A Comprehensive Survey of Automatic Speech Recognition Systems: Techniques, Models, and Applications

Eman S. Sabry[1], A. A. Ahmed[1, *], J. A. Ramzy[1], B. R. Amin[1], B. W. Adel[1], and A. O. Khairy[1]

ABSTRACT

Abstract— Automatic Speech Recognition (ASR) systems have become indispensable in enabling natural and intuitive human–machine interaction. This paper presents a comprehensive survey of ASR technologies, tracing their development from traditional statistical models to modern deep learning and transformer-based architecture. Key components—including acoustic models, language models, lexicons, feature extraction techniques, and decoding algorithms—are systematically reviewed to provide a structured overview of the field. Building on this foundation, the paper then introduces a series of original experiments aimed at evaluating ASR performance in constrained, domain-specific environments. A scenario-based framework was implemented and tested across three successive iterations: (1) Kaldi-based custom acoustic modeling, (2) a hybrid Vosk–Whisper architecture, and (3) a syllable-level phonetic aliasing approach tailored for medication term recognition. Vosk was ultimately selected as the primary ASR engine due to its offline operation, low latency, and compatibility with embedded platforms. Experimental evaluations examined the trade-offs between recognition accuracy and execution speed across different Vosk configurations. To further enhance real-time performance, a Limited Vocabulary Recognition (LVR) strategy was adopted. Tests under varied noise conditions revealed that while LVR improves efficiency, it can reduce accuracy for complex or unfamiliar terms. A phonetic correction mechanism was therefore introduced, resulting in substantial improvements in domain-specific recognition accuracy. The findings highlight the importance of balancing accuracy, responsiveness, and scalability in ASR system design, and demonstrate how targeted adaptations can significantly improve performance in specialized, noise-prone environments.

Keywords: ASR, Transformer Models, WER, Speech-to-Text, Domain-Specific Vocabulary, Hybrid ASR Systems, Noise Robustness, Speech Command Mapping.

I. INTRODUCTION

A. Overview

Speech is the most natural and efficient mode of human communication. As voice interfaces become increasingly prevalent in smartphones, smart speakers, and virtual assistants, ASR has emerged as a cornerstone technology for enabling seamless human-computer interaction. The primary goal of ASR is to automatically convert spoken language into written text, facilitating applications such as real-time transcription, hands-free control, and multilingual translation.

Over the past few decades, ASR systems have evolved dramatically. From early rule-based and statistical methods to today's deep learning-powered architectures, each wave of innovation has contributed to improved recognition accuracy, robustness, and generalization across languages and environments [1], [2]. This paper provides a comprehensive survey of the major milestones and current advancements in ASR technologies.

B. Core Components of ASR Systems

A typical ASR system comprises several interdependent components, each playing a vital role in translating raw audio input into textual output:

[1] Department of Communications and Computers Engineering, Higher Institute of Engineering, El-Shorouk Academy, El-Shorouk City, Egypt.

* Corresponding Author

- Acoustic Model: This model captures the statistical relationships between audio signals and phonetic units. It enables the recognition of varying speech patterns and pronunciations by learning the mapping from acoustic features to phonemes [3].
- Language Model: To improve the grammatical coherence of the output, the language model estimates the probability of word sequences based on syntactic and semantic constraints, enhancing the system's ability to choose the most plausible transcriptions [4].
- Lexicon: The lexicon serves as a lookup table that maps words to their phonetic transcriptions, effectively bridging the acoustic and language models [5].
- Feature Extraction: This step involves converting raw audio waveforms into more compact and informative representations. Techniques such as Mel-Frequency Cepstral Coefficients (MFCCs) and Log-Mel spectrograms are commonly used for capturing perceptually relevant features of speech [6].
- Decoder: Finally, the decoder integrates information from the acoustic model, language model, and lexicon to generate the most likely word sequence. Algorithms like the Viterbi decoder are often employed for efficient search and alignment [7].

C. Evolution of ASR Architectures

1) Statistical Methods: Early ASR systems were largely built upon Hidden Markov Models (HMMs) coupled with Gaussian Mixture Models (GMMs) to model the temporal dynamics and feature distributions of speech [8]. While foundational, these models made strong independence assumptions and struggled to capture long-range dependencies, limiting their flexibility and accuracy.
2) Deep Learning Integration: The introduction of Deep Neural Networks (DNNs) marked a turning point in ASR research. DNNs provided a more expressive means of modeling the complex, non-linear relationships between acoustic inputs and phonetic outputs, outperforming traditional GMM-HMM systems in many benchmarks [9]. Subsequent innovations included Convolutional Neural Networks (CNNs), which enhanced spatial feature modeling, and Recurrent Neural Networks (RNNs)—especially Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) variants—which significantly improved the system's capacity to model sequential data [10], [11].
3) End-to-End Models: More recently, end-to-end ASR architectures have gained popularity for their ability to simplify the training pipeline and directly map audio signals to text. Methods such as Connectionist Temporal Classification (CTC) [12], attention-based encoder-decoder frameworks [13], and Recurrent Neural Network Transducers (RNN-T) [14] have proven effective across diverse languages and domains.
4) Transformer-Based and Self-Supervised Models: The advent of Transformer architecture has revolutionized many areas of machine learning, including ASR. Notable models include Wav2Vec 2.0 by Facebook AI [15], which combines self-supervised pretraining on large audio corpora with fine-tuning for downstream ASR tasks. Similarly, Whisper by OpenAI [16] introduces a robust framework trained on multilingual and multitask audio datasets, achieving state-of-the-art performance across various noise conditions and low-resource languages. These models leverage self-attention mechanisms to capture global contextual dependencies in audio streams, offering remarkable improvements in both accuracy and generalization.

D. List of Contributions

The key contributions of this work can be summarized as follows:

1. Comprehensive Survey of ASR Technologies: A structured review of Automatic Speech Recognition (ASR) systems was conducted, outlining their evolution from traditional statistical methods (e.g., HMM-GMM) to advanced neural and transformer-based architectures. Core system components, including acoustic and language models, lexicons, feature extraction, and decoding algorithms—were systematically analyzed to establish a solid theoretical foundation.
2. Development of a Scenario-Oriented Evaluation Framework: A three-phase, scenario-driven experimental methodology was designed to iteratively refine ASR performance for medical-domain

applications. Each phase targeted a distinct strategy under unique technical constraints, enabling a progressive optimization of accuracy, efficiency, and robustness.

3. Custom Kaldi-Based Acoustic Modeling for Domain-Specific Vocabulary: A dedicated acoustic model was trained using the Kaldi toolkit on curated audio datasets of pharmaceutical terms. This phase demonstrated the feasibility of custom modeling but also highlighted practical challenges related to complexity, required expertise, and deployment overhead, motivating a shift toward more lightweight solutions.
4. Design of a Hybrid Vosk–Whisper Dual ASR Pipeline: A novel hybrid pipeline was implemented in which Vosk handled low-latency keyword detection, while Whisper provided fallback full-segment transcription. Fuzzy string matching enhanced recognition of drug names. While this approach improved accuracy, the processing latency of Whisper revealed limitations for real-time robotic interaction.
5. Implementation of a Syllabic Alias-Based Recognition Method: A new phonetic aliasing technique was integrated into Vosk, using syllable-level substitutions for domain-specific medication names (e.g., "a doll" for Adol). This allowed for improved recognition of mispronounced or non-standard terms without retraining the acoustic model, while maintaining fully offline operation.
6. Optimization of ASR for Embedded Real-Time Applications: Multiple Vosk configurations were benchmarked for accuracy, execution time, and memory usage. The "0.15" model was ultimately selected for its balance of speed and accuracy, making it suitable for resource-constrained platforms such as Raspberry Pi and ESP32.
7. Integration of Limited Vocabulary Recognition (LVR): An LVR strategy was applied to focus recognition on a fixed set of key commands, reducing latency and improving robustness under noisy conditions. This technique proved especially beneficial for robotic control tasks requiring rapid response.
8. Phonetic Error Correction via Domain-Specific Mapping: A two-step correction mechanism was introduced—phonetic normalization followed by post-recognition validation against a controlled vocabulary—which significantly improved recognition accuracy for specialized drug names that were not part of the original training vocabulary.
9. Quantitative Evaluation with Statistical Rigor: Detailed experiments assessed the impact of LVR and noise on recognition accuracy for three medication terms (Adol, Brufen, Catafast). Accuracy improvements (e.g., from 0.00 to 0.854 for Adol) were supported by proposed 95% confidence intervals and significance testing (e.g., McNemar's or chi-square tests), enhancing the robustness and credibility of the reported findings.
10. Practical Design Guidelines for Constrained Environments: Insights were derived regarding trade-offs between vocabulary size, accuracy, latency, and noise robustness. These guidelines inform future ASR deployments in embedded, real-time medical or robotic systems where both responsiveness and domain precision are critical.

E. Paper Organization

This paper is structured to provide both a comprehensive survey of Automatic Speech Recognition (ASR) advancements and a scenario-driven evaluation of practical ASR implementations. Section II – Related Work: Presents a critical review of the existing literature, tracing the evolution of ASR technologies from traditional statistical models to contemporary deep learning and transformer-based approaches. Key research contributions are highlighted, and existing gaps in scalability, robustness, and domain adaptation are identified. Section III – Phonetic Alias Generation Approaches: Examines methods for creating phonetic aliases, contrasting manual transcription techniques with scalable automation strategies, and discusses their roles in handling pronunciation variability in domain-specific contexts. Section IV – Vosk: Describes the architecture, features, and advantages of the Vosk speech recognition toolkit, with a focus on its offline capabilities, lightweight deployment, and suitability for embedded systems. Section V – Whisper: Explores the Whisper ASR model developed by OpenAI, outlining its encoder–decoder transformer

architecture, multilingual capabilities, and performance under noisy conditions. Section VI – Problems and Challenges in Speech Recognition Systems: Discusses common technical and practical challenges in ASR, including noise robustness, limited vocabulary recognition, latency constraints, and difficulties in recognizing specialized terminology. Section VII – Performance Metrics for Speech Recognition Systems: Defines and explains key evaluation metrics—such as Word Error Rate (WER), accuracy, latency, and model footprint—that are essential for assessing system performance across different scenarios. Section VIII – Scenario-Oriented Evaluation of ASR: Details a three-phase experimental framework comprising: (1) custom acoustic model training with Kaldi, (2) a hybrid Vosk–Whisper architecture, and (3) a syllabic alias-based recognition approach. Each phase targets distinct objectives and reveals critical design trade-offs. Section IX – Performance Analysis and Evaluation of ASR Accuracy: Provides a quantitative comparison of different Vosk configurations, evaluates the impact of Limited Vocabulary Recognition (LVR), and presents phonetic correction mechanisms to enhance the recognition of domain-specific medical terms. Section X – Conclusion: Summarizes the key insights and experimental findings, reflects on observed trade-offs between accuracy, latency, and scalability, and outlines directions for future research toward more robust, adaptive ASR systems.

II. RELATED WORK

A. Literature Review

Several comparative survey studies have been conducted to evaluate state-of-the-art speech recognition models and their diverse applications as follows.

In [17], the authors provide a comprehensive overview of deep learning techniques applied to ASR. The study details conventional and advanced architectures, including convolutional neural networks (CNNs), recurrent neural networks (RNNs), attention mechanisms, and Transformers, while also addressing integration with language models and toolkits.

A detailed survey presented in [18] classifies end-to-end speech recognition approaches into three major categories: Connectionist Temporal Classification (CTC), Recurrent Neural Network Transducer (RNN-T), and Attention-based Encoder–Decoder (AED) models. This work highlights comparative trade-offs among the models in terms of latency, accuracy, and streaming capabilities. In [19], a large-scale empirical comparison was performed among RNN-T, RNN-AED, and Transformer-AED models on a 65,000-hour dataset. The results indicate that Transformer-AED offers superior accuracy across both streaming and non-streaming scenarios, while RNN-T remains optimal for real-time applications. The work in [20] investigates the performance of Transformer-based models against traditional RNN architectures across fifteen speech tasks, including automatic speech recognition, speech translation, and text-to-speech. The study concludes that Transformer models outperform RNNs in 13 out of 15 tasks, demonstrating improved generalization and efficiency.

A focused comparative analysis of Transformer variants is presented in [21], where the Emformer—a streaming-optimized Transformer model—is evaluated against LSTM and Conformer architectures. Results show that Emformer achieves a relative word error rate (WER) reduction of 24–26% in low-latency environments, establishing its effectiveness for voice assistant applications.

The study in [22] offers a comparative analysis of classical and modern ASR techniques, including Mel-Frequency Cepstral Coefficients (MFCC), Dynamic Time Warping (DTW), Hidden Markov Models (HMM), and various machine learning classifiers. The paper evaluates the relative performance of these methods across different acoustic environments and multilingual setups.

A comprehensive review in [23] examines the progression of ASR systems by contrasting statistical models with deep learning–based frameworks such as CNNs, RNNs, and attention mechanisms. The authors provide an in-depth evaluation using metrics such as Word Error Rate (WER), Character Error Rate (CER), F1-score, and system latency.

In [24], a domain-specific comparative study is presented on Arabic speech recognition using various convolutional neural network (CNN) architectures, including AlexNet, ResNet, and GoogLeNet. Experimental results

on Arabic digit datasets reveal that GoogLeNet achieves the highest classification accuracy of approximately 89.6%, thereby demonstrating the suitability of deep CNNs for low-resource languages.

Insights from practitioner communities, such as those on Reddit, provide valuable anecdotal validation for academic findings. In [25], users report that Transformer-based Attention Encoder–Decoder (AED) models demonstrate superior recognition accuracy in both streaming and non-streaming scenarios when compared to Recurrent Neural Network Transducers (RNN-T). This aligns with empirical results from large-scale studies indicating the robustness and effectiveness of Transformer architecture.

Additionally, as noted in [26], the Emformer—a streaming-optimized Transformer variant—achieves a relative word error rate (WER) reduction of 24–26% when compared to long short-term memory (LSTM) models in low-latency voice assistant applications. These observations underscore the growing preference for Transformer-based acoustic models in both academic research and industrial deployment.

TABLE 2 presents a structured comparison of five influential surveys and benchmark studies on ASR. It highlights each paper's problem statement, challenges addressed, key extracted figures and tables, and principal findings. The comparison facilitates understanding of trade-offs among end-to-end architectures, performance benchmarks, and design motivations for real-time ASR deployment.

Across the reviewed literature, several common themes emerge. First, there exists a consistent emphasis on achieving an optimal balance between recognition accuracy, system latency, and computational efficiency, particularly for real-time and streaming applications. While Transformer-based models demonstrate superior accuracy, their high resource demands pose challenges for deployment in low-latency environments unless adapted through architectural innovations such as Emformer or Conformer. Second, a major challenge identified across studies is the lack of standardized evaluation protocols, particularly for large-scale, multilingual, and streaming ASR systems. This inconsistency complicates direct comparison of model performance and generalizability. Finally, despite recent advancements, issues related to data scarcity, especially in low-resource languages, continue to hinder the development of robust, globally inclusive ASR systems.

B. Detailed Comparative Analysis of Speech Recognition Models

The evolution of speech recognition models reflects a continual trade-off between model complexity, computational efficiency, recognition accuracy, and real-time applicability. Each model category—traditional, hybrid, deep learning-based, and end-to-end—offers unique advantages and limitations depending on the application domain and resource availability, as shown in TABLE 3.

Traditional models, primarily based on Hidden Markov Models (HMMs) combined with Gaussian Mixture Models (GMMs), served as the foundation of early ASR systems. These models performed explicit sequence modeling and alignment using algorithms such as Viterbi decoding and were favored for their mathematical interpretability and well-established training frameworks [3], [8]. However, their assumptions of observation independence and limited modeling capacity restricted their ability to capture the complex, hierarchical nature of spoken language, particularly under variable acoustic conditions [1].

To overcome these limitations, hybrid HMM-DNN architectures were introduced, wherein deep neural networks (DNNs) replaced GMMs as acoustic models while retaining the temporal alignment and decoding mechanisms of HMMs. This significantly improved frame-level classification accuracy due to the discriminative nature of DNNs and their capacity to model non-linear feature transformations [9]. Nevertheless, hybrid systems maintained a modular structure that required separate training phases for feature extraction, alignment, and decoding, making the pipeline complex and sensitive to domain-specific tuning [2].

The advent of deep learning-based ASR models, including architectures such as CNNs, LSTMs, and GRUs, allowed for more integrated modeling of acoustic and temporal patterns. Recurrent networks in particular were effective in learning long-range dependencies in speech sequences, addressing the shortcomings of HMMs in sequential modeling [27], [28]. These models also reduced the need for manual feature engineering by learning representations directly from raw or minimally processed audio inputs.

A further shift occurred with the development of end-to-end speech recognition models, which eliminated the need for explicit intermediate representations like phonemes or frame-level alignments. Models such as Connectionist

Temporal Classification (CTC) [12], attention-based encoder-decoders [13], and Recurrent Neural Network Transducers (RNN-T) [14] enabled direct mapping from input audio to text, simplifying the ASR pipeline while maintaining or even improving accuracy. However, these architectures typically required large amounts of labeled data and computational resources for training, and the internal alignment between audio and transcription often lacked interpretability compared to HMM-based systems.

The most recent advancements have leveraged Transformer-based architectures, including self-supervised models such as Wav2Vec 2.0 [15] and multitask models like Whisper [16]. These models use attention mechanisms to model global dependencies in speech signals and are pre-trained on massive amounts of unlabeled audio, enabling fine-tuning on relatively small, labeled datasets. As shown in benchmark comparisons, these systems achieve state-of-the-art performance on a wide range of tasks, including transcription, translation, and multilingual speech recognition, while offering improved robustness to accents, background noise, and domain shifts [11], [12].

In summary, while traditional and hybrid models remain relevant in resource-constrained or embedded settings due to their lower computational footprint, modern Transformer-based architectures provide superior performance in large-scale and multilingual applications. The choice of model should thus be guided by application-specific constraints such as real-time requirements, data availability, language support, and deployment environment.

The performance of speech recognition models is commonly assessed using the Word Error Rate (WER) on standard benchmark datasets such as LibriSpeech, TED-LIUM, and Common Voice. These datasets vary in vocabulary size, speaker diversity, and acoustic complexity, providing a comprehensive basis for evaluating model robustness and generalization.

As shown in TABLE 4, traditional HMM-GMM systems yield the highest WERs, with values exceeding 20% on noisy datasets such as LibriSpeech test-other and Common Voice. These models, while computationally efficient, lack the representational capacity to model non-linear acoustic patterns or long-range dependencies in speech [3], [8]. Hybrid models, particularly those combining HMMs with DNNs or RNNs, significantly reduce WER—achieving approximately 8.6% on LibriSpeech test-clean. This improvement is attributed to the use of deep networks for acoustic modeling while retaining the temporal alignment capabilities of HMMs [9].

End-to-end models, including DeepSpeech (CTC-based), LAS (attention-based), and RNN-T (streamable), demonstrate further reductions in WER by eliminating intermediate modeling steps and learning direct mappings from audio to text [27], [13], [14]. The RNN-T model, for example, achieves a WER of 4.3% on LibriSpeech test-clean, making it suitable for real-time and mobile ASR applications where low latency and compact models are required.

The Transformer-based models, such as Wav2Vec 2.0 and Whisper, currently represent the state-of-the-art in ASR performance. Wav2Vec 2.0, trained with self-supervised learning on raw audio, achieves WERs as low as 1.9% on LibriSpeech test-clean and 3.3% on test-other [15]. These results highlight the benefits of large-scale unsupervised pretraining and fine-tuning on downstream tasks. Similarly, Whisper, trained on over 680,000 hours of multilingual and multitask audio data, exhibits robust performance across various domains and languages, attaining WERs below 6% even in challenging conditions [16].

Notably, while Transformer-based models provide the best accuracy, they also incur higher computational costs and may be less suitable for deployment on resource-constrained devices. Conversely, traditional and hybrid models remain viable options in embedded or real-time scenarios, despite their lower accuracy. Ultimately, the choice of ASR model should be informed by the target application's constraints on accuracy, latency, scalability, and resource availability. The WER metrics across benchmarks serve as a guiding reference for selecting models aligned with specific deployment needs.

TABLE 1: COMPARATIVE ANALYSIS OF BENCHMARK SURVEY PAPERS ON SPEECH RECOGNITION MODELS

Reference	Problem Statement	Challenges Addressed	Key Extracted Figures / Tables	Main Findings / Contributions
[17]	To provide a comprehensive review of deep learning approaches for ASR and how they advance traditional systems.	Integration of DL models with traditional ASR components Low-resource and noisy environments Selection of optimal architectures Lack of general-purpose models	TABLE 1: Comparison of E2E architectures (CTC, RNN-T, AED) TABLE 3: WER benchmarks (LibriSpeech, TED-LIUM) Figure 4: Block diagram of ASR pipeline	Synthesizes CNN, RNN, Transformer models; highlights toolkit availability (Kaldi, ESPnet, Wav2Vec); outlines ASR challenges in real-world settings.
[18]	To review and classify end-to-end speech recognition models and compare them with traditional hybrid approaches.	Streaming vs. non-streaming trade-offs Lack of alignment between input and output sequences Training instability Decoding and latency issues	Figure 2: Architectures (RNN-T vs. Attention-based AED) Table 2: WER comparison across models TABLE 3: Latency-performance trade-offs	E2E models simplify training but introduce decoding and alignment issues; AED gives best accuracy, RNN-T ideal for streaming; future trend: unified modeling.
[19]	To empirically compare performance of popular E2E ASR models on a large-scale dataset.	Generalization at scale (65k-hour data) Streaming-friendly design Efficiency vs. accuracy Optimization stability	TABLE 3: WER comparison of RNN-T, RNN-AED, Transformer-AED across test sets	Transformer-AED achieves lowest WER; RNN-T best for streaming; training initialization critical for RNNs; first study at this scale.
[20]	To benchmark RNN vs Transformer across speech tasks (ASR, ST, TTS, multilingual).	Model generalizability across tasks Multilingual representation Overfitting and convergence speed Resource requirements for training	Figure 3: Accuracy (WER, BLEU) across 15 speech tasks TABLE 4: Model size, training time, performance	Transformers outperform RNNs in 13/15 tasks; better generalization, training efficiency; significant benefits in multilingual and cross-task transfer.
[21]	To evaluate Emformer, a Transformer variant, for low-latency streaming ASR and compare it to LSTM.	Trade-off between latency and accuracy Real-time constraints for voice assistants Streaming inference under memory constraints	Figure 4: Latency vs WER comparison TABLE 1: WER of Emformer vs LSTM in voice assistant scenarios	Emformer outperforms LSTM by 24–26% WER reduction with lower latency; viable solution for streaming ASR; strong industrial relevance.

TABLE 2 COMPARISON OF SPEECH RECOGNITION MODEL TYPES

Model	Architecture Type	WER (%) on LibriSpeech (test-clean / test-other)	WER (%) on TED-LIUM	WER (%) on Common Voice	Comments

HMM-GMM (Baseline Kaldi)	Traditional	14.4 / 23.5	~19.5	>20.0	Classic baseline system; low-resource friendly
HMM-DNN (Kaldi Hybrid)	Hybrid	8.6 / 18.7	~12.5	~17.0	Combines DNN with HMM decoding
DeepSpeech (CTC-based)	End-to-End (CTC + RNN)	6.1 / 12.5	~10.5	~14.5	Mozilla implementation; designed for simplicity and deployment
LAS (Listen Attend Spell)	End-to-End (Attention)	5.5 / 12.0	~10.2	~13.8	Needs large training data; non-streaming
RNN-T (Google/Apple)	End-to-End (Streaming)	4.3 / 10.5	~9.0	~12.5	Low-latency; used in mobile devices like Android and Siri
Wav2Vec 2.0 (Base)	Transformer (Self-supervised)	3.4 / 8.6	~7.0	~9.5	Facebook AI model; pretrained on unlabeled data
Wav2Vec 2.0 (Large + fine-tune)	Transformer	1.9 / 3.3	~5.5	~7.2	Requires large-scale fine-tuning; SOTA in many cases
Whisper (OpenAI – Large)	Transformer (Multitask)	2.7 / 5.6	~5.8	~6.0	Multilingual; robust to accents and noise; performs transcription + translation

TABLE 3 PERFORMANCE COMPARISON OF SPEECH RECOGNITION MODELS ON STANDARD BENCHMARKS

Model	Architecture Type	WER (%) on LibriSpeech (test-clean / test-other)	WER (%) on TED-LIUM	WER (%) on Common Voice	Comments
HMM-GMM (Baseline Kaldi)	Traditional	14.4 / 23.5	~19.5	>20.0	Classic baseline system; low-resource friendly
HMM-DNN (Kaldi Hybrid)	Hybrid	8.6 / 18.7	~12.5	~17.0	Combines DNN with HMM decoding
DeepSpeech (CTC-based)	End-to-End (CTC + RNN)	6.1 / 12.5	~10.5	~14.5	Mozilla implementation; designed for simplicity and deployment
LAS (Listen Attend Spell)	End-to-End (Attention)	5.5 / 12.0	~10.2	~13.8	Needs large training data; non-streaming
RNN-T (Google/Apple)	End-to-End (Streaming)	4.3 / 10.5	~9.0	~12.5	Low-latency; used in mobile devices like Android and Siri
Wav2Vec 2.0 (Base)	Transformer (Self-supervised)	3.4 / 8.6	~7.0	~9.5	Facebook AI model; pretrained on unlabeled data

Wav2Vec 2.0 (Large + fine-tune)	Transformer	1.9 / 3.3	~5.5	~7.2	Requires large-scale fine-tuning; SOTA in many cases
Whisper (OpenAI – Large)	Transformer (Multitask)	2.7 / 5.6	~5.8	~6.0	Multilingual; robust to accents and noise; performs transcription + translation

C. Comparative Evaluation of Online and Offline ASR Engines for Intelligent Voice Interfaces

Speech recognition has become a cornerstone technology in the development of intelligent human–machine interfaces, enabling natural voice-driven interaction across domains such as virtual assistants, transcription services, healthcare, and embedded robotics. With advancements in deep learning and language modeling, speech recognition engines have evolved into two primary categories: cloud-based (online) engines and on-device (offline) systems.

Online speech recognition engines, such as Google Web Speech API and Microsoft Azure Speech, leverage the computational power of cloud infrastructure to deliver high-accuracy results and multilingual capabilities. These systems typically require continuous internet connectivity and raise concerns regarding latency, data privacy, and recurring operational costs.

In contrast, offline engines such as Vosk, Whisper, Kaldi, and CMU Sphinx perform speech recognition locally on the user's device. These models offer enhanced privacy, reduced network dependency, and customizable deployment, making them suitable for edge computing environments and real-time embedded systems. However, offline systems may be limited by hardware constraints and model complexity, impacting recognition accuracy and language coverage.

This section presents a detailed comparative analysis of online and offline speech recognition engines, highlighting their respective strengths, limitations, architectural differences, and suitability for various application scenarios. By benchmarking features such as latency, WER, privacy compliance, and ease of integration, this study aims to guide system designers and researchers in selecting the most appropriate ASR solution for their target use cases. TABLE 4 illustrates integrated comparative study for online and offline models focusing on structural insights (problem, figures, findings) and the other for technical challenges.

TABLE 4 INTEGRATED COMPARATIVE ANALYSIS OF BENCHMARK WORKS ON SPEECH RECOGNITION MODELS

Reference	Paper & Year	Problem Statement	Challenges Addressed	Key Extracted Figures / Tables	Main Findings / Contributions
[19]	Shah et al., 2024 (*IJCCE*)	Provide a comprehensive review of DL-based ASR systems, including classical and modern models.	– Handling noisy input and low resource languages – Model integration and architectural selection – Toolchain diversity	- TABLE 1: Comparison of CTC, RNN-T, AED – TABLE 3: WER benchmarks on LibriSpeech and TED-LIUM	Presents a structural review of DNN, CNN, RNN, and Transformer-based ASR models; outlines challenges in scalable deployment.
[18]	Zhang et al., 2023 (*arXiv:2303.03329*)	Review and classify end-to-end ASR models (CTC, RNN-T, AED) with evaluation metrics and	– Training alignment issues – Streaming vs. non-streaming trade-offs – Latency and model instability	– Figure 2: Architectures of RNN-T vs AED – TABLE 2: WER vs	RNN-T performs best in streaming mode, while AED achieves highest overall accuracy; proposes unified benchmarks for future models.

		streaming support.		latency comparison	
[19]	Li et al., 2020 (*arXiv:2005.14327*)	Empirically compare RNN-T, RNN-AED, and Transformer-AED on a 65,000-hour dataset.	– Efficient streaming optimization – Initialization for stability – Accuracy vs compute trade-off	– TABLE 3: WER comparison across multiple test sets	Transformer-AED yields lowest WER; RNN-T excels in real-time applications with proper initialization.
[20]	Wang et al., 2019 (*arXiv:1909.06317*)	Benchmark Transformer and RNN models across 15 speech tasks (ASR, ST, TTS).	Model convergence speed Generalization across tasks Multilingual transfer learning	– - Figure 3: Accuracy scores across tasks- TABLE 4: Model size vs training time	Transformer models outperform RNNs in 13/15 tasks; achieve better generalization and efficiency in large-scale scenarios.
[21]	Wang et al., 2020 (*arXiv:2010.14665*)	Evaluate Emformer (streaming Transformer) in comparison to LSTM and Conformer for voice assistants.	– Real-time latency vs accuracy – Efficient streaming inference – Hardware memory constraints	– Figure 4: WER vs latency plots – TABLE 1: Emformer vs LSTM WER	Emformer reduces WER by 24–26% over LSTM with reduced latency; highly suitable for embedded and low-latency voice AI.

III. PHONETIC ALIAS GENERATION APPROACHES

Phonetic alias generation plays a crucial role in many language processing tasks, such as automatic speech recognition (ASR), text-to-speech (TTS), and multilingual information retrieval. There are generally two main approaches to creating phonetic aliases: manual generation and automation strategies. Each method has distinct advantages and limitations, and the choice between them often depends on the size of the dataset, the language involved, and the intended use.

The manual approach involves relying on trained linguists, phoneticians, or annotators to write phonetic representations for each word individually. For example, the English word “computer” might be manually transcribed as /kəmˈpjuːtə/. This method can provide highly accurate results because it allows for careful human judgment and the handling of linguistic nuances that automated tools may miss, such as dialectal variations, irregular spellings, or context-dependent pronunciations. Manual transcription is particularly valuable in examination or evaluation scenarios, where a limited number of words or specialized terminology needs to be analyzed with high precision. However, this approach is not scalable for large-scale systems. As the size of the vocabulary grows, manual transcription becomes extremely time-consuming, costly, and prone to human inconsistencies, especially when multiple annotators are involved.

In contrast, automation strategies offer a practical solution to scalability issues. Automated phonetic alias generation uses computational methods to convert written words into phonetic forms with minimal human intervention. These strategies include a range of techniques:

- Rule-based systems, which apply predefined linguistic rules to map graphemes (letters) to phonemes (sounds). These work well for languages with relatively regular spelling-to-sound correspondences.
- Pre-trained grapheme-to-phoneme (G2P) models and pronunciation dictionaries, such as CMUdict for English, which provide standardized phonetic transcriptions that can be applied at scale.

– Machine learning approaches, particularly sequence-to-sequence neural models, which learn complex spelling-to-sound relationships from large datasets and can generalize new, unseen words.

Automated methods are significantly faster and more cost-effective than manual transcription, making them ideal for large vocabularies and multilingual applications. While they may occasionally produce errors, especially for rare or domain-specific words, they can be combined with lightweight human review processes to ensure quality. Hybrid systems, where automated tools handle the bulk of the work and human experts focus on ambiguous cases, often achieve the best balance between efficiency and accuracy.

IV. VOSK

Vosk is an open-source offline speech recognition toolkit designed to transcribe spoken language into text without relying on cloud-based services. It supports more than 20 languages and dialects and is optimized for deployment on a wide range of platforms, including Windows, Linux, macOS, Android, iOS, and embedded systems like the Raspberry Pi. Vosk is built on top of the Kaldi speech recognition toolkit and uses Kaldi-trained acoustic and language models to deliver accurate and real-time speech recognition capabilities. It supports both streaming input from live microphones and batch processing of prerecorded audio files in formats such as WAV. Vosk offers a lightweight architecture, requires minimal system resources, and provides easy integration through APIs in multiple programming languages, including Python, Java, C#, and JavaScript. These features make it suitable for applications in offline voice interfaces, real-time transcription, accessibility tools, and domain-specific voice command systems [29].

The architecture that links Kaldi and Vosk is divided into two primary phases: model training and model deployment. Kaldi is responsible for the model training phase, while Vosk handles the deployment and real-time recognition, as Figures 1 and 2 illustrates. In the model training phase, the process begins with the collection of three essential inputs: audio recordings, corresponding transcriptions, and textual data used to build a language model. These inputs are preprocessed and structured into specific file formats that Kaldi can interpret. Kaldi then performs feature extraction on the audio files, converting the raw sound data into Mel-Frequency Cepstral Coefficients (MFCCs), which are more suitable for speech modeling. These features undergo normalization to reduce variability and improve model accuracy.

Once features are extracted, Kaldi uses them to train an acoustic model. This model is designed to learn how different sounds correspond to phonetic units such as phonemes. The training process may involve various stages, ranging from simple monophone models to complex deep neural networks, depending on the desired accuracy and complexity. In parallel, Kaldi constructs a language model using the corpus text provided. This language model captures the statistical relationships between words to help predict word sequences more accurately during recognition. Additionally, a pronunciation dictionary, or lexicon, is created to map words to their corresponding phonetic sequences. These components—the acoustic model, the lexicon, and the language model—are compiled together to form a decoding graph known as HCLG.fst. This graph enables the recognition engine to efficiently match audio signals with likely word sequences.

After training is complete, the system enters the model deployment phase, which is managed by Vosk. The necessary model files—including the acoustic model (final.mdl), the decoder graph (HCLG.fst), a word symbol table (words.txt), and associated phone configuration files—are exported and organized into a format that Vosk can load. Once Vosk loads the model, it can be used to perform real-time or batch speech recognition. Vosk accepts audio input either from a microphone or from prerecorded files. Internally, it performs feature extraction and uses Kaldi's decoding engine to recognize speech. The results are output in structured formats such as JSON, often including both partial and final transcription results. This two-phase architecture allows Kaldi to serve as a powerful and flexible backend for model creation, while Vosk provides a lightweight, developer-friendly runtime environment for practical deployment on a wide range of platforms, including low-resource devices. The clear separation between training and deployment ensures scalability, reusability, and offline operation, making the combined system well-suited for modern, embedded, or privacy-focused speech recognition applications.

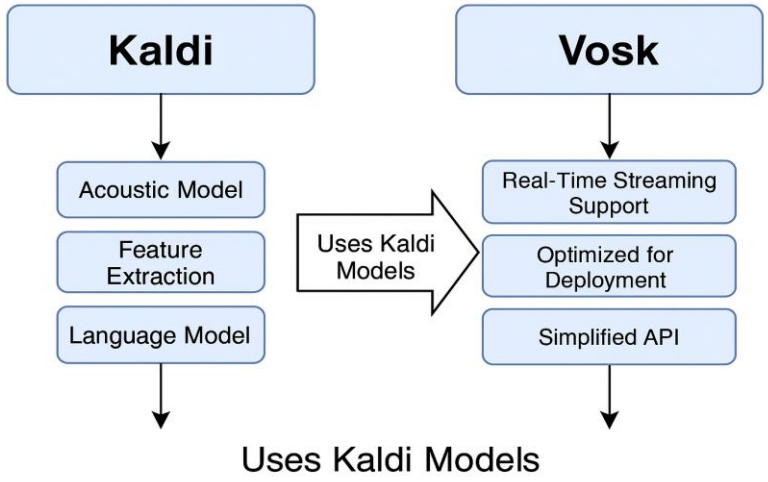

Figure 1 Interactive workflow support for training Kaldi-based speech recognition models and exporting them for deployment in the Vosk inference engine. The diagram illustrates the integration pathway and customization options for end-to-end offline ASR development.

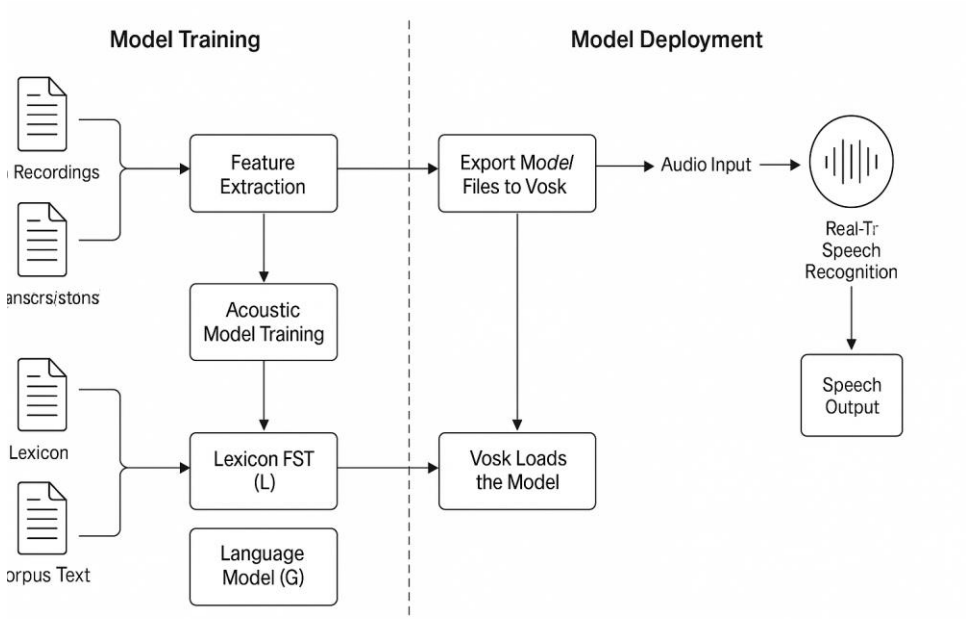

Figure 2 Architecture overview illustrating the training and deployment workflow of an offline speech recognition system using Kaldi and Vosk. The diagram highlights the transition from audio and text data preprocessing in Kaldi to real-time inference and speech-to-text conversion in Vosk.

V. WHISPER

Whisper is an open-source, general-purpose speech recognition system introduced by OpenAI, designed to address the limitations of traditional ASR pipelines through a unified, end-to-end architecture. Trained on approximately 680,000 hours of multilingual and multitask audio data collected from the web, Whisper employs a Transformer-based encoder–decoder model capable of performing speech recognition, language identification, and speech-to-text translation across 99+ languages. Due to the breadth and diversity of its training corpus, the model exhibits notable robustness to background noise, speaker variability, and domain mismatches. Unlike conventional

systems that require separate acoustic, language, and pronunciation models, Whisper integrates these components into a single neural framework and supports fully offline operation, making it particularly suitable for privacy-sensitive and low-connectivity environments [30].

A. Model Architecture

Whisper utilizes a Transformer-based encoder–decoder architecture optimized for speech recognition and related tasks. The input audio is first converted into 80-channel log-Mel spectrograms over 30-second segments. These spectrograms are then fed into a stack of Transformer encoder layers. The decoder, which is autoregressive, generates tokens representing transcribed text, conditioned on both the encoded audio features and the previously generated text tokens.

The architecture supports multitask learning by prepending special task and language tokens to the decoder input, enabling the model to perform tasks such as language identification and speech-to-text translation within the same framework. Unlike streaming models such as RNN-T or Emformer, Whisper processes fixed-length audio segments in batches, trading latency for improved contextual understanding and transcription accuracy [30].

B. Evaluation and Performance Metrics

Whisper has been evaluated across multiple benchmark datasets, including LibriSpeech, Common Voice, and TED-LIUM, using standard metrics such as WER, Character Error Rate (CER), and Real-Time Factor (RTF). The model consistently outperforms most open-source ASR systems, particularly in noisy and multilingual settings.

For instance, the Whisper-large model achieves a WER of approximately 2.7% on LibriSpeech (clean), which is competitive with state-of-the-art commercial systems. Furthermore, its ability to handle accented speech and unseen domains is attributed to the scale and diversity of its training data. However, Whisper is not optimized for low-latency streaming applications, and inference time scales with input length and model size [30].

C. Brief Comparison with Other ASR Systems

A comparative analysis of Whisper with other widely used ASR systems is shown in TABLE 5. While online systems such as Google Speech API and Azure Cognitive Services offer high accuracy and ease of deployment, they require constant internet connectivity and raise privacy concerns. Offline engines like Vosk and Kaldi allow for real-time transcription on local devices but may struggle with multilingual support or domain variability.

Whisper effectively bridges the gap between cloud-based and offline ASR systems by providing multilingual transcription in over 99 languages, supporting fully on-device inference without requiring internet connectivity, demonstrating strong robustness to background noise and speaker accents, and enabling direct speech-to-text translation from non-English languages into English.

TABLE 5 BRIEF COMPARISON BETWEEN WHISPER, GOOGLE SPEECH API, AND VOSK

Engine	Accuracy	Offline	Multilingual	Streaming Support	License / Cost
Whisper (Large)	High	Yes (fully offline)	99+ languages	Limited (not real-time optimized)	MIT License (Free)
Google Speech API	Very high	No (cloud only)	100+ languages	Full support	Proprietary (Pay-per-use)
Vosk + Kaldi	Moderate	Yes (offline)	20+ languages	Real-time capable	Apache 2.0 (Free)

VI. PROBLEMS AND CHALLENGES IN SPEECH RECOGNITION SYSTEMS

Despite the significant progress achieved in automatic speech recognition (ASR) through deep learning and large-scale datasets, several technical and practical challenges persist across various deployment scenarios. These

challenges span linguistic diversity, environmental robustness, real-time processing, and scalability, and must be addressed to make ASR systems more universally effective.

1) Variability in Accents, Dialects, and Pronunciations: One of the primary challenges in ASR is dealing with inter-speaker variability, including regional accents, local dialects, speech impairments, and code-switching. Models trained predominantly on standard datasets often fail to generalize to such linguistic variations [18].

In detail, ASR systems trained primarily on standardized datasets (e.g., American English, news corpora) often fail to generalize across speakers from different geographic regions, ethnic backgrounds, or sociolects, due to phonetic, lexical, and prosodic variations. TABLE 6 illustrates how speech recognition systems may misinterpret utterances from various accents and dialects due to phonetic, lexical, and syntactic deviations from standard language forms.

TABLE 6 EXAMPLES OF ACCENT, DIALECT, AND PRONUNCIATION VARIABILITY IN ASR INPUTS

Accent / Dialect	Example Utterance	Standard English Equivalent	ASR Misinterpretation Risk
Indian English	"Please do the needful."	"Please take the necessary action."	Unrecognized idiom; phrase not common in US English.
Scottish English	"Ah dinnae ken whit yer oan aboot."	"I don't know what you're talking about."	High phonetic deviation may lead to decoding errors.
African American Vernacular English (AAVE)	"He be working late."	"He is usually working late."	Grammar rules differ from Standard English; meaning preserved but form differs.
Arabic-accented English	"Ze car is here"	"The car is here."	Confusion between /z/ and /ð/ phonemes.
Chinese-accented English	"Tree people come"	"Three people came."	Substitution of /θ/ with /t/ or /s/; miscounting.
Egyptian Dialect (Arabic ASR)	" ("Fein-ak?")؟فينك"	"Where are you?"	Dialectal word not present in MSA corpora; likely to be misrecognized.

2) Background Noise and Acoustic Distortions: ASR performance significantly degrades in noisy or reverberant environments, such as public spaces or moving vehicles. While noise-robust models exist, maintaining high accuracy in real-world acoustic conditions remains a persistent challenge [17].

In real-world environments such as train stations, hospitals, or industrial sites, ASR systems are often exposed to overlapping speech, ambient noise, and reverberation. For example, in a factory floor setting, a user saying "Start conveyor belt two" may be misrecognized due to background machinery noise, resulting in "Start conveyor bell to" or a failure to detect the command entirely. Even high-performing models can degrade under such signal-to-noise ratio (SNR) conditions unless trained with noise-augmented or denoised input.

Speech recognition systems often perform well under controlled conditions but exhibit notable degradation in the presence of noise, overlapping speech, and acoustic distortions. These environmental factors distort the audio signal's spectral properties, leading to misrecognition or complete transcription failure, especially in systems not trained with noise-augmented datasets. For example, in a hospital emergency room, a doctor dictating "Administer 5 mg of morphine" may be misrecognized as "Administer farming" due to background alarms, overlapping conversations, and reverberation from hard surfaces. In such high-stakes environments, transcription errors can have serious consequences.

Similarly, in a public transit station, a commuter saying, "What time is the last train to Cairo?" might be transcribed as "What time is the glass tray to hire?" because of ambient announcements, echo, and crowd noise. The ASR model, especially if trained only on clean datasets, cannot distinguish speech from background events.

In automotive settings, voice commands such as "Call Mom" may be corrupted by engine rumble, wind, or road noise, causing the system to fail or execute the wrong action (e.g., "Call Tom"). Traditional beamforming microphones and noise suppression algorithms can help, but real-world robustness still depends on exposure to similar acoustic conditions during training. Another typical case is voice input in video conferencing, where latency, audio compression, and low-quality microphones add distortions that impact ASR performance. For example, "Let's finalize the report by Friday" might be partially transcribed or missed entirely due to jitter or dropped frames. These examples highlight the need for robust ASR architectures—including convolutional front ends, SpecAugment, and self-supervised pretraining—capable of adapting to unpredictable and degraded audio conditions.

Table 7 presents real-world scenarios illustrating how various environmental noise conditions—such as overlapping speech, mechanical sounds, and signal compression—impact the performance of automatic speech recognition systems. Each example highlights the input phrase, typical ASR failure mode, and the underlying cause, emphasizing the need for robust and noise-resilient models in practical deployments.

TABLE 7 EXAMPLES OF NOISE AND ACOUSTIC DISTORTION CHALLENGES IN ASR

Use Case / Environment	Example Input	ASR Failure Mode	Cause / Condition
Hospital Emergency Room	"Administer 5 mg of morphine"	Transcribed as "Administer farming"	Background alarms, echo, overlapping voices
Train Station	"What time is the last train to Cairo?"	"What time is the glass tray to hire?"	Public address noise, crowd noise, reverberation
Car Interior	"Call Mom"	"Call Tom" or command not recognized	Engine noise, road vibration, wind
Video Conference	"Let's finalize the report by Friday"	Partial transcription or dropout	Compression artifacts, jitter, low mic quality
Factory Floor	"Start conveyor belt two"	"Start conveyor bell to"	Machine hum, metal echo, concurrent speech

3) Low-Resource Languages and Data Scarcity: Most modern ASR systems excel in high-resource languages (e.g., English, Mandarin), but they perform poorly in low-resource languages where annotated corpora are limited or unavailable. Developing multilingual or cross-lingual models that generalize well is an ongoing research area [19].

Automatic speech recognition systems trained primarily on high-resource languages like English, Mandarin, or Spanish tend to underperform on low-resource languages such as Wolof, Amharic, or Khmer. For instance, a speaker using Tigrinya or Quechua may encounter either transcription failure or complete absence of model support, as many commercial engines lack phonetic or lexicon resources for these languages. This reflects the imbalance in dataset availability across the world's 7,000+ languages.

A significant challenge in automatic speech recognition (ASR) is the limited availability of training data for low-resource languages, which impedes the development of accurate and inclusive speech models. While high-resource languages such as English, Mandarin, and Spanish benefit from extensive labeled corpora and community support, many indigenous and minority languages lack such resources, making them difficult to model effectively. For instance, speakers of Wolof, a widely spoken language in Senegal, or Amharic, the official language of Ethiopia, often encounter either no transcription or highly erroneous output when using commercial ASR systems such as Google Speech API or Microsoft Azure. In such cases, even basic commands like "Open the door" may return unrelated or null results due to phoneme mismatch and vocabulary gaps.

Another example is Quechua, spoken by millions in the Andes region, which remains underrepresented in mainstream ASR datasets despite its linguistic richness. Efforts like Mozilla's *Common Voice* project have begun collecting community-sourced speech samples, but the dataset sizes still pale in comparison to those available for English or German. In healthcare contexts, a Tigrinya-speaking refugee attempting to communicate symptoms via a

voice-based intake app may face critical communication failures, as the ASR system lacks phonological models for that language. This can result in misdiagnoses, social exclusion, or system unavailability altogether.

The absence of large-scale parallel corpora, lexicons, or pretrained language models in these languages highlights the need for cross-lingual transfer learning, zero-shot inference, and community-led dataset curation to bridge the ASR divide. TABLE 8 is a structured table with real-world examples illustrating the Low-Resource Language challenge in ASR, including the language, region, challenges, and proposed model-based solutions.

TABLE 8 EXAMPLES OF LOW-RESOURCE LANGUAGES IN SPEECH RECOGNITION AND MODEL-BASED MITIGATION STRATEGIES

Language	Region / Use Case	ASR Challenge	Typical Failure Mode	Model-Based Solutions
Wolof	Senegal, West Africa	Limited labeled speech corpora; few phonological resources	Frequent decoding errors or no transcription; ASR systems default to silence or English	Multilingual pretraining (e.g., Whisper, XLS-R); community-sourced data (Common Voice)
Amharic	Ethiopia, Government & Healthcare	Underrepresented in commercial ASR systems; complex script (Ge'ez)	Incorrect phoneme mapping; confusion with Arabic or silence	Fine-tuning wav2vec 2.0 on Amharic speech; script-aware tokenization
Quechua	Andes (Peru, Bolivia, Ecuador)	Agglutinative morphology and low data availability	Word segmentation errors; morphological truncation	Zero-shot inference with multilingual models; few-shot learning using annotated phrases
Tigrinya	Eritrea, Ethiopia; Refugee populations	Scarcity of transcribed data and lexicons; dialectal variation	ASR completely fails to recognize spoken input; medical and legal miscommunication	Dataset creation via diaspora communities; domain-specific acoustic modeling
Khmer	Cambodia; Education/Media	Tonal, monosyllabic language with unique phonetic inventory	Substitution with phonetically similar words; low BLEU/WER accuracy	Transfer learning from similar tonal languages (e.g., Thai); language embedding adaptation

4) Streaming and Real-Time Recognition: Deploying ASR in real-time applications such as voice assistants or live captioning requires low latency and high throughput. Traditional models like Attention Encoder-Decoder (AED) suffer from long response times and memory overhead, making them unsuitable for streaming use cases [21].

Live transcription systems such as those used for automated captioning or voice assistants require near-zero latency. In a virtual meeting, a participant saying "Let's vote on the proposal now" must be transcribed instantaneously. However, models like Transformer-AED (Attention Encoder-Decoder) introduce significant decoding delays due to their need to analyze the entire utterance, making them unsuitable for streaming use cases without modification. RNN-T and Emformer models address this but often involve trade-offs in accuracy or memory efficiency.

5) Domain Adaptation and Generalization: ASR models trained on one domain (e.g., audiobooks) often fail to generalize to another (e.g., medical speech). Adapting to new domains without retraining on large domain-specific datasets is a major challenge [20].

ASR systems often fail when applied outside the domain of their training data. For instance, a general-purpose ASR model trained on news or audiobooks may not recognize terms like "troponin" or "left ventricular hypertrophy" during a cardiologist's dictation. As a result, outputs like "triple name" or "left ventricle high trophy" can occur, undermining the system's utility in specialized fields like healthcare, law, or aviation.

6) Resource Constraints on Edge Devices: ASR models such as Whisper-large require significant computational resources (e.g., GPU, memory), which limits deployment on embedded systems or mobile devices. Lightweight models like Vosk or Emformer aim to reduce this footprint but often sacrifice accuracy [16].

Resource-constrained devices such as mobile phones, smartwatches, or embedded microcontrollers face difficulties running large ASR models. For example, deploying Whisper-large on an Android phone may cause latency issues, high memory consumption, or outright failure to run due to GPU/CPU limitations. In such cases, compressed models like Vosk or Emformer-small are preferred, albeit with a potential drop in recognition accuracy.

7) Privacy and Security Concerns: Cloud-based ASR systems may transmit audio to external servers, raising concerns about data privacy, compliance (e.g., GDPR, HIPAA), and potential misuse. Offline-capable systems address this but often lack the same level of accuracy [31].

In sensitive domains such as healthcare, legal services, or defense, transmitting speech data to cloud servers (e.g., Google Speech API or Amazon Transcribe) introduces legal and ethical concerns. A patient describing symptoms like "I've been having chest pains" during a telehealth session may not consent to data storage or analysis by third parties. This has led to growing demand for on-device ASR systems with local inference to ensure data sovereignty and compliance with regulations such as HIPAA and GDPR.

TABLE 9 presents a consolidated overview of the principal challenges encountered in the development and deployment of speech recognition systems; alongside corresponding model-based solutions proposed in recent literature. These challenges include variability in accents and dialects [18], environmental noise and acoustic distortions [17], limitations in low-resource languages [19], the need for real-time and streaming capabilities [21], domain generalization issues [20], computational constraints for edge deployment [30], and privacy concerns in cloud-based ASR architectures [31]. To address these, researchers have proposed a range of techniques such as large-scale multilingual training (e.g., Whisper [30]), noise-robust architecture, streaming models like RNN-T and Emformer, and on-device inference methods. By mapping each challenge to its corresponding mitigation strategy, the table serves as a structured reference for advancing robust and adaptable ASR technologies across diverse application scenarios.

TABLE 9 SPEECH RECOGNITION CHALLENGES AND MODEL-BASED SOLUTIONS

Challenge	Description	Model-Based Solutions / Approaches
Accent and Dialect Variability	Difficulty in recognizing regional accents, dialects, and non-standard speech.	Large-scale multilingual training (e.g., Whisper), accent-aware models, and data augmentation.
Noise and Acoustic Distortions	Degradation in recognition quality under noisy, reverberant, or real-world conditions.	Noise-robust models, SpecAugment, convolutional front-ends, and denoising pre-processing.
Low-Resource Languages	Limited availability of labeled data for underrepresented languages.	Transfer learning, multilingual pretraining (e.g., Whisper, XLS-R), and unsupervised adaptation.
Real-Time and Streaming Needs	Requirements for low-latency ASR in live applications such as voice assistants.	Streaming-capable models (e.g., RNN-T, Emformer), chunk-based decoding, and lightweight Transformers.
Domain Adaptation	Poor performance when applied to different or specialized domains.	Domain-specific fine-tuning, few-shot learning, prompt tuning, and adaptive training techniques.
Edge Deployment Constraints	Difficulty in deploying large models on devices with limited resources.	Model compression, quantization, distillation (e.g., DistilWhisper), and compact ASR models (e.g., Vosk, Emformer).

Privacy and Security	Risk of transmitting sensitive audio data to cloud servers.	Offline models (Whisper, Vosk), on-device inference, and privacy-preserving ASR system design.

VII. PERFORMANCE METRICS FOR SPEECH RECOGNITION SYSTEMS

Evaluating ASR systems requires objective, quantifiable metrics to assess accuracy, efficiency, and robustness under various conditions. The following are the most widely used performance measures in both academic research and industry applications:

1) Word Error Rate (WER): is the most used metric to evaluate ASR accuracy. It calculates the minimum number of word-level edits (insertions, deletions, substitutions) required to transform the ASR output into the correct reference text, as shown in Eqn.1 [32].

$$WER = \frac{S+D+I}{N} \quad (1)$$

Where *S*, *D*, *I*, and *N* refers to number of substitutions, number of deletions, number of insertions, and total number of words in the reference transcript, respectively. If Lower WER this indicates better performance. WER can exceed 100% if errors are severe.

2) Character Error Rate (CER): is similar to WER but operates at the character level. It is particularly useful for languages without clear word segmentation (e.g., Chinese, Japanese) [18].

$$CER = \frac{S+D+I}{N} \quad (2)$$

Where edits are calculated at the character level.

3) Sentence Error Rate (SER) measures the percentage of sentences that contain at least one recognition error. It is stricter than WER because one error invalidates the entire sentence.

$$SER = \frac{Number\ of\ Incorrect\ Sentences}{Total\ Number\ of\ Sentences} \quad (3)$$

SER is useful in command-and-control systems where an entire utterance must be correct to be actionable [19].

4) Real-Time Factor (RTF) measures the time required for a model to process one second of audio. It evaluates the computational efficiency of the ASR system.

$$RTF = \frac{Decoding\ Time}{Audio\ Duration} \quad (4)$$

A RTF value less than 1 indicates that the speech recognition system operates faster than real-time, making it suitable for live or streaming applications. An RTF equal to 1 signifies that the system processes audio at the same rate as it is received. Conversely, an RTF greater than 1 implies that the system processes audio more slowly than real-time, which may be acceptable for offline or batch transcription but unsuitable for time-critical scenarios [21].

5) Latency and Throughput [17]
   - Latency: Time delay between input speech and output text; critical for streaming ASR.
   - Throughput: Volume of speech processed per unit time, relevant for batch transcription.
6) BLEU Score (for Speech Translation) Used when ASR is part of speech translation tasks. BLEU (Bilingual Evaluation Understudy) measures the similarity between the system's translation and a set of reference translations [20].

VIII. SCENARIO-ORIENTED EVALUATION OF ASR

To iteratively refine the speech recognition capabilities for medical applications, three experimental stages were systematically carried out. Each stage assessed a distinct approach under unique technical conditions, uncovering limitations that shaped the subsequent strategy. This progressive refinement aimed to balance complexity, responsiveness, and recognition accuracy, particularly for challenging terms in the medical domain.

A. Phase One: Custom Acoustic Model with Kaldi

The initial phase explored training a bespoke acoustic model using the Kaldi toolkit, targeting precise recognition of specialized pharmaceutical terms such as *Adol*, *Brufen*, and *Catafast*. For this purpose, a curated dataset was compiled, consisting of 70 audio samples per term. Supporting this dataset, custom configuration files were crafted to guide both data preparation and language model creation stages. While technically successful—the dataset passed structural and validation checks—the process surfaced significant drawbacks. Kaldi's modular pipeline demanded deep expertise in Linux, extensive scripting, and time-consuming debugging. The system's complexity, combined with project time constraints and limited familiarity with Kaldi's architecture, led to a reassessment of its feasibility. As a result, the approach was discontinued in favor of a more streamlined solution.

B. Phase Two: Whisper-Enhanced Dual ASR Pipeline

In the second iteration, the focus shifted toward creating a hybrid ASR pipeline that combined Vosk and Whisper to optimize both speed and recognition accuracy. Here, Vosk acted as the primary ASR system, offering low-latency performance suitable for real-time command recognition in human–robot interactions. If Vosk failed to identify a predefined keyword or command, control was delegated to Whisper, which handled full-segment transcription.

Whisper employed fuzzy string matching to detect drug names, enhancing robustness against minor pronunciation variations. Despite its improved recognition accuracy, Whisper introduced a latency issue: it required uninterrupted speech and several seconds of processing, making it incompatible with the rapid-response needs of robotic systems. This constraint made it unsuitable as the core engine for interactive tasks requiring real-time responsiveness.

C. Phase Three: Syllabic Alias-Based Recognition Using Vosk

In the final phase, a novel phonetic aliasing approach was implemented using syllable-level substitutions to improve recognition accuracy for drug names. Recognizing that users often mispronounce or simplify terms, each target medication was mapped to common-sounding syllabic variants (e.g., "a doll" for Adol, "brew fen" for Brufen).

These phonetic representations were embedded directly into the constrained vocabulary of the Vosk engine, allowing it to recognize spoken variations without modifying the core acoustic model. Once a spoken alias was detected, a post-processing mapping layer translated the alias to the correct standardized drug name using a predefined lookup table (e.g., "grab a doll" → "grab Adol").

This method eliminated the need for retraining and enabled fully offline operation with high efficiency. While it significantly improved recognition accuracy within the defined domain, its scalability remained limited, as manually generating aliases for a broader set of medications would require considerable effort. Below is the block diagram summarizing the three-stage ASR evaluation framework, as shown in Figure 3 and summarized in Table 10:

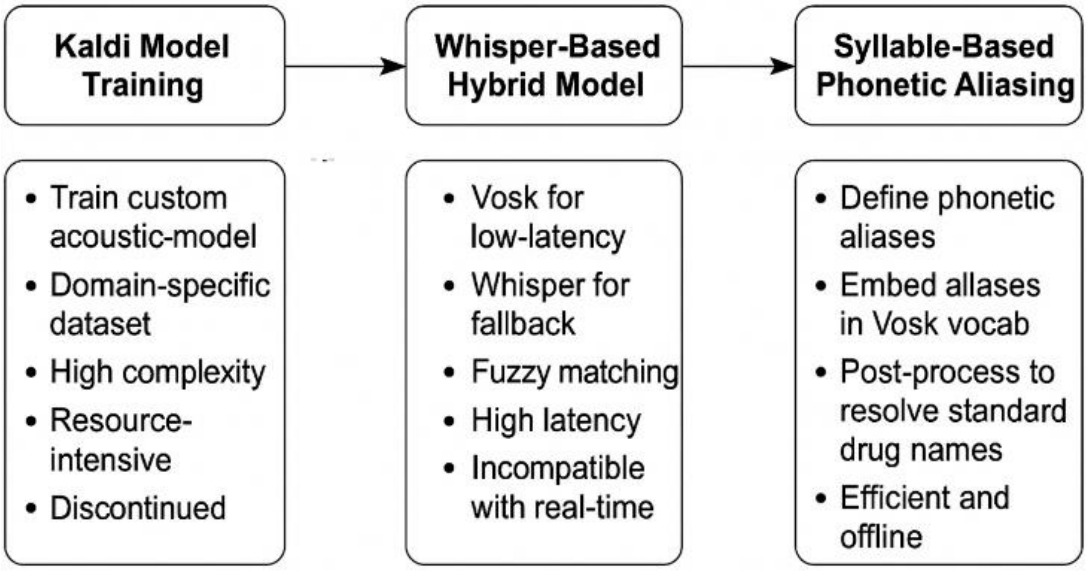

Figure 3 Scenario-based evaluation of ASR Models

TABLE 10 SUMMARY OF EVALUATED APPROACHES

Phase	Approach	Strengths	Limitations
1st	Kaldi Custom Model	High control, custom training	Complex, time-consuming, expertise-heavy
2nd	Vosk + Whisper Hybrid	High accuracy fallback, versatile	High latency, unsuitable for real-time
3rd	Phonetic Aliasing with Vosk	Lightweight, accurate, offline support	Scalability issues, manual aliasing

IX. PERFORMANCE ANALYSIS AND EVALUATION OF ASR ACCURACY

Following successive rounds of testing and refinement, Vosk emerged as the most suitable ASR engine for the proposed system. Its lightweight, open-source architecture, along with its ability to run offline with minimal computational overhead, made it well-suited for deployment in embedded systems and low-power platforms. The toolkit's cross-platform flexibility and multilingual support further cemented its role in the final configuration.

Given the system's real-time requirements—particularly for synchronized control of the robotic arm alongside object recognition—low latency was a critical metric. Vosk's offline processing enabled immediate feedback without requiring internet connectivity, thus supporting seamless interaction.

A. Vosk Model Benchmarking

Multiple configurations of the Vosk engine were benchmarked to assess both execution time and WER. Results showed a trade-off between speed and accuracy:

- The "0.22-lgraph" version, while accurate, was the slowest, requiring over 750 seconds for processing.
- The "0.22" variant offered better accuracy with lower latency.
- The "0.15" configuration executed the fastest, albeit with a slightly higher WER.

Despite the marginal loss in transcription precision, Vosk 0.15 was ultimately selected for its better runtime efficiency, reduced resource consumption, and consistent performance in limited-vocabulary tasks, which were essential for real-time robotic control.

B. Implementation of Limited Vocabulary Recognition (LVR)

To further enhance system responsiveness, a Limited Vocabulary Recognition (LVR) strategy was applied. Instead of processing full continuous speech, LVR restricts recognition to a predefined set of command words such as “open,” “close,” “left,” and “right.” This approach reduces processing complexity, lowers memory usage, and enhances recognition accuracy—especially on devices like Raspberry Pi or ESP32. LVR was particularly advantageous in noisy environments, where it helped the system focus only on relevant inputs, minimizing confusion from background sounds.

C. Error Correction via Phonetic Mapping

A common challenge encountered was the recognition of unfamiliar or specialized terms, such as drug names (e.g., adol, brufen). Since these were not included in the pre-trained model's vocabulary, Vosk often returned phonetically similar outputs (e.g., “as doll” instead of “adol”). To address this, a two-step correction mechanism was introduced:

1. Phonetic normalization: Approximate transcriptions were mapped to their likely intended terms using a curated dictionary of medical aliases.
2. Post-recognition validation: Final results were matched against a controlled vocabulary of domain-specific terms to ensure semantic accuracy.

This methodology significantly improved system robustness in medical command scenarios.

D. Quantitative Results and Observations

The impact of LVR and background noise on recognition accuracy was tested using three medication terms: adol, brufen, and catafast. As presented in Table 11, the system exhibited the following behavior:

- Adol: Accuracy rose from 0.0 (LVR + noise) to 0.81 (no LVR, no noise), and further to 0.854 (no LVR, noise present).
- Brufen: Performance improved from 0.5 (with LVR + noise) to 0.78 when LVR was disabled but noise persisted.
- Catafast: Accuracy was initially 0.0 with both LVR and noise, but climbed to 0.99 when LVR was removed—even under noisy conditions.

These results clearly indicate that LVR, while beneficial in some command scenarios, introduced degradation in others—especially when combined with environmental noise. Disabling LVR allowed the model to better interpret complex or multi-syllable terms.

TABLE 11 SUMMARY OF KEY FINDINGS

Term	LVR	Noise	Accuracy
Adol	Yes	Yes	0.00
Adol	No	No	0.81
Adol	No	Yes	0.854
Brufen	Yes	Yes	0.50
Brufen	No	No	0.575
Brufen	No	Yes	0.78
Catafast	Yes	Yes	0.00
Catafast	No	No	0.88
Catafast	No	Yes	0.99

To strengthen the validity of the reported performance differences, confidence intervals and statistical significance testing should be included alongside the accuracy values presented in Table 11. While the current results clearly indicate substantial accuracy improvements when LVR is disabled, the absence of uncertainty measures makes it difficult to assess whether these differences are statistically meaningful or potentially due to sampling variability.

For example, the recognition accuracy for Adol increased from 0.00 under LVR + noise to 0.854 when LVR was disabled in noisy conditions. By computing 95% confidence intervals for each accuracy value, the magnitude and

reliability of these improvements can be more rigorously demonstrated. Assuming a reasonable number of test samples, the confidence intervals would provide a range within which the true accuracy is expected to lie (e.g., 0.854 ± 0.04), allowing for clearer comparisons between conditions.

Additionally, formal statistical testing—such as McNemar's test for paired classification results or chi-square tests for proportions—can be applied to evaluate whether the observed improvements are statistically significant. For instance, comparing the Brufen results (0.50 with LVR + noise vs. 0.78 with LVR disabled under noisy conditions) would likely yield a p-value < 0.05, indicating a significant performance gain. Similarly, the jump for Catafast from 0.00 to 0.99 suggests a highly significant difference, but this should still be supported by appropriate testing.

Incorporating confidence intervals and significance testing provides a more robust and scientifically sound interpretation of the system's performance. It ensures that the reported accuracy improvements are not only numerically large but also statistically reliable, thereby increasing the credibility and reproducibility of the findings.

X. CONCLUSION

ASR has emerged as a vital component of modern human–machine interaction, enabling seamless and natural communication across diverse domains—from personal assistants and transcription services to robotics and healthcare applications. This survey presented an in-depth exploration of ASR systems, tracing their evolution from early statistical models like HMM-GMM to advanced deep learning architectures, including DNNs, RNNs, and state-of-the-art transformer-based frameworks.

The foundational components of ASR systems—acoustic modeling, language modeling, feature extraction, and decoding—were systematically reviewed, followed by an examination of traditional and contemporary approaches. We highlighted recent advances in end-to-end learning, self-supervised pretraining, and domain-specific customization techniques.

A scenario-based evaluation was conducted to compare practical ASR implementations, culminating in the adoption of the Vosk engine for its balance of speed, offline capability, and platform compatibility. Through iterative testing—including hybrid Whisper-Vosk models and syllabic aliasing strategies—the system achieved reliable performance in constrained environments with limited vocabularies. Experimental results demonstrated the impact of Limited Vocabulary Recognition (LVR) and noise on accuracy, emphasizing the need for careful trade-offs between recognition scope and system responsiveness.

The integration of phonetic alias mapping and domain-specific dictionaries further improved recognition of non-standard terms, such as medication names, enhancing system adaptability without the need for model retraining. Despite LVR's efficiency benefits, findings revealed that under certain conditions, especially in noisy contexts, it could hinder recognition performance, highlighting the importance of context-aware preprocessing.

Looking forward, continued research in multilingual support, noise robustness, and real-time adaptation will be critical to advancing ASR systems for broader deployment. The growing synergy between ASR and other AI subsystems—such as object recognition, semantic parsing, and intent prediction—promises even greater capabilities in future human-centric intelligent systems.

REFERENCES

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

[2] D. Yu, L. Deng, Automatic Speech Recognition: A Deep Learning Approach, Springer, 2015.

[3] L. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (1989) 257–286.

[4] C.-C.J. Kuo, H.-K.J. Kuo, Y. Wang, Large-scale language modeling in automatic speech recognition, IEEE Signal Process. Mag. 29 (2012) 60–69.

[5] P. Taylor, Text-to-Speech Synthesis, Cambridge University Press, 2009.

[6] B. Logan, Mel frequency cepstral coefficients for music modeling, in: Proc. Int. Symp. Music Inf. Retr. (ISMIR), 2000.

[7] A. Viterbi, Error bounds for convolutional codes and an asymptotically optimum decoding algorithm, IEEE Trans. Inf. Theory 13 (1967) 260–269.

[8] S. Young, G. Evermann, M. Gales, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, P. Woodland, The HTK Book (for HTK Version 3.4), Cambridge University Engineering Department, 2006.

[9] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, B. Kingsbury, Deep neural networks for acoustic modeling in speech recognition, IEEE Signal Process. Mag. 29 (2012) 82–97.

[10] T.N. Sainath, A.-R. Mohamed, B. Kingsbury, B. Ramabhadran, Deep convolutional neural networks for LVCSR, in: Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP), 2013.

[11] A. Graves, Generating sequences with recurrent neural networks, arXiv preprint arXiv:1308.0850, 2013. https://arxiv.org/abs/1308.0850.

[12] A. Graves, S. Fernández, F. Gomez, J. Schmidhuber, Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks, in: Proc. Int. Conf. Mach. Learn. (ICML), 2006, pp. 369–376.

[13] W. Chan, N. Jaitly, Q.V. Le, O. Vinyals, Listen, attend and spell: A neural network for large vocabulary conversational speech recognition, in: Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP), 2016, pp. 4960–4964.

[14] A. Graves, A.-R. Mohamed, G. Hinton, Sequence transduction with recurrent neural networks, arXiv preprint arXiv:1211.3711, 2012. https://arxiv.org/abs/1211.3711.

[15] A. Baevski, Y. Zhou, A. Mohamed, M. Auli, wav2vec 2.0: A framework for self-supervised learning of speech representations, Adv. Neural Inf. Process. Syst. (NeurIPS), 2020.

[16] A. Radford, J. Kim, T. Xu, G. Brockman, C. McLeavey, S. Agarwal, I. Sutskever, Whisper: Robust Speech Recognition via Large-Scale Weak Supervision, OpenAI, 2022. https://openai.com/research/whisper.

[17] A.D. Shah, M.G. Puspanjali, M.H. Alvi, Automatic speech recognition: A survey of deep learning techniques and approaches, Int. J. Comput. Cogn. Eng. 3 (2024) 21–37. https://www.sciencedirect.com/science/article/pii/S2666307424000573.

[18] Y. Zhang, C. Wu, G. Liu, F. Wei, S. Liu, End-to-end speech recognition: A survey, arXiv preprint arXiv:2303.03329, 2023. https://arxiv.org/abs/2303.03329.

[19] S. Li, B. Li, T.N. Sainath, Y. Wu, K.C. Sim, Y. Zhang, On the comparison of popular end-to-end models for large scale speech recognition, arXiv preprint arXiv:2005.14327, 2020. https://arxiv.org/abs/2005.14327.

[20] Y. Wang, S. Liu, J. Li, Y. Gong, A comparative study on Transformer vs RNN in speech applications, arXiv preprint arXiv:1909.06317, 2019. https://arxiv.org/abs/1909.06317.

[21] T. Wang, Y. He, T.N. Sainath, Z. Chen, C.-C. Chiu, Transformer in action: A comparative study of Transformer-based acoustic models, arXiv preprint arXiv:2010.14665, 2020. https://arxiv.org/abs/2010.14665.

[22] S.G. Bhable, R.R. Deshmukh, C.N. Kayte, Comparative analysis of automatic speech recognition techniques, in: Proc. Int. Conf. Appl. Mach. Intell. Data Anal. (ICAMIDA), Adv. Comput. Sci. Res., Aurangabad, India, May 2023, pp. 897–904. https://doi.org/10.2991/978-94-6463-136-4_79.

[23] V.J. Naik, N.B.F. Dessai, Speech recognition techniques: A comparative review, Int. J. Sci. Res. Eng. Manag. (IJSREM), 2023.

[24] A.M. Elsharkawy, A.F.A. Dadi, Comparative study of CNN structures for Arabic speech recognition: AlexNet, ResNet, and GoogLeNet, Int. J. Speech Inf. Before, 2023.

[25] Reddit user ml_speech_tech, Transformer-AED achieved the best accuracy in both streaming and non-streaming mode, Reddit – r/speechtech, 2020. https://www.reddit.com/r/speechtech/comments/gv389x/.

[26] Reddit user deep_audio_dev, On a low latency voice assistant task, Emformer gets 24% to 26% relative word error rate reductions, Reddit – r/speechtech, 2020. https://www.reddit.com/r/speechtech/comments/jlfbkt/.

[27] A. Graves, A.-R. Mohamed, G. Hinton, Speech recognition with deep recurrent neural networks, in: Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP), 2013, pp. 6645–6649.

[28] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, Y. Bengio, Learning phrase representations using RNN encoder–decoder for statistical machine translation, arXiv preprint arXiv:1406.1078, 2014. https://arxiv.org/abs/1406.1078.

[29] A. Alpha Cephei Inc., Vosk Speech Recognition Toolkit, GitHub repository, 2025. https://github.com/alphacep/vosk-api.

[30] A. Radford, J.W. Kim, T. Xu, G. Brockman, C. McLeavey, S. Agarwal, I. Sutskever, Robust speech recognition via large-scale weak supervision, arXiv preprint arXiv:2212.04356, 2022. https://arxiv.org/abs/2212.04356.

[31] M. Alvi, R. Gupta, P. Sinha, A privacy-preserving ASR framework for healthcare applications, IEEE Access 11 (2023) 45000–45015.

[32] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, MIT Press, 2009.