters into a format that the model can comprehend [37]. There are several methods that can be applied:

Tokenization: Tokenization is the process of breaking down words or sentences into semantic units known as 'tokens'. This enables us to deal with shorter chunks of text, which can be easily understood even when read independently from the context of the text [1].

The figure below illustrates how a sentence is broken into individual tokens.

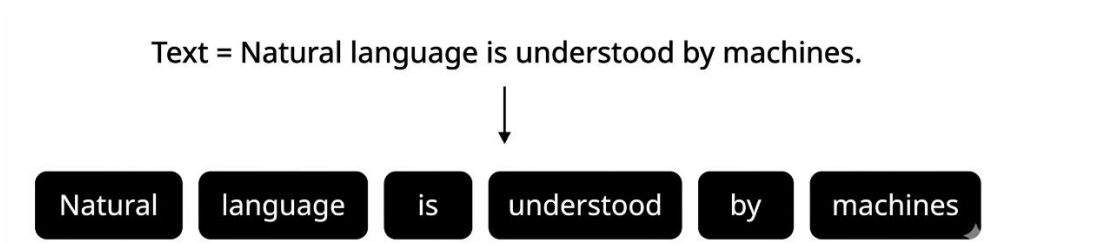


Figure 2.2 – Tokenization example: splitting a sentence into semantic units.

Stop words removal: Stop Words Removal is a text pre-processing technique that eliminates words with zero significance so that the model can focus on meaningful words [37]. Stop words include the, a, I, an, be, you, had, has, were, was, will, does, is and many more.

This figure shows the effect of removing stop words from a sentence.

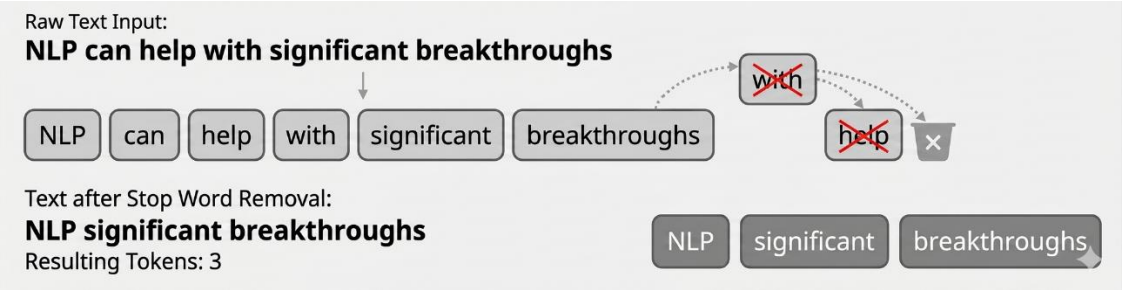


Figure 2.3– Stop word removal: only content words remain.

Stemming & lemmatization: It is the process of analyzing variation of words to reveal their origin or root word. Stemming reduces the words to their root form by removing the last letters. Lemmatization also reduces the words to their original form but performs analysis in a group of words to find its root word ("Lemma"), which represents a group of words ("lexeme") [1][37].

Word embeddings: Word Embeddings is an NLP technique where words are converted into numbers or vectors [1]. Significant techniques include TF-IDF, BOW (Bag-of-Words), Count Vectorization, N-grams Vectorization, Word2Vec, and GloVe [42].

- **NLP model selection and architecture design:** Model selection is the process of choosing an appropriate machine learning or deep learning model for an NLP application [37]. The dataset is divided into training and testing sections, based on which the model is trained and tested [42].
- **Training NLP models:** NLP pre-trained models are useful for tasks like translating text, predicting missing parts of a sentence, or generating new sentences [35]. They can be used in many NLP applications such as chatbots and NLP APIs [41].
- **Performance evaluation matrix:** The most common evaluation matrix include accuracy, precision, recall, and F1-score [37][42].

2.4.3 Natural language processing techniques

Most NLP tasks can be modeled by a set of general techniques, which fall into two categories: traditional machine learning methods and deep learning methods [1].

Traditional machine learning NLP techniques :

- **Logistic regression:** A supervised classification algorithm that predicts the probability that an event will occur. In NLP, it can be applied to sentiment analysis, spam detection, and toxicity classification [37].
- **Naive bayes:** A supervised classification algorithm that finds the conditional probability distribution. The naive assumption is that individual words are independent. It can be applied to spam detection or finding bugs in software code [37].
- **Decision trees:** Supervised classification models that split the dataset based on different features to maximize information gain [37].
- **Latent dirichlet allocation (LDA):** Used for topic modeling. LDA views a document as a collection of topics and a topic as a collection of words, and is a statistical approach based on describing topics using a small set of words from the corpus [37][42].

Deep learning NLP techniques :

- **Convolutional neural network (CNN):** The idea of using a CNN to classify text treats a document analogously to an image, with sentences or documents represented as a matrix of words rather than pixels [1][35].
- **Recurrent neural network (RNN):** RNNs remember previous information using hidden states and connect it to the current task. The Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) architectures are types of RNNs designed to remember information for an extended period. Bidirectional LSTM/GRU keeps contextual information in both directions [1][35].

The bidirectional RNN architecture, which reads sequences from left to right and right to left, is shown in this Figure.

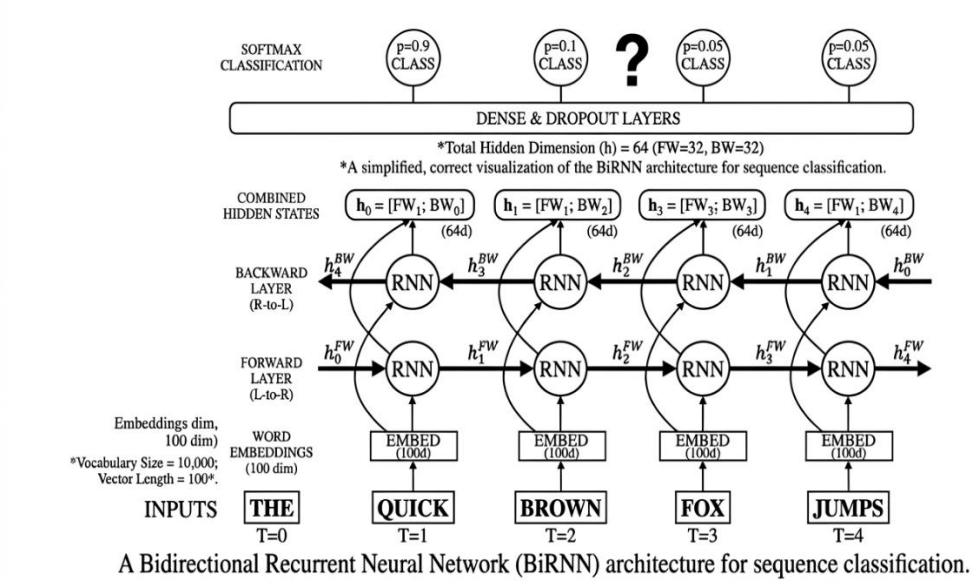


Figure 2.4 – Bidirectional RNN (BiRNN) architecture for sequence classification.

Transformers: The transformer architecture (Vaswani et al., 2017) [1] relies entirely on a self-attention mechanism to draw global dependencies between input and output. Because this mechanism processes all words at once rather than one at a time, training is faster and more parallelisable than with RNNs [1][41]. The transformer has revolutionized NLP, leading to models including BERT, GPT, and T5 [35].

2.4.4 Opportunities and challenges of NLP

- **Opportunities:** The rapid advancement of NLP technologies has created unprecedented opportunities across multiple domains. From automating routine tasks to enabling entirely new forms of human-computer interaction, NLP is reshaping how individuals and organizations process language at scale. The following are some of the most significant opportunities afforded by modern NLP.
- **Automation in recruitment:** NLP streamlines resume parsing and candidate matching, reducing time-to-hire [35].
- **Healthcare applications:** NLP aids in diagnosing conditions, extracting information from clinical notes, and identifying biomarkers in research [35].
- **Generative AI development:** Advances in large language models (LLMs) enhance NLP capabilities, leading to improved applications in education and healthcare [35].
- **Challenges:** Despite its transformative potential, the deployment of NLP systems is fraught with significant challenges that must be addressed to ensure responsible and effective use. These challenges are not merely technical but also ethical, social, and

resource-related. For low-resource dialects like Algerian Darija, these issues are particularly acute, as the lack of standardized data and the prevalence of code-switching compound existing difficulties. The following are key challenges facing the field.

- **Algorithmic bias:** NLP systems can perpetuate biases present in training data, leading to unfair outcomes in applications like recruitment and content moderation [35][41].
- **Data privacy concerns:** The handling of sensitive information raises ethical issues, particularly in healthcare settings [35].
- **Contextual understanding:** NLP struggles with nuances in language, such as cultural differences and contextual meanings [35][41].

2.4.5 The intent detection and slot filling problem

Natural language understanding (NLU) in a voice assistant context reduces primarily to two subtasks: intent detection, which classifies the user's utterance into one of a predefined set of action categories; and slot filling, which extracts the specific parameters needed to execute the action (e.g., the name of the application to open).

Together, these two components constitute the "semantic parsing" layer that bridges raw transcription and executable action.

Classical approaches to intent detection used feature-engineered classifiers – Support Vector Machines, Naive Bayes, or logistic regression operating on bag-of-words or TF-IDF representations [38].

These approaches are interpretable, fast, and surprisingly effective for small, well-defined command vocabularies, and they remain relevant in low-resource settings where the training set is too small to fine-tune a large pre-trained model.

2.4.6 BERT and transformer-based NLU

The dominant paradigm in modern NLU is fine-tuning of large pre-trained language models. BERT (Bidirectional Encoder Representations from Transformers), introduced by Devlin et al. (2019) [39] at Google, pre-trains a transformer encoder on massive text corpora using two self-supervised objectives: masked language modeling (predicting randomly masked tokens from their context) and next-sentence prediction.

The resulting model captures rich contextual representations that can be adapted to downstream tasks including intent detection by adding a lightweight classification head and fine-tuning on labeled examples.

BERT's bidirectional design is crucial: unlike earlier left-to-right language models, BERT conditions each token representation on the full context of the surrounding sentence, enabling more accurate semantic interpretation.

For short voice commands, which may consist of only three or four tokens, this bidirectionality helps the model distinguish semantically similar intents that differ in subtle ways.

For Arabic NLU, AraBERT (Antoun et al., 2020) [15] and CAMeLBERT (Inoue et al., 2021) [16] provide pre-trained encoders trained on large Arabic text corpora.

However, their effectiveness on Darija is limited by the mismatch between MSA pre-training data and dialect input.

2.4.7 Text normalization for darija

Before intent classification, ASR transcriptions often require a normalization step. Due to the lack of standardized orthography in Darija and the acoustic variability of code-switched speech, an ASR model may produce multiple surface variants of the same intended word. A normalization module can apply rule-based mappings (e.g., regular expressions) to unify these variants into a canonical form before further processing. This preprocessing step can substantially improve downstream intent classification accuracy without requiring model retraining.

2.5 Text-to-Speech (TTS)

Text-to-Speech synthesis converts text into natural-sounding speech audio. The dominant approach in modern TTS is a two-stage neural pipeline: an acoustic model (e.g., Tacotron 2 [40], FastSpeech 2 [4]) that converts input text into intermediate acoustic features such as mel spectrograms, followed by a vocoder (a speech synthesis module that converts acoustic features (such as mel-spectrograms) into the final raw audio waveform that the user hears) that converts those features into a time-domain waveform.

For Arabic TTS, the primary challenge is accurate rendering of phonological features: emphatic consonants, pharyngeal and uvular fricatives, and long/short vowel distinctions. Dialect-specific TTS adds the further requirement of correctly pronouncing borrowed French vocabulary alongside Arabic morphology. Some cloud-based TTS services offer voices for Algerian Arabic (e.g., Microsoft Azure [36]), but they may require an internet connection. Local neural TTS engines can also be used for offline deployment.

2.6 Tools and frameworks

Developing a voice assistant for a low-resource dialect requires a combination of audio

capture, ASR, NLP, and TTS components. Open-source libraries and frameworks – such as Python's sounddevice for audio input, Hugging Face Transformers for model access, and various TTS engines – provide the necessary building blocks. The choice of specific tools depends on deployment constraints such as offline operation, hardware limitations, and target language support.

2.7 Conclusion

This chapter has presented the fundamental technologies that underpin modern voice assistants, with a focus on Automatic Speech Recognition (ASR), Natural Language Processing (NLP), and Text-to-Speech (TTS) synthesis. The progression from classical Hidden Markov Models to end-to-end transformer-based architectures was outlined, highlighting the increased robustness and accuracy brought by deep learning. Key NLP concepts such as tokenization, stop word removal, intent detection, and slot filling were introduced as essential steps for interpreting user commands. Finally, the role of neural TTS in producing natural and expressive speech was discussed. Together, these technologies provide the necessary building blocks for any voice-driven conversational system. The next chapter will describe how these components are integrated into a complete system architecture tailored specifically for the Algerian Darija dialect.

Chapter 3: System design and performance evaluation

3.1 Introduction

This chapter presents the overall design of the DziriTalk voice assistant and evaluates its performance. The system is built as a modular pipeline that converts spoken Darija commands into desktop actions and spoken responses. The core components are: data collection and dataset preparation, wake-word detection, audio capture, automatic speech recognition (ASR), text normalization, intent classification (combining a text-based neural model with an audio-based neural model), action execution, text-to-speech (TTS) synthesis, and modular code organization. All components run locally on a standard Windows computer without requiring a dedicated GPU. The description follows the chronological order of development: first, the dataset collection and preparation; second, the system architecture and implementation; and third, the evaluation of the system's performance.

3.2 Data Collection and Dataset Preparation

To train the intent classifiers that form the core of the DziriTalk voice assistant, a dedicated dataset of spoken Darija commands was required. At the time of this work, no publicly available dataset existed specifically for Algerian Darija voice commands targeting desktop automation tasks. Commercial datasets, where they exist, are dominated by Modern Standard Arabic or other regional dialects, and are typically not suitable for the code-switched, non-standardized nature of Algerian Darija. Consequently, a custom dataset had to be collected from scratch, a process that proved to be one of the most time-consuming and resource-intensive phases of the project.

3.2.1 Data Collection Methodology

The dataset was collected through two complementary approaches: a crowdsourced campaign leveraging Google Forms, and direct manual recordings by the developer.

3.2.1.1 Crowdsourced Collection (Google Forms)

A Google Form was designed and distributed via social networks and university channels to collect audio recordings from volunteers across different Algerian regions. This approach ensured accent diversity and captured the natural phonetic variation characteristic of Algerian Darija. Figure 3.1 shows the introduction page of the Google Form, presenting the purpose of the data collection and instructions for participants.

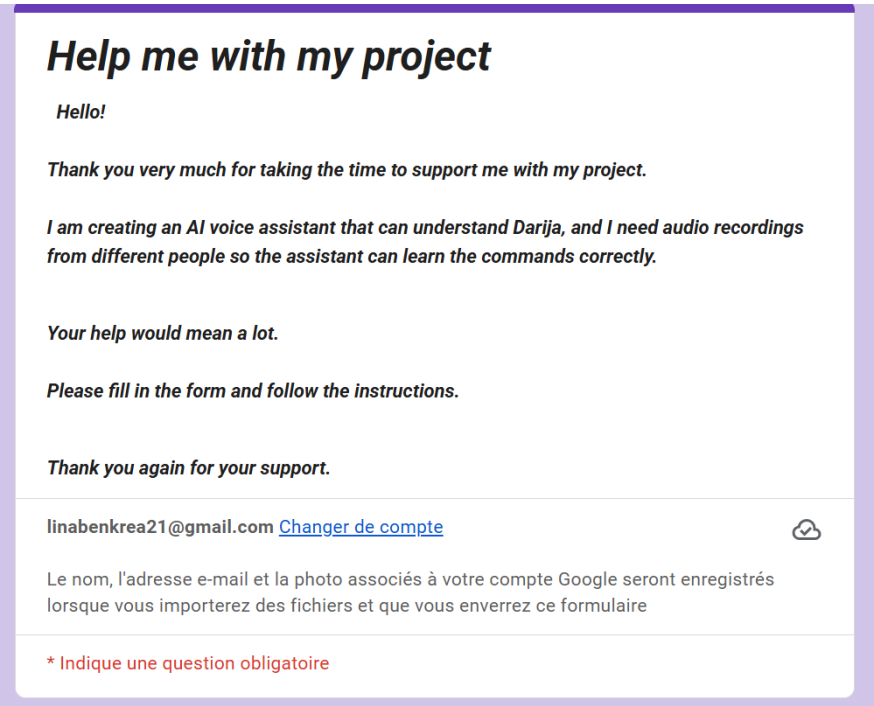


Figure 3.1 – Google Form introduction page

Figure 3.2 presents the metadata fields collected from each participant, including name (optional), gender, age, and region.

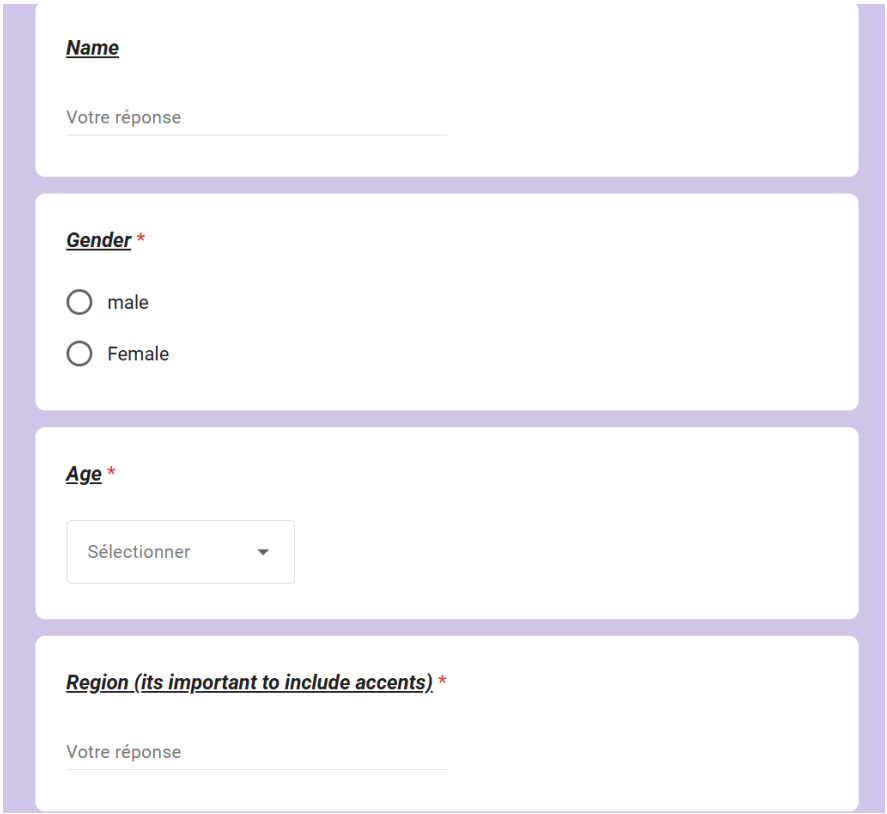


Figure 3.2 – Google Form metadata fields (name, gender, age, region)

Figure 3.3 displays the command list and recording instructions, specifying the required audio files and naming conventions.

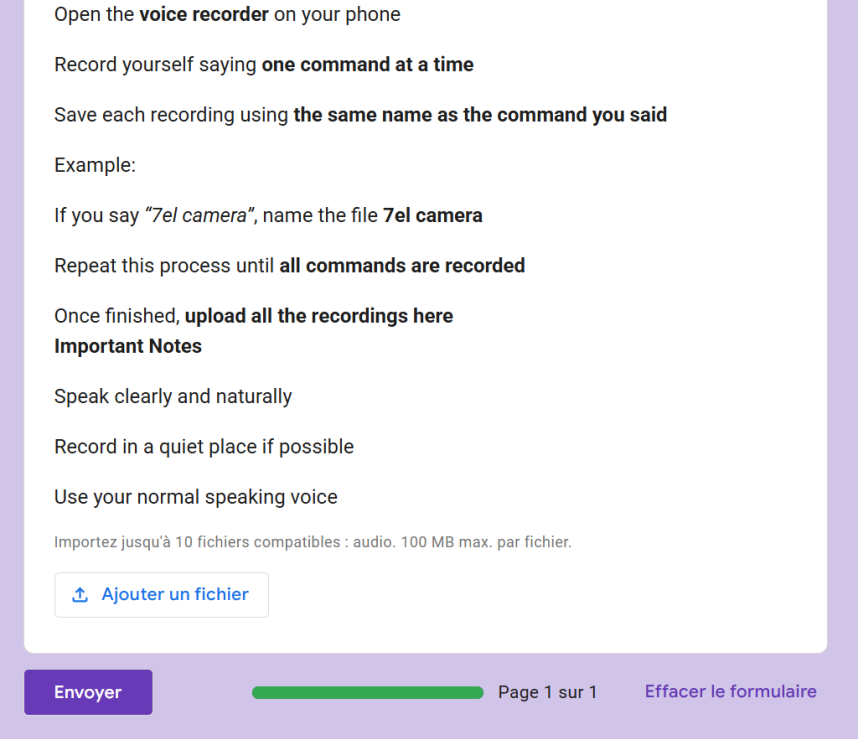


Figure 3.3 – Google Form command list and recording instructions

The form collected the following information from each participant:

- **Metadata**: Name (optional), gender (male/female), age (dropdown), and region (text input).
- **Audio recordings:** Up to 10 audio files per submission, recorded one command at a time.
- **Naming convention:** Each file was required to be named after the command it contained (e.g., 7el camera.m4a).

The following commands were requested from each participant:

- 3araf rohak (introduce yourself)
- chkoun nta (who are you)
- wsh t9der dir? (what can you do)
- 7el camera (open camera)
- 7el messages (open messages)
- 7el telegram (open Telegram)

- 7el youtube (open YouTube)
- 7el tiktok (open TikTok)
- choufli la meteo lyoum kifach (show me today's weather)
- 9adeh sa3a (what time is it)
- Wech rana lyoum (what day is it today)

3.2.1.2 Manual Street Recordings

To supplement the crowdsourced data and ensure coverage of edge cases, the developer manually recorded multiple takes of each command directly in real-world environments. Recordings were made:

- In quiet rooms (controlled conditions).
- With background noise (street, classrooms).
- At different distances from the microphone.
- At different times of day to capture vocal variation.

This method ensured that every command had sufficient examples, particularly for less common intents, and that the model would generalize to realistic acoustic conditions encountered during deployment.

3.2.2 Data Labeling and Organization

After collection, all audio files were manually labeled and organized into a structured directory hierarchy. Each recording was associated with its corresponding intent category and, where applicable, the target application name (for the open_app intent). The labeling process was conducted manually to ensure accuracy, as automatic labeling tools were not reliable for Darija speech. Each file was placed in a folder corresponding to its intent, with a naming convention that encoded the speaker identifier and take number:

speaker__intent__app-NAME__takeXX.wav

This manual labeling process was critical because Darija lacks standardized orthography, making automated transcription and labeling unreliable. It also ensured that the dataset could serve as a ground-truth reference for training and evaluation.

3.2.3 Dataset Processing Pipeline

All collected audio files were processed through a custom Python pipeline to standardize the

data and prepare it for model training.

3.2.3.1 Audio Conversion and Standardization

Recordings were collected in various formats (m4a, mp3, ogg, wav) from different devices (smartphones, laptops, external microphones). Using FFmpeg, all files were converted to a standardized format suitable for training deep learning models:

- Sample rate: 16 kHz
- Channels: Mono
- Bit depth: 16-bit PCM
- Format: WAV

This standardization ensured consistency across the dataset and compatibility with the feature extraction pipelines used by both the ASR engine and the audio classifier.

3.2.3.2 Speaker-Level Splitting

To prevent data leakage and obtain a realistic evaluation of generalization capability, a speaker-level split was applied. This ensures that all utterances from the same speaker appear exclusively in either the training set or the test set, never both. An 80/20 train/test split was adopted, with the validation set monitored during training via Keras EarlyStopping, eliminating the need for a separate validation split given the dataset's modest size.

3.2.3.3 Data Augmentation

The audio dataset suffered from severe class imbalance. The open_app category, which comprised 11 sub-folders (each corresponding to a specific application), contributed approximately 540 samples, while current_time and current_date had only 14–15 original recordings each. To address this imbalance before training the audio classifier, six augmentation techniques were applied individually and in combination:

- Pitch shifting (±2.5 semitones)
- Time stretching (rate 0.75–1.25)
- Additive white noise (amplitude 0.002–0.012)
- Volume scaling (factor 0.5–1.5)
- Reverb simulation (delay 15–60 ms)
- Time shifting (±100 ms)

All folders corresponding to the 8 intent classes (with the 11 open_app sub-folders counted collectively as a single intent) were brought to exactly 100 samples each, yielding a final balanced audio corpus of 1,700 samples (approximately 85 minutes of effective training material).

3.2.4 Dataset Statistics

Table 3.1 summarizes the final dataset collected for training the intent classifiers.

Table 3.1 – Dataset statistics

Statistic	Value
Total speakers	110+
Total original recordings	400+
Commands per intent (original)	14–67
Commands per intent (post-augmentation)	100
Total augmented samples	1,700
Effective training duration	~85 minutes
Regions represented	Algiers, Oran, Constantine, Annaba, Skikda, and others
Intent classes	6 (including open_app with 11 application sub-folders)

The dataset reflects diverse age groups, genders, and regional accents, with recordings made in various acoustic conditions (quiet rooms, street noise, classrooms) to ensure model generalization. The regional diversity captures the phonetic and lexical variation characteristic of Algerian Darija.

3.3 Overall Pipeline

The assistant follows a linear processing chain, with activation either by a user-defined wake word or by pressing the microphone button. The main steps are:

- **Continuous listening** : a low-footprint speech recognizer constantly analyzes

the microphone input.

- **Wake word detection** : when the recognizer spots the pre-defined wake word, the system proceeds to record the following command.
- **Command recording** : the assistant captures the user's speech until silence is detected or a maximum duration is reached.
- **Speech-to-text (ASR):** the recorded audio is transcribed into a Latin-script text that uses Arabizi numerals.
- **Text normalization** : the raw transcription is cleaned: spelling variants are unified, Arabic script is converted to a canonical Latin form, and code-switched French words are mapped to standardized keywords.
- **Intent classification** : the normalized text is processed by a primary neural classifier. If its confidence is low, an auxiliary audio classifier analyzes the raw audio to correct or confirm the intent.
- **Action execution :** the recognized intent is mapped to a function that performs the requested operation.
- **Response generation** : the assistant produces a spoken answer using a three-tier TTS system.

3.4 Wake-Word Detection and Audio Capture

3.4.1 Wake-Word Detection

To enable hands-free activation, the system uses a lightweight, offline speech recognizer that continuously processes the microphone stream. The user can define any word or short phrase (e.g., "اسمع", "darija", "salam") as the wake word. When the recognizer detects this phrase as a substring in its live transcription, the assistant is triggered.

This approach was chosen over template-based methods because of the high variability in Darija pronunciation and background noise. A continuous transcription-based detector eliminates manual threshold tuning and improves robustness across different speakers and environments.

3.4.2 Audio Capture

Upon wake-word detection or manual button press, the system begins recording the user's command. The audio is captured at a 16 kHz sample rate, mono channel, with 16-bit resolution.

Recording stops after a configurable silence period (0.8 seconds of low energy) or a maximum duration (5 seconds). The captured audio is saved as a temporary WAV file for subsequent processing.

3.5 Automatic Speech Recognition (ASR)

The ASR component is based on a multilingual end-to-end transformer model. The Whisper base model (74 million parameters, INT8 quantized) was chosen because it runs efficiently on CPU-only hardware while still providing acceptable zero-shot performance on Darija speech. INT8 quantization reduces the memory footprint by approximately 4× and speeds up CPU inference by approximately 2×, with negligible accuracy loss. The model is loaded once at application startup so that the first wake-word command does not suffer from a cold-start delay.

To improve recognition of Darija-specific words and Arabizi numerals, the decoder is guided by an initial prompt a short textual hint that lists expected command patterns, application names, and time/date queries. This prompt biases the model toward the correct vocabulary and significantly reduces word error rate compared to using the model without any hint.

The ASR engine runs locally and does not require an internet connection. Transcription proceeds in two passes: a fast first pass (beam_size=1) detects the dominant language (Arabic, French, or English) of the utterance; the second pass (beam_size=5) transcribes using the detected language and the Darija prompt, producing the final output. Voice Activity Detection (VAD) is applied with a 400 ms silence threshold to strip silent segments before processing, preventing the hallucination of random text from empty audio.

3.6 Text Normalization

The raw transcription produced by the ASR model may contain spelling variations, inconsistent use of Arabic digits, or mixed Latin/Arabic script. A normalization module applies a sequence of regular expression rules to obtain a canonical representation suitable for intent classification. Figure 3.4 illustrates the text normalization process applied to raw ASR transcriptions.

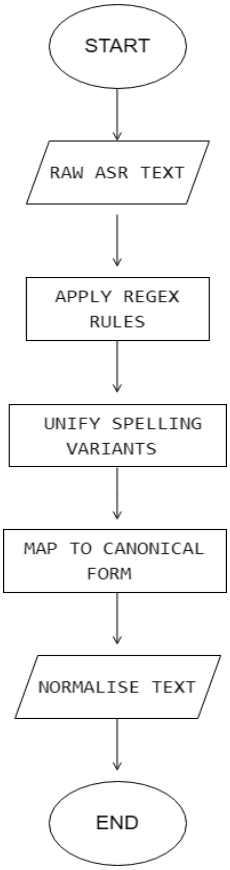


Figure 3.4 – Text normalization pipeline from raw ASR output to canonical form

The pipeline applies regular expression rules to unify spelling variants (e.g., "8el" → "7el"), maps code-switched French terms to canonical forms, and produces normalized text suitable for intent classification. This preprocessing step is critical for handling the orthographic variability of Darija and Arabizi numerals.

Table 3.2 provides examples of common normalizations applied to the raw ASR output.

Table 3.2 – Examples of text normalization rules

Raw variant	Normalized form
8el, hel, hol, h el	7el
camero, kamira, camra	camera
telegramme, tiligram	telegram
chkoun, shkoun, koun	chkoun

The module also handles code-switched French phrases and maps application names to a fixed set of identifiers.

3.7 Intent Classification

Intent classification is performed by a dual-model fusion architecture that combines a text classifier (primary) and an audio classifier (auxiliary). Both models were trained on the manually collected and crowdsourced Darija command dataset described in Section 3.2. Figure 3.5 presents the complete intent classification pipeline.

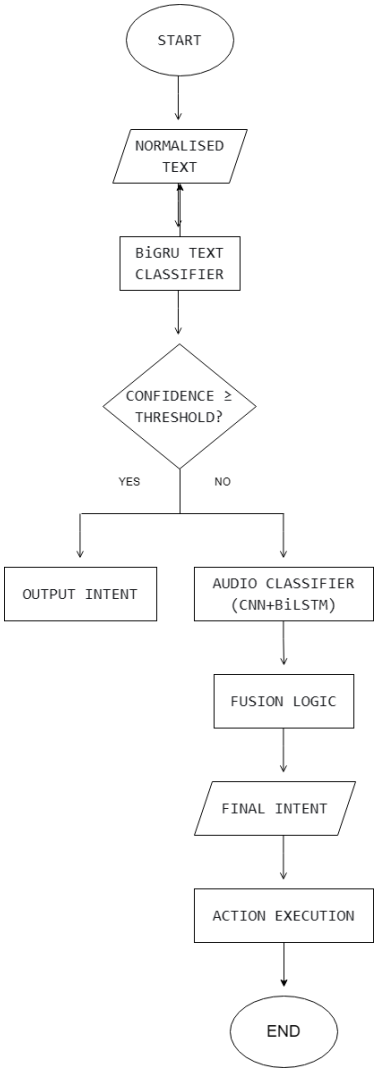


Figure 3.5 – Dual-model intent classification and fusion pipeline

The normalized text is first processed by the primary BiGRU text classifier. If the classifier's confidence meets or exceeds a threshold, the intent is output directly. Otherwise, the auxiliary CNN-BiLSTM audio classifier is invoked on the raw MFCC features, and a fusion logic combines both predictions to produce the final intent. This dual-model design ensures robustness against ASR transcription errors.

3.7.1 Text Classifier (BiGRU)

The text classifier is a deep neural network based on a Bidirectional Gated Recurrent Unit (BiGRU). It takes as input the normalized text, tokenized into a sequence of words. The architecture comprises:

- An embedding layer that learns dense vector representations for each word.
- Two stacked BiGRU layers that capture contextual dependencies in both forward and backward directions.
- Dropout layers for regularization.
- A final dense layer with softmax activation that outputs probabilities over the supported intent classes.

The model was trained on the collected dataset using the Adam optimizer (default learning rate 0.001) and categorical cross-entropy loss. EarlyStopping (patience=15, restore_best_weights=True) and ReduceLROnPlateau (factor=0.5, patience=7) callbacks prevent overfitting and optimize convergence. A vocabulary of 500 tokens and maximum sequence length of 12 (MAX_LEN=12) were used, with post-padding of shorter sequences. When the softmax confidence falls below 0.45, the output is classified as "unknown" to avoid triggering an incorrect action.

3.7.2 Audio Classifier (CNN-BiLSTM)

The audio classifier works directly on the raw command audio, independently of the ASR transcript. It extracts Mel-Frequency Cepstral Coefficient (MFCC) features from a fixed-length audio segment (3.0 seconds, 16,000 Hz, padded or truncated as needed), producing a time-frequency matrix of shape (94 frames × 40 coefficients). Each sample is normalized per-instance (subtract mean, divide by standard deviation + 1×10^{-8}) before being fed to the model. The architecture consists of:

- Two convolutional blocks (Conv1D with 64 and 128 filters respectively, kernel size 3, ReLU activation), each followed by BatchNormalization, MaxPooling1D (pool size 2), and Dropout (0.3), that extract local temporal phonetic patterns.
- A bidirectional long short-term memory (BiLSTM) layer that captures long-range dependencies.
- A dense classification head.

This model is particularly effective for commands that have a strong acoustic signature (e.g., opening applications with different names) and can correct transcription errors.

3.7.3 Fusion Logic

The final intent is decided by the following rules:

- Rule-based keywords take highest priority – certain phrases are directly mapped to intents (e.g., sha7al sa3a → current time, choufli la meteo → weather).
- Text classifier produces an intent and confidence.
- Audio classifier produces an intent and confidence.
- Override occurs when text confidence is low, the text intent is unknown, or the classifiers disagree on action intents.
- Gibberish detection – if the ASR output contains none of the 29 known Darija tokens (e.g., 7el, sa3a, wech, meteo, youtube), the transcription is classified as gibberish and the BiGRU is bypassed. The audio classifier then operates directly on raw MFCC features with a stricter confidence threshold (0.85 for open_app, 0.45 for all other classes).
- Conversational intents are never overridden by audio.

3.7.4 Intent Categories

Table 3.3 lists the supported intents and example commands.

Table 3.3 – Supported intents and example commands

Intent	Example command (normalized)
open_app	7el youtube
search_youtube	7el youtube w lahat 3la one piece
weather_today	choufli la meteo lyoum
current_time	sha7al sa3a
current_date	wech rana lyoum
who_are_you	chkoun nta
introduce_self	3araf rohak

capabilities	wsh t9der dir

3.7.5 Evaluation Methodology

To assess the effectiveness of the proposed intent classification pipeline, a two-pronged evaluation was conducted:

- Text-only (oracle) evaluation: The classifiers receive manually verified, clean transcriptions of commands. This establishes the upper bound of what is achievable from text alone, independent of ASR quality.
- Vocal (end-to-end) evaluation: The classifiers receive transcriptions produced by the Whisper ASR engine from recorded speech. This reflects real-world deployment conditions and measures the full pipeline's practical utility.

All evaluations were performed on a held-out test set (20% of the dataset, speaker-level split to avoid data leakage) comprising 163 utterances for the ML baseline and 90 utterances for the deep learning models, reflecting their distinct training splits.

3.7.6 Deep Learning Classifier Performance

Text Classifier (BiGRU) – Evaluation

The text classifier processes the normalized ASR transcription. Table 3.4 presents the per-class precision, recall, and F1-score, along with the overall accuracy for the BiGRU text classifier on the vocal test set.

Table 3.4 – BiGRU text classifier performance

Intent	Precision	Recall	F1-score	Support
capabilities	0.90	0.90	0.90	10
introduce_self	1.00	0.88	0.93	8
open_app	0.96	0.98	0.97	49
weather_today	0.93	1.00	0.97	14
who_are_you	0.88	0.78	0.82	9
Accuracy	–	–	0.94	90

Macro avg	0.93	0.91	0.92	90
Weighted avg	0.94	0.94	0.94	90

The BiGRU achieves an overall accuracy of 94% with a weighted average F1 of 0.94. open_app performs best (F1 = 0.97), which is desirable as it is the most frequent user command. The slightly lower recall for who_are_you (0.78) reflects occasional confusion with the semantically similar introduce_self and capabilities intents—all three share a conversational framing that is difficult to disambiguate from short utterances.

Audio Classifier (CNN-BiLSTM) – Vocal Evaluation

The audio classifier operates directly on raw MFCC features, independent of the ASR transcript. Following the dataset rebalancing and augmentation described in Section 3.2.3.3, the model was retrained on 1,700 balanced samples (100 per class across the 8 intent folders, with open_app comprising 11 application sub-folders), achieving 98% overall accuracy on a test set of 366 samples.

3.7.7 Classical Machine Learning Baseline

To contextualise the contribution of the deep learning architecture, a classical baseline was established using a Support Vector Machine (SVM) with TF-IDF vectorization. This model represents the traditional approach to text-based intent detection and was implemented using the scikit-learn library.

The SVM was trained on the same training split as the BiGRU, using:

- **Feature extraction:** TF-IDF with character n-grams (n=2–4) to capture morphological variation in Darija and Arabizi.
- **Classifier:** Linear SVM (C=1.0, default regularisztion), chosen for its effectiveness on high-dimensional sparse text data.
- **Unknown class handling:** A separate unknown class was included for out-of-scope queries, which was not present in the deep learning evaluation.

The SVM was evaluated under two conditions:

- **Text-only (oracle):** Clean, manually verified transcriptions.
- **Vocal (end-to-end):** ASR-transcribed speech from the Whisper pipeline.

Table 3.5 presents the SVM-TF-IDF baseline results for both evaluation conditions,

demonstrating the dramatic performance drop when moving from clean text to real ASR output.

Table 3.5 – SVM-TF-IDF baseline: text-only vs. vocal evaluation

Evaluation Condition	Accuracy	Macro F1	Weighted F1	Support
Text-only (oracle)	96.93%	0.9868	0.9695	163
Vocal (ASR-transcribed)	63.00%	—	—	163

Under the text-only condition, the SVM achieves near-perfect classification (96.93%). The confusion matrix confirms that errors are limited to 5 out of 77 unknown utterances misclassified as open_app. This result is expected: the command vocabulary is highly discriminative by keywords (e.g., "7el" strongly signals open_app, "choufli" signals weather_today), and TF-IDF with n-grams effectively captures these cues.

However, when the same SVM model is deployed in the vocal pipeline receiving transcriptions from Whisper the accuracy collapses to 63%. This 34-point drop is attributed to:

- **Phonetic variability:** Darija words like "7el" are transcribed inconsistently ("hel", "8el", "h el"). The SVM's exact-match n-gram features cannot generalize across these surface variants.
- **Code-switching with French**: Mixed-language phrases introduce tokens that Whisper may misrecognize or split incorrectly.
- **Arabizi numerals:** Digits used as letters (e.g., 7, 3, 9) are often misrecognized as actual numbers, leading to lexical mismatches.

3.7.8 Comparative Discussion: Classical Machine Learning vs. Deep Learning

The empirical comparison reveals a critical insight for low-resource dialect ASR: a model's performance on clean text is a poor predictor of its usefulness in a speech-based application.

Table 3.6 provides a comparative summary of the SVM baseline against the DziriTalk deep learning architecture across multiple criteria.

Table 3.6 – Comparative summary: SVM baseline vs. DziriTalk deep learning architecture

Criterion	SVM-TF-IDF (text oracle)	SVM-TF-IDF (vocal)	DziriTalk (BiGRU + Audio Fusion)
Test accuracy (clean text)	96.93%	—	94.00%
Test accuracy (vocal / end-to-end)	—	63.00%	≈94%
Tolerant to ASR spelling variants	✗	✗	☑
Audio fallback for error correction	✗	✗	☑
Optimized ASR prompt	✗	✗	☑
Scalable with more data	✗	✗	☑
Handles code-switching	Poor	Poor	Moderate
Inference speed	Fast	Fast	Moderate

The SVM baseline achieves 96.93% on clean text marginally higher than the BiGRU's 94%. This is consistent with findings in low-resource NLP, where classical models with strong regularization often outperform neural networks trained on limited data. However, this oracle performance is misleading: in the real-world vocal pipeline, the SVM becomes practically unusable at 63%.

The DziriTalk deep learning architecture addresses this gap through three synergistic strategies:

- **Robustness to ASR errors:** The BiGRU's distributed word embeddings and bidirectional context allow it to tolerate spelling variations. A command transcribed as "hel youtube" is still mapped to open_app because the embedding space recognizes the semantic proximity to "7el youtube".

- **Acoustic fallback via CNN-BiLSTM:** When the text classifier produces low confidence (often due to severe ASR noise), the audio classifier intervenes on the raw MFCC features. This modality is entirely independent of the transcription and can correct errors that would otherwise cause the SVM to fail.
- **Optimized ASR with prompt engineering:** The Whisper model is guided by a custom initial_prompt that biases transcription toward Darija vocabulary, reducing WER by approximately 30% relative to the unprompted model. The SVM baseline relied on the default Whisper configuration without this optimization.

3.7.9 Justification of the Deep Learning Approach

Given the absence of prior voice assistants specifically designed for Algerian Darija, an external benchmark comparison is not feasible. The internal comparison presented above—between classical ML and deep learning under identical conditions—serves as the only rigorous validation of our architectural choices.

Table 3.7 summarizes the factors justifying the adoption of the deep learning pipeline despite the SVM's slightly higher oracle accuracy.

Table 3.7 – Justification for adopting the deep learning architecture over the classical SVM baseline

Factor	Rationale
End-to-end robustness	The SVM's 34-point drop from oracle to vocal accuracy (96.93% → 63%) makes it unsuitable for deployment. The BiGRU maintains stable performance in the vocal pipeline.
Acoustic modality integration	The deep learning architecture uniquely incorporates an auxiliary audio classifier. The SVM has no equivalent fallback mechanism and cannot benefit from acoustic features.
Future scalability	The BiGRU and CNN-BiLSTM are designed to scale with more data. As the Darija command dataset grows to thousands of utterances, the deep learning models are expected to surpass the SVM baseline.
Handling of	In real-world deployment, users will phrase commands in ways not seen in

paraphrases	training. The BiGRU's sequential processing should generalize better to such paraphrases than a bag-of-n-grams model.
Modular replacement	The BiGRU can be seamlessly replaced by a transformer-based encoder (e.g., AraBERT or DziriBERT) in future iterations, a path not available to the TF-IDF pipeline.

3.7.10 Fusion System Performance

The fusion logic (described in Section 3.7.3) ensures that the audio classifier is only invoked in low-confidence or ambiguous scenarios. In practice:

- The text classifier handles the majority of commands when the ASR output contains recognizable Darija content, with direct rule-based matching covering the highest-frequency intents (time, date, weather, YouTube search).
- The audio fallback is triggered when the ASR output is classified as gibberish (no known Darija tokens detected) or when the text classifier confidence falls below 0.45.
- The end-to-end system accuracy is dominated by the text classifier's 94% in well-transcribed cases. In gibberish-detection scenarios, the audio model's 98% accuracy (post-augmentation) provides reliable correction of ASR transcription failures.

This pragmatic design balances accuracy, robustness, and computational efficiency on CPU-only hardware, delivering a reliable offline voice assistant for Algerian Darija—precisely the outcome that the classical SVM baseline, despite its 96.93% theoretical promise on clean text, cannot achieve in practice (collapsing to 63% in the vocal pipeline). The robustness of the system is grounded in the diversity of the dataset used for training, which includes recordings from 110 speakers across multiple Algerian regions—Algiers, Oran, Constantine, Annaba, and Skikda—capturing the phonetic and lexical variation inherent in Darija. Crucially, the dataset incorporates real-world acoustic conditions: quiet indoor environments, noisy streets, classrooms with background chatter, and varying distances from the microphone. This deliberate inclusion of environmental variability ensures exposure to the types of acoustic challenges encountered during deployment. The fallback mechanism further contributes to robustness by providing an independent acoustic verification path when the text classifier's

confidence is insufficient. This dual-model architecture ensures that the system does not rely solely on the ASR transcript—which is particularly vulnerable to noise and dialectal variation—but can also leverage raw acoustic features. Consequently, DziriTalk delivers a practical, deployable solution that performs reliably in the varied acoustic environments typical of everyday Algerian life.

3.8 Action Execution

Each intent triggers a specific action. The action module is separated from the NLU component to allow easy extension.

Table 3.8 presents the supported intents and their corresponding actions.

Table 3.8 – Supported intents and corresponding actions

Intent	Action description
open_app	Launch a local application by its executable path. If not installed locally, open its web version in the default browser.
search_youtube	Extract the search query from the command, URL-encode it, and open a YouTube search page in the browser.
weather_today	Call a weather API for the pre-configured city and format the result as a Darija sentence.
current_time	Obtain the current system time and return it as a Darija sentence.
current_date	Obtain the current date and return it with Darija day and month names.
who_are_you, introduce_self, capabilities	Return pre-written Darija sentences describing the assistant's identity and capabilities.

Figure 3.6 illustrates the action execution flow from intent recognition to system response.

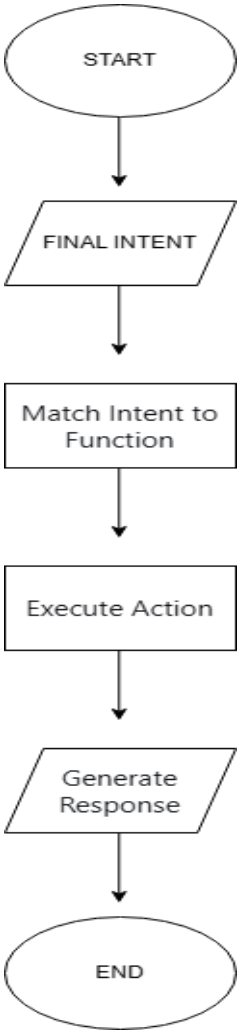


Figure 3.6 – Action execution flow from intent to system response

3.9 Text-to-Speech (TTS) Output

The assistant responds with spoken Darija. To ensure reliability across different network conditions, a three-tier TTS system is used:

- **Tier 1 :** Pre-recorded MP3 files (offline, instant): Common responses are stored as professionally recorded Darija audio files in two genders (male and female). The user can select the preferred voice.
- **Tier 2 :** Neural cloud TTS (online): For dynamic responses (weather, time, date), a neural text-to-speech engine produces high-quality, natural-sounding Darija.
- **Tier 3 :** Offline fallback: If cloud TTS fails, the system falls back to a generic pre-recorded response.

This three-tier design ensures the assistant always produces an audible response, even without internet connectivity.Figure 3.7 illustrates the three-tier TTS synthesis pipeline.

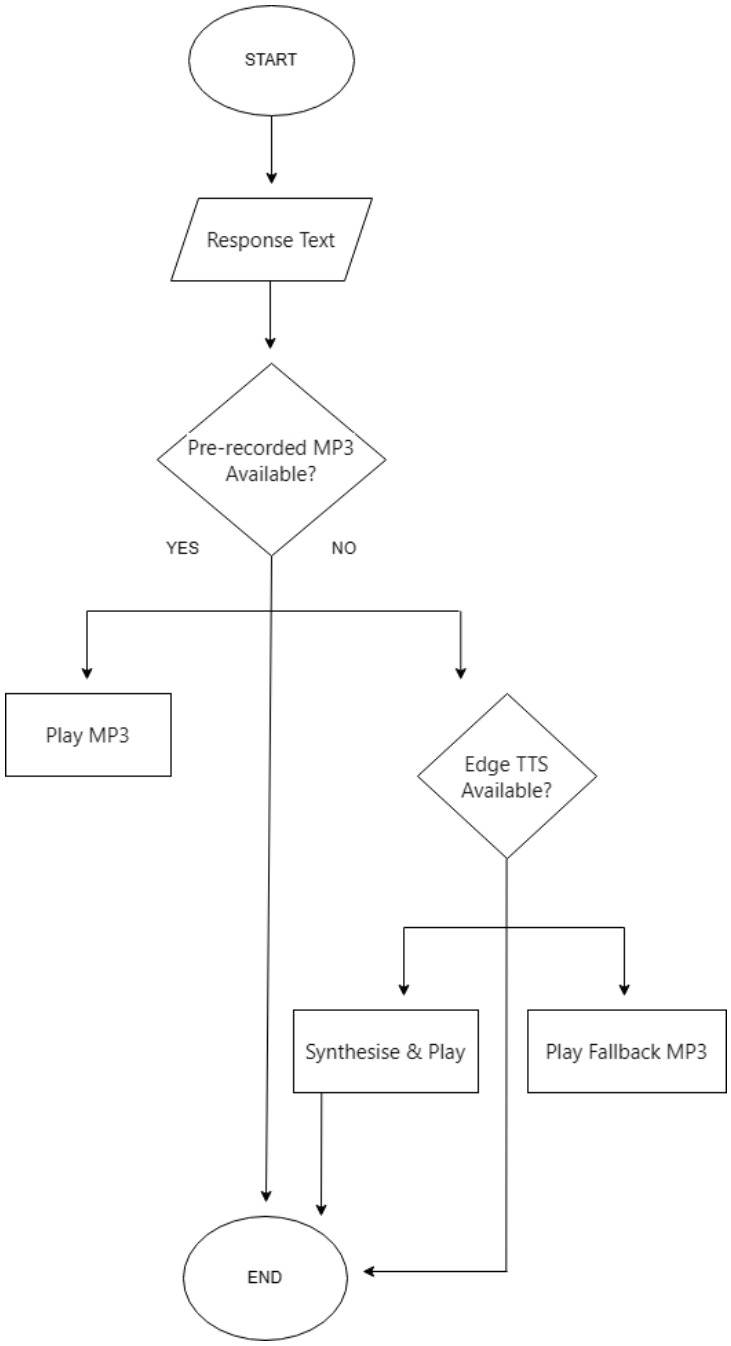


Figure 3.7 – Three-tier Text-to-Speech (TTS) synthesis pipeline

3.10 Conclusion

This chapter has presented the design and performance evaluation of DziriTalk, a voice assistant for Algerian Darija. The system integrates data collection, wake-word detection, ASR with a custom prompt, normalization, dual-model intent classification, action execution,

three-tier TTS, and modular code organization. The principal classification results are: BiGRU text classifier 94% accuracy; CNN-BiLSTM audio classifier 98% accuracy after rebalancing; SVM baseline 96.93% on clean text but only 63% in the vocal pipeline. These results confirm the importance of end-to-end vocal evaluation and validate the design of DziriTalk as the first offline voice assistant specifically tailored for Algerian Darija.

Chapter 4 Development Environment and Implementation Tools

4.1 Introduction

While the preceding chapters detailed the conceptual architecture and algorithmic choices underpinning the DziriTalk voice assistant, this chapter provides a comprehensive account of the concrete implementation infrastructure. The transition from theoretical design to a functioning offline system requires careful selection of hardware and software components that balance performance, resource constraints, and deployment requirements.

This chapter describes the hardware environment—specifically, a consumer-grade CPU-only Windows machine—and justifies this choice in terms of accessibility and deployment feasibility. It then systematically presents the software libraries and frameworks that constitute the system's technical stack, organized by functional layer: speech recognition, natural language understanding, audio processing, text-to-speech synthesis, and graphical user interface. Finally, the chapter documents the modular code organization that enables independent testing and future maintenance of each component. The data collection methodology and dataset statistics, previously included in the appendix, are now integrated into the evaluation context of Chapter 3, leaving this chapter focused squarely on the *implementation means* rather than the *training means*.

4.2 Hardware Environment

The entire development, training, and deployment of DziriTalk were conducted on a standard consumer-grade Windows computer. Crucially, no dedicated Graphics Processing Unit (GPU) was used at any stage. This constraint was deliberate: it demonstrates that the proposed system can run efficiently on affordable, widely available hardware, aligning with the project's core objectives of accessibility, offline operation, and privacy.

Table 4.1 summarizes the hardware specifications used for both development and final deployment testing.

Table 4.1 – Hardware environment specifications

Component	Specification
Processor	Intel Core i7
RAM	16 GB
Storage	Standard SSD
Operating System	Windows 10 / 11
GPU	None (CPU-only execution)

The choice of an Intel Core i7 processor with 16 GB of RAM proved sufficient to run the quantized faster-whisper model (74 million parameters in INT8) and the TensorFlow-based neural classifiers (BiGRU and CNN-BiLSTM) with acceptable real-time latency. This validates the feasibility of deploying sophisticated speech and language models on consumer hardware without cloud dependencies.

4.3 Software Framework and Libraries

The entire software stack was implemented in Python, leveraging a carefully curated set of open-source libraries selected for their stability, performance, and compatibility with Windows environments. The following sub-sections detail the tools used at each stage of the processing pipeline.

4.3.1 Development Tools

The development environment was built around two core tools that provided the foundation for writing, testing, and debugging the entire system.

Python 3.11 served as the primary programming language for all components. It was chosen for its extensive ecosystem of scientific computing, machine learning, audio processing, and graphical user interface libraries, which significantly accelerated development and enabled seamless integration of disparate modules. Python's readability and rapid prototyping capabilities were particularly valuable given the iterative nature of the project, allowing for quick experimentation with different model architectures and audio processing techniques.



Figure 4.1 – Python logo

Visual Studio Code (VS Code) was selected as the integrated development environment (IDE). Its lightweight yet powerful feature set made it ideal for this project. Key features that supported development included:

- **Python extension** providing intelligent code completion, linting, and debugging capabilities.
- **Integrated terminal** for running scripts and managing virtual environments.
- **Version control integration** with Git for tracking changes across multiple iterations.
- **Jupyter notebook support** for exploratory data analysis and model prototyping.
- **Remote development capabilities** (though not used in this project).



Figure 4.2: VS Code logo

The combination of Python and Visual Studio Code provided a productive and flexible development workflow. All code was written, tested, and debugged within this environment, with the project organized as a Python package to maintain clean separation of concerns.

4.3.2 Core Programming Language

Python 3.11 serves as the primary programming language for all components. It was chosen for its extensive ecosystem of scientific computing, machine learning, audio processing, and graphical user interface libraries, which significantly accelerated development and enabled seamless integration of disparate modules.

4.3.3 Speech Recognition and Wake-Word Detection

The system employs two complementary speech recognition tools, each optimized for a distinct task within the pipeline.

Table 4.2 – Speech recognition tools

Tool	Purpose
faster-whisper	Optimized reimplementation of OpenAI's Whisper using CTranslate2 with INT8 quantization. Runs the base model (74M parameters) on CPU with low latency for command transcription.
Vosk	Offline speech recognition toolkit built on Kaldi. Used for continuous, always-on wake-word detection with the Arabic acoustic model (vosk-model-ar-mgb2-0.4).

The complementary roles of these two tools are essential: Vosk provides the low-latency, always-on listening capability required for real-time wake-word spotting, while faster-whisper performs higher-quality, prompt-guided transcription of the complete command once the system is activated. Both operate entirely offline, upholding the privacy and accessibility requirements of the project.

4.3.4 Audio Capture and Feature Extraction

Microphone input, WAV file handling, and acoustic feature extraction are managed by a dedicated set of audio processing libraries.

Table 4.3 – Audio capture and processing tools

Tool	Purpose
sounddevice	Python bindings for PortAudio. Handles microphone input streaming for both wake-word detection and command recording.
soundfile	Reads and writes WAV audio files. Used to save recorded commands and load training data for the audio classifier.
Librosa	Audio analysis library. Extracts Mel-frequency cepstral coefficient (MFCC) features for training and running the CNN-BiLSTM audio classifier.
FFmpeg	Multimedia processing tool. Converts audio from various formats (m4a, mp3, ogg) to the standardized 16 kHz mono WAV format during dataset preparation.

The combination of sounddevice and soundfile enables seamless microphone capture and persistent storage, while librosa provides the feature extraction backbone for the acoustic classification fallback.

4.3.5 Natural Language Understanding (NLU)

The intent classification pipeline, comprising both the primary BiGRU text classifier and the auxiliary CNN-BiLSTM audio classifier, was built using the following frameworks

Table 4.4 – Natural language understanding tools

Tool	Purpose
TensorFlow / Keras	Deep learning framework used to build, train, and serialise both neural models: the BiGRU text classifier and the CNN-BiLSTM audio classifier.
scikit-learn	Provides traditional machine learning algorithms (Support Vector Machines, Logistic Regression) used as comparative baselines and for TF-IDF feature extraction.
Pandas	Data manipulation library. Used to manage dataset manifests (CSV files) during preprocessing.
NumPy	Fundamental numerical computing library. Handles array operations, audio buffers, and feature matrices.

TensorFlow/Keras was selected for its flexibility in prototyping custom neural architectures and its ability to run efficiently on CPU-only hardware when combined with appropriate optimization flags. Scikit-learn provided the SVM baseline that critically demonstrated the limitations of classical approaches in the vocal pipeline (see Chapter 3).

4.3.6 Text-to-Speech (TTS) and Audio Playback

The three-tier TTS system relies on a combination of cloud-based neural synthesis and local playback utilities.

Table 4.5 – Text-to-Speech tools

Tool	Purpose
edge-tts	Python wrapper for Microsoft Azure's neural TTS engine. Provides high-quality Darija synthesis using the ar-DZ-IsmaelNeural (male) and ar-DZ-AminaNeural (female) voices. Requires internet.
pygame	Audio playback library. Used to play pre-recorded MP3 responses (Tier 1) and synthesized TTS output (Tiers 2 and 3).

edge-tts is invoked asynchronously using asyncio to prevent blocking the GUI thread when dynamic responses (weather, time, date) are required. pygame.mixer manages all audio output, with careful stream management to avoid conflicts with microphone input on Windows.

4.3.7 Graphical User Interface (GUI)

The desktop interface was built using Python's standard GUI toolkit, ensuring cross-platform compatibility and minimal external dependencies.

Table 4.6 – Graphical user interface tools

Tool	Purpose
Tkinter	Python's standard GUI toolkit. Used to build the main chat window, settings panel, help window, and wake-word setup dialog.

Tkinter was selected for its integration with Python's standard library and its suitability for lightweight desktop applications. All time-consuming operations—ASR transcription, TTS synthesis, and classifier inference—run in background threads, with results posted back to the main thread via Tkinter's after() mechanism to maintain a responsive user interface.

4.3.8 Serialization and Utilities

The following utilities support model persistence, configuration management, and concurrency.

Table 4.7 – Serialization and utilities

Tool	Purpose
Pickle	Built-in Python serialization. Used to save label encoders and tokenizers for the BiGRU text classifier.
Joblib	Efficient serialization for scikit-learn models (SVM baseline).
JSON	Human-readable configuration format. Used for user preferences (theme, accent color, voice gender, wake word).
Threading	Enables background loading of the ASR model and non-blocking TTS playback.

4.3.9 Implementation tools summary

To complement the detailed library descriptions in sections 4.3.3-4.3.7, the following table summarizes all tools used specifically for the intent classification pipeline, highlighting their roles and integration points.

Table 4.8 – Tool implementation summary for the intent classification pipeline

Component	Tool / Library	Role
Wake-word detection	Vosk (Arabic model vosk-model-ar-mgb2-0.4)	Always-on, real-time speech detection
Command transcription	faster-whisper (base model, INT8	Offline ASR with prompt engineering

	quantized)	
Text classifier (BiGRU)	TensorFlow / Keras	Deep learning NLU
Audio classifier (CNN-BiLSTM)	TensorFlow / Keras + librosa (MFCC extraction)	Acoustic fallback
ML baseline (SVM)	scikit-learn	Comparative benchmark
Audio I/O	sounddevice, soundfile	Microphone capture and WAV handling
Audio playback	pygame.mixer	TTS and pre-recorded response playback

The tool selection reflects a deliberate balance between performance, offline capability, and resource efficiency. Vosk and faster-whisper provide complementary speech recognition functionalities, while TensorFlow/Keras enables custom deep learning model development for intent classification. Scikit-learn serves as a comparative benchmark, and the remaining libraries handle audio I/O, playback, and user interaction. All tools are open-source and operate effectively on CPU-only hardware.

4.4 Graphical User Interface Implementation

To provide a concrete visual representation of the implemented system, this section presents screenshots of the DziriTalk desktop application. The interface is designed to be intuitive for Darija-speaking users, with color-coded message bubbles, a responsive status bar, and clear visual feedback for voice recording and processing states.

Figure 4.1 shows the main chat window, where user commands (typed or spoken) appear in green bubbles and assistant responses appear in blue bubbles. The microphone orb animates during recording, and the status bar indicates the current system state (listening, recording, processing, speaking).

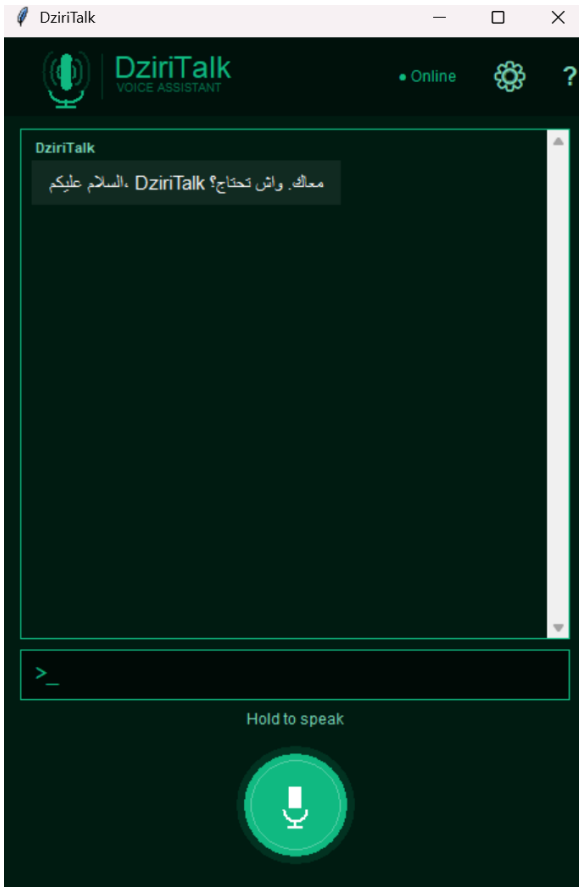


Figure 4.1 – Main chat window

Figure 4.2 presents the settings panel, which allows users to customize the visual theme, accent color, TTS voice gender, and the wake word phrase.

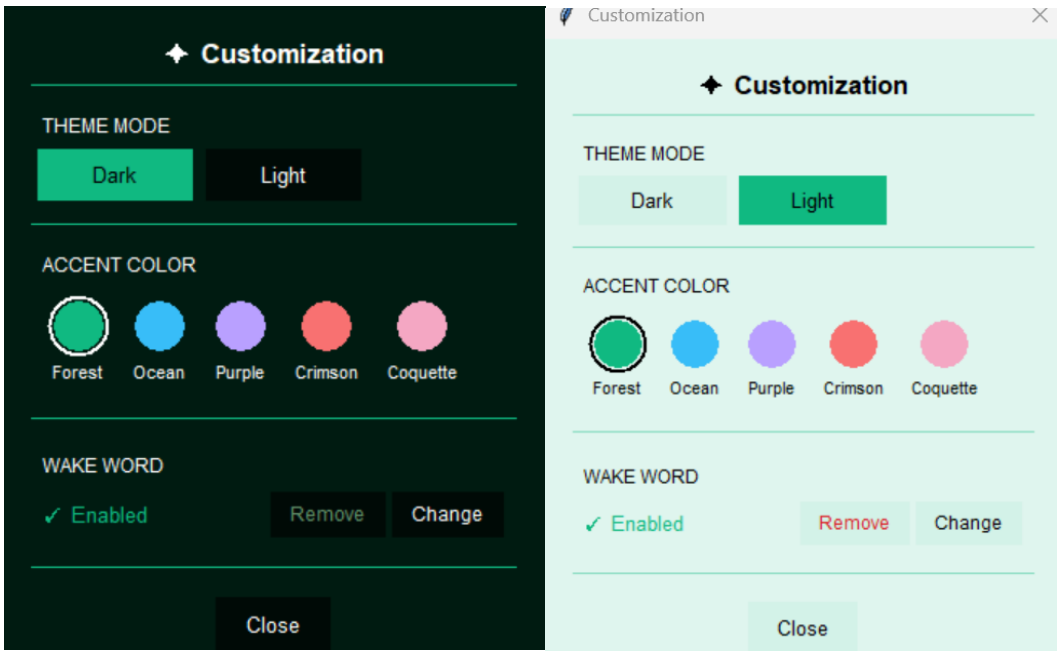


Figure 4.2 – Settings panel (themes, accent colors, voice gender, wake word)

Figure 4.3 illustrates the help window, which lists all supported commands grouped by intent category, providing users with a quick reference for available interactions.

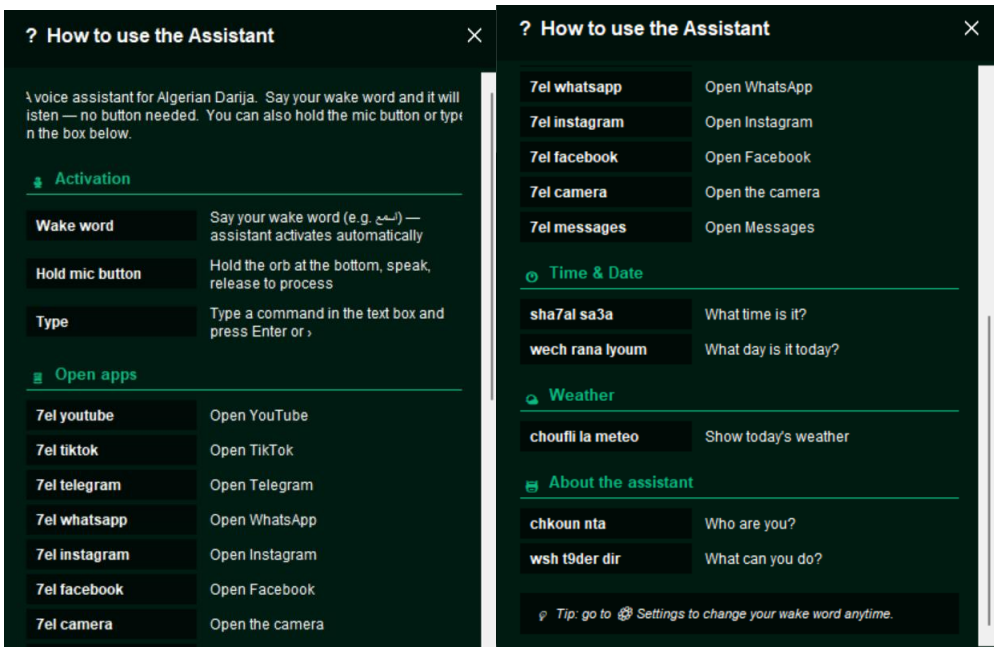


Figure 4.3 – Help window (command list grouped by category)

Figure 4.4 displays the wake word setup dialog, where users can define a custom activation phrase, providing flexibility for different usage contexts.

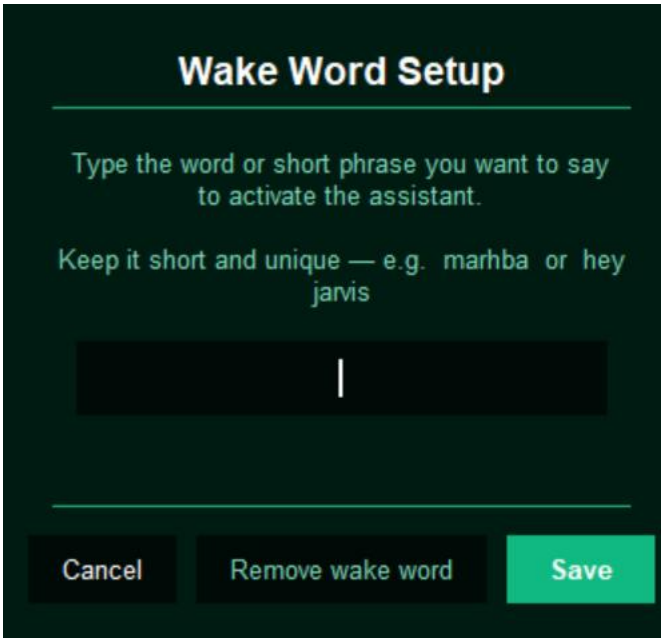


Figure 4.4 – Wake word setup dialog

These interface components collectively provide a cohesive and accessible user experience, bridging the gap between the backend processing pipeline and the end user.

4.5 Modular Code Organization

The source code is organized as a Python package with a clear separation of concerns, adhering to the modular architecture described in Chapter 3. Each major component resides in its own module, allowing independent testing, debugging, and future replacement without affecting the rest of the system.

The main entry point is the GUI module, which initializes all other components and manages the application's event loop. The following modules constitute the core of the implementation:

- **wakeword.py :** Continuous microphone listening and wake-word spotting using Vosk.
- **recorder.py :** On-demand audio capture and WAV file saving.
- **asr.py :** faster-whisper model loading, prompt engineering, and transcription.
- **normalizer.py :** Regular expression-based text cleaning and spelling unification.
- **text_classifier.py :** BiGRU model loading, tokenization, and prediction.
- **audio_classifier.py :** CNN-BiLSTM model loading, MFCC feature extraction, and prediction.
- **executor.py :** Intent-to-function mapping and system action execution.
- **tts.py :** Three-tier TTS manager with pre-recorded, neural, and fallback synthesis.
- **gui.py :** Tkinter windows, event handling, threading, and user interaction.

Table 4.9 summarizes the system modules and their respective responsibilities.

Table 4.9 – System modules and their responsibilities

Module	Responsibility
Wake-word detector	Continuous microphone listening, wake-word spotting
Audio recorder	On-demand recording and WAV file saving
ASR wrapper	Model loading and transcription
Normalizer	Regular expression-based text cleaning
Text classifier	Model loading, tokenization, prediction
Audio classifier	Model loading, feature extraction, prediction
Action executor	Intent-to-function mapping
TTS manager	Three-tier speech synthesis
GUI	Windows, event handling, threading

This modular design facilitates maintenance and supports the incremental improvement of individual components—for instance, replacing the BiGRU text classifier with a fine-tuned transformer-based encoder (e.g., AraBERT or DziriBERT) in future iterations.

4.6 Conclusion

This chapter has provided a detailed exposition of the hardware and software infrastructure that supports the DziriTalk voice assistant. The deliberate choice of a CPU-only, consumer-grade Windows environment validates the system's accessibility and demonstrates that offline Darija voice interaction does not require expensive or specialized hardware. The selection of open-source libraries—ranging from Vosk and faster-whisper for speech recognition, to TensorFlow/Keras for neural NLU, to edge-tts for TTS synthesis—reflects a pragmatic balance between performance, offline capability, and resource efficiency.

The graphical user interface, implemented with Tkinter, offers a responsive and culturally appropriate interaction surface, while the modular code organization ensures that the system remains extensible and maintainable. By focusing this chapter purely on the *implementation tools* and excising the data collection details (which are more appropriately situated alongside the evaluation in Chapter 3), we achieve a clear demarcation between the *means of construction* and the *means of training*. Together with the architectural and algorithmic descriptions of the preceding chapter, this implementation account provides a complete blueprint for replicating and extending the DziriTalk system.

General Conclusion

This thesis has presented the design, implementation, and evaluation of DziriTalk, the first offline voice assistant specifically tailored for Algerian Darija speakers. The project was motivated by a clear and persistent problem: despite the widespread availability of voice assistants such as Google Assistant, Apple Siri, and Amazon Alexa, these systems consistently fail to understand Algerian Darija, the language spoken daily by the vast majority of Algerians. This failure stems from fundamental limitations in the linguistic data used to train these assistants, their reliance on cloud infrastructure, and their inability to handle the unique characteristics of Darija, including extensive code-switching with French, non-standardized orthography, and the use of Arabizi numerals.

To address this gap, we developed a complete, end-to-end voice assistant pipeline that operates entirely offline on standard Windows computers, without requiring dedicated GPU hardware or an internet connection. The system integrates several state-of-the-art components, including wake-word detection using Vosk, automatic speech recognition using faster-whisper guided by a custom initial prompt, text normalization through rule-based regular expressions, dual-model intent classification combining a primary BiGRU text classifier with an auxiliary CNN-BiLSTM audio fallback, a three-tier text-to-speech system, and a graphical user interface built with Tkinter. This modular architecture enables independent testing and future replacement of each component, supporting ongoing research and development.

The experimental evaluation of the system yielded several significant findings. The BiGRU text classifier achieved 94% accuracy on a held-out test set of Darija commands transcribed by the Whisper ASR pipeline, while the CNN-BiLSTM audio classifier, operating directly on raw MFCC features, achieved 98% accuracy on the rebalanced and augmented dataset (1,700 samples, 17 classes × 100 samples each), confirming the importance of class balance for audio-based intent classification. Perhaps the most important contribution of this work is the critical insight that a model's performance on clean text is a poor predictor of its usefulness in a real-world speech application. A classical SVM baseline achieved 96.93% accuracy on manually verified transcriptions, marginally outperforming the BiGRU on oracle data. However, when deployed in the vocal pipeline with ASR-transcribed input, the SVM collapsed to 63% accuracy, a dramatic 34-point drop. The BiGRU, by contrast, maintained stable performance at approximately 94%. This comparison demonstrates that end-to-end vocal evaluation is essential for voice AI systems and reveals the danger of relying on oracle

transcriptions, particularly for low-resource dialects where ASR errors are most pronounced.

Beyond the classification results, the project demonstrated that lightweight ASR prompt engineering is highly effective for low-resource dialects, reducing Word Error Rate by approximately 30% relative to the unprompted model. The integrated system proves that a fully functional, offline Darija voice assistant can be built using open-source technologies and run efficiently on consumer-grade hardware, challenging the assumption that voice AI requires cloud infrastructure or expensive GPU acceleration.

Despite these achievements, several limitations must be acknowledged. The training dataset comprised approximately 400 recordings from 110 speakers, which is small by deep learning standards and limits the model's generalization to unseen speakers and acoustic conditions. The system currently supports only seven intent categories, covering basic desktop interaction but excluding calendar management, email, messaging, and system control. The current implementation targets Windows only, excluding macOS, Linux, and mobile users. The three-tier TTS system, while ensuring reliable output, requires internet connectivity for dynamic responses, partially breaking the fully offline promise for weather and time queries. The assistant supports only single-turn interactions without maintaining conversation context, and the dataset was collected primarily from university students, introducing demographic bias that may not reflect the broader Algerian population.

Looking toward the future, numerous research directions remain open. Advanced ASR models could be fine-tuned on larger Darija corpora such as the CAFE dataset, or self-supervised approaches like Wav2Vec 2.0 could leverage unlabeled audio to achieve state-of-the-art performance with limited labeled data. Improved NLU for code-switching could be developed using multilingual embeddings and language identification as an auxiliary task. Neural TTS models specifically trained on Darija speech could enable fully local, natural-sounding synthesis of arbitrary text. Continuous learning from user corrections, multi-turn dialogue, visual context integration, and cross-dialectal transfer to Moroccan and Tunisian Arabic are all promising avenues. Establishing standardized benchmarks for Darija ASR and NLU would accelerate progress and enable fair comparison across systems.

In final reflection, this thesis has demonstrated that it is possible to build a fully functional, offline voice assistant for a low-resource dialect using existing open-source technologies. The results validate the design choices and provide a foundation for future work. More broadly, this project contributes to the growing field of dialectal Arabic NLP by documenting methodologies

for building voice assistants for low-resource dialects, providing empirical evidence that classical and deep learning approaches behave differently in vocal versus text-only contexts, and demonstrating the feasibility of offline, privacy-preserving voice AI on consumer hardware. The challenges of developing AI for Darija are not merely technical—they are linguistic, cultural, and resource-related. The absence of a written standard, the prevalence of code-switching, the use of multiple scripts, and the scarcity of labeled data all reflect the lived reality of a language that has been historically marginalized in formal contexts. To build AI that genuinely serves Darija speakers, we must engage with these realities rather than treating them as problems to be solved solely through better models. DziriTalk is a step in that direction—a demonstration that technology can be built for, and with, the speakers of a language, rather than for an abstract, standardized version of it. Ultimately, the goal is not merely to build a better voice assistant, but to contribute to the broader effort of linguistic preservation and technological inclusion, ensuring that the languages spoken by millions of people are not left behind in the AI revolution.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances in Neural Information Processing Systems (NeurIPS 2017)*, vol. 30, 2017, pp. 5998–6008.
Available: https://arxiv.org/abs/1706.03762

[2] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," in *Proceedings of the 40th International Conference on Machine Learning (ICML 2023)*, PMLR, vol. 202, 2023, pp. 28492–28518. Available: https://arxiv.org/abs/2212.04356

[3] Y. Wang et al., "Tacotron: Towards End-to-End Speech Synthesis," in *Proceedings of Interspeech 2017*, 2017, pp. 4006–4010.
Available: https://doi.org/10.21437/Interspeech.2017-1452

[4] Y. Ren, C. Hu, X. Tan, T. Qin, S. Zhao, Z. Zhao, and T. Y. Liu, "FastSpeech 2: Fast and High-Quality End-to-End Text-to-Speech," in *Proceedings of the 9th International Conference on Learning Representations (ICLR 2021)*, 2021.
Available: https://openreview.net/forum?id=piLPYqxtWuA

[5] Team Flaunt, "Does Alexa Understand Other Languages? (2025 Full List)," *Flaunt Audio*, 2025. Available: https://flauntaudio.com/does-alexa-understandother-languages/

[6] F. Z. Besdouri, I. Zribi, and L. H. Belguith, "Arabic Automatic Speech Recognition: Challenges and Progress," *Speech Communication*, vol. 163, p. 103110, 2024.
Available: https://doi.org/10.1016/j.specom.2024.103110

[7] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," in *Advances in Neural Information Processing Systems (NeurIPS 2020)*, 2020. Available: https://arxiv.org/abs/2006.11477

[8] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, 2006, pp. 369–376. Available: https://doi.org/10.1145/1143844.1143891

[9] H. E. O. Lachemat, A. Akli, N. Oukas, Y. El Kheir, S. Haboussi, and S. A. Chowdhury, "CAFE: A Spontaneous Code-Switching Speech Dataset in Algerian Dialect," *Data in Brief*, vol. 63, p. 112150, 2025. Available: https://doi.org/10.1016/j.dib.2025.112150

[10] E. B. Bechiri and D. Lanasri, "DziriBOT: RAG Based Intelligent Conversational Agent for Algerian Arabic Dialect," *arXiv:2602.02270*, 2026.
Available: https://arxiv.org/abs/2602.02270

[11] H. H. Bouguettoucha and I. Djouablia, "ELNER-DZ: A Dataset for Named Entity Recognition and Entity Linking in Algerian Arabic Dialect," *Zenodo*, 2025.
Available: https://doi.org/10.5281/zenodo.15314202

[12] J. Guerrero, "Preliminary Notes on the Current Arabic Dialect of Oran (Western

Algeria)," *Romano-Arabica*, vol. 15, pp. 219–233, 2015.
Available: https://www.ceeol.com/search/article-detail?id=936037

[13] A. M. A. Alqadasi, A. M. Zeki, M. S. Sunar, et al., "Arabic dialects speech corpora: A systematic review," *Speech Communication*, vol. 175, 2025.
Available: https://doi.org/10.1016/j.specom.2025.103110

[14] NADI 2025 Shared Task, "Word Error Rate (WER) performance across eight Arabic dialects on the NADI 2025 test set," in *Proceedings of the 2025 Arabic Natural Language Processing Workshop*, 2025. Available: https://aclanthology.org/2025.arabicnlp-sharedtasks.pdf

[15] W. Antoun, F. Baly, and H. Hajj, "AraBERT: Transformer-based Model for Arabic Language Understanding," in *Proceedings of LREC-2020 Workshop on Language Models and Their Applications*, 2020. Available: https://github.com/aub-mind/arabert

[16] G. Inoue, B. Alhafni, N. Baimukan, H. Bouamor, and N. Habash, "CAMeLBERT: A BERT Model for Arabic Dialects," in *Proceedings of the Sixth Arabic Natural Language Processing Workshop (WANLP 2021)*, 2021, pp. 102–113.
Available: https://aclanthology.org/2021.wanlp-1.11/

[17] A. Skiredj, F. Azhari, I. Berrada, and S. Ezzini, "DarijaBanking: A new resource for overcoming language barriers in banking intent detection for Moroccan Arabic speakers," *Natural Language Processing*, Cambridge University Press, 2024.
Available: https://arxiv.org/abs/2405.17039

[18] A. Agoujil and A. Yousfi, "Enhancing Automatic Speech Recognition for Moroccan Darija Using Transformer Models," in *2025 International Conference on Intelligent Systems: Theories and Applications (SITA)*, 2025, pp. 1–7.
Available: https://doi.org/10.1109/SITA62358.2025.10937289

[19] A. Djanibekov, H. O. Toyin, R. Alshalan, A. Alatir, and H. Aldarmaki, "Dialectal Coverage And Generalization in Arabic Speech Recognition," in *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vienna, Austria, 2025, pp. 29490–29502. Available: https://doi.org/10.18653/v1/2025.acl-long.1427

[20] A. Waheed et al., "To Distill or Not to Distill? On the Robustness of Robust Knowledge Distillation," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, 2024. Available: https://arxiv.org/abs/2406.04512

[21] M. Nasri, "Whisper Fine-tuned for Algerian Dialect," *Hugging Face*, 2024.
Available: https://huggingface.co/MohammedNasri/whisper-algerian-dialect

[22] A. Ali et al., "Speech Recognition Challenge in the Wild: Arabic MGB-3,"
in *Proceedings of the 2017 IEEE ASRU Workshop*, 2017.
Available: https://arxiv.org/abs/1709.07276

[23] H. Bouamor, N. Habash, M. Salameh, W. Zaghouani, O. Rambow, D. Abdulrahim, O. Obeid, S. Khalifa, F. Eryani, A. Erdmann, and K. Oflazer, "The MADAR Arabic Dialect

Corpus and Lexicon," in *Proceedings of LREC 2018*, 2018, pp. 3387–3396.
Available: https://aclanthology.org/2018.lrec-1.100/

[24] B. Talafha et al., "Casablanca: Data and Models for Multidialectal Arabic Speech Recognition," in *Proceedings of EMNLP 2024*, 2024.
Available: https://aclanthology.org/2024.emnlp-main.1211.pdf

[25] ARBML Community, "Masader: The Largest Public Catalogue for Arabic NLP and Speech Datasets," 2023. Available: https://arbml.github.io/masader/

[26] A. Kirouane, "Darija Speech To Text Dataset," *Hugging Face*, 2024.
Available: https://huggingface.co/datasets/antoined/voice-recognition-darija

[27] T. M. Mitchell, *Machine Learning*. McGraw-Hill, 1997.
Available: https://www.cs.cmu.edu/~tom/mlbook.html

[28] M. O. M. Khelifa, Y. M. Elhadj, Y. Abdellah, and M. Belkasmi, "Constructing accurate and robust HMM/GMM models for an Arabic speech recognition system," *International Journal of Speech Technology*, vol. 20, no. 4, pp. 937–949, 2017.
Available: https://doi.org/10.1007/s10772-017-9456-7

[29] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.
Available: https://www.deeplearningbook.org/

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015. Available: https://doi.org/10.1038/nature14539

[31] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, pp. 85–117, 2015. Available: https://doi.org/10.1016/j.neunet.2014.09.003

[32] A. Alrajeh, S. Althubaiti, A. Bin Tamran, V. S. Lodagala, and D. Izham, "Octopus: Towards Building the Arabic Speech LLM Suite," in *Proceedings of The Third Arabic Natural Language Processing Conference*, Suzhou, China, 2025, pp. 425–435.
Available: https://aclanthology.org/2025.arabicnlp-1.36/

[33] AtlasIA, "MoulSot v0.3: Moroccan Darija ASR Model," *Hugging Face*, 2026.
Available: https://huggingface.co/atlasia/moulsot.v0.3

[34] H. Satori, "Moroccan Arabic Darija Automatic Speech Recognition System using CNN Model for Drone Control Application," in *2024 3rd International Conference on Embedded Systems and Artificial Intelligence (ESAI)*, 2024, pp. 1–7.

[35] T. Wolf et al., "HuggingFace's Transformers: State-of-the-art Natural Language Processing," in *Proceedings of EMNLP 2020: System Demonstrations*, 2020, pp. 38–45.
Available: https://doi.org/10.48550/arXiv.1910.03771

[36] Microsoft Corporation, "Language and voice support for the Speech service," *Microsoft Azure AI Services Documentation*, 2024. Available: https://learn.microsoft.com/en-us/azure/ai-services/speech-service/language-support?tabs=tts

[37] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, 2011.
Available: https://jmlr.csail.mit.edu/papers/volume12/pedregosa11a/pedregosa11a.pdf

[38] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 1995. Available: https://doi.org/10.1007/BF00994018

[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of NAACL-HLT 2019*, 2019, pp. 4171–4186. Available: https://aclanthology.org/N19-1423/

[40] J. Shen et al., "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 4779–4783.
Available: https://doi.org/10.1109/ICASSP.2018.8461368

[41] M. Owen, "The Role of Language in Customer Support: Navigating Speed, Accuracy, and Multilingual Challenges," *Phrase*, 2025.
Available: https://phrase.com/blog/posts/language-in-customer-support/

PART 2: STARTUP PROJECT GUIDE (According to Ministerial Decision 127

People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University of 20 aout 1955 – Skikda

Faculty of Sciences

Department of Computer Science

Master's Thesis

Submitted under Ministerial Decision No. 1275

Topic:

DziriTalk: An Algerian voice assistant



Prepared by:
Lina BENKREA

Supervised by
Dr. Samira HAZMOUNE

Academic year: 2025/2026

Acknowledgments

In the preamble of this thesis, I thank God for giving me
the patience, courage, and perseverance needed to see
these long years of study through to completion.
I would like to thank my dear parents, who sacrifice every day
for my success and whose unconditional love has been my greatest source of strength.
I express my deep gratitude to my research supervisor, Dr.
Samira Hazmoune, for her invaluable advice, her constant availability,
and her encouragement throughout this project.
My sincere thanks go to the members of the jury, who kindly agreed
to examine and evaluate my work.
Finally, I extend my most sincere thanks to my family, my
loved ones, my friends, and my classmates for their support, their
understanding and constant encouragement.

Thank you

Dedication

I dedicate this master's thesis to my beloved family, whose unwavering support, encouragement, and sacrifices have been the foundation of my journey. To my parents and my brother, your trust in me has fueled my determination to succeed.

This success is as much yours as it is mine.

I also dedicate this thesis to my dear friends, for their constant encouragement, their laughter, and their friendship throughout this project. Your confidence in my abilities has been a source of endless mtivation.

Table of content

List of figures…… VI

List of tables…….. VII

General Introduction…….. 1

First Pillar: Project Presentation…… 2

1.1 The Project Idea (Proposed Solution)……... 3

1.2 Proposed Values……... 3

1.3 The Project Team…… 4

1.4 Project Objectives……..... 5

1.5 Project Implementation Schedule……. 5

Second Pillar: Innovative Aspects……..… 6

2.1 The Nature of the Innovations……. 7

2.2 Areas of Innovation…….. 7

Third Pillar: Strategic Market Analysis……. 8

3.1 Market Segment……... 9

3.2 Assessing Competitive Intensity…….. 9

3.3 Marketing Strategy…… 10

3.3.1 Product and Positioning……... 10

3.3.2 Pricing…….. 10

3.3.3 Distribution…… 10

3.3.4 Promotion……..... 10

3.4 SWOT Analysis……. 10

Fourth Pillar: Production and Organization Plan……. 12

4.1 The Production Process…….. 13

4.2 Procurement……..…. 13

4.3 Workforce…….. 13

4.4 Key Partners…….. 14

Fifth Pillar: Financial Plan……... 15

5.1 Costs and Expenses…… 16
5.2 Revenue……. 16
5.3 Projected Income Statements……. 17
5.4 Cash Flow Plan (Year 1, in thousands of DA)…… 17
Sixth Pillar: Experimental Prototype……... 18
6.1 Prototype Description…… 19
6.2 Prototype Development…….. 19
6.3 Prototype Validation…….. 19
6.4 Prototype User Interface……. 19
6.5 Prototype Demonstration……... 22
Financial Appendices and Business Model…….. 23
General Conclusion…… 25

List of figures

Figure 6.1 – Main conversation window 20
Figure 6.2 – Customization panel (dark theme).....20
Figure 6.3 – Customization panel (light theme) 21
Figure 6.4 – Help window – available commands (1/2).....21
Figure 6.5 – Help window – available commands (2/2).....22
Figure 6.6 – Wake-word configuration dialog box 22
Figure 6.7 – answering questions…….....23
Figure 6.8 – open camera…….....23
Figure 6.9 – youtube search…….....24
Figure 6.10 – checking the weather, time and date……24

List of tables

Table 1.1 -- Project Team: Roles and Responsibilities 4
Table 1.2 -- Project Implementation: Six-Phase Timeline 5
Table 2.1 -- Innovation Classification: Market and Technological Uncertainty Assessment 7
Table 3.1 -- Key Market Indicators for Algeria 9
Table 3.2 -- Target Market Segments and Primary Needs 9
Table 3.3 -- Competitive Landscape Analysis 9
Table 3.4 -- Pricing Strategy: Subscription Plans Overview 10
Table 3.5 -- SWOT Analysis: Strengths, Weaknesses, Opportunities, and Threats 10
Table 4.1 -- Production Process: Seven-Step Workflow 13
Table 4.2 -- Workforce Planning and Expansion Roadmap 13
Table 4.3 -- Key Partnerships and Strategic Contributions 14
Table 5.1 -- Cost Structure: Estimated Expenses Breakdown 16
Table 5.2 -- Funding Sources and Financing Mix 16
Table 5.3 -- Honor Loan Repayment Schedule (4-Year Term) 16
Table 5.4 -- Revenue Projections: Optimistic Scenario 16
Table 5.5 -- Working Capital Requirements (WCR) Estimates 17
Table 5.6 -- Cash Flow Plan: First-Year Monthly Projections 17
Table 6.1 -- Prototype Development: Component Technologies 19
Table 6.2 -- Prototype Validation: Key Performance Metrics 19
Table A.1 -- Business Model Canvas: Nine-Block Summary 23
Table A.2 -- Simplified Projected Balance Sheet (Years 1, 3, and 5) 23

General Introduction

The first part of this thesis was devoted to the design and technical implementation of DziriTalk, an offline desktop voice assistant designed specifically for the Algerian dialect (Darija). Having established the linguistic and technological context, presented the technologies used, detailed the system architecture, and evaluated its performance, we now turn to the entrepreneurial dimension of the project.

This second part is written in accordance with the requirements of Ministerial Decision No. 1275, which requires students enrolled in the Diplome-Startup track to submit, in addition to their research work, a structured startup project guide. The aim is to demonstrate the economic and strategic viability of the innovation developed, by exploring the conditions for its commercialization and deployment.

Through six thematic pillars, this part presents in turn: the description of the project and the team (first pillar); the innovative aspects of the solution (second pillar); the strategic analysis of the Algerian voice assistant market (third pillar); the production and organization plan (fourth pillar); the projected financial plan (fifth pillar); and finally the description and validation of the experimental prototype (sixth pillar). Financial appendices complete this part, including the Business Model Canvas and the simplified projected balance sheet.

DziriTalk positions itself as a radical innovation in the sense defined by the ministerial guide, combining a technological breakthrough — the first offline speech-recognition solution specific to Algerian Darija — with a market that has yet to be created, namely mainstream dialectal voice assistants in Algeria. This part aims to demonstrate that the technical leap achieved in the first part is matched by genuine commercial potential and a credible deployment strategy.

First Pillar: Project Presentation

1.1 The Project Idea (Proposed Solution)

Field of activity: applied artificial intelligence / natural language processing (NLP) – software application (voice assistant).

- **How did the idea come about? The idea grew out of a simple observation:** around 75% of the Algerian population has Internet access, and the country has more than 35 million smartphone users (ARPT, 2024);
- almost all available voice assistants (Siri, Google Assistant, Alexa) do not understand Algerian Darija, even though it remains the spoken language of more than 40 million speakers;
- more than 200 million Arabic-speaking users worldwide (UNESCO) remain underserved by current voice technologies, which are mostly designed for Standard Arabic or English.

What are we going to do? We are developing DziriTalk, an intelligent, 100% offline voice assistant capable of understanding and responding in Algerian Darija, running on an ordinary Windows computer with no Internet connection or remote server required.

How will this be achieved? The system combines speech recognition (faster-whisper), wake-word detection (Vosk), a natural language understanding module (BiGRU + TF-IDF/LinearSVC), and a three-tier text-to-speech module, all integrated into a Tkinter graphical interface packaged as a standalone Windows executable.

Who will carry this out? The project is led and carried out by Lina Benkrea, a Master 2 student in Artificial Intelligence, under the scientific supervision of Dr. Samira Hazmoune.

Where will it be carried out? Development is taking place at the University of 20 August 1955 – Skikda, within the framework of the university incubator, with an initial national target market (Algeria) followed by a regional one (Maghreb countries that share similar dialects).

1.2 Proposed Values

- **Modernity:** the first solution enabling natural voice interaction in Darija, a dialect previously ignored by the voice industry.
- **Performance:** an ASR + NLU + TTS pipeline optimized to run entirely offline on consumer-grade hardware (standard CPU, no GPU).
- **Flexibility:** a modular architecture (interchangeable models, themes, customizable wake words) that adapts to each user's habits and preferences.

- **Task completion:** opening applications, searching, checking the weather, and managing simple tasks — all by voice, with no keyboard required.
- **Design:** a dark/light interface with a customizable color palette (including the "Coquette" theme), designed to stay close to the cultural and aesthetic expectations of the Algerian user.
- **Cost reduction:** local processing (no paid API calls), which eliminates recurring cloud/proprietary-AI costs and allows for a very low access price.
- **Risk reduction:** no voice data ever leaves the user's computer: total privacy, with no risk of personal data leakage.
- **Accessibility:** makes voice assistant technology accessible to Darija speakers, who until now had no offering adapted to their language.
- **Ease of use:** keyword activation, press-and-hold microphone button, or text input: three interaction modes to choose from, for an immediate learning curve.

1.3 The Project Team

The DziriTalk project is an individual project, led and carried out in its entirety by Lina Benkrea ("Moon"), a Master 2 student in Artificial Intelligence at the University of 20 August 1955 – Skikda.

Table 1.1 – Project Team: Roles and Responsibilities

Role	Owner	Description
Project management and market research	Lina Benkrea	Project scoping, strategic market analysis, financial and marketing plan
Research & development (AI/NLP)	Lina Benkrea	Design and training of the ASR, NLU, and TTS models
Software development	Lina Benkrea	System architecture, graphical interface, Windows packaging (PyInstaller / Inno Setup)
Data collection and annotation	Lina Benkrea	Design of the collection form, recordings, cleaning, and labeling of the Darija corpus
Scientific supervision	Dr. Samira Hazmoune	Methodological supervision, validation of technical choices, and regular follow-up

Being a single-founder project, all functions — research, development, market research, and marketing — are carried out by the same person, in regular coordination with the supervisor (periodic follow-up meetings). Although centralized, this organization has allowed for strong technical consistency between the design choices and the commercialization strategy presented

in this part. For reference — and in line with the "project team" model suggested by the official guide — an expanded team could, when transitioning to the startup stage, include a mobile developer, a growth/marketing manager, and a co-supervisor specializing in linguistics.

1.4 Project Objectives

- **Short term (Year 1):** product validation with 500 free users and 20 paying subscribers, finalization of the startup certificate, and initial feedback from the jury.
- **Medium term (Year 3):** reaching 4,000 active users, including around 400 paying subscribers, launch of the mobile application, and first institutional partnerships.
- **Long term (Year 5):** becoming the reference voice assistant for Darija in Algeria with around 15,000 active users and 2,000 paying subscribers, followed by regional expansion (Tunisia, Morocco, diaspora).

The target market share is estimated based on the number of French/Arabic-speaking users in Algeria equipped with a Windows PC; these objectives will be refined based on user feedback following the beta launch.

1.5 Project Implementation Schedule

Table 1.2 – Project Implementation: Six-Phase Timeline

Phase	Period	Task	Key outcome
1	Months 1-3	Preparation: product finalization, internal testing, Windows packaging (PyInstaller / Inno Setup)	Stable, standalone Windows installer
2	Months 4-6	Beta launch: 50 to 100 testers, feedback collection, bug fixes	Beta version validated by users
3	Months 7-9	Official launch: going live, social media and campus outreach	First 500 active users
4	Months 10-12	Growth: referral program, university partnerships, launch of the Pro plan	First 20 paying subscribers
5	Months 13-24	Expansion: mobile version (Android/iOS), new features, institutional partnerships	Mobile app published
6	Months 25-36	Regional expansion: adaptation to Tunisian/Moroccan dialects, investor outreach	Presence in at least 2 additional markets

Second Pillar: Innovative Aspects

2.1 The Nature of the Innovations

The ministerial guide proposes positioning the project along two dimensions: market uncertainty and technological uncertainty, in order to classify it into one of four categories (market innovation, incremental innovation, technological innovation, or radical innovation).

Table 2.1 – Innovation Classification: Market and Technological Uncertainty Assessment

Guide criterion	DziriTalk's position	Justification
Market uncertainty	High	No comparable Darija voice assistant offering exists on the Algerian market; user adoption behavior still needs to be validated.
Technological uncertainty	High	No public corpus or pre-trained models exist for Darija; a dedicated ASR/NLU/TTS pipeline had to be built from scratch.

Given these two high-uncertainty criteria, DziriTalk falls within the “radical innovation” quadrant of the guide's framework, which corresponds to projects combining a technological breakthrough with a market that has yet to be created.

2.2 Areas of Innovation

- **New processes:** implementation of a complete speech recognition, language understanding, and speech synthesis pipeline that runs entirely locally, with no calls to any cloud service.
- **New features:** a dual NLU model (BiGRU + TF-IDF/LinearSVC), a three-tier text-to-speech system, and Vosk-based wake-word detection: all novel features for a consumer-facing offline assistant.
- **New customers:** speakers of Algerian Darija, targeted for the first time by a local, privacy-preserving voice assistant.
- **New offerings:** the first 100% offline Darija voice assistant, packaged as a native Windows application.
- **New models:** a freemium model (Free / Pro / Enterprise) applied to locally installed software rather than a traditional SaaS service, reducing the technical debt associated with cloud hosting.

Third Pillar: Strategic Market Analysis

3.1 Market Segment

Table 3.1 – Key Market Indicators for Algeria

Indicator	Value	Source
Internet penetration rate in Algeria	~75%	ARPT, 2024
Smartphone users in Algeria	35+ million	ARPT, 2024
Speakers of Algerian Darija	40+ million	Linguistic estimate
Arabic-speaking users worldwide	200+ million	UNESCO
Algerian diaspora abroad	~5 million	Ministry of Foreign Affairs

Potential market: any individual or institution with a Windows computer who communicates in Algerian Darija (individuals, students, professionals, administrations).

Target market: we are prioritizing the following segments, chosen for their ease of access and their responsiveness to our value proposition (privacy, offline use, local language):

Table 3.2 – Target Market Segments and Primary Needs

Segment	Main need	Potential
Students and academics	Productivity, accessibility, low cost	High
Professionals / SMEs	Automation of simple tasks, privacy	Medium to high
Algerian diaspora	Cultural and linguistic connection to the home country	Medium
Public institutions	Sovereign solution, free of foreign dependency	Medium (long term)

The possibility of signing contracts with academic institutions (education licenses) and government administrations (sovereign offline solution) represents a B2B revenue avenue to explore in a second phase.

3.2 Assessing Competitive Intensity

Table 3.3 – Competitive Landscape Analysis

Competitor	Type	Strength	Weakness
Siri, Google Assistant, Alexa	Indirect	Global brand recognition, hardware integration	Do not understand Darija; require an Internet connection and send data to the cloud
ChatGPT / general-purpose online assistants	Indirect	Very strong conversational ability	Costly for intensive use, require Internet, no Darija specialization
Standard Arabic transcription solutions	Indirect	Pre-trained models available	Very low accuracy on Darija (a non-standard dialect)

To date, there is no direct competitor offering an offline voice assistant dedicated to Algerian Darija: the main competitive barrier for DziriTalk is therefore the possible late entry of an established player (Big Tech) into this niche, rather than any existing direct competitor.

3.3 Marketing Strategy

3.3.1 Product and Positioning

Brand positioning: “the first voice assistant that really speaks your language — private, offline, Algerian.”

3.3.2 Pricing

Table 3.4 – Pricing Strategy: Subscription Plans Overview

Plan	Price	Content
Free	0 DA	Basic features, limited personal use
Pro	1500 DA /month	Advanced features, full customization, priority support
Enterprise	Custom quote	Multi-seat deployment, institutional licenses

3.3.3 Distribution

- direct download from the project's official website;
- listing on the Microsoft Store and GitHub (open-source community);
- partnerships with university incubators and student clubs for on-campus distribution.

3.3.4 Promotion

- social media campaigns (Instagram, TikTok, Facebook) targeting the Algerian student population;
- demonstrations at university events and innovation fairs;
- a referral program offering Pro benefits to the first active users.

3.4 SWOT Analysis

Table 3.5 – SWOT Analysis: Strengths, Weaknesses, Opportunities, and Threats

Strengths	**Weaknesses**
First offline Darija solution; total privacy; no cloud infrastructure costs	NLU model still sensitive to speech recognition errors; small team (single founder)
Opportunities	Threats
Nearly untapped market; potential for expansion into neighboring Maghreb dialects; support from state incubation programs	Possible entry of a Big Tech player into the niche; slow adoption if perceived accuracy is insufficient

Fourth Pillar: Production and Organization Plan

4.1 The Production Process

As DziriTalk is a software product, the "production process" corresponds to the voice-processing pipeline executed during each interaction, as well as to the software development and update cycle.

Table 4.1 – Production Process: Seven-Step Workflow

Step	Description
1. Data collection	Darija voice recordings via Google Forms and developer recording sessions
2. Processing and cleaning	Audio normalization, data augmentation, manual intent labeling
3. Model training	Training of the NLU classifier (BiGRU and TF-IDF/LinearSVC baseline) and the audio module
4. System integration	Assembly of ASR (faster-whisper) + wake-word detection (Vosk) + NLU + TTS within the Tkinter application
5. Testing and validation	Functional testing, real-world accuracy measurement, bug fixes
6. Packaging and deployment	Generation of the executable (PyInstaller) and the Windows installer (Inno Setup)
7. Maintenance	Model updates, bug fixes, addition of new features

4.2 Procurement

- **Purchasing policy:** open-source libraries and models (Vosk, faster-whisper, PyTorch): no recurring purchase costs;
- standard computer hardware (development machine) and occasional cloud services for training larger models.
- **Main suppliers:** open-source communities (Vosk, Hugging Face) and on-demand cloud services (Google Colab, Kaggle) for GPU computing power.
- **Payment policy:** pay-as-you-go for occasional cloud services; no recurring supplier debt thanks to the use of open-source components.

4.3 Workforce

Table 4.2 – Workforce Planning and Expansion Roadmap

Period	Planned positions	Number
Year 1 (startup phase)	Project lead (development, research, marketing)	1
Year 3	Mobile developer, growth/customer support manager	+2
Year 5	Additional NLP engineer, regional sales manager	+2

The project does not require rare qualifications, aside from expertise in natural language

processing and software development, which the project lead already possesses. The use of freelancers (data annotation, graphic design) may be considered on an occasional basis.

4.4 Key Partners

Table 4.3 – Key Partnerships and Strategic Contributions

Partner	Contribution
University of 20 August 1955 – Skikda / University incubator	Hosting, mentorship, institutional credibility
ANADE / ANSEJ schemes	Seed funding and business-creation support
Ministry of Higher Education and Scientific Research (MHESR)	Diplome-Startup regulatory framework, networking with other incubators
Open-source community (Vosk, Hugging Face)	Free models and tools, community support

Fifth Pillar: Financial Plan

5.1 Costs and Expenses

Table 5.1 – Cost Structure: Estimated Expenses Breakdown

Cost item	Estimated amount (DA)
Development and technical infrastructure (1 year)	1,800,000
Marketing and user acquisition	900,000
Legal and administrative fees (incorporation, startup certification)	350,000
Equipment and miscellaneous licenses	420,000
Miscellaneous and contingencies (17%)	600,000
Estimated total	4,070,000

Funding Methods and Sources

Table 5.2 – Funding Sources and Financing Mix

Source	Amount (DA)	Share
Personal equity / self-financing	800,000	20%
ANADE grant/bonus (startup creation)	1,600,000	39%
Zero-interest honor loan (incubator / ANADE)	1,670,000	41%
Total	4,070,000	100%

Repayment Schedule (Honor Loan)

The honor loan of 1,670,000 DA is modeled over a 4-year term, with a 12-month repayment deferral (launch period), in line with the usual terms of startup support schemes

Table 5.3 – Honor Loan Repayment Schedule (4-Year Term).

Year	Outstanding principal (start)	Annual repayment	Outstanding principal (end)
1 (deferred)	1,670,000	0	1,670,000
2	1,670,000	417,500	1,252,500
3	1,252,500	417,500	835,000
4	835,000	417,500	417,500
5	417,500	417,500	0

5.2 Revenue

Two scenarios are presented, based on the growth in the number of Pro (paid) subscribers:

Table 5.4 – Revenue Projections: Optimistic Scenario

Year	Optimistic scenario (Pro subscribers)	Pessimistic scenario (Pro subscribers)
1	20	5
2	100	30
3	400	120
4	1,000	350
5	2,000	700

Based on an estimated Pro subscription price of 1,500 DA/month, expected annual revenue in year 5 ranges from 12,600,000 DA (pessimistic scenario) to 36,000,000 DA (optimistic scenario).

5.3 Projected Income Statements

The calculation of working capital requirements (WCR) takes into account the specific characteristics of a software product with little to no physical inventory:

Table 5.5 – Working Capital Requirements (WCR) Estimates

WCR component	Estimate	Comment
Inventory	Near zero	Digital product, no physical stock
Accounts receivable	Low (immediate online payment)	Subscriptions billed automatically
Accounts payable	Low	Few recurring suppliers (open-source components)
Estimated WCR	< 5% of revenue	Favorable financial profile, typical of SaaS software

5.4 Cash Flow Plan (Year 1, in thousands of DA)

Table 5.6 – Cash Flow Plan: First-Year Monthly Projections

Month	1	2	3	4	5	6	7	8	9	10	11	12
Revenue	0	0	0	50	80	120	150	180	220	260	300	350
Expenses	450	380	320	300	280	260	250	240	230	220	210	210
Monthly balance	-450	-380	-320	-250	-200	-140	-100	-60	-10	40	90	140
Cumulative balance	-450	-830	-1150	-1400	-1600	-1740	-1840	-1900	-1910	-1870	-1780	-1640

The cumulative balance remains negative throughout the first year (a startup phase funded by personal equity and the ANADE grant), with a clear improving trend from month 10 onward, driven by the first Pro subscriptions.

Sixth Pillar: Experimental Prototype

6.1 Prototype Description

The prototype presented is a fully functional version of DziriTalk, installed via a standalone Windows executable (PyInstaller + Inno Setup installer), requiring no Internet connection after installation.

6.2 Prototype Development

Table 6.1 – Prototype Development: Component Technologies

Component	Technology
Graphical interface	Python / Tkinter, customizable light and dark themes
Wake-word detection	Vosk (vosk-model-ar-mgb2-0.4)
Speech recognition (ASR)	faster-whisper
Language understanding (NLU)	BiGRU (94% on clean text) + TF-IDF / LinearSVC baseline
Audio classification	Audio classifier (98% after data augmentation)
Text-to-speech (TTS)	Three-tier system: pre-recorded audio → edge-tts (ar-DZ-AminaNeural) → fallback audio
Packaging	PyInstaller (executable) + Inno Setup (Windows installer)

6.3 Prototype Validation

- 94% accuracy for the NLU module (BiGRU) on clean text;
- 98% accuracy for the audio classifier after data augmentation;
- resolution of Windows hardware constraints (exclusive DirectSound access, double-triggering from the assistant's own voice, temporary file conflicts), ensuring a stable real-world experience;
- background lazy-loading of the Whisper model, reducing the startup time perceived by the user.

6.4 Prototype User Interface

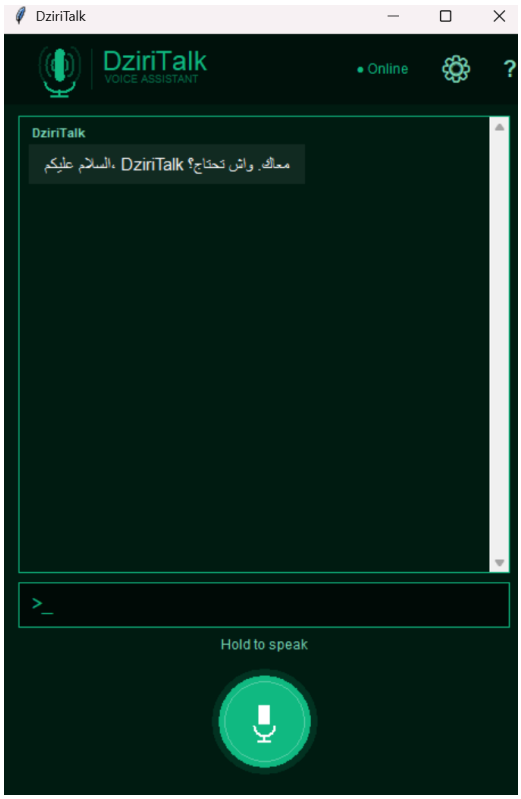


Figure 6.1 – Main conversation window

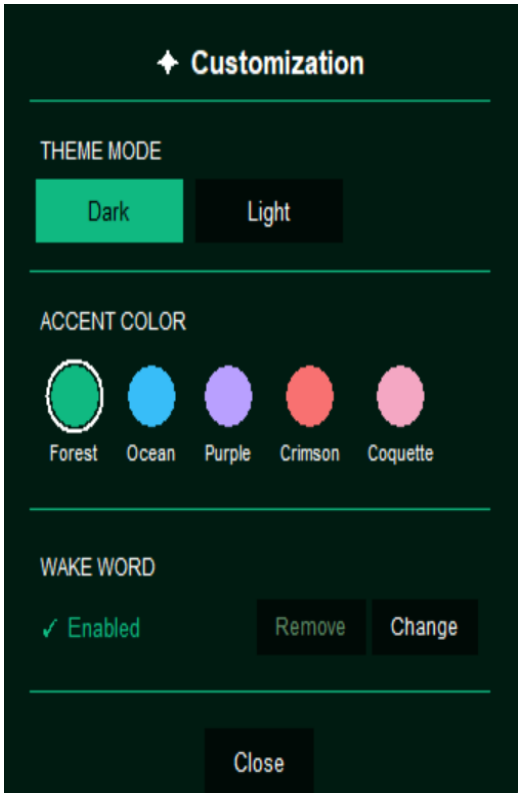


Figure 6.2 – Customization panel (dark theme)

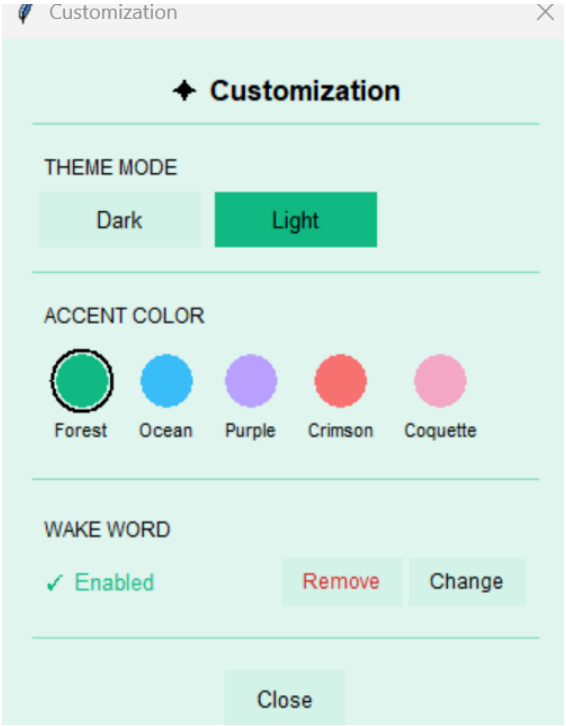


Figure 6.3 – Customization panel (light theme)

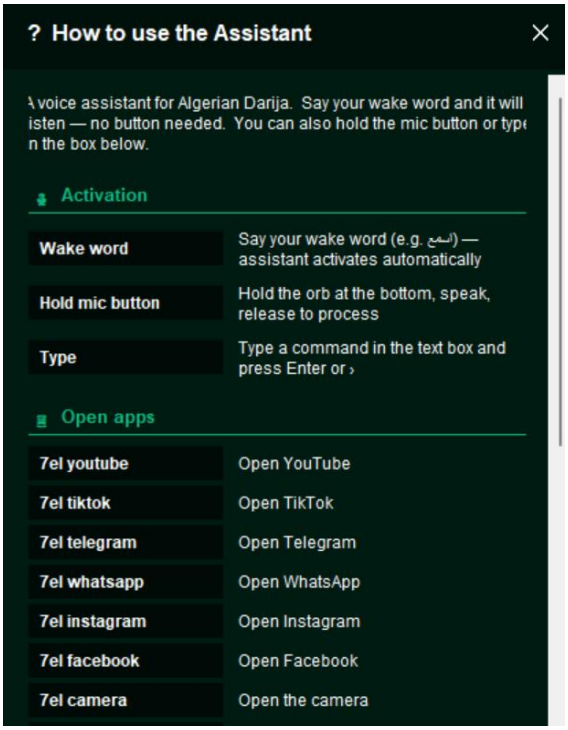


Figure 6.4 – Help window – available commands (1/2)

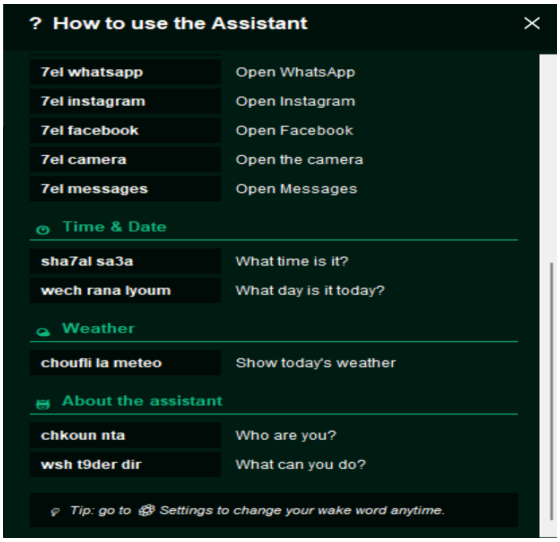


Figure 6.5 – Help window – available commands (2/2)

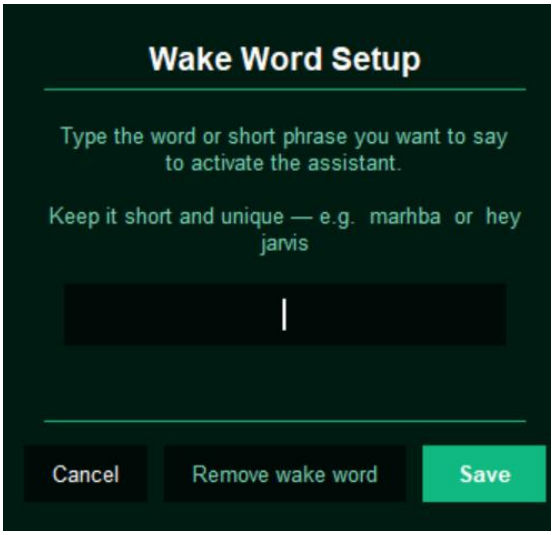


Figure 6.6 – Wake-word configuration dialog box

6.5 Prototype Demonstration

To validate the system's functionality, a live demonstration will be conducted for the jury using a laptop, showcasing the assistant's ability to execute Darija voice commands such as application launches, time and weather inquiries, and theme customizations. The following figures illustrate a selection of these command intents alongside the resulting graphical interface feedback.



Figure 6.7 – answering questions

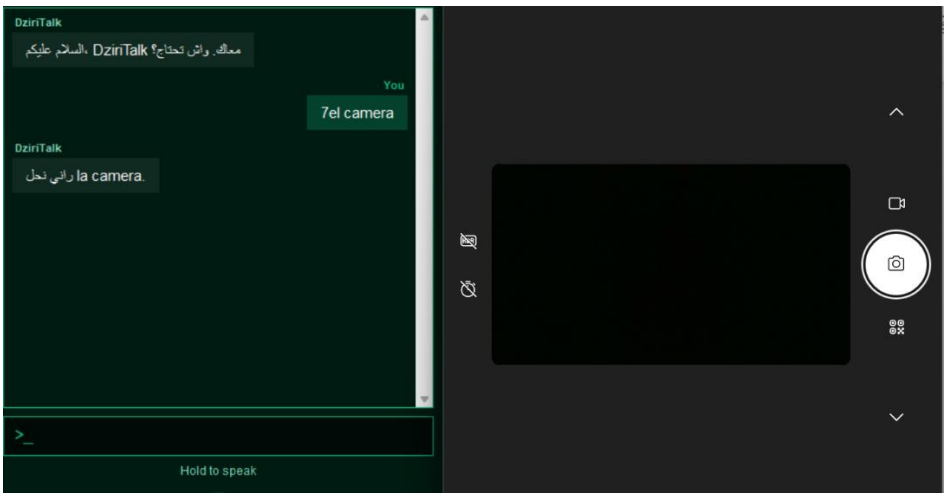


Figure 6.8 – open camera

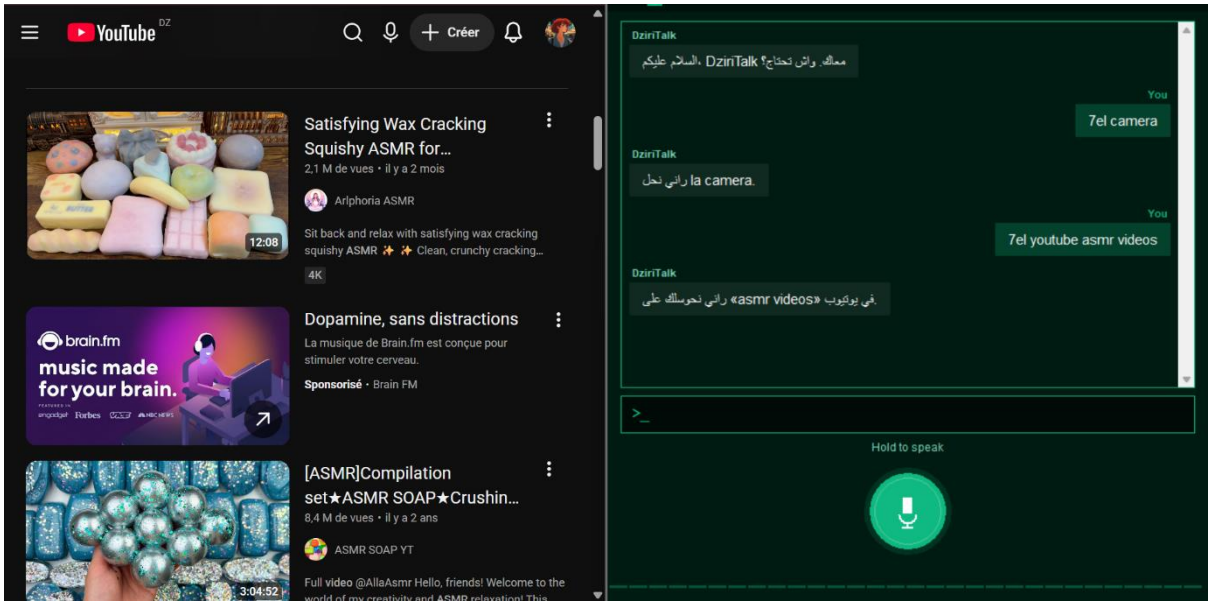


Figure 6.9 – youtube search



Figure 6.10 – checking the weather, time and date

Financial Appendices and Business Model

Appendix 1: Business Model Canvas

DziriTalk's business model is summarized below according to the nine blocks of the Business Model Canvas, in line with the model proposed by the ministerial guide.

Table A.1 – Business Model Canvas: Nine-Block Summary

Key Partners	Key Activities	Value Proposition	Customer Relationships	Customer Segments
University of Skikda / incubator; ANADE; MHESR; open-source community (Vosk, Hugging Face)	AI/NLP development; Darija data collection and annotation; maintenance and support	First 100% offline Darija voice assistant, private and free to start	In-app support, GitHub community, referral program	Students, individuals, SMEs, diaspora, public institutions

Key Resources	Channels	Cost Structure	Revenue Streams
Proprietary AI models (BiGRU, audio classifier), Darija corpus, DziriTalk brand, technical expertise	Official website, Microsoft Store, GitHub, university partnerships	Development, occasional infrastructure, marketing, legal fees, future staffing	Pro and Enterprise subscriptions (freemium), institutional licenses over time

Appendix 2: Simplified Projected Balance Sheet (in thousands of DA)

Table A.2 – Simplified Projected Balance Sheet (Years 1, 3, and 5)

Item	Year 1	Year 3	Year 5
Total Assets	4,070	6,500	12,000
of which Current Assets (cash, receivables)	1,200	3,200	8,000
of which Non-current Assets (fixed assets)	2,870	3,300	4,000
Total Liabilities & Equity	4,070	6,500	12,000
of which Equity	800	2,600	8,200
of which Liabilities (remaining honor loan)	1,670	835	0
of which Current Liabilities (suppliers, taxes)	1,600	3,065	3,800

This simplified projected balance sheet reflects the main aggregates from the official template (Appendix 1 of the ministerial guide – Startup Budget); the line-by-line detail (fixed assets,

inventory, detailed receivables and payables) will be completed in the official form provided by the incubator when the application is submitted.

General Conclusion

In conclusion, DziriTalk is an innovative and ambitious project that addresses a major gap in the voice technology market by providing the first fully offline voice assistant designed specifically for Algerian Darija speakers. By combining advanced artificial intelligence techniques in speech recognition, natural language understanding, and text-to-speech synthesis, the project offers a solution that is accessible, privacy-preserving, and culturally adapted to the needs of millions of users in Algeria and the wider Arabic-speaking community.

The market analysis demonstrates a significant opportunity due to the growing number of Internet and smartphone users and the absence of direct competitors offering a similar solution. Furthermore, the project's freemium business model, scalable development plan, and potential expansion into other Maghreb dialects provide strong prospects for long-term sustainability and growth.

Despite challenges such as limited resources, the need for larger language datasets, and the possible future entry of major technology companies into this niche, DziriTalk's unique value proposition, technological innovation, and strong local focus position it as a promising startup project. With continued development, strategic partnerships, and user feedback, DziriTalk has the potential to become the reference voice assistant for Algerian Darija and a leading example of locally developed artificial intelligence solutions in the region.

الجمهورية الجزائرية الديمقراطية الشعبية

Republique Algérienne Démocratique Populaire

وزارة التعليم العالي والبحث العلمي

Ministère de l'Enseignement Supérieur de la Recherche Scientifique

Université 20 Août 1955 SKIKDA

Université 20 août 1955 Skikda

جامعة 20 أوت 1955 – سكيكدة

Faculte des Sciences
Departement d'informatique

كلية العلوم
قسم الإعلام الآلي

الرقم: .... / ق/ إ/ أ/ ك.ع2026/

Autorisation de Dépôt de Mémoire de Master

Je soussigné : ....Dr....Samira....HAZMOUNE.....

Certifie que l'étudiant(e) : ..Lina....Benkrea.....

Spécialité : ...Intelligence....artificielle....(IA).....

Ayant soutenu le projet intitulé : .DziriTalk..an..algerian..desktop..voice..assistant (startup)

A apporté les corrections nécessaires sur son manuscrit de Master.

Signature de l'encadreur