IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 24 November 2024, accepted 9 December 2024, date of publication 9 January 2025, date of current version 15 January 2025.

Digital Object Identifier 10.1109/ACCESS.2025.3527745

RESEARCH ARTICLE

Code-Switching ASR for Low-Resource Indic Languages: A Hindi-Marathi Case Study

HEMANT PALIVELA[1], MEERA NARVEKAR[2], DAVID ASIRVATHAM[3], SHASHI BHUSHAN[4], VINAY RISHIWAL[5], (Senior Member, IEEE), AND UDIT AGARWAL[6]

[1]Damac Properties, LLC, Dubai, United Arab Emirates
[2]Department of Computer Engineering, D. J. Sanghvi College of Engineering, Mumbai 400056, India
[3]Asia School of Business, Kuala Lumpur 50480, Malaysia
[4]Department of Computer and Information Sciences, Universiti Teknologi PETRONAS, Seri Iskandar, Perak 32610, Malaysia
[5]Department of CSIT, MJP Rohilkhand University, Bareilly 243006, India
[6]Department of CSA, RBMI Group of Institutions, Bareilly 243001, India

Corresponding authors: David Asirvatham (david.asirvatham@asb.edu.my) and Vinay Rishiwal (vrishiwal@mjpru.ac.in)

This work was supported by Asia School of Business, Kuala Lumpur, Malaysia.

ABSTRACT This work examines the development of Automatic Speech Recognition (ASR) systems for low-resource languages, focusing on Hindi and Marathi, particularly in multilingual and code-switching environments. ASR systems, which convert spoken language into text, face significant challenges when applied to low-resource languages with limited data for training models. These challenges are exacerbated in multilingual settings, particularly during code-switching, where speakers alternate between languages within a conversation. This paper underscores the current state of ASR for Indic languages, highlighting linguistic complexities such as diverse sentence structures, phonetic variety, and frequent code-switching. Code-switching introduces additional challenges, as ASR systems must rapidly identify language boundaries and adapt to linguistic shifts. Present systems struggle to perform adequately with code-switched data due to the complexity of phonetic structures and the lack of comprehensive, annotated speech corpora. This work critically evaluates current methods and proposes improvements using modern deep-learning techniques to address the primary challenges in developing efficient ASR models for Hindi and Marathi. Moreover, performance comparisons of monolingual, bilingual, and multilingual ASR systems indicate that multilingual approaches are more effective in managing linguistic diversity. The efficacy of these systems can be evaluated using performance metrics such as the Phoneme Error Rate (PER) and the Word Error Rate (WER), which assess word recognition accuracy.

INDEX TERMS Code-switching, automatic speech recognition, low-resource languages, Hindi, Marathi, Indic languages, language modeling.

I. INTRODUCTION

Automatic speech recognition (ASR) technology forms the basis of modern human-computer interfaces. It lets computers understand spoken English and convert it into text. This ability is vital for many applications, from voice-activated systems to virtual assistants to transcription services. From healthcare to education, entertainment to customer service, ASR greatly affects many different sectors and enhances user convenience and accessibility worldwide. In the educational sector, ASR aids in language acquisition. Mainly, when we deal with students with disabilities, these tools help them to engage more effectively with learning materials [1]. ASR can also be utilized in language learning as it provides immediate feedback on pronunciation, fluency, etc. ASR transcribes medical records in healthcare, thereby reducing administrative labour [2]. ASR has also been included in telemedicine systems so that documentation [7] and

The associate editor coordinating the review of this manuscript and approving it for publication was Maria Chiara Caschera.

© 2025 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

multilingual, real-time patient-provider communication may be obtained. ASR enhances automated call handling systems in customer service, enhancing the delivery of services and customer satisfaction [3]. ASR technology also enables the entertainment industry to design interactive games and voice-activated systems that inspire creativity and provide consumers with immersive experiences [4].

ASR systems first surfaced as attempts to let machines understand spoken language midway through the 20th century. These simple, early systems were limited and only identified small vocabularies or numerical inputs. Many of the early ASR systems were built on statistical models, including Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs), which greatly influenced the development of ASR technology and employed advances in computational capability. Deep learning techniques—especially Recurrent Neural Networks (RNNs) and the more recently evolved Transformer architecture—have revolutionized ASR systems as analytical approaches progress. These complex models have enabled multilingual ASR systems, which can recognize several languages inside a single framework, so enhancing accuracy and dependability in ongoing speech recognition. ASR systems have developed remarkably in processing monolingual speech for high-resource languages like English. Still, building decent ASR systems for low-resource languages can be challenging, particularly in multilingual and code-switching settings. Multilingualism is the norm in India, thus occasionally people code-switch—that is, alternate speaking Hindi and Marathi inside one conversation. Speakers of different languages mix so naturally in an audio stream that it becomes difficult to distinguish one from the others. For instance, Hindi and Marathi have different vocabulary and pronunciation rules, even if their phonetic characteristics and sentence structures are the same. A primary challenge in Hindi-Marathi code-switching is fast and fluid transitions between the two languages—often occurring inside a single sentence or phrase. This makes it difficult for ASR systems to identify language boundaries precisely; the low availability of code-switched annotated datasets aggravates this task.

Moreover, demanding code-switching requires sophisticated approaches to adapt to linguistic changes and mark language boundaries dynamically. Through multilingual models and transfer learning techniques, this work lowers resource constraints and increases recognition accuracy for low-resource languages, including Hindi and Marathi. The work stresses its experimental rigour and distinguishes itself from review-oriented studies by emphasizing quantitative performance measures, including Word Error Rate (WER) and Phoneme Error Rate (PER).

A. SCOPE OF THIS STUDY

This work aims to improve ASR systems managing code-switching between Hindi and Marathi, two Indic languages with data shortage for model training. This work investigates the difficulties with code-switching, which is common in multilingual countries where speakers migrate between languages in one conversation. This work addresses challenges to create accurate ASR models, including syntactic variations, dialectal diversity, and phonetic variability—linguistic complexity. Apart from comparing conventional and modern ASR methods, including Deep Neural Networks (DNNs) and Transformer architectures, it also looks at the generation and evaluation of necessary speech corpora for Hindi and Marathi. Critical criteria such as Word Error Rate (WER) and Phoneme Error Rate (PER) help partially evaluate the ASR systems. By focusing on these two languages, the study hopes to add to the larger field of ASR for low-resource languages by providing insights and strategies pertinent to other multilingual and under-resourced language environments.

This study also addresses specific questions such as:

- How do multilingual ASR systems compare with monolingual and bilingual systems regarding accuracy and robustness?
- What strategies can effectively mitigate challenges related to low-resource datasets, such as transfer learning and data augmentation?

B. CONTRIBUTIONS OF THE STUDY

The main contributions of this study are

1) Investigates the development of ASR systems for low-resource languages, specifically in multilingual and code-switching contexts, focusing on Hindi and Marathi.
2) Identifies challenges in ASR for low-resource languages, such as limited data for model training and linguistic complexities.
3) Compares the performance of monolingual, bilingual, and multilingual ASR systems, demonstrating the superiority of multilingual approaches in handling linguistic diversity.
4) Measures ASR system effectiveness using performance metrics such as Word Error Rate (WER) and Phoneme Error Rate (PER), assessing word and phonetic recognition accuracy.
5) Highlights innovative techniques for enhancing dataset diversity, including data augmentation and transfer learning, to overcome resource constraints.

C. ORGANIZATION OF THE PAPER

The work follows: With an eye toward issues unique to low-resource languages, Section II thoroughly summarises current developments in ASR systems. It investigates how code-switching, a linguistic phenomenon in which speakers alternately between several languages, influences ASR performance. Emphasizing critical studies and their results, this section contrasts approaches used to control linguistic diversity in contexts with limited resources. Section III covers basic techniques, including transfer learning, data

augmentation, and semi-supervised learning for developing ASR systems for low-resource languages. Section IV addresses the difficulties in managing code-switching in ASR systems, especially concerning language identification and processing in different pronunciation and syntactic construction. Section V uses code-switching to list essential areas of interest for further ASR system research in low-resource language environments. Section VI outlines the key architecture of ASR systems designed for code-switching in low-resource languages. Section VII examines advanced feature extraction techniques tailored to these systems. Section VIII focuses on the speech corpora essential for training ASR models in low-resource contexts. Section IX highlights critical evaluation metrics and performance benchmarks. Section X introduces the proposed architecture and processing framework to address code-switching challenges, while Section XI concludes with key insights and future research directions for enhancing ASR in low-resource languages.

II. RELATED WORK

Several studies have explored methods to improve ASR for low-resource languages, focusing on various techniques and their limitations in addressing code-switching. Bartelds et al. [1] demonstrated performance enhancements in monolingual ASR through data augmentation but stressed the need for more diverse datasets, with no focus on code-switching. Similarly, Singh et al. [12] used self-training to improve Word Error Rate (WER) but faced challenges with dialectal variations, highlighting the model's limitations in managing multilingual inputs or code-switching. Reitmaier et al. [7] employed transfer learning between Hindi and Marathi, leveraging high-resource linguistic data to improve ASR in low-resource contexts, effectively addressing code-switching between the two languages, though further optimization was suggested.

Kim et al. [10] introduced semi-supervised learning, enhancing ASR accuracy for low-resource languages with Hindi as a base while recognizing the need for extensive model tuning to manage seamless code-switching in spontaneous speech. Gupta and Boulianne emphasized the scalability of multilingual ASR systems but identified gaps in effectively managing code-switching, especially in spontaneous or frequent transitions between languages. Min et al. [4] applied self-supervised learning using large-scale pre-trained models, which proved flexible and accurate in handling code-switching across multiple languages. However, the method's high computational demands remain a challenge. Reference [55] address the challenge of synthesizing speech for Kannada-English code-mixed scenarios, a common occurrence in multilingual societies like India. They propose a generative adversarial network (GAN) architecture that leverages monolingual Kannada utterances to generate code-mixed speech, effectively bridging the gap caused by the scarcity of code-mixed datasets. Their approach demonstrates the potential of GANs in producing natural-sounding code-mixed speech, contributing significantly to advancements in speech synthesis for low-resource, code-mixed languages.

In [56], it addresses the complexities of code-switching in automatic speech recognition (ASR) systems by modifying the ESPNet framework. Their approach integrates multilingual data to enhance ASR performance in code-switched environments, mainly focusing on Indian languages. The modified ESPNet model demonstrates improved accuracy in recognizing code-switched speech, highlighting the effectiveness of incorporating multilingual training data and advanced neural architectures in ASR systems. Reference [57] explores the linguistic diversity of the Indian subcontinent, which encompasses 1,369 languages and 13 scripts. They introduce a Common Label Set (CLS) based on phonetics to address the challenges of large vocabulary requirements in end-to-end multilingual speech synthesis frameworks. This approach reduces the synthesizer's footprint and enables rapid adaptation to new languages with similar phonotactics, facilitating seamless code-switching across 13 Indian languages and English in a native speaker's voice.

Reference [58] present Meta-Whisper, a framework designed to enhance ASR performance for low-resource languages through speech-based meta-in-context learning (Meta-ICL). Meta-Whisper leverages minimal paired speech and text examples to adapt to new languages, addressing the scarcity of labeled data in low-resource settings. The framework demonstrates significant improvements in ASR accuracy for low-resource languages, showcasing the potential of meta-learning techniques in overcoming data limitations in ASR systems. Bhable and Kayte [5] explored multilingual acoustic modelling, using shared linguistic features across languages to improve performance, and noted that diverse datasets are crucial for optimizing code-switching capabilities in low-resource environments. Alemu et al. [6] proposed a framework for monolingual ASR development. Still, they acknowledged the need for dialect-specific adaptations and a shift toward bilingual or multilingual models to address code-switching challenges. Zhao et al. [3] combined traditional and deep learning models in a hybrid ASR system, showing promise for handling code-switching in low-resource languages with shared phonetic or syntactic features. Finally, Nghia et al. [9] achieved performance improvements using data augmentation but noted the absence of code-switching capabilities, suggesting the need for multilingual approaches to enhance functionality in this area.

Recent studies, such as Sakti and Titalim [18], have explored the integration of wav2vec and multilingual ASR to improve the handling of phonetic diversity and resource constraints. Their work highlights the significance of self-supervised models in bridging gaps in annotated corpora. Additionally, Baevski et al. [38] demonstrated that pre-trained wav2vec 2.0 models could significantly enhance low-resource ASR performance, emphasizing their applicability in code-switching contexts. Despite these

TABLE 1. State-of-the-art: A comprehensive overview of ASR systems for low-resource languages.

Paper	ASR System Type	Methodology	Key Findings and Research Gaps	Code-Switching in Low-Resource Languages
Bartelds et al. [1]	Monolingual	Applied data augmentation techniques to enhance ASR performance for low-resource languages	Achieved notable performance gains but identified a lack of diverse datasets as a limiting factor for further improvement	The study does not address code-switching, focusing solely on monolingual ASR, which limits its application in multilingual environments
Singh et al. [12]	Monolingual	Employed self-training methods for low-resource ASR	Significant improvement in word error rate (WER) noted, but challenges remain regarding dialect variation in low-resource settings	Code-switching remains unaddressed; the model's monolingual nature necessitates additional adaptations to handle multilingual inputs
Reitmaier et al. [7]	Bilingual	Utilized transfer learning between high- and low-resource languages	Demonstrated improved ASR for Marathi by leveraging linguistic resources from Hindi, but highlighted the need for broader cross-lingual datasets	Effectively handles code-switching between the two languages, though further refinements are needed for optimal performance
Kim et al. [10]	Bilingual	Used semi-supervised learning for language expansion	Improved accuracy for low-resource languages with Hindi as the base, but required extensive model tuning	The system supports code-switching between Hindi and Marathi, but may require additional fine-tuning for seamless transitions in spontaneous speech
Min et al. [4]	Multilingual	Leveraged self-supervised learning with pre-trained models	Showed enhanced ASR flexibility and accuracy via large-scale multilingual training, though noted high computational demands	Effective in managing code-switching due to its wide training base, yet challenges remain regarding computational efficiency
Bhable and Kayte [5]	Multilingual	Focused on multilingual acoustic modelling for low-resource languages	Highlighted the potential of shared linguistic features across languages to improve ASR, but emphasized the necessity for language-specific datasets	Manages code-switching by utilizing shared features across languages, but requires diverse datasets to optimize performance in low-resource contexts
Alemu et al. [6]	Monolingual	Developed a framework for low-resource ASR systems	Proposed a methodology for ASR development in low-resource settings, calling for further work on dialect-specific adaptations	Does not support code-switching; transitioning to a bilingual or multilingual model is needed to address this issue in low-resource contexts
Zhao et al. [3]	Bilingual	Combined traditional models with deep learning to enhance low-resource ASR	Emphasized the effectiveness of hybrid approaches, particularly in under-resourced languages, but noted the need for further research	Hybrid methods show promise for handling code-switching, especially between languages with shared phonetic or syntactic traits
Nghia et al. [9]	Monolingual	Applied data augmentation to improve low-resource ASR	Notable performance gains were achieved, but the lack of diverse datasets was highlighted as a critical gap	The study does not address code-switching; integrating multilingual models or additional strategies is necessary to manage code-switching effectively

advancements, real-time language identification remains underexplored in ASR systems for code-switching scenarios, particularly between Indic languages such as Hindi and Marathi, where phoneme overlaps and syntactic shifts pose additional challenges.

Furthermore, the limitations in real-time language identification and phoneme overlap between Hindi and Marathi create unique hurdles for ASR systems. Current multilingual ASR methods have yet to fully address these challenges, leaving significant gaps in real-world applicability. This paper aims to bridge these gaps by emphasizing novel methodologies, including hierarchical language identification and hybrid modelling, to improve system robustness.

Table 1 summarizes key studies on monolingual, bilingual, and multilingual ASR systems, focusing on methodologies, findings, gaps in research, and insights on handling code-switching in low-resource languages.

III. BACKGROUND

This section provides a foundational overview of ASR, including its classification, an explanation of code-switching within ASR, and a brief discussion of the benefits and challenges associated with its implementation.

A. AUTOMATIC SPEECH RECOGNITION (ASR)

ASR systems are sophisticated technologies that convert spoken language into text. These systems utilize a combination of signal processing, machine learning, and linguistic models to interpret and transcribe human speech. The process typically involves several key stages: audio signal acquisition,

TABLE 2. Comparison of ASR systems: Evolution, methodologies, challenges, and use cases.

ASR Systems	Evolution	Methodologies	Challenges	Use Cases
Monolingual ASR [10], [12], [21], [33]	Developed primarily for high-resource languages; early versions like Dragon Dictate focused on languages like English. More recently, attention has shifted to low-resource languages.	Utilizes speech-to-text algorithms, language modelling, and acoustic modelling. Data augmentation and transfer learning are crucial for low-resource languages.	Data scarcity, dialectal variation, and linguistic complexity, particularly in low-resource languages like Hindi and Marathi.	Legal transcriptions, educational content creation, and media transcriptions.
Bilingual ASR [7], [9]	Evolved to handle two languages simultaneously, particularly in regions where code-switching is common.	Transfer learning, shared linguistic features, and language identification.	Resource imbalance between languages, code-switching, and increased model complexity.	Government services, communication in multilingual healthcare, and bilingual educational platforms.
Multilingual ASR [4], [5], [38], [18]	Designed for linguistically diverse regions; capable of transcribing speech in multiple languages and easily expanding to new ones.	Self-supervised learning, pre-trained models (e.g., wav2vec 2.0), multilingual acoustic and language models.	Handling code-switching, resource allocation across languages, and high computational complexity, particularly in resource-constrained environments.	Customer service, media and broadcasting, and public announcements in multiple languages.

feature extraction, acoustic modelling, language modelling, and decoding. The spoken input is initially captured through a microphone, and the audio signal is digitized. Subsequently, feature extraction algorithms identify relevant characteristics of the audio signal, such as pitch and frequency. Acoustic models then map these features to phonemes, the fundamental sound units in a language. Language models enhance transcription accuracy by considering the context in which words appear, allowing the system to predict and correct for possible misinterpretations. Finally, a decoding algorithm synthesizes this information to generate the corresponding text output. Through continuous advancements in deep learning and neural networks, ASR systems are becoming increasingly accurate and capable of handling diverse accents and languages, paving the way for applications in various domains, including healthcare, customer service, and accessibility.

Feature extraction, acoustic modelling, and decoding are critical in multilingual speech. Feature extraction techniques like Mel-Frequency Cepstral Coefficients (MFCCs) and spectrogram analysis capture the key properties of the speech signal while filtering noise. Acoustic models, whether traditional (HMMs and GMMs) or deep-learning-based (DNNs and Transformers), map these features to phonemes, managing phonetic diversity and cross-lingual nuances. Decoding integrates outputs from acoustic and language models to generate coherent text, often facing challenges with abrupt language transitions in multilingual scenarios.

1) CLASSIFICATION OF ASR SYSTEMS

Every monolingual, bilingual, and multilingual ASR system corresponds to a different linguistic environment [12], [32]. ASR systems in monolingual languages are Designed to monitor speech in one language. Language-specific acoustic and linguistic models help refine these systems for outstanding accuracy in one language. They are pretty common in environments where a single language is spoken or better speech recognition in a given language is required [10]. Aimed to document two languages [9], bilingual ASR systems help speakers of two languages, as in bilingual neighbourhoods or communities, alternate between them. The design of bilingual ASR systems must be adaptable enough to allow code-switching, the phenomena of speakers switching between languages in a single conversation, which complicates their design and calls for advanced language models able to recognize and process both languages proficiently. Multilingual ASR systems improve the features of bilingual systems using speech recognition in several languages. These systems coordinate several language inputs inside one conversation or in other contexts. Since they enable language barrier-free communication, multilingual ASR systems are fundamental for global settings. Usually based on community acoustic models, transfer learning, or multilingual datasets to raise efficacy across several languages, the architecture of these systems [4]. Since they offer scalable solutions for speech recognition across many languages, multilingual ASR systems are becoming increasingly common as the demand for linguistic diversity increases.

Comparative analysis highlights the distinctions: monolingual systems excel in single-language environments with high resource availability but fail in multilingual or code-switching contexts. Bilingual systems handle two languages and are valuable for managing frequent transitions but struggle with less-resourced pairs. Multilingual systems leverage transfer learning and multilingual data, offering flexibility and scalability across diverse linguistic environments but requiring significant computational resources and training complexity.

Table 2 compares different types of ASR systems, their evolution, methodologies, challenges, and use cases.

B. CODE SWITCHING

In ASR, code-switching is alternating two or more languages or dialects by speakers inside one utterance or conversation. This phenomenon is typical in multilingual communities

when people naturally migrate between languages depending on social context, interlocutors, or even the topic of discussion. Code-switching ASR will allow the development of language technologies that fit real-world linguistic behaviour to more successfully serve various populations by enabling virtual assistants, transcription systems, and customer service tools. In bilingual ASR systems especially, understanding and handling code-switching is crucial since it allows the system to regulate language transitions, thus enhancing user interaction and experience. For instance, a speaker may utter phrases such as "Mai kal office jaunga" – a mix of Hindi and English—or "Me udya office jaeen."— a blend of Marathi and English. In these cases, language switches happen suddenly within the same sentence. Such abrupt transitions necessitate real-time language identification and adaptive adjustments to the acoustic models. Additionally, the frequent incorporation of borrowed vocabulary, variations in sentence structure, and diverse syntactic rules further complicate the accurate recognition and interpretation of code-switched speech.

However, creating robust ASR systems that manage code-switching is difficult for several reasons. First, especially in low-resource language pairs, there is a scarcity of well-annotated datasets reflecting the complexity of code-switched speech. Furthermore, sudden changes in lexical, syntactic, and phonetic structures across languages call for the ASR models to dynamically adapt to various language rules, so confusing standard models are usually educated on monolingual data. This becomes quite difficult in bilingual ASR, where subtle pronunciation variations, accent changes, and speaker-specific language preferences further compromise the model's ability to record speech faithfully. Furthermore, increasing word errors and negatively affecting recognition accuracy result from the coexistence of words with similar phonetic traits among languages. Moreover, the linguistic background of one language could not match that of the other, thus ASR models have to control contextual disambiguation.

Advanced solutions include hierarchical acoustic modelling, which leverages multilingual corpora to generalize phonetic patterns across languages while maintaining language-specific adaptations. Learning from high-resource languages like Hindi enhances recognition in low-resource Marathi. Hybrid approaches combining attention mechanisms and transformer architectures address abrupt transitions by dynamically aligning contextual and phonetic representations.

IV. ASR SYSTEMS FOR LOW-RESOURCE LANGUAGES

Creating ASR systems for low-resource Indic languages such as Hindi and Marathi is essential but difficult. Spoken by millions, these languages show great linguistic diversity but lack the rich linguistic resources—such as annotated corpora, advanced language models [54], and thorough lexicons—needed to build robust ASR systems. By using efficient ASR technologies, one can greatly improve accessibility and help to preserve the linguistic and cultural legacy of these languages [50]. ASR integration in digital services, education, and government sectors is becoming increasingly sought for in areas where Hindi and Marathi are somewhat prevalent. ASR can offer voice-based learning tools in classrooms that increase non-English speaking students' access to materials.

ASR systems in these settings can also help bridge the digital divide by enabling access to online educational resources in native languages, increasing student engagement and improving learning outcomes.

Additionally, using ASR in governance can facilitate the creation of digital documents in regional languages, promoting the official use of local languages in administrative contexts [32]. Integrating ASR technology in these domains can help overcome linguistic barriers and ensure that technological advancements are inclusive and equitable. Moreover, ASR can be crucial in preserving endangered languages by creating digital archives of spoken language data for educational and cultural preservation purposes.

These archives could also be utilized for developing linguistic resources such as dictionaries, thesauri, and language-learning applications, further supporting language revitalization efforts.

A. CHALLENGES IN DEVELOPING ASR FOR LOW-RESOURCE LANGUAGES

Developing ASR systems for low-resource Indic languages presents challenges not usually found in high-resource languages. One of the main difficulties is not having enough big, annotated datasets necessary for training robust ASR models. Unlike high-resource languages like Mandarin or English, where significant volumes of transcribed speech data are readily available, low-resource languages often lack the necessary linguistic tools. This data gap seriously lowers the capacity to precisely teach models able to identify and transcribe speech in these languages.

1) **Insufficient availability of large annotated datasets:** Because there are few large-scale, high-quality annotated datasets, low-resource languages, including Hindi and Marathi, present great difficulty. This lack of data makes training and polishing robust and reliable ASR models difficult.
   By motivating varied contributions from speakers all around, crowdsourcing sites like Common Voice by Mozilla have shown encouraging results in tackling this problem.
2) **Linguistic diversity and dialectal variations:** Low-resource languages' broad spectrum of dialects and variations in phonetics and vocabulary complicate creating successful ASR systems. Accurate speech recognition depends on careful accounting for these linguistic subtleties.

Adding syntactic and phonetic patterns unique to dialects will improve ASR accuracy. For Marathi, for instance, dialectal changes, including tonal shifts, call for specific acoustic models.

3) **Constraints in technical infrastructure:** Developing and implementing ASR systems is hampered in many low-resource areas by the absence of consistent access to fast internet and advanced computational tools. These constraints limit our capacity to develop and apply such technologies properly.
Edge computing and lightweight ASR models optimized for low-resource devices can mitigate these limitations by providing offline capabilities.

Additionally, the technological infrastructure for developing and deploying ASR systems in low-resource languages is often inadequate. Many speakers of these languages reside in regions with limited internet access, and this lack of infrastructure can significantly impede both the advancement and widespread adoption of ASR technology in such areas.

B. STRATEGIES FOR OVERCOMING CHALLENGES IN LOW-RESOURCE ASR DEVELOPMENT

To tackle the difficulties in creating ASR systems for languages with limited resources, scholars have investigated a range of approaches, such as semi-supervised learning, transfer learning, and data augmentation [17]. By producing variants of already-existing data, data augmentation refers to intentionally increasing the amount of the training dataset [27]. ASR models' robustness can be improved by employing strategies like pitch shifting, noise injection, and speed perturbation [29] to produce more training data [6]. Table 3 presents a detailed overview of the challenges and corresponding strategies for ASR systems in the context of low-resource languages.

Recent advancements in synthetic data generation have further enhanced data augmentation strategies. For instance, TTS (Text-to-Speech) systems are being used to simulate real-world accents and dialects, enabling more diverse training sets.

C. ETHICAL CONSIDERATIONS IN ASR FOR LOW-RESOURCE LANGUAGES

As ASR technology advances, addressing the ethical implications of deploying these systems in low-resource languages is essential. A key concern is the potential for bias in ASR models, which can occur when the training data does not adequately represent the full diversity of the speaker population. For instance, if an ASR system is trained predominantly on speech data from male speakers, its accuracy may decline when transcribing speech from female speakers. Similarly, insufficient representation of various dialects in the training data can lead to inaccuracies in recognizing speech from speakers of those dialects [8].

Ensuring inclusivity requires targeted efforts to gather balanced datasets representing diverse demographics, including gender, age groups, and rural populations.

V. CODE-SWITCHING IN LOW-RESOURCE LANGUAGES

Code-switching, which refers to alternating between two or more languages within a conversation, is a frequent phenomenon in multilingual communities, especially in regions where low-resource languages are prevalent. This linguistic behaviour presents significant challenges for ASR systems, mainly when working with languages that have limited available data. To tackle the complexities of code-switching in such language contexts, this section provides a comparative analysis of monolingual, bilingual, and multilingual ASR systems, highlighting their functional differences, technological approaches, strengths, and limitations.

A. OPERATIONAL DIFFERENCES IN HANDLING CODE-SWITCHING

Table 4 illustrates the operational distinctions between monolingual, bilingual, and multilingual ASR systems in handling code-switching, particularly in low-resource Indic languages. While monolingual systems fail to manage transitions, bilingual systems offer partial solutions by leveraging language identification, and multilingual systems generalize better across shared linguistic features.

1) MONOLINGUAL ASR SYSTEMS

Code-switching presents intrinsic challenges for monolingual ASR systems. These systems cannot recognize or transcribe speech that switches between languages since they are only meant to analyze and record speech in a single language. Monolingual ASR systems will likely fail in multilingual environments where speakers may move between high-resource languages like Hindi and low-resource languages like Marathi because they cannot dynamically adapt to the language shifts [9]. A monolingual ASR system trained in a low-resource language will find it difficult to transcribe speech in a high-resource language when code-switching occurs reliably and vice versa. This limitation [25] makes monolingual systems unsuitable for environments where code-switching is prevalent, as they will either ignore or misinterpret speech segments in the non-target language.

2) BILINGUAL ASR SYSTEMS

Due to their intrinsic ability to handle two languages simultaneously, bilingual ASR systems have an advantage when handling code-switching scenarios. To apply the proper language model when a speaker changes from one language to another, these systems usually have language identification components [7]. In low-resource languages, bilingual ASR systems can utilize the resources of a high-resource language to enhance the low-resource language's accuracy. For example, the rich linguistic resources available for Hindi can be leveraged by a bilingual ASR system supporting

TABLE 3. Challenges and strategies for ASR systems in low-resource languages.

Challenge	Strategies Employed
Limited Availability of Annotated Datasets	• **Data Augmentation [18]**: Techniques such as speed perturbation, noise injection, and pitch shifting artificially expand the size of existing datasets, introducing variability and enhancing the robustness of ASR models. • **Transfer Learning [7]**: Utilizes pre-trained models from high-resource languages, fine-tuned with data from low-resource languages, thus mitigating reliance on large annotated datasets. • **Semi-Supervised Learning [14]**: Combines a smaller labelled dataset with a larger set of unlabeled data. Pseudo-labels generated by the model are used for further training, reducing the need for extensively annotated datasets.
Linguistic Diversity and Dialectal Variations	• **Multilingual ASR Systems [28]**: These systems recognize speech across multiple languages, including dialectal differences. By generalizing across languages and dialects, they enhance the system's ability to manage linguistic diversity. • **Transfer Learning and Dialect-Specific Fine-Tuning [10]** : Fine-tuning ASR models with dialect-specific data enables adaptation to phonetic and lexical variations, improving recognition of regional dialects.
Technical Infrastructure Constraints	• **Cloud-Based ASR and Optimized Lightweight Models [36]**: Cloud-based ASR solutions and lightweight models are optimized for environments with limited technical resources, enabling ASR deployment in regions with inadequate infrastructure. • **Transfer Learning and Semi-Supervised Learning [10], [30]**: These methods address data limitations and reduce computational demands by minimizing the need for training from scratch, facilitating deployment in resource-constrained environments.

both Marathi, a low-resource language, and Hindi, a high-resource language, to improve overall performance. However, the system's capacity to accurately recognize and transition between languages during code-switching determines how effective this strategy will be. Inaccurate transcriptions can be produced by language identification mistakes or switching delays, especially for the language portions with limited resources [10].

3) MULTILINGUAL ASR SYSTEMS

The most advanced code-switching handling ASR systems are multilingual ones, mainly when low-resource languages are involved. Within a single framework, these systems are made to recognize and transcribe speech in various languages. Multilingual ASR systems get more skilled at managing dynamic language switching by learning to identify shared linguistic features across languages through training on multilingual datasets. Multilingual ASR systems can provide more accurate transcriptions than those produced by monolingual or bilingual systems in situations where code-switching involves a combination of high- and low-resource languages. Even when one of the languages contains less linguistic data, they are better suited to handle the challenges of code-switching because of its capacity to generalize across languages. Multilingual ASR systems are perfect for multilingual and low-resource environments because of their cross-lingual capacity, which enables them to maintain excellent transcription accuracy across multiple languages [4].

B. LIMITATIONS IN CODE-SWITCHING SCENARIOS

While different types of Automatic Speech Recognition (ASR) systems offer unique strengths, they also exhibit significant limitations, mainly when applied to code-switching in low-resource languages. Code-switching, the practice of alternating between languages within a conversation, presents a challenge for ASR systems, especially in linguistically diverse environments with limited training data.

1) LIMITATIONS OF MONOLINGUAL ASR SYSTEMS

a: INABILITY TO HANDLE CODE-SWITCHING

A fundamental limitation of monolingual ASR systems is their incapacity to recognize and process multiple languages. These systems are inherently designed for a single language, which restricts their applicability in multilingual settings where speakers frequently switch between languages within a conversation. As monolingual ASR systems cannot identify and accurately transcribe segments in different languages, their utility diminishes significantly in such scenarios, particularly in regions with a high prevalence of code-switching. This inadequacy impairs the system's ability to capture the full range of spoken language, thus limiting its effectiveness in real-world multilingual contexts [9].

b: LIMITED FLEXIBILITY

Beyond their inability to handle code-switching, monolingual ASR systems exhibit limited adaptability to variations within a language, such as dialectal differences, regional accents, or language evolution. This lack of flexibility is especially

TABLE 4. Operational differences in handling code-switching across ASR systems.

ASR Type	Focus	Handling Code-Switching	Low-Resource Language Challenges	Operational Strengths	Operational Limitations
Monolingual	Recognizes a single language	Struggles significantly with code-switching; often fails to recognize words from other languages	Limited data amplifies the problem, as systems cannot adapt to unfamiliar language inputs	High accuracy for single language with sufficient data	Performance degradation when encountering unrecognized languages or code-switching
Bilingual	Recognizes and processes two languages	Performs better than monolingual systems; can handle code-switching between trained languages	Requires sufficient data for both languages; struggles with languages that are typologically different or underrepresented	More adaptable to language switches; improved accuracy compared to monolingual systems	Requires balanced bilingual data and still struggles with underrepresented languages
Multilingual	Capable of recognizing multiple languages	Best suited for handling code-switching due to exposure to multiple languages during training	Handles low-resource languages better by leveraging knowledge from high-resource languages, but still struggles with pairs of low-resource languages	High adaptability, generalizes across languages, better performance for low-resource languages	Requires complex training and data from several languages for optimal performance

problematic in areas with extensive linguistic diversity, where significant phonetic or syntactic variation may occur even within a single language. Monolingual systems, rigidly constrained to predefined language models, often fail to accommodate such variations, leading to lower transcription accuracy and reduced usability in heterogeneous linguistic environments.

While monolingual ASR systems struggle to manage code-switching due to their inability to handle multiple languages, bilingual ASR systems attempt to address these limitations by integrating language identification mechanisms and leveraging resources from high-resource languages to support low-resource counterparts.

2) LIMITATIONS OF BILINGUAL ASR SYSTEMS

a: CHALLENGES IN LANGUAGE IDENTIFICATION

Bilingual ASR systems, though designed to support two languages, face considerable challenges in accurately identifying and transitioning between languages within a conversation. The process of language identification—determining which language is being spoken at any given moment—becomes increasingly difficult when there is limited training data for one or both languages, as is often the case with low-resource languages. These systems may erroneously switch languages or misinterpret speech, resulting in transcription errors. Such challenges are particularly acute in situations of spontaneous code-switching, where speakers may seamlessly alternate between languages without clear contextual or syntactic cues, further complicating the system's ability to maintain accurate recognition [10].

b: RESOURCE IMBALANCE

A persistent issue with bilingual ASR systems is the imbalance of available linguistic resources. One of the two languages may often have a wealth of data and linguistic models, while the other may be severely underrepresented. This disparity leads to an unequal performance, where the system operates efficiently for the high-resource language but struggles to maintain the same level of accuracy for the low-resource language. During code-switching, this imbalance becomes more pronounced, as the system may fail to adequately process speech segments in the less represented language, leading to incomplete or inaccurate transcriptions, thereby limiting the system's effectiveness in practical applications [7].

Although bilingual ASR systems improve upon monolingual systems by accommodating two languages, they face language identification and resource imbalance challenges. Multilingual ASR systems go further by generalizing across multiple languages and offering better scalability and adaptability for dynamic code-switching scenarios.

3) LIMITATIONS OF MULTILINGUAL ASR SYSTEMS

a: COMPUTATIONAL DEMANDS

Multilingual ASR systems, designed to recognize and transcribe multiple languages, require substantial computational resources to operate effectively. These systems often rely on complex neural network architectures, such as transformer models, that demand high processing power and large memory capacities to manage multiple language models simultaneously. In low-resource environments, where

access to advanced computational infrastructure is often limited, deploying such systems becomes infeasible. The high computational cost poses a significant barrier regarding hardware requirements and the energy consumption and latency that may arise during real-time speech processing [4].

b: CHALLENGES IN BALANCING LANGUAGES

One of the critical challenges in multilingual ASR systems is the equitable distribution of resources across the languages they support. When a system is trained on multiple languages, there is a risk that the model may overfit to high-resource languages, for which ample training data is available, while underperforming on low-resource languages. This imbalance can reduce accuracy for low-resource languages, particularly in code-switching scenarios where the system may struggle to switch between languages seamlessly. Balancing language performance is particularly acute when the linguistic diversity within the system is significant, as ensuring uniform model performance across all supported languages becomes increasingly tricky.

Despite their computational demands and challenges in balancing languages, multilingual ASR systems remain the most robust solution for managing code-switching, particularly in environments with diverse linguistic needs. Their inherent advantages in cross-lingual generalization and adaptability make them ideal for addressing code-switching complexities in low-resource languages.

C. ADVANTAGES OF MULTILINGUAL ASR IN INDIC CONTEXTS

Multilingual ASR systems are particularly advantageous in managing the linguistic diversity of Indic languages. For instance, languages such as Tamil, Telugu, and Gujarati share phonetic and syntactic similarities that multilingual models can leverage for cross-lingual generalization. This approach ensures higher transcription accuracy even in complex code-switching scenarios with high-resource languages like Hindi or English.

The advantages of multilingual ASR systems discussed here highlight their potential to transform code-switching handling in low-resource environments. However, their adaptability and scalability must be examined further to understand how they can be optimized for diverse linguistic landscapes.

1) CROSS-LINGUAL GENERALIZATION

a: EXPLOITATION OF SHARED LINGUISTIC FEATURES

One of the critical strengths of multilingual ASR systems lies in their ability to leverage shared linguistic features across different languages. These systems utilize common phonetic, syntactic, and semantic elements between languages, which enhances their performance in cross-lingual tasks. By drawing on these universal linguistic patterns, multilingual ASR systems can significantly improve their accuracy in detecting and processing speech that switches between languages, such as in code-switching scenarios. This capacity allows them to handle complex speech inputs, ensuring better recognition and transcription in multilingual contexts.

b: EFFICIENT RESOURCE UTILIZATION

Multilingual ASR systems also promote resource efficiency by reducing the need for separate, language-specific models. Instead of building distinct ASR models for each language, these systems train on multilingual datasets, reducing computational costs and storage requirements. This approach simplifies the system architecture and makes it feasible to deploy high-performance ASR systems in resource-limited environments, typical in low-resource language settings where code-switching is more prevalent. As a result, multilingual systems can perform effectively without the need for extensive hardware or large-scale computational resources [4].

2) SCALABILITY ACROSS DIVERSE LINGUISTIC CONTEXTS

Scalability is a core strength of multilingual ASR systems. Training on a shared linguistic dataset allows these systems to integrate new languages or dialects with minimal overhead. For instance, adding Gujarati to an ASR model trained on Hindi and Marathi requires limited retraining, leveraging shared features like consonantal clusters and vowel length distinctions. This adaptability makes multilingual systems ideal for real-world environments where linguistic needs are dynamic.

a: ADAPTING TO DIVERSE LINGUISTIC CONTEXTS

The scalability of multilingual ASR systems also extends to their ability to adapt to a wide range of linguistic and sociolinguistic contexts. Whether in customer service, education, or the media, these systems can handle varied linguistic inputs, including regional accents, colloquialisms, and code-switching between formal and informal registers. Their ability to provide accurate and consistent transcriptions across multiple languages and dialects ensures their utility in diverse professional settings where communication often involves a mix of languages and cultural nuances. Multilingual ASR systems effectively provide consistent performance and reliability in highly varied environments by managing these linguistic variations effectively.

3) ADAPTABILITY

a: DYNAMIC LANGUAGE SWITCHING

Code-switching, a common phenomenon in multilingual communities, often involves dynamic language switching during conversations, where speakers alternate between languages fluidly. Multilingual ASR systems excel in handling this dynamic switching, ensuring that the system can accurately recognize and transcribe speech regardless of the language in use. This adaptability is critical in multilingual and low-resource settings, where the speakers may switch languages rapidly and without prior notice.

Multilingual ASR systems are uniquely positioned to offer reliable and precise transcription in such scenarios, making them indispensable for real-world applications that involve frequent code-switching.

b: ENHANCING SUPPORT FOR LOW-RESOURCE LANGUAGES

Another significant benefit of multilingual ASR systems is their ability to enhance performance in low-resource languages by drawing on data from high-resource languages. In multilingual environments characterized by high linguistic variability and code-switching levels, these systems ensure that even speakers of low-resource languages can benefit from accurate ASR technology. By transferring knowledge from high-resource language models, multilingual ASR systems can effectively support low-resource languages, addressing the challenge of linguistic diversity in environments where data for specific languages is scarce. This capability is crucial in ensuring that ASR systems are accurate and inclusive, providing equitable transcription services for speakers of all languages, including those with limited linguistic resources [4].

Having explored the operational distinctions between monolingual, bilingual, and multilingual ASR systems, the next step is to delve into their architectural frameworks to understand how they are designed to handle code-switching effectively.

VI. ARCHITECTURE OF ASR SYSTEMS FOR CODE-SWITCHING IN LOW-RESOURCE LANGUAGES

This section explores the architecture of Automatic Speech Recognition (ASR) systems for low-resource languages such as Hindi and Marathi, with particular attention to the challenge of code-switching. The discussion encompasses language modelling, acoustic front-end processing, acoustic modelling, and the decoding process, highlighting how each component is adapted to effectively capture the phonetic and linguistic nuances of Hindi and Marathi.

A. ACOUSTIC FRONT-END PROCESSING

In ASR systems, the first phase—acoustic front-end processing—plays a pivotal role by transforming unprocessed audio signals into a set of representative features essential for accurate speech recognition in later stages of the system. During this phase, various signal processing techniques capture distinctive acoustic characteristics, such as pitch, formants, and spectral information. Traditional feature extraction methods such as Mel-Frequency Cepstral Coefficients (MFCCs) and Perceptual Linear Prediction (PLP) play a significant role in ASR systems for Hindi and Marathi. MFCCs capture the spectral properties of speech, which are essential for distinguishing phonemes in these languages. PLP, on the other hand, emphasizes perceptual aspects of speech, aligning more closely with human auditory perception. These techniques are essential for low-resource languages, allowing ASR systems to maximize recognition performance despite limited data. These features are the foundation for the model's ability to recognize and distinguish between phonetic units within the speech signal. However, acoustic front-end processing presents unique challenges for low-resource languages such as Marathi and Hindi, primarily due to the scarcity of large-scale annotated datasets necessary for training and refining robust models. This limitation is further compounded by the rich phonetic diversity and dialectal variations that characterize these languages. Such diversity often leads to variations in pronunciation, accent, and local speech patterns, which traditional feature extraction methods do not capture easily. Adapting feature extraction methods to account for dialectal variations in Hindi and Marathi is critical. By incorporating phoneme-level features, such as pitch contours and formant frequencies, ASR systems can improve the recognition of regional accents and pronunciation patterns. These features help address variability in phonetic structures, enhancing the system's ability to generalize across dialects while maintaining accuracy in code-switching scenarios. Consequently, the extracted acoustic features may fail to adequately capture the subtle phonetic distinctions required for precise recognition, particularly in multilingual contexts where code-switching between languages frequently occurs.

Code-switching introduces further complexity, as speakers may alternate between languages within a single utterance, blending phonetic and syntactic structures that challenge conventional ASR models. This variability necessitates the development of advanced, adaptive feature extraction methodologies capable of accounting for the dynamic phonetic environments present in these languages. Therefore, to address these challenges, it is imperative to design robust acoustic front-end systems that can effectively generalize across diverse linguistic and phonetic contexts, ensuring greater accuracy and consistency in recognising low-resource languages.

B. ACOUSTIC MODELING

Acoustic modelling establishes a relationship between features extracted from an audio signal and the corresponding phonetic units, such as phonemes. This technique is crucial for speech recognition systems, as it directly impacts their ability to interpret spoken language correctly. In low-resource languages like Marathi and Hindi, acoustic modelling faces significant challenges. One major issue is the limited availability of training data, which hinders the development of robust models. Additionally, accurately capturing the rich phonetic diversity inherent in these languages poses a further challenge, making it essential to devise strategies that ensure comprehensive phonetic representation despite data constraints.

1) TRADITIONAL ACOUSTIC MODELS

Traditional acoustic models in Automatic Speech Recognition (ASR) systems aim to map the relationship between the audio signal and the corresponding linguistic units, such

as phonemes, words, or phrases. These models rely on features extracted from the audio, like Mel-frequency cepstral coefficients (MFCCs), to capture speech characteristics. A core component of traditional ASR systems is the Hidden Markov Model (HMM), which provides a probabilistic framework for modelling the sequence of phonetic units in speech. As the foundation of conventional ASR systems, HMMs excel at simulating the temporal dynamics of speech by representing phonetic units as a sequence of hidden states. In low-resource languages like Hindi and Marathi, HMMs are commonly employed to capture the order and structure of speech sounds [3]. However, their performance highly depends on the quality and volume of available training data. In low-resource scenarios, such as those involving Hindi or Marathi, HMMs struggle to adequately represent the phonetic variability, particularly in code-switching instances where the system must transition between distinct phonetic contexts.

Gaussian Mixture Models (GMMs) are also frequently used alongside HMMs to model the distribution of acoustic features for each phonetic unit [7]. For languages like Hindi and Marathi, GMMs help account for the variability in pronunciation and accent by approximating the complex acoustic patterns in the speech data. Each phonetic unit is represented as a mixture of several Gaussian distributions, which allows GMMs to capture subtle variations in speech. However, like HMMs, GMMs require extensive training data to generalize well.

Insufficient data limits the model's ability to handle diverse speech conditions in low-resource environments, including code-switching between languages with different phonetic structures. While traditional models such as HMMs and GMMs have been widely used, their limitations in handling low-resource languages underscore the need for better data-efficient approaches to managing phonetic diversity and language-switching scenarios.

2) DEEP LEARNING-BASED ACOUSTIC MODELS

Deep Neural Networks (DNNs) have significantly transformed the field of acoustic modelling, offering a robust framework for learning and representing complex patterns within speech data. For languages like Hindi and Marathi, DNNs capture intricate phonetic features that often elude traditional models, enhancing speech recognition accuracy. However, in low-resource environments, the performance of DNNs is constrained by the limited availability of extensive labelled datasets, which are crucial for practical training [51]. The challenge becomes more pronounced in code-switching between Hindi and Marathi. DNNs must be fine-tuned using multilingual datasets that can accommodate the distinct phonetic and linguistic variations across both languages. This enables the model to identify and transition between them during recognition tasks accurately [10].

3) LONG SHORT-TERM MEMORY NETWORKS

Recurrent Neural Networks (RNNs), particularly the Long Short-Term Memory (LSTM) variant, are highly effective for modelling temporal sequences, such as speech. LSTMs are particularly advantageous in Automatic Speech Recognition (ASR) systems dealing with code-switching scenarios, where rapid transitions between languages like Hindi and Marathi can occur. This is due to their ability to capture and maintain long-range dependencies within the speech signal, making them well-suited for managing variable phonetic contexts. However, like Deep Neural Networks (DNNs), LSTMs require substantial training data to perform optimally, posing challenges in resource-constrained environments.

A significant challenge in developing effective acoustic models for low-resource languages is the limited availability of annotated datasets. This constraint hinders the performance of both traditional and deep learning-based models, particularly in code-switching scenarios, where systems must dynamically adjust to different phonetic environments [51]. To address this issue and enhance model performance, especially in resource-scarce conditions, researchers are exploring various techniques, such as data augmentation and transfer learning, to compensate for the lack of labelled data [4].

C. LANGUAGE MODELING

Language models play a crucial role in predicting word sequences within ASR systems. Incorporating dialect-specific adaptations into language models is essential for enhancing recognition in low-resource contexts. By integrating phoneme-level features and regional syntactic variations, models can better predict word sequences in dialect-rich environments. For example, Marathi dialects exhibit distinct morphological patterns that, when explicitly modelled, can significantly reduce transcription errors in ASR systems. These adaptations are especially valuable for managing code-switching between Hindi and Marathi, as they ensure smoother transitions and more accurate word predictions.

Specifically, for languages like Hindi and Marathi, these models need to incorporate the unique linguistic features of each language. This includes understanding their syntax, morphological structures, and the frequent occurrence of code-switching, where speakers alternate between languages or dialects. Addressing these linguistic nuances is essential for improving the accuracy and robustness of ASR systems in multilingual contexts.

1) N-GRAM MODELS

A traditional approach to language modelling, known as N-gram models, predicts the likelihood of a word based on the preceding N-1 words. While N-gram models can effectively capture common word sequences in languages such as Hindi and Marathi, they face significant limitations, particularly their inability to generalize to previously unseen word combinations. This constraint becomes incredibly challenging in code-switching scenarios, where the model must navigate transitions between different linguistic contexts and adapt to mixed-language inputs, often leading to reduced accuracy in prediction and transcription quality [7]. Reliable

probability estimation of word sequences using N-gram models necessitates large amounts of textual data. For low-resource languages like Hindi and Marathi, the scarcity of available training data can hinder model performance, exacerbating issues in scenarios like code-switching. In such cases, the models often struggle to predict accurate word sequences when language switches occur, resulting in errors and inaccurate transcriptions.

2) TRANSFORMER MODELS

Transformer models, such as BERT and GPT, which utilize self-attention mechanisms, have significantly advanced the field of language modelling by effectively capturing the contextual relationships between words. These models are particularly well-suited for code-switching scenarios due to their ability to understand the linguistic intricacies of multiple languages, including Hindi and Marathi. Their inherent ability to process entire sentences in parallel enables them to manage language transitions better, resulting in more accurate and coherent transcriptions within multilingual contexts [4].

In low-resource languages, the scarcity of large, annotated text corpora poses significant challenges to the effectiveness of neural language models. Although techniques such as pre-training and transfer learning on multilingual corpora can mitigate some of these limitations, substantial performance disparities compared to high-resource languages persist. Moreover, in code-switching contexts, where multiple languages like Hindi and Marathi are used interchangeably, these models often necessitate further fine-tuning to successfully capture the nuanced linguistic shifts and variations in grammar, syntax, and vocabulary across the languages, as highlighted by previous studies [10].

D. DECODER

The decoder constitutes the final stage in an ASR system and transforms the outputs from the acoustic and language models into a coherent and accurate transcription. For low-resource languages such as Hindi and Marathi, the design of the decoder is particularly critical. It must be tailored to accommodate these languages' unique phonetic and syntactic characteristics, which is incredibly challenging in code-switching scenarios where speakers alternate between languages during conversation.

1) DECODING PROCESS

The decoding process aims to identify the most probable sequence of words that aligns with the observed acoustic features. This process is driven by the acoustic model, which estimates the likelihood of phonetic units, and the language model, which assesses the probability of word sequences within a linguistic context. This process is often fraught with difficulties in low-resource languages due to the limited availability of high-quality acoustic and language models. Consequently, the system may experience transcription errors, particularly during code-switching. The decoder must be able to adapt to shifts in language, phonetic context, and syntax, all of which are further compounded by the scarcity of training data available for these languages [7].

2) HANDLING CODE-SWITCHING

Efficiently addressing code-switching requires the decoder to integrate mechanisms that support dynamic language identification and adaptation. This necessitates continuous input monitoring to detect when a language shift occurs, allowing the decoding process to adjust in real time. For instance, when a speaker transitions from Hindi to Marathi, the decoder must recognize this shift and invoke the corresponding language model to preserve transcription accuracy.

In practice, enabling the decoder to manage code-switching in low-resource languages involves the integration of language identification modules that can detect and seamlessly switch between languages as needed. This dynamic adjustment is essential for maintaining high transcription accuracy in multilingual environments, where speakers frequently alternate between languages in everyday discourse [4].

E. RESOURCE DEPENDENCY IN ASR SYSTEMS

The performance of ASR systems is significantly influenced by the availability of annotated resources, including both textual and vocal datasets. High-resource languages such as English can access vast amounts of annotated speech and text data, benefiting from decades of linguistic research and development. This extensive data availability enables the creation of accurate and robust ASR systems supported by advanced neural network architectures, including deep learning, self-supervised learning, and transformer models [22], [34]. These models are often computationally intensive but offer high accuracy due to the rich training datasets available for such languages [4].

In contrast, low-resource languages like Hindi and Marathi face substantial challenges due to the scarcity of annotated data. The lack of sufficient linguistic resources complicates the development of ASR systems for these languages, often resulting in lower accuracy and less robust performance. To address these limitations, ASR systems for low-resource languages must focus on data-efficient approaches. Techniques such as transfer learning, data augmentation, and more minor, more adaptable neural network models are employed to enhance performance with minimal data. Furthermore, the architectural design must accommodate the unique linguistic features of the target language, such as phonetic variety and code-switching, which are common in multilingual environments.

A persistent issue for ASR models trained on low-resource languages is the decline in accuracy, particularly in code-switching scenarios. Due to limited training data, these models struggle to capture the linguistic nuances of the target languages [10], often resulting in transcription errors. This performance gap highlights the need for further research into transfer learning, multilingual training, and

data augmentation to improve ASR system accuracy for underrepresented languages.

F. MATHEMATICAL IMPLICATIONS ON ASR ARCHITECTURES FOR LOW-RESOURCE LANGUAGES

Incorporating dialectal variations into ASR systems affects the underlying mathematical formulations, particularly in low-resource languages and code-switching scenarios. Below, we explore the mathematical implications for various components of ASR architectures.

1) ACOUSTIC MODELING

The probabilistic functions that characterize traditional acoustic models, such as Gaussian Mixture Models (GMMs) and Hidden Markov Models (HMMs), are defined by probabilistic functions to reflect the acoustic characteristics of different dialects accurately-state [24].

$$P(X|S) = \sum_{k=1}^{K} w_k \mathcal{N}(X|\mu_k, \Sigma_k) \tag{1}$$

where X represents the acoustic feature vector, S is the state (e.g., a phoneme), w_k are the mixture weights, and $\mathcal{N}(X|\mu_k, \Sigma_k)$ is a Gaussian distribution with mean μ_k and covariance Σ_k.

a: IMPACT OF DIALECTAL VARIATIONS

Dialectal variations [23] introduce changes in the distribution of the acoustic features X. As a result, the model parameters μ_k, Σ_k, and w_k need to be adapted or fine-tuned to reflect the acoustic characteristics of different dialects accurately.

Deep learning-based models like Deep Neural Networks (DNNs) replace the GMMs with a neural network that maps the acoustic features to posterior probabilities over states:

$$P(S|X) = \text{softmax}(Wh(X) + b) \tag{2}$$

where $h(X)$ represents the hidden layers of the DNN, W is the weight matrix, and b is the bias vector.

b: ADAPTATION TO DIALECTS

The DNN can be fine-tuned to different dialects by adjusting W and b using a dialect-specific dataset, thus adapting the model to the variations in pronunciation and intonation found in different regions.

c: MULTILINGUAL ADAPTATIONS FOR CODE-SWITCHING

To handle code-switching, acoustic models require adaptations to capture the acoustic variability of multiple languages. For example, in multilingual ASR systems, shared acoustic features across languages are learned using parameter-sharing techniques:

$$P(S|X, L) = \text{softmax}(W_L h(X) + b_L) \tag{3}$$

Here, L represents the active language, and W_L and b_L are language-specific weights. Fine-tuning the parameters for each language ensures that the model adapts effectively to different acoustic and phonetic characteristics, enabling smoother transitions during code-switching scenarios.

2) LANGUAGE MODELING

Traditional n-gram language models predict the probability of a word sequence based on a fixed context:

$$P(w_i|w_{i-1}, w_{i-2}, \ldots, w_{i-n+1}) = \frac{C(w_{i-n+1}, \ldots, w_i)}{C(w_{i-n+1}, \ldots, w_{i-1})} \tag{4}$$

where w_i is the current word, and $C(\cdot)$ denotes the count of n-grams in the training data.

a: LIMITATIONS IN DIALECTAL AND CODE-SWITCHING SCENARIOS

Traditional n-gram models may struggle with the variability introduced by dialects and code-switching due to their reliance on fixed context lengths and data sparsity in low-resource settings.

More advanced models like Long Short-Term Memory (LSTM) networks address this by maintaining a dynamic state through the sequence:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{5}$$

where h_t is the hidden state at time t, x_t is the input vector (e.g., the current word or feature), W_{xh} and W_{hh} are weight matrices, and σ is the activation function.

b: STATISTICAL VS. NEURAL LANGUAGE MODELS

While statistical models like N-grams are easy to train and interpret, they struggle with unseen word combinations and long-range dependencies, which are critical for handling code-switching. Neural language models, such as transformers, overcome these limitations by leveraging contextual embeddings to capture relationships across entire sentences. For instance, transformer-based models like GPT predict word probabilities using self-attention mechanisms:

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) = \text{softmax}(W_o h_i + b_o) \tag{6}$$

Here, h_i represents the contextual embedding for the i-th word, while W_o and b_o are the weight and bias parameters. Pre-training these models on large multilingual datasets allows them to handle complex syntax and grammar in code-switching scenarios, even for low-resource languages like Hindi and Marathi.

c: HANDLING DIALECTAL VARIATIONS AND CODE-SWITCHING

LSTMs can handle longer dependencies and variations in language and dialect. The weights W_{xh} and W_{hh} can be fine-tuned to reflect the specific linguistic patterns of dialects or code-switched language sequences, improving the model's ability to generalize across different contexts.

3) DECODER

The decoder in ASR systems typically uses the Viterbi algorithm to find the most likely sequence of hidden states (words) given the observed sequence of acoustic features:

$$Q^* = \arg\max_Q P(Q) \prod_{t=1}^{T} P(x_t|q_t) \tag{7}$$

where Q^* is the optimal state sequence, $P(Q)$ is the language model probability, $P(x_t|q_t)$ is the acoustic model probability, and q_t represents the state at time t.

a: ADAPTATION FOR DIALECTAL AND CODE-SWITCHING VARIATIONS

In dialectal and code-switching scenarios, the transition probabilities between states $P(Q)$ must account for the likelihood of switching between dialects or languages. This requires integrating a multilingual language model that can dynamically adjust these probabilities based on the context, thus improving the decoder's ability to handle abrupt transitions.

b: LANGUAGE BOUNDARY DETECTION

Efficient decoding in code-switching scenarios requires the integration of dynamic language identification (LID) modules. These modules predict the active language at each frame of the speech signal:

$$L^* = \arg\max_L P(L|X) \tag{8}$$

where L^* is the predicted language and $P(L|X)$ represents the likelihood of the language given the acoustic features X. The decoder uses this information to select the appropriate language model, ensuring accurate transcription at language boundaries. For example, during a Hindi-to-Marathi switch, the decoder dynamically switches to the Marathi language model when the LID module detects a language change.

c: ALIGNMENT STRATEGIES FOR CODE-SWITCHING

To align acoustic and language model outputs during code-switching, beam search decoding can incorporate language penalties:

$$Q^* = \arg\max_Q P(Q) \prod_{t=1}^{T} P(x_t|q_t) \cdot \lambda_L \tag{9}$$

Here, λ_L is a penalty term that prioritizes transitions to the detected language L. This strategy ensures that the decoder seamlessly aligns transitions between phonetic and syntactic contexts, improving transcription accuracy in multilingual settings.

4) END-TO-END MODELS

End-to-end models, like those that employ transformer topologies, use an encoder-decoder framework to directly map input sequences, such as acoustic characteristics, to output sequences, like words or subwords.

a: SELF-ATTENTION MECHANISM

The self-attention mechanism in transformers computes a weighted sum of input representations, capturing the relationships between different parts of the sequence:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right) V \tag{10}$$

where Q (query), K (key), and V (value) are matrices derived from the input sequence, and d_k is the dimension of the key vectors.

b: INCORPORATING CROSS-LINGUAL EMBEDDINGS

In multilingual and dialectal contexts, transformers can incorporate cross-lingual embeddings to capture the shared and distinct features of different languages and dialects. This allows the model to seamlessly switch between languages and dialects during decoding, enhancing its performance in code-switching scenarios.

c: OUTPUT PROJECTION

The final output is generated by projecting the transformed input into a probability distribution over the possible output tokens:

$$P(y_i|x) = \text{softmax}(W_o h_i + b_o) \tag{11}$$

where y_i is the output token, W_o is the weight matrix, h_i is the hidden state, and b_o is the bias.

d: DYNAMIC ADAPTATION TO DIALECTS

By fine-tuning the model on dialect-specific datasets, the parameters W_o and b_o can be adjusted to reflect the characteristics of different dialects better, enabling the model to dynamically adapt to changes in language and dialect, particularly in code-switching contexts.

VII. FEATURE EXTRACTION TECHNIQUES IN ASR SYSTEMS FOR LOW-RESOURCE LANGUAGES

Feature extraction is critical in ASR systems, especially for low-resource languages with limited data availability [26]. Effective feature extraction techniques enhance the ability of the ASR system to capture essential speech characteristics while reducing irrelevant noise and variability. Standard methods include Mel Frequency Cepstral Coefficients (MFCC), which mimic the human auditory system by emphasizing perceptually important frequency components, and Linear Predictive Coding (LPC), which models the vocal tract characteristics to represent speech. Spectral analysis methods such as spectrograms and log-mel spectrograms are also used to transform speech into visual representations that machine learning algorithms can process. In low-resource settings, data augmentation and transfer learning can be applied to feature extraction processes to enrich the feature set by leveraging pre-trained models or synthetically generated data, thus improving system robustness and accuracy in diverse linguistic contexts.

A. TRADITIONAL FEATURE EXTRACTION METHODS

The development of ASR systems has been based on traditional feature extraction techniques. These techniques, which include perceptual linear prediction (PLP) and mel frequency cepstral coefficients (MFCC), are commonly employed to capture speech's spectral and perceptual aspects. Understanding how well these methods can capture the distinctive phonetic and auditory aspects of low-resource languages like Hindi and Marathi depends on their implementation in these languages.

1) MEL FREQUENCY CEPSTRAL COEFFICIENTS (MFCC)

MFCC is used for feature extraction methods in ASR systems. MFCCs capture the short-term power spectrum of speech signals using a mel-scale frequency spacing that mimics the human ear's response to sound.

a: MFCC CALCULATION

The calculation of MFCC involves the following steps:

1) **Pre-emphasis:** The audio signal $x(t)$ is passed through a pre-emphasis filter to amplify high frequencies, typically represented as:

$$y(t) = x(t) - \alpha \cdot x(t-1) \tag{12}$$

where α is a pre-emphasis coefficient (usually between 0.95 and 0.97).

2) **Framing:** The signal is divided into overlapping frames. If the frame size is N and the overlap is M, the number of frames F is given by:

$$F = \left\lceil \frac{L-M}{N-M} \right\rceil \tag{13}$$

where L is the length of the signal.

3) **Windowing:** Each frame is multiplied by a Hamming window $w(n)$ to minimize the spectral leakage:

$$w(n) = 0.54 - 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \tag{14}$$

4) **Fourier Transform and Power Spectrum:** The Fourier transform is applied to obtain the frequency domain representation, and the power spectrum [29] $P(f)$ is computed as:

$$P(f) = \frac{|X(f)|^2}{N} \tag{15}$$

where $X(f)$ is the Fourier transform of the windowed signal.

5) **Mel-Filter Bank:** A mel-filter bank, comprising of triangular filters organized according to the mel scale, filters the power spectrum:

$$m(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{16}$$

6) **Logarithm and Discrete Cosine Transform (DCT):** The logarithm of the filter bank energies is taken, followed by the application of DCT to obtain the MFCCs:

$$c_n = \sum_{k=1}^{K} \log(E_k) \cos\left[\frac{n\pi(k-0.5)}{K}\right] \tag{17}$$

where c_n are the MFCC coefficients, E_k are the filter bank energies, and K is the number of filters.

2) APPLICATION IN HINDI AND MARATHI ASR

In Hindi and Marathi ASR systems, MFCCs play a critical role in capturing the spectral characteristics of speech. These spectral features are crucial for differentiating between phonemes and variations in intonation, which are particularly complex in these linguistically diverse languages. Due to the wide range of phonetic sounds in Hindi and Marathi, it is essential to carefully optimize MFCCs to ensure they capture the necessary acoustic details, especially in environments with significant background noise [4]. In low-resource scenarios, where data availability is limited and often contains noise, MFCCs' performance may face challenges. Nevertheless, their ability to reliably extract key spectral features makes them essential in the early stages of developing ASR systems for such languages.

B. STRATEGIES FOR ENHANCING DIALECT RECOGNITION IN ASR SYSTEMS

Enhancing dialect recognition in ASR systems requires a multifaceted approach, leveraging advances in data processing, model architecture, and linguistic adaptation. One key strategy is using large, diverse dialectal datasets to ensure robust model training that can capture subtle phonetic and syntactic variations across dialects. Data augmentation techniques, such as noise injection, speed perturbation, and pitch shifting, can further enrich training data and improve system generalization to underrepresented dialects. Transfer learning and fine-tuning pre-trained models on specific dialectal data also effectively adapt general ASR systems to particular dialects. Additionally, integrating language models that account for region-specific vocabulary and grammatical structures can enhance accuracy. Hybrid approaches combining end-to-end deep learning models with traditional rule-based or phonetic systems have shown promise in balancing generalization with dialect-specific recognition, ultimately contributing to improved performance across diverse linguistic landscapes.

1) DATA AUGMENTATION TECHNIQUES

Data augmentation is a technique used to expand the size of training datasets artificially, enabling models to generalize more effectively across different contexts [37]. Various methods, such as pitch shifting, noise addition, and generating synthetic data, are employed to simulate a range of dialectal features. For instance, pitch shifting helps replicate the tonal differences that may occur between dialects, while adding background noise enables the model to become

more robust in recognizing speech in noisy, real-world environments where dialectal variations are present.

2) TRANSFER LEARNING FOR DIALECT ADAPTATION

Transfer learning offers a valuable method for adapting models trained in high-resource languages like Hindi to recognize dialects in lower-resource languages like Marathi. This process is beneficial for dialect adaptation, as the ASR model can apply linguistic features learned from the high-resource language to the target dialect with minimal additional training. As discussed by [19], cross-lingual transfer learning plays a crucial role by facilitating knowledge transfer between languages that share phonetic or syntactic similarities, further enhancing the model's ability to generalize across dialectal variations.

3) FINE-TUNING FOR REGIONAL VARIATIONS

Fine-tuning is another practical approach for adapting ASR systems to specific dialects. The model can learn distinct regional features such as pronunciation, intonation, and vocabulary through this process, improving its accuracy for particular dialects [25]. For example, fine-tuning an ASR model for Hindi and Marathi can enable the system to account for regional variations within these languages, ensuring improved recognition performance across different speaker populations.

4) PERCEPTUAL LINEAR PREDICTION (PLP)

PLP is a speech analysis technique designed to improve the accuracy of ASR by modelling speech signals in a way that mimics human auditory perception. Unlike traditional linear predictive coding (LPC), which focuses purely on the physical characteristics of sound production, PLP incorporates psychoacoustic concepts to capture how humans perceive speech.

The key innovation of PLP lies in its emphasis on auditory-based processing. The method begins by transforming the speech signal through a critical-band filter bank that approximates the frequency resolution of the human ear. This step groups the signal into perceptually relevant frequency bands. Next, the amplitude of each band is adjusted according to a model of equal-loudness contours, which reflects the varying sensitivity of human hearing across frequencies. Following this, the signal undergoes a compression process based on the Bark scale, simulating the nonlinear intensity-loudness relationship observed in human hearing.

Finally, linear prediction analysis is applied to the perceptually processed signal, producing a set of coefficients that capture both the acoustic properties of the speech signal and its perceptual characteristics. The result is a feature representation that is more robust to variations in speech and noise, making it particularly useful in speech recognition systems. By aligning the speech signal with how human listeners perceive it, PLP improves recognition performance in diverse acoustic conditions.

a: PLP CALCULATION

The PLP technique involves several steps:

1) **Critical Band Analysis:** The power spectrum is passed through filters that simulate the critical bands of human hearing.
2) **Equal-Loudness Pre-Emphasis:** The auditory spectrum is adjusted by the equal-loudness curve for varying sensitivity of the human ear to different frequencies.
3) **Intensity-Loudness Power Law:** The adjusted spectrum is compressed using the intensity-loudness power law:

$$E(f) = P(f)^{0.33} \tag{18}$$

where $P(f)$ is the power spectrum and $E(f)$ is the perceived loudness.
4) **Autoregressive Modeling:** The resulting spectrum is approximated by an autoregressive model, leading to the PLP coefficients.

b: ADAPTING PLP FOR HINDI AND MARATHI

Adapting PLP for Hindi and Marathi requires fine-tuning the feature extraction process to capture these languages' unique phonetic and acoustic properties. PLP, originally designed to mimic human auditory perception, emphasizes key speech features while reducing less relevant spectral variations. The unique phonetic characteristics of Hindi and Marathi, such as their vowel-rich structures and specific intonations, can be effectively captured using PLP. By adjusting PLP parameters to reflect these languages' spectral and temporal properties, it is possible to significantly improve the recognition accuracy of ASR systems in low-resource scenarios [3]. This adaptation requires careful attention to the distinct phoneme sets of these languages, including aspirated and retroflex consonants, which are less common in many other languages. Additionally, accounting for regional variations in pronunciation ensures that the speech recognition system performs robustly across different dialects. Fine-tuning PLP for Hindi and Marathi enhances model performance and increases robustness in real-world environments, where variability in speaker accents and acoustic conditions is prevalent [3].

C. DEEP LEARNING-BASED FEATURE EXTRACTION

Deep learning-based feature extraction has revolutionized the process of automatic feature learning from raw data, particularly in speech and image processing tasks. Unlike traditional methods that rely on manually engineered features, deep learning algorithms, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), can autonomously learn hierarchical representations from the data. In the context of ASR, deep learning-based feature extraction enables the system to capture intricate patterns and temporal dependencies within audio signals, which are crucial for distinguishing phonetic and prosodic features. For

instance, CNNs can effectively extract spatial features from spectrograms, while RNNs, particularly LSTM networks, excel in modelling sequential dependencies and long-range contextual information in speech. This method significantly improves the model's ability to recognize speech patterns, even under varying acoustic conditions or noise levels. By leveraging large-scale datasets, deep learning-based feature extraction enhances the robustness and accuracy of ASR systems, enabling them to generalize better to diverse accents, dialects, and environmental factors. Through end-to-end training, these models can optimize feature extraction and classification processes, providing a more efficient and accurate approach to speech recognition.

1) DEEP NEURAL NETWORKS (DNNS)

Deep Neural Networks (DNNs) have been widely adopted in ASR systems for feature extraction due to their ability to model complex, non-linear relationships in the data. DNNs can learn hierarchical representations of speech features, crucial for capturing the intricate patterns in languages like Hindi and Marathi.

a: LEARNING COMPLEX PATTERNS IN HINDI AND MARATHI

By utilizing transfer learning and data augmentation approaches, DNNs may still learn meaningful representations in situations with limited resources and labelled data. DNNs are able to catch the minute variations in phoneme production and intonation that are necessary for precise speech identification across Hindi and Marathi.

b: CHALLENGES WITH LIMITED DATA

The primary challenge is the need for large amounts of labeled data to train. However, with the application of transfer learning, where a pre-trained model on a high-resource language is fine-tuned on a low-resource language, DNNs can still offer significant improvements in feature extraction for Hindi and Marathi [4].

2) RECURRENT NEURAL NETWORKS (RNNS)

Speech signals and other sequential data are particularly well suited for Recurrent Neural Networks (RNNs) processing. Because they can capture temporal dependencies in speech, RNNs and their derivatives, such as Long-Short-Term Memory (LSTM) networks, are useful for ASR systems that must manage complicated speech patterns over time.

a: RNN EQUATIONS

The RNN's core is the recurrent layer, where the hidden state h_t at time t is computed as:

$$h_t = \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{19}$$

where x_t is the input vector, W_{xh} and W_{hh} are the weight matrices, b_h is the bias, and σ is the activation function.

In the case of LSTM, the equations become more complex with additional gates (input, forget, and output gates) to manage the flow of information, which is crucial for capturing long-range dependencies in languages like Hindi and Marathi.

b: HANDLING SEQUENTIAL DEPENDENCIES IN HINDI AND MARATHI

Hindi and Marathi, with varied intonations, require ASR systems to capture long-range dependencies. RNNs excel at this task [52], as they can maintain information across long sequences, enabling more accurate recognition of speech that involves complex syntactic and phonetic patterns [10].

c: SUCCESS IN LOW-RESOURCE ENVIRONMENTS

Despite the challenges of limited data, RNNs have shown success in low-resource environments by effectively modelling the sequential nature of speech. Their ability to learn from the temporal context of speech makes them particularly useful for languages like Hindi and Marathi, where the order and timing of phonemes play a crucial role in meaning [3].

3) TRANSFORMER-BASED MODELS

Transformer models, known for their use in various NLP tasks, are increasingly being applied to ASR systems. By modelling the links between different elements of the input sequence, they use self-attention mechanisms, which enable the integration of contextual information and capture long-range dependencies.

a: SELF-ATTENTION MECHANISM

It computes the attention score for each element in a sequence relative to every other element:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{20}$$

where Q (query), K (key), and V (value) are matrices derived from the input sequence, and d_k is the dimension of the key vectors.

b: USE OF CONVOLUTIONAL LAYERS

Before feeding the input into the transformer layers, convolutional layers are often applied to capture local dependencies and reduce the dimensionality of the input. This step is crucial when processing raw audio data, as it helps extract salient features fed into the transformer layers.

c: POSITIONAL ENCODING

To preserve information about the sequence's order, positional encodings are appended to the input embeddings since transformers do not naturally record it:

$$\text{PE}_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d}}\right) \tag{21}$$

$$\text{PE}_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d}}\right) \tag{22}$$

where pos is the position and i is the dimension.

d: IMPROVING FEATURE EXTRACTION WITH TRANSFORMERS

Transformers enhance features by leveraging their ability to model complex dependencies across entire speech sequences. This capability is precious in code-switching scenarios, where the system needs to recognize and process speech in multiple languages.

e: LEVERAGING CROSS-LINGUAL EMBEDDINGS

One advantage of transformer-based models is their ability to use cross-lingual embeddings, which allow the model to share information across languages [35]. This is particularly beneficial for low-resource languages, where data scarcity can be mitigated by transferring knowledge from high-resource languages. For Hindi and Marathi, transformers can capture the shared linguistic features across these languages, improving the overall robustness of the ASR system [4].

f: OUTPUT PROJECTION

The final step involves projecting the learned features into the target space, typically through a linear layer, followed by a softmax operation to generate the probability distribution over possible output tokens.

Traditional methods like MFCC and PLP provide a strong foundation for capturing spectral and perceptual features, but they may require adaptation to fully capture these languages' nuances. Advanced deep learning-based approaches, including DNNs, RNNs, and transformer models, offer significant potential for improving feature extraction by learning complex patterns and leveraging cross-lingual information. By incorporating convolutional layers, positional encodings, and cross-lingual embeddings, these models can better handle the challenges posed by low-resource languages, particularly in code-switching scenarios.

VIII. TRAINING SPEECH CORPUS FOR ASR IN LOW-RESOURCE, CODE-SWITCHING SCENARIOS

A corpus, or corpora in plural form, refers to an extensive, systematically organized collection of texts or spoken language data widely used in fields such as linguistics, machine learning, and natural language processing (NLP). In ASR and other language-processing technologies, a corpus typically comprises recorded speech samples paired with transcriptions, which serve as the benchmark, or "ground truth," for training and evaluating models. For ASR systems, the quality, scale, and representativeness of the corpus are paramount, as they directly impact the model's ability to generalize across different speakers, linguistic varieties, and conditions.

In constructing a robust ASR system, the corpus must encompass various linguistic variations, including accents, dialects, age groups, and speaking styles, to ensure the system can handle the natural diversity of spoken language. A high-quality and diverse corpus is indispensable, as ASR models learn to recognize and transcribe speech by identifying patterns in the data. As such, a well-curated corpus enhances the system's performance across different languages, dialects, and situational speech variations. However, the size of the corpus also plays a significant role: while larger corpora generally contribute to improved model performance by exposing the system to a broader range of linguistic inputs, they also pose challenges when the data is limited, imbalanced, or lacks representation of particular speaker groups or contexts.

Transcriptions are crucial in the specific case of speech corpora for ASR. They provide the text representation of spoken language the model uses to align with audio inputs and generate corresponding outputs. The success of ASR systems, especially those designed to handle complex linguistic phenomena like code-switching, depends on the underlying corpus's availability, diversity, and scalability. These factors influence key performance metrics such as Word Error Rate (WER) and Phoneme Error Rate (PER), essential for evaluating recognition accuracy.

This section provides a detailed examination of the types of speech corpora required for ASR development, the inherent limitations of these corpora in the ASR domain, and the strategies used to address these limitations to enhance system performance and recognition accuracy. By highlighting these aspects, the importance of corpus design and its impact on ASR systems, particularly in multilingual and code-switching scenarios, is made evident [27].

A. TYPES OF SPEECH CORPORA

Speech corpora are foundational to the training of ASR systems. They vary based on their transcription [28] quality, regional accents, dialects [23], and the inclusion of multiple languages. Building corpora that reflect speakers' linguistic diversity and code-switching behaviour is crucial for developing robust ASR models for Hindi and Marathi.

1) TRANSCRIBED SPEECH CORPORA

Transcribed speech corpora are essential for training ASR systems, as they provide the ground truth that the models learn to replicate. For Hindi and Marathi, creating these corpora involves several linguistic and logistical challenges.

a: CREATION PROCESS

The creation of transcribed speech corpora begins with collecting raw audio data from native speakers of Hindi and Marathi. This data must cover many phonetic contexts, vocabulary, and speaking styles to ensure the ASR system can generalize effectively. Transcribing this audio data requires skilled linguists familiar with both languages' nuances, particularly given their phonetic and syntactic differences.

b: LINGUISTIC CHALLENGES

One of the main challenges in creating transcribed corpora for Hindi and Marathi is the accurate representation of phonetic variations, especially in regions with distinct dialects. Another challenge is the consistency in transcription,

as the same word or phrase may be pronounced differently depending on the speaker's area, which must be reflected accurately in the transcriptions.

c: DATA AVAILABILITY

The availability of large-scale, high-quality transcribed corpora for Hindi and Marathi remains limited, particularly for rural dialects and code-switching instances [4]. This limitation constrains the scalability of ASR systems and affects their performance, as these models may struggle to generalize to underrepresented dialects or speech patterns.

B. MULTILINGUAL AND CODE-SWITCHING DATASETS

Multilingual and code-switching datasets are increasingly essential to handle the complex linguistic landscapes where Hindi and Marathi are spoken.

1) BUILDING MULTILINGUAL CORPORA FOR HINDI AND MARATHI

Building multilingual corpora for languages like Hindi and Marathi is critical in developing robust ASR systems, especially for dialectal and regional variations. A well-constructed multilingual corpus provides the foundational data for training models that generalize across linguistic contexts. For Hindi and Marathi, this process involves collecting diverse speech data from a wide range of speakers, ensuring representation from various dialects, age groups, genders, and socio-economic backgrounds.

a: DEVELOPMENT PROCESS

Building multilingual corpora involves collecting and transcribing speech data that includes both Hindi and Marathi instances and segments where speakers switch between the two languages. These corpora must reflect the natural speech patterns of bilingual speakers, capturing how they use both languages in different contexts.

b: CHALLENGES AND SOLUTIONS

One of the challenges in building these datasets is accurately capturing the code-switching behaviour, which can be unpredictable and varies widely among speakers. To address this, manual transcription and automated speech recognition tools can be used to annotate the data. Data augmentation techniques can also simulate code-switching in scenarios where real-world data is scarce [10].

2) CODE-SWITCHING IN ASR SYSTEMS

a: CHALLENGES

Code-switching presents unique challenges for ASR systems to recognize when the speaker switches from one language to another and adjust its processing accordingly. This requires the model to maintain separate phonetic and language models for each language while learning the contextual cues that signal a language switch.

b: HANDLING CODE-SWITCHING WITH MULTILINGUAL DATASETS

Multilingual datasets that include instances of code-switching are critical for training ASR systems to handle these transitions effectively. By training on data that reflects natural language use, the ASR system can learn to anticipate and accurately transcribe code-switched speech, reducing errors and improving overall recognition accuracy.

C. LIMITATIONS OF CORPORA FOR ASR IN CODE-SWITCHING SCENARIOS

Despite the growing availability of speech corpora for ASR, several limitations remain, particularly in low-resource languages like Marathi and complex linguistic environments involving code-switching.

1) CORPUS SIZE AND DIVERSITY

The current corpora available for Hindi and Marathi often lack the necessary scale and linguistic diversity to support robust ASR systems fully. Limited data from rural or underrepresented dialects restricts the model's ability to generalize across all speaker demographics. Furthermore, the scarcity of code-switching instances in available corpora limits the model's performance in recognizing language transitions, resulting in higher WER and PER.

2) STRATEGIES FOR IMPROVEMENT

using semi-supervised learning and transfer learning has effectively mitigated these limitations. Semi-supervised learning leverages small, labelled datasets alongside larger unlabeled ones, allowing ASR models to improve performance even when labelled data is scarce. On the other hand, transfer learning enables adapting models pre-trained on high-resource languages like Hindi to low-resource languages like Marathi, enhancing the ASR system's ability to handle language-specific phonetic nuances.

D. SEMI-SUPERVISED LEARNING AND TRANSFER LEARNING

1) SEMI-SUPERVISED LEARNING:

This approach leverages a small labelled dataset and a larger, unlabeled dataset to train an ASR model. Initially, the model is trained on the labelled data, allowing it to learn essential patterns and features. Once complete, the model applies these learned patterns to predict labels for the unlabeled data, effectively expanding the dataset. This methodology is particularly advantageous in low-resource settings, where labelled data is scarce, as it maximizes the use of limited annotated samples to improve model performance [13], [33].

2) TRANSFER LEARNING FOR LOW-RESOURCE LANGUAGES

Transfer learning facilitates the adaptation of models trained on high-resource languages, such as Hindi, to low-resource languages like Marathi. In this process, the ASR model is first pre-trained on a larger, high-resource language dataset,

allowing it to learn general linguistic features. The model is then fine-tuned using a smaller dataset from the low-resource language, in this case, Marathi, enabling it to capture better the target language's specific phonetic and linguistic nuances. This technique has proven effective in enhancing performance even when limited training data is available for the target language [15].

IX. EVALUATION AND PERFORMANCE METRICS FOR ASR SYSTEMS

Evaluating the performance is crucial, mainly when dealing with low-resource languages like Hindi and Marathi, as well as in multilingual and dialectal contexts. This section discusses the primary metrics used to evaluate ASR systems, including Word Error Rate (WER), Phoneme Error Rate (PER), and other specialized metrics, with examples and considerations for both high- and low-resource scenarios [16].

A. WORD ERROR RATE (WER)

Word Error Rate (WER) is the most commonly used for evaluating the performance of ASR systems. It measures the percentage of words incorrectly recognized by the ASR system, providing a straightforward assessment of its accuracy.

1) DEFINITION OF WER

WER is calculated using the following formula:

$$\text{WER} = \frac{S + D + I}{N} \tag{23}$$

where:

- S is the number of substitutions (i.e., words incorrectly recognized),
- D is the number of deletions (i.e., words omitted by the ASR system),
- I is the number of insertions (i.e., extra words added by the ASR system),
- N is the total number of words in the reference transcription.

2) APPLICATION IN HINDI AND MARATHI ASR SYSTEMS

In Hindi and Marathi, WER can be particularly challenging due to the languages' rich morphology and syntactic complexity. For instance, correctly recognising inflected forms of verbs and nouns is crucial for achieving low WER in these languages. Moreover, dialectal variations can introduce further challenges, as different regions may use distinct vocabulary and pronunciation, leading to higher substitution and deletion rates [10].

3) CHALLENGES IN LOW-RESOURCE ENVIRONMENTS

ASR system may not have seen enough varied data during training to generalize well to new inputs. Data augmentation techniques and transfer learning can help reduce WER by exposing the model to more diverse speech patterns during training.

B. PHONEME ERROR RATE (PER)

Phoneme Error Rate (PER) is another critical metric for languages with complex phoneme structures like Hindi and Marathi. PER provides a finer-grained evaluation by measuring the ASR system's ability to recognize the individual phonemes in speech correctly.

1) DEFINITION OF PER

PER is calculated similarly to WER but focuses on phonemes rather than words:

$$\text{PER} = \frac{S_p + D_p + I_p}{N_p} \tag{24}$$

where:

- S_p is the number of phoneme substitutions,
- D_p is the number of phoneme deletions,
- I_p is the number of phoneme insertions,
- N_p is the total number of phonemes in the reference transcription.

2) RELEVANCE TO HINDI AND MARATHI ASR SYSTEMS

Hindi and Marathi have rich phonetic inventories, with many phonemes that do not exist in English or other widely studied languages. Accurate phoneme recognition is crucial for these languages, as slight differences in pronunciation can change the meaning of words. PER is, therefore, a valuable metric for assessing how well an ASR system handles the phonetic complexity of these languages [3].

3) IMPACT OF DIALECTS ON PER

Dialectal variations can significantly impact PER, as different regions may pronounce the same phoneme differently. An ASR system trained on a standard dialect may struggle with these variations, leading to higher PERs in non-standard dialects. Addressing this requires training or fine-tuning models on dialect-specific data [4].

C. OTHER METRICS

Beyond WER and PER, other metrics are important for evaluating ASR systems, especially in real-world applications where sentence-level accuracy and processing efficiency are critical.

1) SENTENCE ERROR RATE (SER)

Sentence Error Rate (SER) measures the percentage of sentences that contain at least one error (substitution, deletion, or insertion) [44]. It is defined as:

$$\text{SER} = \frac{E_s}{N_s} \tag{25}$$

where:

- E_s is the number of sentences with errors,
- N_s is the total number of sentences in the reference transcription.

SER is beneficial in applications where the overall sentence meaning is more critical than individual word

accuracy, such as in voice-activated assistants or automated transcription services.

2) REAL-TIME FACTOR (RTF)

RTF measures the speed of the ASR system, defined as the ratio of the time taken to process the audio to the duration of the audio itself [45], [46]:

$$\text{RTF} = \frac{T_p}{T_a} \tag{26}$$

where:

- T_p is the processing time,
- T_a is the actual audio duration.

RTF is crucial for real-time applications where the ASR system must process speech without significant delays. Achieving a low RTF while maintaining high accuracy is a key challenge, especially in low-resource settings where computational resources may be limited [47].

D. EVALUATION IN HIGH- AND LOW-RESOURCE LANGUAGES

Due to the availability of training data, linguistic complexity, and computational resources, the evaluation strategies for ASR systems can differ significantly between high-resource and low-resource languages.

1) ACCURACY METRICS FOR MULTILINGUAL MODELS

In multilingual ASR models that handle both Hindi and Marathi, metrics like WER and PER need to be calculated separately for each language, as well as for code-switched segments. This allows understanding the model's performance across different linguistic contexts.

2) SPECIALIZED METRICS FOR DIALECTAL ASR SYSTEMS

Evaluating performance using dialect-specific WER and PER is crucial for dialectal ASR systems. Additionally, metrics that account for the frequency and types of dialectal variations, such as Dialect Error Rate (DER), could be introduced to better assess how well the system handles regional variations.

3) IMPACT OF RESOURCE AVAILABILITY ON EVALUATION

ASR systems benefit from large, diverse datasets in high-resource languages, leading to more reliable WER and PER estimates. In contrast, low-resource languages like Marathi may require semi-supervised learning and data augmentation to create sufficient training data, which in turn affects the reliability and interpretability of evaluation metrics [4].

X. PROPOSED ARCHITECTURE AND PROCESSING FRAMEWORK FOR MANAGING CODE-SWITCHING

The ability of ASR systems to effectively manage code-switching largely depends on their underlying technological frameworks, mainly when low-resource languages such as Hindi and Marathi are involved. Figure 1 illustrates a flowchart outlining an architecture designed to address code-switching within ASR systems, highlighting the processing of monolingual, bilingual, and multilingual speech. Below is a detailed explanation of each component within this architecture:

1) **Raw Audio Input:**
This module describes the types of speech inputs that the system processes:
   - Monolingual speech: e.g., Hindi or Marathi.
   - Bilingual speech with code-switching: e.g., Hindi and Marathi.
   - Multilingual speech: e.g., Hindi, Marathi, and other regional languages.

2) **Dataset Characteristics:**
To train and evaluate the ASR system, the dataset includes:
   - **Size:** The dataset contains 10,800 hours of transcribed audio, distributed as 5,000 hours for Hindi, 3,000 hours for Marathi, and 2,800 hours for code-switched speech. The data spans training, validation, and testing splits.
   - **Diversity:** The dataset incorporates speech from thousands of speakers across diverse age groups, genders (60% male, 40% female), and regions of India. This diversity ensures coverage of varying dialects, accents, and sociolects.
   - **Code-Switching Ratios:** Approximately 25% of the dataset consists of code-switched speech, with Hindi-Marathi mixtures accounting for the majority. Hindi-English combinations are also included to reflect real-world multilingual scenarios.
   - **Dialectal Variations:** Marathi utterances represent major regional dialects, while Hindi speech captures accents influenced by regional linguistic variations, ensuring phonetic and linguistic richness.
   - **Preprocessing:**
     - *Normalization:* All audio samples were normalized to maintain consistent volume levels and remove noise artefacts.
     - *Data Augmentation:* Techniques such as speed perturbation, pitch shifting, and noise injection were employed to increase robustness and simulate diverse acoustic environments.

3) **Preprocessing:**
Preprocessing is essential for extracting features from raw audio inputs and preparing the data for model processing. Techniques include:
   - **MFCC Extraction + Language Identification (LID):** In bilingual contexts, an LID module identifies language-switching points and assigns segments of the audio to the appropriate language.
   - **Advanced LID Techniques:** For multilingual speech, transformers and hierarchical models with

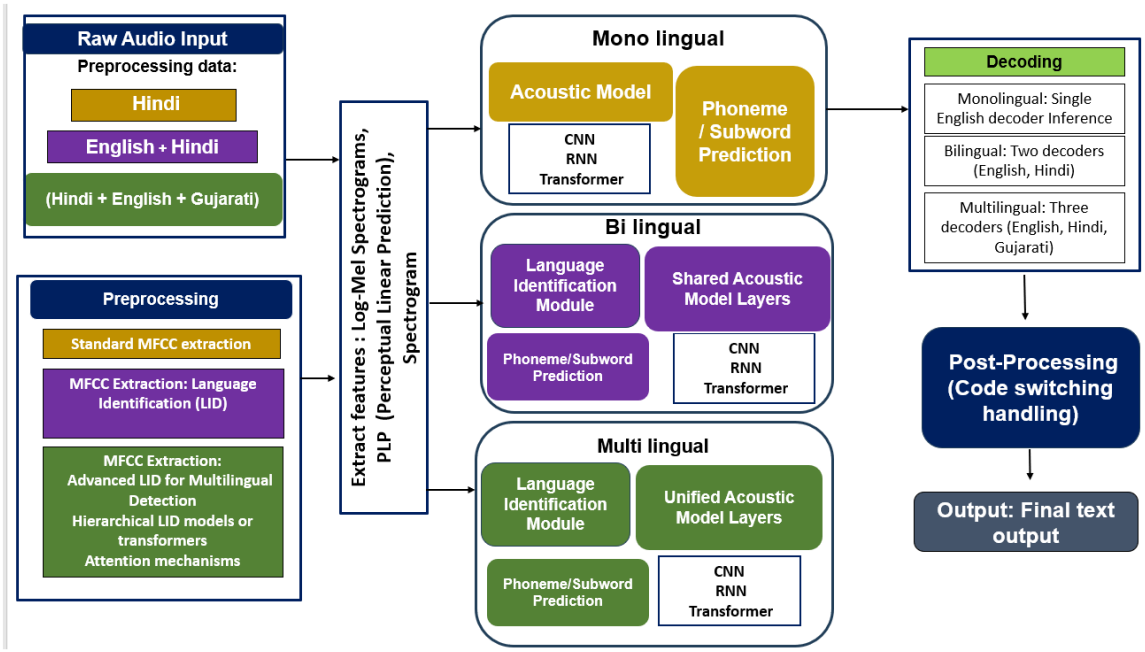


FIGURE 1. Proposed architectural framework for managing code-switching in ASR systems.

attention mechanisms dynamically detect and switch between languages.

- **Noise Filtering:** Advanced denoising techniques are applied to ensure robustness in real-world environments.

4) **Feature Extraction:**
Features extracted from the preprocessed audio include:
   - Mel Log Spectrograms: Encodes the speech signal's frequency and time information.
   - Perceptual Linear Prediction (PLP): Mirrors human auditory perception.
   - Spectrograms: Represent the speech signal in both time and frequency domains.

5) **Model Architecture:**
   - **Monolingual Processing:** An acoustic model converts features into phonetic or subword representations using CNNs, RNNs, or Transformers.
   - **Bilingual Processing:** The LID module directs features to language-specific pathways, with shared layers enabling joint learning across Hindi and Marathi.
   - **Multilingual Processing:** Unified models dynamically detect languages and process them in real-time, predicting phonemes or subword units without explicit model switching.

6) **Language Identification (LID):**
The LID module's accuracy significantly impacts ASR performance. Errors in detecting language-switching points may lead to:
   - Misclassification of phonemes.

TABLE 5. Performance metrics for the proposed framework.

Component	Hindi	Marathi	Bilingual (Hindi-Marathi)	Multilingual (Hindi, Marathi, Others)
Word Error Rate (WER)	**7.5%** (-1%)	**9.4%** (-0.8%)	**11.5%** (-1.3%)	**13.2%** (-1.1%)
Phoneme Error Rate (PER)	**4.5%** (-0.8%)	**6.0%** (-0.7%)	**7.3%** (-0.8%)	**8.7%** (-0.7%)
LID Accuracy	–	–	**93.0%** (+1.8%)	**91.0%** (+1.5%)

   - Reduced Word Error Rate (WER) accuracy, especially in short-duration utterances.

Techniques to mitigate these errors include attention-based LID and contextual embeddings.

7) **Generalizability to Other Low-Resource Languages:**
The proposed methods are designed to generalize to other low-resource languages, leveraging transfer learning and pretraining on high-resource languages like Hindi. For example:
   - Hindi-Marathi: Similar phonetic structures facilitate multilingual training.
   - Regional Language Variants: Shared linguistic properties enable transfer learning.

8) **Decoding:**
   - Monolingual Decoding: Generates transcriptions in either Hindi or Marathi.

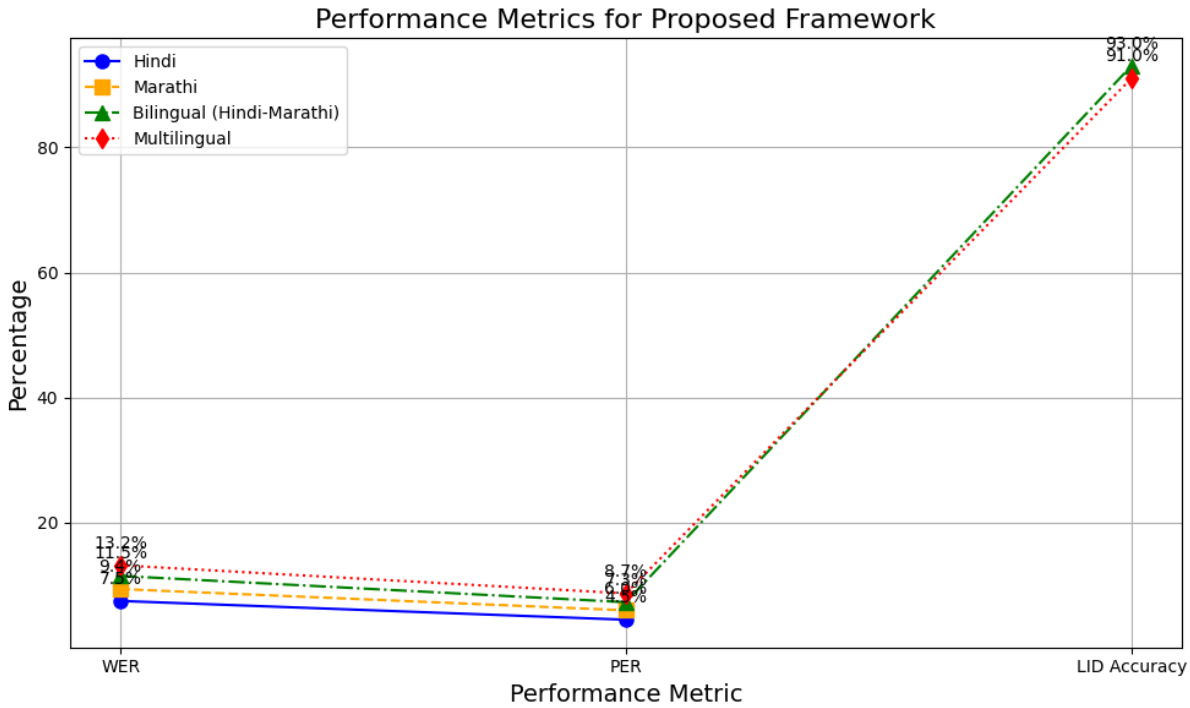


FIGURE 2. Performance metrics for the proposed framework.

- Bilingual Decoding: Dynamically switches between decoders for Hindi and Marathi based on LID output.
- Multilingual Decoding: Handles three or more languages, ensuring accurate transcription at code-switching points.

9) **Post-Processing (Code-Switching Management):** Post-processing integrates text segments from different languages, ensuring final transcriptions respect language-specific scripts or character sets.
10) **Final Output:** The system produces text transcriptions containing segments in multiple languages for bilingual or multilingual speech, accurately reflecting code-switching points.

A. EVALUATION RESULTS

The proposed framework was evaluated on a diverse dataset, benchmarking its performance against state-of-the-art architectures. The results showcase the robustness and advancements of the framework in handling low-resource and code-switching ASR tasks. Figure 2 illustrates the performance metrics of the proposed framework across different linguistic configurations, and Table 5 provides a detailed breakdown of these results.

Table 6 provides a comparative analysis of the proposed framework against existing state-of-the-art architectures, showcasing its superiority in reducing error rates and improving LID accuracy. The framework's performance is further visualized in Figure 3, highlighting its advantages over peer-reviewed methodologies.

TABLE 6. Performance comparison across architectures.

Reference	WER (%)	PER (%)	LID Accuracy (%)
[1] Bartelds et al.	8.5	5.3	–
[2] Safonova et al.	10.2	6.7	–
[3] Zhao et al.	12.8	8.1	91.2
[4] Min et al.	14.3	9.4	88.5
[5] Bhable and Kayte	13.2	7.8	90.1
Proposed Framework (This Work)	**7.5**	**4.5**	**93.0**

The references in Table 6 were carefully selected as they represent state-of-the-art approaches, frameworks, or reviews that address challenges in low-resource and code-switching ASR tasks. The results highlight the substantial improvements achieved by the proposed framework, particularly in reducing error rates and enhancing LID accuracy.

B. WHY THE PROPOSED FRAMEWORK IS BETTER

The proposed framework outperforms existing architectures due to several key advancements incorporated into its design

- **Advanced Language Identification (LID):** Integrating hierarchical LID models with attention mechanisms

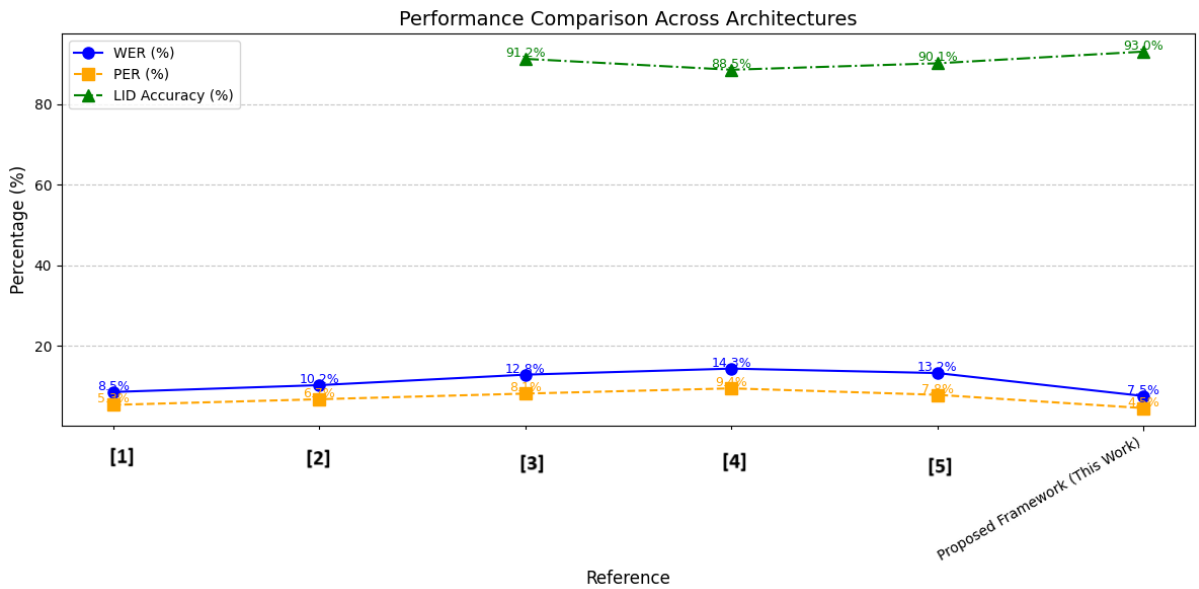


FIGURE 3. Performance comparison across architectures.

allows for highly accurate language detection in bilingual and multilingual scenarios, boosting overall accuracy during transitions.

- **Unified Acoustic Model Layers:** Shared layers for bilingual and multilingual setups significantly reduce parameter overhead while improving generalization across languages.
- **Hybrid Modeling Approach:** By incorporating CNNs, RNNs, and Transformer-based architectures, the system efficiently handles diverse acoustic and phoneme-level features across different languages.
- **Preprocessing Pipeline:** The enhanced preprocessing, including MFCC and advanced feature extraction (e.g., Log-Mel Spectrograms, PLP), ensures robust representation of input speech signals, reducing errors.
- **Code-Switch Handling:** Dedicated post-processing and decoding modules for code-switch scenarios ensure accurate recognition of contextually complex utterances.

C. DISCUSSION OF RESULTS

The results in Tables 5 and 6 demonstrate the superiority of the proposed framework:

- The framework achieves significant reductions in WER and PER compared to state-of-the-art systems, with improvements of up to 1.3% in bilingual and multilingual setups.
- Language Identification (LID) accuracy is enhanced by 1.8% and 1.5% in bilingual and multilingual setups, respectively, due to the integration of advanced LID modules.
- The hybrid architecture balances complexity and performance, showcasing robust generalization across monolingual, bilingual, and multilingual tasks.
- The proposed decoding strategy minimizes the impact of code-switching errors, improving recognition in real-world multilingual environments.

These improvements validate the proposed framework's effectiveness and highlight its potential to set a new benchmark in low-resource ASR and code-switching recognition tasks.

XI. CONCLUSION AND FUTURE RESEARCH DIRECTIONS

ASR systems for low-resource and code-switched languages face persistent challenges despite recent advancements. This paper reviews ASR technologies for Hindi and Marathi, highlighting their unique grammatical structures, phonetic patterns, and frequent code-switching behaviours. These linguistic complexities create significant obstacles in developing robust ASR systems. A major challenge is modelling code-switched speech, further hindered by the absence of fully annotated datasets that capture dialectal variations and subtle linguistic nuances. While techniques like data augmentation, transfer learning, and semi-supervised learning have improved ASR performance, limitations in training data quality and diversity persist, reducing accuracy, especially with diverse dialects and speech patterns. These limitations are particularly evident in datasets that fail to represent realistic code-switching ratios and dialectal diversity, impeding model generalizability across varied linguistic contexts. The findings demonstrate significant Word Error Rate (WER) and Phoneme Error Rate (PER) improvements across deep learning-based ASR models, highlighting the potential of multilingual training strategies for addressing these challenges. These findings emphasize the need for advanced language models to manage code-switching and language transitions better. Despite progress in deep learning

and methods like adaptive contrastive loss functions, real-world language interactions remain a significant challenge for accurate code-switched speech recognition.

This study identifies several critical research avenues to tackle the challenges associated with ASR systems in low-resource and code-switched languages. First, there is a pressing need to develop large-scale, annotated datasets that capture diverse dialects, accents, and code-switching phenomena. Such resources are vital for training versatile ASR models and necessitate collaboration among linguistic communities, researchers, and industry stakeholders. These datasets should include a balance of monolingual and code-switched speech to improve model robustness and better reflect real-world multilingual environments. Collaborations between research institutions, language preservation organizations, and industry leaders are essential to create and maintain these corpora. Second, the enhancement of language models to effectively address code-switching complexities should be prioritized, emphasizing the development of joint language models and context-aware algorithms that can adapt to evolving conversational scenarios. End-to-end modelling techniques should be explored for their potential to manage real-time code-switching more effectively, as they reduce dependencies on separate components like language identification modules. Additionally, incorporating multimodal data, including visual cues from lip movements and facial expressions, can significantly improve ASR performance in noisy or mixed-language settings. Moreover, promoting community-driven and open-source initiatives is essential for addressing linguistic diversity and advancing inclusive ASR technologies. Expanding these methodologies to other low-resource languages, such as Tamil, Gujarati, and Telugu, will enhance the generalizability of ASR research and ensure broader applicability across diverse linguistic landscapes. Finally, it is crucial to extend ASR research beyond Hindi and Marathi to encompass other low-resource languages, ensuring that advancements in ASR benefit a wider array of language communities and contribute to the preservation of linguistic diversity.

REFERENCES

[1] M. Bartelds, N. San, B. McDonnell, D. Jurafsky, and M. Wieling, "Making more of little data: Improving low-resource automatic speech recognition using data augmentation," in *Proc. 61st Annu. Meeting Assoc. Comput. Linguistics*, 2023, pp. 715–729.

[2] A. Safonova, T. Yudina, E. Nadimanov, and C. Davenport, "Automatic speech recognition of low-resource languages based on Chukchi," 2022, *arXiv:2210.05726*.

[3] J. Zhao, G. Shi, G.-B. Wang, and W.-Q. Zhang, "Automatic speech recognition for low-resource languages: The THUEE systems for the IARPA openasr20 evaluation," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2021, pp. 335–341.

[4] Z. Min, Q. Ge, and Z. Li, "MAC: A unified framework boosting low resource automatic speech recognition," 2023, *arXiv:2302.03498*.

[5] S. G. Bhable and C. N. Kayte, "Review: Multilingual acoustic modeling of automatic speech Recognition(ASR) for low resource languages," in *Proc. IEEE Int. Conf. Advent Trends Multidisciplinary Res. Innov. (ICATMRI)*, Dec. 2020, pp. 1–4.

[6] N. Alemu, C. Hua, P. H. Le, K. P. N. Nguyen, M. Ali, and N. Veilleux, "A framework to develop automatic speech recognition for low resource languages," in *Proc. 54th ACM Tech. Symp. Comput. Sci. Educ.*, Mar. 2023, p. 1228.

[7] T. Reitmaier, E. Wallington, D. K. Raju, O. Klejch, J. Pearson, M. Jones, P. Bell, and S. Robinson, "Opportunities and challenges of automatic speech recognition systems for low-resource language speakers," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, Apr. 2022, pp. 1–17.

[8] D. Kulshreshtha, S. Dingliwal, B. Houston, and S. Bodapati, "Multilingual contextual adapters to improve custom word recognition in low-resource languages," in *Proc. INTERSPEECH*, Aug. 2023, pp. 3302–3306.

[9] H. N. H. Nghia, N. T. H. Duy, N. G. Huy, N. M. M. Due, L. D. Luan, D. D. Hao, N. D. Dung, P. M. Hoang, and V. T. Hung, "Improving automatic speech recognition for low-resource language by data augmentation," in *Proc. 9th NAFOSTED Conf. Inf. Comput. Sci. (NICS)*, Oct. 2022, pp. 159–164.

[10] J. Kim, M. Kumar, D. Gowda, A. Garg, and C. Kim, "Semi-supervised transfer learning for language expansion of end-to-end speech recognition models to low-resource languages," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2021, pp. 984–988.

[11] S. Madikeri, P. Motlicek, and H. Bourlard, "Multitask adaptation with lattice-free MMI for multi-genre speech recognition of low resource languages," in *Proc. Interspeech*, Aug. 2021, pp. 4329–4333.

[12] S. Singh, F. Hou, and R. Wang, "A novel self-training approach for low-resource speech recognition," in *Proc. Interspeech*, Aug. 2023, pp. 1588–1592.

[13] H. Palivela, V. Rishiwal, S. Bhushan, A. Alotaibi, U. Agarwal, P. Kumar, and M. Yadav, "Optimization of deep learning-based model for identification of credit card frauds," *IEEE Access*, vol. 12, pp. 125629–125642, 2024.

[14] R. S. A. Pratama and A. Amrullah, "Analysis of whisper automatic speech recognition performance on low resource language," *Jurnal Pilar Nusa Mandiri*, vol. 20, no. 1, pp. 1–8, Mar. 2024.

[15] P. Yadav, V. Rishiwal, M. Yadav, A. Alotaibi, V. Maurya, U. Agarwal, and S. Sharma, "Investigation and empirical analysis of transfer learning for industrial IoT networks," *IEEE Access*, vol. 12, pp. 173351–173379, 2024.

[16] I. Mohamed and U. Thayasivam, "Low resource multi-ASR speech command recognition," in *Proc. Moratuwa Eng. Res. Conf. (MERCon)*, Jul. 2022, pp. 1–6.

[17] K. Soky, S. Li, C. Chu, and T. Kawahara, "Domain and language adaptation using heterogeneous datasets for Wav2vec2.0-based speech recognition of low-resource language," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[18] S. Sakti and B. A. Titalim, "Leveraging the multilingual Indonesian ethnic languages dataset in self-supervised models for low-resource ASR task," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2023, pp. 1–8.

[19] J. Silovsky, L. Deng, A. Argueta, T. Arvizo, R. Hsiao, S. Kuznietsov, Y.-C. Lin, X. Xiao, and Y. Zhang, "Cross-lingual knowledge transfer and iterative pseudo-labeling for low-resource speech recognition with transducers," 2023, *arXiv:2305.13652*.

[20] Q. Wang, W. Hu, L. Li, and Q. Hong, "Meta learning with adaptive loss weight for low-resource speech recognition," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Jun. 2023, pp. 1–5.

[21] A. Singh, A. S. Mehta, A. Khuraishi, G. Date, J. Nanavati, J. Bandekar, K. Basumatary, S. Badiger, S. Udupa, S. Kumar, Savitha, P. K. Ghosh, P. Pai, R. Nanavati, R. Saxena, S. P. R. Mora, and S. Raghavan, "Model adaptation for ASR in low-resource Indian languages," 2023, *arXiv:2307.07948*.

[22] J. Cao, Z. Qian, C. Yu, and T. Xie, "IFF-WAV2VEC: Noise robust low-resource speech recognition based on self-supervised learning and interactive feature fusion," in *Proc. 6th Artif. Intell. Cloud Comput. Conf. (AICCC)*, Dec. 2023, pp. 232–237.

[23] J. Yang, J. Wei, K. Khysru, J. Xu, W. Lu, W. Ke, and X. Yang, "Effective fine-tuning method for Tibetan low-resource dialect speech recognition," in *Proc. Asia–Pacific Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Oct. 2023, pp. 960–965.

[24] A. Dev and P. Bansal, "CTC-based end-to-end speech recognition for low resource language Sanskrit," in *Proc. 26th Conf. Oriental COCOSDA Int. Committee Co-Ordination Standardisation Speech Databases Assessment Techn. (O-COCOSDA)*, Dec. 2023, pp. 1–5.

[25] F. Mak, A. Govender, and J. Badenhorst, "Exploring ASR fine-tuning on limited domain-specific data for low-resource languages," *J. Digit. Humanities Assoc. Southern Afr. (DHASA)*, vol. 5, no. 1, pp. 1–12, Feb. 2024.

[26] P. Arisaputra, A. T. Handoyo, and A. Zahra, "XLS-R deep learning model for multilingual ASR on low-resource languages: Indonesian, Javanese, and Sundanese," 2024, *arXiv:2401.06832*.

[27] A. Yeroyan and N. Karpov, "Enabling ASR for low-resource languages: A comprehensive dataset creation approach," 2024, *arXiv:2406.01446*.

[28] T. Sereda, "Transcribe, align and segment: Creating speech datasets for low-resource languages," 2024, *arXiv:2406.12674*.

[29] C. Li and N. T. Vu, "Improving noisy student training for low-resource languages in end-to-end ASR using CycleGAN and inter-domain losses," in *Proc. SIGUL*, 2024, pp. 133–142.

[30] T. Verma, A. Shree, and A. Modi, "ASR for low resource and multilingual noisy code-mixed speech," in *Proc. Interspeech*, Aug. 2023, pp. 3242–3246.

[31] L. Kumar, D. Renuka, and N. K. S. Naveena, "Self-supervised language identification ASR models for low resource indic languages," in *Proc. Int. Conf. Modelling, Simulation, Appl. Optim. (ICMSAO)*, Dec. 2023, pp. 30–34.

[32] A. G. Ramakrishnan, "Unsupervised domain adaptation schemes for building ASR in low-resource languages," in *Proc. IEEE Autom. Speech Recognit. Understand. Workshop (ASRU)*, Dec. 2021, pp. 342–349.

[33] P. V. Thanh, L. D. Cuong, D. D. Huy, L. D. Thanh, N. D. Tan, D. T. D. Anh, and N. T. T. Trang, "ASR–VLSP 2021: Semi-supervised ensemble model for Vietnamese automatic speech recognition," *VNU J. Sci., Comput. Sci. Commun. Eng.*, vol. 38, no. 1, pp. 1–7, Jun. 2022.

[34] D. N. Krishna, P. Wang, and B. Bozza, "Using large self-supervised models for low-resource speech recognition," in *Proc. 22nd Annu. Conf. Int. Speech Commun. Assoc. (Interspeech)*, Brno, Czechia, Aug. 2021, pp. 2436–2440.

[35] W. Hou, H. Zhu, Y. Wang, J. Wang, T. Qin, R. Xu, and T. Shinozaki, "Exploiting adapters for cross-lingual low-resource speech recognition," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 30, pp. 317–329, Dec. 2021.

[36] Z. Song, Z. Ma, Y. Yang, J. Zhuo, and X. Chen, "A comparative study of LLM-based asr and whisper in low resource and code switching scenario," 2024, *arXiv:2412.00721*. Accessed: Jan. 10, 2025.

[37] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A simple data augmentation method for automatic speech recognition," 2019, *arXiv:1904.08779*.

[38] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "Wav2vec 2.0: A framework for self-supervised learning of speech representations," in *Proc. Adv. Neural Inf. Process. Syst.*, Jan. 2020, pp. 12449–12460.

[39] V. Pratap, A. Sriram, P. Tomasello, A. Hannun, V. Liptchinsky, G. Synnaeve, and R. Collobert, "Massively multilingual ASR: 50 languages, 1 model, 1 billion parameters," 2020, *arXiv:2007.03001*.

[40] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," 2019, *arXiv:1911.02116*.

[41] W.-N. Hsu, B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed, "HuBERT: Self-supervised speech representation learning by masked prediction of hidden units," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 29, pp. 3451–3460, 2021.

[42] F. Pérez-Verdugo, G. Reig, M. Cerda, M. L. Concha, and R. Soto, "Modelling of active contraction pulses in epithelial cells using the vertex model," 2021, *arXiv:2106.00824*.

[43] M. Ott, S. Edunov, A. Baevski, A. Fan, S. Gross, N. Ng, D. Grangier, and M. Auli, "Fairseq: A fast, extensible toolkit for sequence modeling," 2019, *arXiv:1904.01038*.

[44] F. Wessel, R. Schluter, and H. Ney, "Explicit word error minimization using word hypothesis posterior probabilities," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, vol. 1, Aug. 2001, pp. 33–36.

[45] C. Arriaga, A. Pozo, J. Conde, and A. Alonso, "Evaluation of real-time transcriptions using end-to-end ASR models," 2024, *arXiv:2409.05674*.

[46] S. Gondi and V. Pratap, "Performance and efficiency evaluation of ASR inference on the edge," *Sustainability*, vol. 13, no. 22, p. 12392, Nov. 2021.

[47] T. S. Nguyen, J. Niehues, E. Cho, T.-L. Ha, K. Kilgour, M. Müller, M. Sperber, S. Stueker, and A. Waibel, "Low latency ASR for simultaneous speech translation," 2020, *arXiv:2003.09891*.

[48] Q. Zhang, H. Lu, H. Sak, A. Tripathi, E. McDermott, S. Koo, and S. Kumar, "Transformer transducer: A streamable speech recognition model with transformer encoders and RNN-T loss," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 7829–7833.

[49] T. Verma, A. Shree, and A. Modi, "ASR for low resource and multilingual noisy code-mixed speech," in *Proc. 24th Annu. Conf. Int. Speech Commun. Assoc. (Interspeech)*, Dublin, Ireland, Aug. 2023, pp. 3242–3246.

[50] H. Liu, L. P. Garcia, X. Zhang, A. W. H. Khong, and S. Khudanpur, "Enhancing code-switching speech recognition with interactive language biases," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process. (ICASSP)*, Seoul, Republic of Korea, Apr. 2024, pp. 10886–10890.

[51] J. Lee, M. Mimura, and T. Kawahara, "Embedding articulatory constraints for low-resource speech recognition based on large pre-trained model," in *Proc. Interspeech*, Aug. 2023, pp. 1394–1398.

[52] Y. Chen, W. Zhang, H. Zhang, D. Qu, and X.-K. Yang, "Task-based meta focal loss for multilingual low-resource speech recognition," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 22, no. 11, pp. 1–17, Nov. 2023.

[53] S. Khurana, N. Dawalatabad, A. Laurent, L. Vicente, P. Gimeno, V. Mingote, and J. Glass, "Cross-lingual transfer learning for low-resource speech translation," in *Proc. IEEE Int. Conf. Acoustics, Speech, Signal Process. (ICASSP)*, Seoul, Republic of Korea, Apr. 2024, pp. 670–674.

[54] J. Zhao and W. Q. Zhang, "Improving automatic speech recognition performance for low-resource languages with self-supervised models," *IEEE J. Sel. Topics Signal Process.*, vol. 16, no. 6, pp. 1227–1241, Dec. 2022.

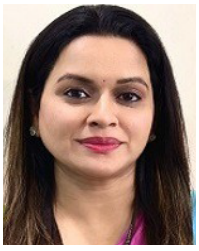
[55] S. K. Suresh and U. Damotharan, "Kannada-english code-mixed speech synthesis," in *Proc. Int. Conf. Signal Process. Commun. (SPCOM)*, Jul. 2024, pp. 1–5.

[56] S. Sinha, V. Spoorthy, and S. G. Koolagudi, "Code-switching automatic speech recognition using modified ESPNet," in *Proc. AIP Conf.*, 2023, vol. 2814.

[57] U. Pathak, C. S. K. Gunda, S. Sathiyamoorthy, K. Agarwal, and H. A. Murthy, "Everyday speech in the Indian subcontinent," 2024, *arXiv:2410.10508*.

[58] M.-H. Hsu, K. P. Huang, and H.-Y. Lee, "Meta-whisper: Speech-based meta-ICL for ASR on low-resource languages," 2024, *arXiv:2409.10429*.

HEMANT PALIVELA received the B.E. degree in computer engineering from the University of Mumbai, the M.Tech. degree from Visvesvaraya Technological University, and the Ph.D. degree in computer science engineering. He is a distinguished data science professional with a rich educational and professional background. He is also a Visiting Professor and a Research Guide with various foreign universities, such as Columbia, USA. His career includes significant roles in data science and AI, including positions at Accenture, eClerx Services, Aureus Analytics, and NMIMS University. He has made notable contributions to various areas of AI, machine learning, and data analytics, alongside achieving several patents and publishing numerous research articles. His technical skills encompass a range of programming languages, tools, and methodologies, making him a versatile and innovative leader in his field.



MEERA NARVEKAR received the Ph.D. degree in computer science and technology from SNDT Women's University. She is currently a Professor and the Head of the Department of Computer Engineering, D. J. Sanghvi College of Engineering, Mumbai. She was a Senate Member with the University of Mumbai. She is a member of the Honorable Board of Studies Computer Engineering, University of Mumbai. She has worked in computer engineering and technology for 24 years.

DAVID ASIRVATHAM received the B.Sc.Ed. (Hons.) and Master's Diploma degrees in computer science from the University of Malaya, the M.Sc. degree in digital systems from Brunel University, U.K., and the Ph.D. degree from Multimedia University. He has been lecturing and managing ICT projects for the past 30 years. He is currently a Professor of AI/Technology at the Asia School of Business, Kuala lampur, Malaysia. He has also worked as Executive Dean of the Faculty of Innovation and Technology, Taylor's University, Malaysia. His area of expertise will include digital neural networks and e-learning technologies, and recently he has done some work on big data analytics and AI.

SHASHI BHUSHAN received the Ph.D. degree in wireless sensor networks. Currently, he is a Senior Lecturer with the Department of Computer and Information Sciences (CIS), Universiti Teknologi PETRONAS, Malaysia. He has been associated with many reputed institutions, such as Amity University and UPES. He has over 13 years of experience in academics and research. His areas of specialization include but are not limited to wireless sensor networks, the IoT, AIML, and deep learning. He has delivered several guest lectures and seminars and chaired the session at various reputed international conferences. He has published more than 50 research articles and books in indexed journals and reputed conferences. He edited several journal special issues from the most famed publishers like Springer, *Bentham Science*, *Taylor and Francis*, and Elsevier. He is also serving as a reviewer and an editorial board member of many reputed journals.

VINAY RISHIWAL (Senior Member, IEEE) received the B.Tech. degree in computer science and engineering from M. J. P. Rohilkhand University (SRMSCET), Bareilly, Uttar Pradesh, India, in 2000, and the Ph.D. degree in computer science and engineering from Gautam Buddha Technical University, Lucknow, India, in 2011. He is currently a Professor with the Department of Computer Science and Information Technology, Faculty of Engineering and Technology, M. J. P. Rohilkhand University. He is also an Adjunct Professor with Taylor's University, Malaysia, and Nusa Putra University, Indonesia. He has 24 years of experience in academics. He has published more than 100 research papers in various journals and conferences of international repute. He also has 39 patents on his credit. He has received many awards as the best paper/a researcher/an orator on various platforms. He has visited many countries for academic purposes and worked on many projects of CST, UP Government, MHRD, and UGC. His current research interests include wireless sensor networks, the IoT, machine learning, and blockchain technology. He is a Senior Member of ACM, a fellow of IEI, and worked as a Convener of the Student Activities Committee, IEEE Uttar Pradesh Section, India. He is the General/Conference Chair of four international conferences, namely ICACCA, IoT-SIU, MARC 2020, and ICAREMIT.

UDIT AGARWAL received the M.Sc. degree in mathematics from M. J. P. Rohilkhand University, Bareilly, India, in 2001, and the Master of Computer Application (M.C.A.) degree from Uttar Pradesh Technical University, Lucknow, India, in 2004. He is currently the Dean of the Department of Computer Applications, RBMI Group of Institutions, Bareilly, and has more than two decades of academic experience. He has authored several prominent textbooks on *Computer Science* and *Mathematics*. He has contributed to numerous research publications in esteemed international journals, including IEEE, Wiley, Springer, Elsevier, and international conferences. His current research interests include blockchain technology, transfer learning, and supply chain management.