



Building an Intelligent Voice Assistant Using Open-Source Speech Recognition Systems

Venkata Baladari

Software Developer, Newark, Delaware, USA
Email ID: vrssp.baladari@gmail.com

Abstract: This study delves into designing intelligent voice assistants through the implementation of open-source speech recognition algorithms. Developers can build AI-powered voice interfaces by utilizing technologies such as Whisper, DeepSpeech, and Kaldi, enabling them to process spoken language, understand user goals, and produce responses. Open-source options provide flexibility and cost-effectiveness, but difficulties persist in terms of accuracy disparities, background noise disturbances, and the computational requirements that accompany them. Protecting user privacy and preventing unauthorized access requires secure data handling practices. Improvements in deep learning, multilingual capabilities, and bespoke customization can improve speech recognition performance and user interaction. Implementing emotion-sensing artificial intelligence and contextual memory capabilities can enhance conversational flow and responsiveness. Responsible AI deployment necessitates the resolution of ethical concerns. Future studies should concentrate on refining AI models to improve real-time processing capabilities and enhance their comprehension of context. Continuous enhancements enable open-source voice assistants to serve as a viable alternative to proprietary systems. This research focuses on important methods, obstacles, and possible improvements in the creation of voice assistants.

Keywords: Voice Assistants, Workflow, Intelligence, Frameworks, Interaction

Introduction

Evolution and Significance of Voice Assistants

The use of voice assistants has revolutionized human interaction with technology, allowing for hands-free, voice-driven communication methods. Systems like Siri, Alexa, and Google Assistant, which are powered by artificial intelligence, rely on sophisticated speech recognition and natural language processing capabilities to interpret and answer user requests [1],[2]. The integration of voice assistants into smartphones, smart homes, and business systems is becoming more widespread, underscoring their expanding role in contemporary digital networks.

Open-Source Speech Recognition and Its Impact on AI Assistants

Proprietary voice assistant solutions currently dominate the market, yet open-source speech recognition frameworks provide developers with a flexible and cost-effective means to develop custom AI assistants. Speech-to-text technologies including Mozilla DeepSpeech, Kaldi, and Whisper offer reliable functionality, allowing for innovation beyond the limitations of commercial platforms [3],[4],[5]. Customizable voice recognition capabilities enabled by open-source models allow researchers and developers to tailor, refine, and merge this technology into numerous applications, all while preserving transparency and management of data confidentiality.



Purpose and Scope of the Study

This research article seeks to investigate the methods and technologies employed in creating a bespoke voice assistant by utilizing open-source speech recognition frameworks. The study explores system architecture, implementation approaches, assesses performance, and addresses security concerns. In addition, the study also aims to offer developers a systematic method for constructing voice-driven interfaces that are tailored to users by examining existing frameworks and their usage.

2. Core Concepts of Voice Assistants

Understanding Voice Assistants and Their Functionality

This system is an artificial intelligence-driven assistant that interprets user voice commands and generates suitable responses or takes corresponding actions. Human-computer interactions are facilitated by these assistants through the use of speech recognition, natural language processing, and machine learning. Examples of commonly used assistants include Amazon Alexa, Apple Siri, and Google Assistant, which are utilized for a wide range of tasks, such as setting reminders and controlling smart home equipment [1],[2].

Essential Components of a Voice Assistant System

A seamless user experience with voice assistants is dependent on several interconnected components functioning in harmony with one another. A key feature of the system is the wake word detection capability, enabling the assistant to remain idle until a designated activation phrase, like "Hey Siri" or "Okay Google," is identified [7]. Upon activation, the system utilizes automatic speech recognition to convert spoken language into written text, facilitating further analysis [6]. The subsequent step entails the use of Natural Language Processing (NLP), wherein AI algorithms scrutinize the text, identify the user's intent, and establish suitable responses [8]. Ensuring contextual relevance is also a function of dialogue management, enabling a series of conversations to stay on topic. The last component, text-to-speech (TTS) synthesis, converts the AI-generated response into a voice that sounds naturally, thereby enhancing the interactive and engaging experience for users [9]. These components are frequently integrated with cloud-based services, APIs, and external databases in order to increase functionality and offer real-time access to information.

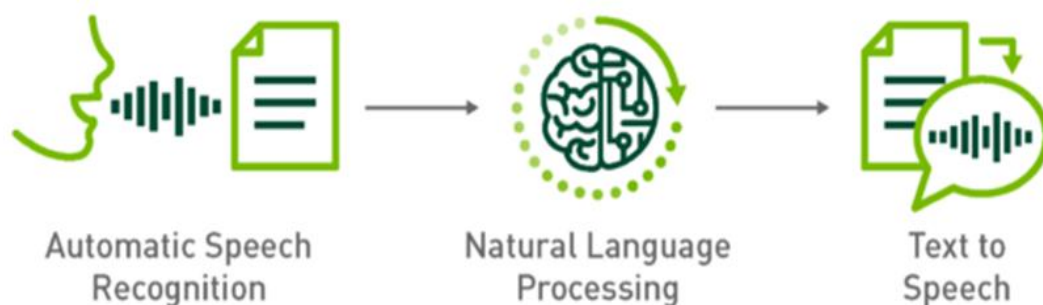


Figure 1: Pipeline for Text-To-Speech Conversational AI

(Accessed from <https://developer.nvidia.com/blog/how-to-deploy-real-time-text-to-speech-applications-on-gpus-using-tensorrt/>)

Comparing Speech Recognition and Speech Synthesis

The interaction between users and voice assistants is underpinned by two crucial technologies: speech recognition and speech synthesis. Automatic Speech Recognition (ASR) is tasked with transcribing spoken language into a format that can be read by computers [7]. This process relies on sophisticated AI algorithms that have been trained on expansive data collections to enhance precision in decoding various accents, speech cadences, and background noise fluctuations. In contrast, speech synthesis, often referred to as Text-to-Speech (TTS), allows voice assistants to produce speech responses with a natural, human-like quality [9]. Current TTS systems employ deep learning and neural networks to generate speech that sounds natural, expressive, and contextually relevant, thereby enhancing the user experience in AI interactions. Speech recognition enables



assistants to comprehend and manage user input, whereas speech synthesis upgrades the experience by facilitating AI-created responses that are more natural and realistic.

Challenges in Developing Custom AI Voice Assistants

Constructing a fully operational custom voice assistant still poses numerous difficulties despite the progress made in artificial intelligence and machine learning. Speech accuracy is a major concern, because differences in accents, dialects, and pronunciation can greatly affect recognition results. Background noise and simultaneous speech pose additional challenges to the system's accurate transcription of commands. Voice assistants face difficulties in understanding phrases with unclear meanings, conversational backgrounds, and users' individual preferences, which typically necessitates the use of advanced natural language understanding models to increase accuracy. Real-time processing is a crucial consideration for voice assistants, which need to provide rapid and pertinent responses while handling substantial computational demands. Ensuring the protection of user data and preventing unauthorized access are crucial for deploying AI in a manner that is ethically sound, as they pose substantial barriers to progress. Developers are increasingly required to provide multilingual support, necessitating the training of AI models on a diverse range of linguistic datasets in order to create assistants that serve users worldwide. To overcome these challenges, ongoing improvements in AI, data processing, and ethics are necessary to create more intelligent, dependable, and accessible voice assistants.

3. Structure and Workflow of a Custom Voice Assistant

Overview of System Architecture

A custom voice assistant's architecture is engineered to enable smooth communication between users and AI-powered systems. The system's foundation is built on a modular structure that combines multiple AI components, such as speech recognition, natural language processing, and response generation capabilities [7],[8],[9]. The architecture's typical composition involves several layers, starting with the capture of voice input, which is then followed by processing modules that convert spoken language into a structured format. The processed inputs are subsequently examined to ascertain the user's intent, produce relevant responses, and convey them through speech synthesis. The system can be deployed either as a standalone device-based assistant for offline functionality or a cloud-hosted model that utilizes advanced AI services to improve speech processing capabilities.

Key Elements of a Voice Assistant Workflow

The initial step in the workflow involves capturing spoken input through a microphone and then converting it into a format that can be understood by machines via Speech-to-Text processing [10]. Speech recognition models like Whisper, Kaldi, Mozilla DeepSpeech, and Vosk are often employed to simplify the process [3],[4],[5]. The accuracy of speech-to-text is impacted by various factors such as background noise, differences in accent, and the software's capacity to distinguish between words with similar sound patterns.

After speech is converted into text, Natural Language Understanding (NLU) algorithms examine the text to identify the underlying meaning and purpose[11]. This stage utilizes methods such as tokenization, Named Entity Recognition (NER), and sentiment analysis to accurately interpret user commands [12].

Upon comprehending the user's query, the system identifies the intended purpose. User input is matched to pre-defined intents, after which the system responds accordingly based on established logic or AI-driven algorithms. The assistant's decision-making abilities are enhanced by developers utilizing rule-based processing, knowledge graphs, and transformer-based models [19].

Converting the text output into spoken words is the last stage of the process. This procedure utilizes speech engines including Tacotron, Festival, Modular Architecture for Research on speech sYnthesis (MaryTTS) to produce natural-sounding voice outputs [13]. Contemporary Text-to-Speech systems also permit customized voice synthesis, allowing developers to create individualized voice interactions matched to particular applications.

4. Developing A Custom Voice Assistant: Step-By-Step Implementation

Configuring the Development Environment

Establishing a suitable development environment is essential prior to constructing a bespoke voice assistant. Developers need to pick suitable coding languages like Python or JavaScript and then install the required



frameworks and libraries. Essential dependencies comprise speech recognition engines, natural language processing libraries, and text-to-speech modules. Moreover, setting up cloud-based AI services or processing capabilities on a device guarantees a scalable and efficient workflow process. Frameworks like TensorFlow, PyTorch can be utilized to simplify the development process [16],[17].

Incorporating Open-Source Speech Recognition Technology

Any voice assistant is built upon the basis of speech recognition technology, which allows it to interpret spoken words as machine-readable text. Developers can improve the assistant's capacity to recognize a range of spoken dialects by incorporating open-source models like Whisper, Mozilla DeepSpeech or Kaldi [3],[4],[5]. These models need to be properly set up to manage differences in accents, languages, and background noise levels. Using additional datasets or transfer learning can also enhance the accuracy of these engines.

Implementing Activation and Wake Word Detection

A voice-activated system ensures the virtual assistant stays dormant until activated by a specific pre-set phrase, like "Hey Assistant" or "Okay AI." This feature minimizes unnecessary processing of unintended speech inputs, thereby optimizing resource efficiency [7]. Developers can use tools like Snowboy, or TensorFlow-based models to implement wake word detection [16],[18]. Fine-tuning these models ensures they accurately distinguish between genuine wake words and unintentional triggers.

Enhancing Speech Recognition with Model Training and Optimization

Training speech recognition models on domain-specific datasets is a crucial step to enhance their overall performance. Fine-tuning enables the system to adjust to multiple accents, sector-specific vocabulary, and often utilized expressions. Developers can employ supervised machine learning methods and dataset enhancement techniques to boost the model's resilience. Implementing noise reduction algorithms and speaker adaptation techniques improves recognition accuracy in real-world scenarios.

Integrating Natural Language Processing (NLP) for Context-Aware Conversations

A voice assistant should possess the ability to comprehend user intent in addition to speech recognition capabilities, and to sustain an understanding of conversational context. Developers can enhance the assistant's capabilities by incorporating Natural Language Processing (NLP) frameworks to better comprehend commands, identify user intent, and provide suitable responses. Meaningful and personalized interactions can be facilitated through the use of dialogue state management and memory retention, which enable the assistant to be contextually aware.

Generating Voice Responses with Text-to-Speech (TTS) Technology

To finish the conversational cycle, a voice assistant needs to change written responses into speech that sounds natural. Text-to-Speech (TTS) synthesis systems like Tacotron, Festival or MaryTTS models enable the production of expressive and human-like voice. Developers can personalize TTS voices by modifying tone, pitch, and speech rate to craft a distinct assistant character. TTS models employing deep learning techniques provide enhanced articulation and a more natural intonation, thus improving the overall user experience.

5. Advancing The Voice Assistant with AI And Machine Learning

Enhancing Speech Recognition Accuracy Using Deep Learning Models

A significant obstacle in voice assistant development is attaining high levels of speech recognition accuracy among various users, accents, and settings. Developers can substantially enhance transcription accuracy by utilizing deep learning-based speech models. Transformer-based models, like Whisper and Recurrent Neural Networks (RNN) with Long Short-Term Memory (LSTM), specifically RNNs utilizing LSTM, play a key role in enhancing automatic speech recognition systems [19],[20],[21]. Pre-trained models can be fine-tuned with domain-specific data to enhance recognition capabilities in specific applications using transfer learning techniques. In addition, self-learning artificial intelligence models can continually adjust to user interactions, thereby improving their performance over time.

Customizing Voice Assistants Through User-Centric Data Adaptation

To deliver a more tailored experience, a voice assistant needs to acquire knowledge from a user's unique preferences and speech characteristics. Reinforcement learning and personalized intent recognition enable machine learning techniques to facilitate adaptive responses from assistants based on prior conversations. Developers can build dynamic AI models by gathering and examining anonymized speech samples and user



behavior patterns, allowing them to craft responses that cater to an individual's distinct needs. Voice biometrics capabilities can be used to refine personalization through speaker recognition and adaptive responses. To safeguard user data security, federated learning and other privacy-conscious methods should be implemented to prevent personalization from undermining data protection.

Expanding Speech Recognition with Multi-Language Capabilities

As global demand continues to rise, voice assistants need to be able to handle multiple languages and dialects. Providing support for multiple languages requires training AI models using a wide range of linguistic data collections to identify and analyze speech from speakers with varied language origins. Technologies such as Multilingual BERT and Wav2Vec speech recognition contribute to enhanced language adaptability [22],[23]. Language switching in assistants can be smoothly implemented by AI-driven technology to switch languages within a conversation. To boost real-time multilingual communication, developers can incorporate speech-to-speech translation tools, allowing users to converse in their native language without manually adjusting their settings.

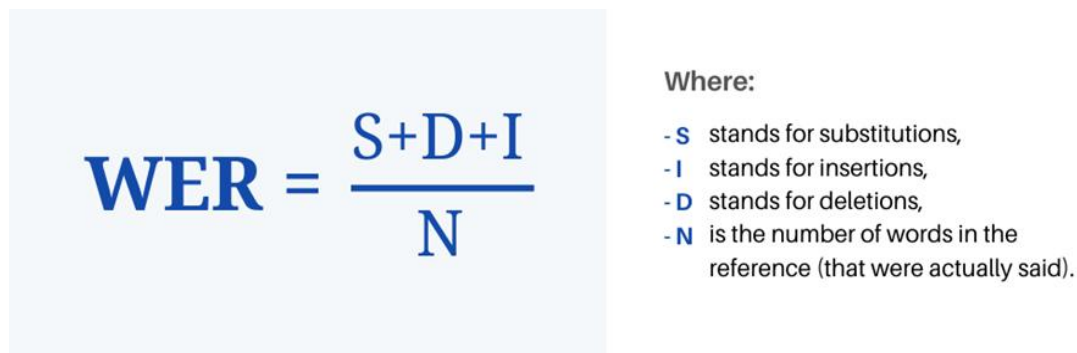
Enhancing Voice Assistants in Noisy Environments with Advanced Filtering Techniques

Poor audio quality and background noise can severely affect a voice assistant's capacity to accurately interpret voice commands. Developers employ sophisticated noise cancellation, speech enhancement, and audio pre-processing methods to enhance voice recognition capabilities. Spectral subtraction, deep neural network-based noise suppression, and adaptive filtering methods assist in eliminating ambient noise while maintaining speech clarity. Echo cancellation and microphone array processing also improve speech capture in environments with multiple speakers. Ensuring voice assistants remain reliable involves the implementation of AI-driven noise reduction methods in environments with difficult acoustic conditions like public areas or noisy homes.

6. Assessing Performance and Benchmarking for Voice Assistants

Measuring the Accuracy of Speech Recognition Systems

Assessing the performance of a voice assistant commences with examining its capacity for speech recognition accuracy. A widely used metric for evaluation is the Word Error Rate (WER), which determines the percentage of misrecognized words in a provided speech input [24]. Furthermore, Phoneme Error Rate (PER) and Sentence Error Rate (SER) aid in measuring the accuracy of pronunciation and the effectiveness of sentence-level recognition [25],[26]. Evaluating speech recognition models under diverse conditions, including different accents, varying background noise levels, and different speaking speeds, leads to a more robust assessment. Developers can utilize open datasets and real-world user interactions to improve speech models and lower recognition errors via ongoing learning methods.



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

Where:

- S stands for substitutions,
- I stands for insertions,
- D stands for deletions,
- N is the number of words in the reference (that were actually said).

Figure 2: Word Error Rate Formula

(Accessed from <https://www.gmrtranscription.com/blog/word-error-rate-mechanism-asr-transcription-and-challenges-in-accuracy-measurement>)

Optimizing Response Time and Reducing Latency

A proficient voice assistant should handle user requests quickly and provide relevant answers without significant lag. Delays in voice assistant interactions can stem from processing voice delays, slow natural language comprehension, or network communication problems when utilizing cloud-based artificial intelligence



services. Developers can improve response time by implementing speech recognition on the device itself, thereby decreasing reliance on remote servers. Implementing model quantization and hardware acceleration can significantly accelerate computations, thereby enhancing the smoothness of interactions. Data caching and asynchronous processing systems also play a role in speeding up response times, ultimately enhancing the user's overall experience.

Enhancing User Experience Through Continuous Testing and Refinement

The level of satisfaction achieved by users is a vital element in determining the success of a voice assistant. Conducting regular usability testing enables the identification of areas for enhancement including voice clarity, context awareness, and conversational flow. Software engineers employ A/B testing, user polls, and practical application testing to collect feedback and adjust the system as needed. Analyzing sentiment in voice interactions can yield valuable insights into an assistant's ability to comprehend and respond to user emotions. The voice assistant can evolve over time by incorporating user input and implementing adaptive learning techniques, which will enable it to provide a more user-friendly and interactive experience.

7. Security, Privacy, And Ethical Considerations In Voice Assistants

Ensuring Data Privacy in Voice-Enabled Systems

The growing reliance on voice assistants has sparked concerns about data protection, as these technologies handle and retain sensitive user data. Developers need to incorporate privacy-by-design principles in order to safeguard user data, with the aim of reducing data collection and securely managing it. Whenever feasible, voice assistants should utilize on-device processing to decrease their dependence on cloud storage and lower the potential for privacy breaches. Users should have the capability to oversee and regulate their own data, including options to remove stored voice recordings and adjust their consent settings.

Preventing Unauthorized Data Collection and Misuse

A major security issue with voice assistants is the potential for unauthorized data access and harvesting. Processing voice data only when explicitly initiated by a wake word reduces the risk of accidental recordings occurring. Voice data access can be monitored using anomaly detection algorithms to detect and mark unusual activity. Developers can also employ differential privacy techniques, which introduce a specified amount of controlled noise to datasets in order to safeguard individual user identities, thus still allowing for AI model enhancements. Conducting regular audits and maintaining transparent privacy policies boosts user trust in how their data is collected, stored and managed.

8. Conclusion

Developing a custom voice assistant with open-source speech recognition models can be an efficient and budget-friendly method for creating AI-powered voice interfaces. Tools like Whisper, DeepSpeech, and Kaldi assist in processing speech, interpreting intent, and creating voice responses. These systems improve user interaction, but they still face difficulties such as inconsistent accuracy, background noise interference, and high computational requirements. Extensive training is necessary for speech models to identify a wide range of accents and languages, which complicates real-time processing unless using sophisticated computer hardware. User data protection requires addressing security and privacy issues to prevent unauthorized access. Developing a deeper understanding of natural language is essential for building conversations that are seamless and responsive to context. Future developments will concentrate on multilingual capabilities, tailored experiences, and enhanced privacy safeguards. Implementing emotion-aware artificial intelligence methods can improve user experience and bolster security. Responsible AI development necessitates prioritizing ethical considerations. As advancements continue, open-source voice assistants have the potential to serve as viable substitutes for commercial voice assistant options.

References

- [1]. D. Singh, G. Kaur, and A. Bansal, "How voice assistants are taking over our lives - A review," J. Emerg. Technol. Innov. Res. (JETIR), vol. 6, no. 5, pp. 355, May 2019. [Online]. Available: www.jetir.org



- [2]. E. Moriuchi, "Okay, Google!: An empirical study on voice assistants on consumer engagement and loyalty," *Psychol. Mark.*, vol. 36, pp. 489–501, 2019. doi: 10.1002/mar.21192.
- [3]. R. Bhat, P. Chandran, J. Jose, V. Dibbur, and P. S. Ajith, "NTP: A Neural Network Topology Profiler," arXiv preprint arXiv:1905.09063, 2019. [Online]. Available: <https://arxiv.org/abs/1905.09063>.
- [4]. M. Ravanelli, T. Parcollet, and Y. Bengio, "The PyTorch-Kaldi Speech Recognition Toolkit," arXiv preprint arXiv:1811.07453, 2019. [Online]. Available: <https://arxiv.org/abs/1811.07453>.
- [5]. M. Jacoby, S. Y. Tan, M. Katanbaf, A. Saffari, H. Saha, Z. Kapetanovic, J. Garland, A. Florita, G. Henze, S. Sarkar, et al., "WHISPER: Wireless Home Identification and Sensing Platform for Energy Reduction," *J. Sensor Actuator Netw.*, vol. 10, no. 4, p. 71, 2021. [Online]. Available: <https://doi.org/10.3390/jsan10040071>.
- [6]. R. Errattahi, A. El Hannani, and H. Ouahmane, "Automatic speech recognition errors detection and correction: A review," *Procedia Computer Science*, vol. 128, pp. 32-37, 2018. DOI: 10.1016/j.procs.2018.03.005.
- [7]. G. Milsap, M. Collard, C. Coogan, Q. Rabbani, Y. Wang, and N. E. Crone, "Keyword spotting using human electrocorticographic recordings," *Frontiers in Neuroscience*, vol. 13, 2019. Available: <https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2019.00060>. doi: 10.3389/fnins.2019.00060.
- [8]. I. Das, B. Sharma, S. S. Rautaray, and M. Pandey, "An Examination System Automation Using Natural Language Processing," 2019 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2019, pp. 1064-1069, doi: 10.1109/ICCES45898.2019.9002048.
- [9]. M. Cotescu, T. Drugman, G. Huybrechts, J. L. Trueba, and A. Moinet, "Voice conversion for whispered speech synthesis," *IEEE Signal Processing Letters*, 2020.
- [10]. A. Trivedi, N. Pant, P. Shah, S. Sonik, and S. Agrawal, "Speech to text and text to speech recognition systems - A review," *IOSR Journal of Computer Engineering (IOSR-JCE)*, vol. 20, no. 2, pp. 36–43, Mar.–Apr. 2018. DOI: 10.9790/0661-2002013643.
- [11]. E. D. Liddy, "Natural Language Processing," in *Encyclopedia of Library and Information Science*, 2nd ed., New York, NY: Marcel Decker, Inc., 2001.
- [12]. A. Goyal, V. Gupta, and M. Kumar, "Recent Named Entity Recognition and Classification techniques: A systematic review," *Comput. Sci. Rev.*, vol. 29, pp. 21–43, 2018, doi: 10.1016/j.cosrev.2018.06.001.
- [13]. Y. Zhang, R. J. Weiss, H. Zen, Y. Wu, Z. Chen, R. J. Skerry-Ryan, Y. Jia, A. Rosenberg, and B. Ramabhadran, "Learning to speak fluently in a foreign language: Multilingual speech synthesis and cross-language voice cloning," arXiv preprint arXiv:1907.04448, 2019.
- [14]. P. Stavropoulou, D. Tsonos, and G. T. Kouroupetroglou, "Language resources and evaluation for the support of the Greek language in the MARY TtS," 2014.
- [15]. Y.-P. P. Chen, C. Johnson, P. Lalbakhsh, T. Caelli, G. Deng, D. Tay, S. Erickson, P. Broadbridge, A. E. Refaie, W. Doube, and M. E. Morris, "Systematic review of virtual speech therapists for speech disorders," *Comput. Speech Lang.*, vol. 37, pp. 98-128, 2016. doi: 10.1016/j.csl.2015.08.005.
- [16]. B. Pang, E. Nijkamp, and Y. N. Wu, "Deep Learning With TensorFlow: A Review," *J. Educ. Behav. Stat.*, vol. 45, no. 2, pp. 227–248, 2020, doi: 10.3102/1076998619872761.
- [17]. A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," arXiv preprint arXiv:1912.01703, 2019.
- [18]. M. Duguleană, V.-A. Briciu, I.-A. Duduman, and O. M. Machidon, "A virtual assistant for natural interactions in museums," *Sustainability*, vol. 12, no. 17, p. 6958, 2020. doi: 10.3390/su12176958.
- [19]. Y. Wang et al., "Transformer-Based Acoustic Modeling for Hybrid Speech Recognition," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Barcelona, Spain, 2020, pp. 6874-6878, doi: 10.1109/ICASSP40776.2020.9054345.
- [20]. R. M. Schmidt, "Recurrent Neural Networks (RNNs): A gentle Introduction and Overview," arXiv preprint arXiv:1912.05911, 2019. [Online]. Available: <https://arxiv.org/abs/1912.05911>.



- [21]. X.-H. Le, H. V. Ho, G. Lee, and S. Jung, "Application of Long Short-Term Memory (LSTM) neural network for flood forecasting," *Water*, vol. 11, no. 7, p. 1387, 2019. doi: 10.3390/w11071387.
- [22]. T. Pires, E. Schlinger, and D. Garrette, "How multilingual is Multilingual BERT?" arXiv preprint arXiv:1906.01502, 2019. [Online]. Available: <https://arxiv.org/abs/1906.01502>.
- [23]. S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised pre-training for speech recognition," arXiv preprint arXiv:1904.05862, 2019. [Online]. Available: <https://arxiv.org/abs/1904.05862>.
- [24]. R. Errattahi, A. El Hannani, and H. Ouahmane, "Automatic speech recognition errors detection and correction: A review," in *Proc. Int. Conf. Nat. Lang. Speech Process. (ICNLSP)*, El Jadida, Morocco, 2018.
- [25]. S. Yolchuyeva, G. Németh, and B. Gyires-Tóth, "Grapheme-to-Phoneme Conversion with Convolutional Neural Networks," *Appl. Sci.*, vol. 9, no. 6, p. 1143, 2019. doi: 10.3390/app9061143.
- [26]. Y. Weng, S. S. Miryala, C. Khatri, R. Wang, H. Zheng, P. Molino, M. Namazifar, A. Papangelis, H. Williams, and F. Bell, "Joint contextual modeling for ASR correction and language understanding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, 2020.

