ChatGPT for Visually Impaired and Blind

Askat Kuzdeuov
Institute of Smart Systems and AI
Nazarbayev University
Astana, Kazakhstan
askat.kuzdeuov@nu.edu.kz

Olzhas Mukayev
Institute of Smart Systems and AI
Nazarbayev University
Astana, Kazakhstan
olzhas.mukayev@nu.edu.kz

Shakhizat Nurgaliyev
Institute of Smart Systems and AI
Nazarbayev University
Astana, Kazakhstan
shakhizat.nurgaliyev@nu.edu.kz

Alisher Kunbolsyn
Institute of Smart Systems and AI
Nazarbayev University
Astana, Kazakhstan
alisher.kunbolsyn@nu.edu.kz

Huseyin Atakan Varol
Institute of Smart Systems and AI
Nazarbayev University
Astana, Kazakhstan
ahvarol@nu.edu.kz

***Abstract*—According to the World Health Organization (WHO), hundreds of million people have some type of visual disability. Vision impairment has a personal impact with lifelong consequences because more than 80% of our perception, cognition, learning, and daily activities are mediated through vision. Moreover, in the era of rapid advancements in artificial intelligence (AI), visually impaired and blind people face challenges at work and in education because of inaccessibility to AI technologies. In this regard, we present an assistive mobile application with an intuitive user interface (UI) for visually impaired and blind people to interact with ChatGPT via natural conversation. The app employs automatic speech recognition (ASR), text-to-speech (TTS), keyword spotting (KWS), voice activity detection (VAD), and a convenient UI to interact with ChatGPT effortlessly. We have made the source code, pre-trained models, and UI publicly available at https://github.com/IS2AI/talk-llm to stimulate the development of assistive mobile applications.**

I. Introduction

According to the World Health Organization (WHO), around 1.3 billion people, about one-sixth of the world's population, have some form of disability. Of these, around 253 million people are visually impaired, of whom about 36 million are blind [1]. In terms of socioeconomic disadvantages, blind and visually impaired individuals face a number of challenges in education, employment, and income. According to a study conducted by the National Federation of the Blind, only 10% of blind children in the United States are literate, compared to more than 90% of sighted children [2]. In terms of employment, a study conducted in the United States found that the unemployment rate for blind people is nearly double that of the general population [3]. Additionally, blind individuals are more likely to work in low-paying jobs with limited opportunities for advancement [4]. They are also more likely to live in poverty and are less likely to own their own homes.

Blind people access information through a variety of assistive technologies, such as screen readers, Braille displays, and audiobooks [5]. Braille displays are devices that convert text into Braille, a system of raised dots that can be read by touch. Screen readers are software programs that read text on a computer or mobile device aloud, allowing blind users to access websites, documents, and other digital content. Blind people can use search engines (e.g., Google) to find information using screen reader software that works with search engines. Some screen readers also have built-in support for Google search, allowing users to perform a search and navigate the results directly within the screen reader software. Nonetheless, access to information in the digital world is cumbersome for the blind and visually impaired. For instance, many websites and documents are not designed to be accessible to screen readers. Often, blind people have to rely on the help of others to access information, which limits their independence and autonomy and has a detrimental effect on their psychological well-being.

Speech has been an essential component of human communication from the earliest times [6]. As human societies evolved and became more complex, language and speech became increasingly important for social interactions, such as negotiation and storytelling. With the introduction of hieroglyphs and cuneiforms, writing has allowed for the preservation of knowledge and cultural traditions and has boosted the abstract thinking abilities of humankind. However, recent advances in artificial intelligence led to the emergence of large-scale language models (LLMs) [7]. For instance, ChatGPT [8] and GPT-4 [9] have demonstrated state-of-the-art performance on a wide range of natural language processing (NLP) tasks such as language translation, text summarizing, question answering, programming, and many others. These models have been pretrained on massive amounts of text data and optimized for dialogues using reinforcement learning from human feedback (RLHF). LLMs with billions of parameters can process natural language at an unprecedented level and achieve comparable or even superior results than human experts in some tasks.

ChatGPT and GPT-4 have the potential to serve as assistive technologies, providing access to information for individuals who are blind or visually impaired. OpenAI has also introduced GPT-4V [10], an enhanced version of GPT-4 with image analysis capabilities. In 2023, OpenAI and Be My Eyes launched a novel tool called Be My AI, designed to illustrate the visual world for visually impaired and blind

individuals [10]. The tool was integrated with GPT-4V into the current Be My Eyes platform, offering explanations of images captured by the user's smartphone. Nearly 16,000 blind and low-vision individuals tested Be My AI, aiding in refining the product's safety and overall user experience. However, Be My AI was accessible only to the beta testers at the time of writing this paper.

OpenAI provides a website and a mobile app (iOS, Android) to engage with ChatGPT. Visually impaired and blind users can utilize screen reader software to access these platforms. Also, OpenAI has recently incorporated a new feature into the ChatGPT mobile app that allows users to communicate through image and voice messages[1]. The voice feature is free, but the image feature is accessible only for Plus and Enterprise users. The voice feature utilizes Whisper [11] to convert spoken language into written text and a text-to-speech model that closely resembles human speech. However, the voice feature utilizes neither KWS nor speaker recognition to activate the conversation when needed. Therefore, it processes any speech inputs, including background conversations. It might impact the conversation because ChatGPT relies on previous information to respond to the present input.

Another concern is data privacy because all requests (text, voice, image) are processed in OpenAI's servers. ChatGPT logs and saves all conversation transcripts, including any personal or confidential information inputted, which may inadvertently leak personal or work details. Another concern is accessibility. OpenAI offers free access to ChatGPT on web and mobile platforms, but this doesn't necessarily ensure stable usage. To have assured, uninterrupted access, a paid subscription is required. As of the writing of this paper, GPT-4 was only available to paying subscribers at a monthly rate of $20. This cost could potentially be inaccessible for visually impaired and blind individuals residing in developing countries due to its relatively high cost. Furthermore, only a limited number of developers had access to the GPT-4V.

In an attempt to address these restrictions, open-source alternatives have been emerging. For instance, Meta AI has introduced LLaMA, which consists of a set of foundational language models with parameter ranges from 7 billion to 65 billion [12]. These models were trained on vast amounts of publicly available data and have achieved state-of-the-art performance on various benchmark datasets. Subsequently, Meta AI launched Llama 2 with a range of pretrained and fine-tuned LLMs with parameters ranging from 7 billion to 70 billion [13]. The fine-tuned LLMs, known as Llama 2-Chat, were optimized for dialogue-based scenarios. These models have exhibited superior performance compared to other open-source chat models across multiple benchmarks. Subsequently, the open-source community implemented fast inference of Llama 2 in pure C/C++ with 2-bit, 3-bit, 4-bit, 5-bit, 6-bit, and 8-bit integer quantization [14]. As a result, these optimized models can run locally on a MacBook and an Android device. Recently, a large language and vision assistant, LLaVA [15], was released as an open-source alternative for GPT-4V. The model attains top-notch performance across 11 benchmark datasets and demonstrates remarkable chat capabilities, effectively emulating the essence of GPT-4V. The open-source community also optimized LLaVA with 4-bit and 5-bit quantization [14]. Another powerful open-source visual language model, CogVLM-17B [16], with 10 billion vision parameters and 7 billion language parameters, was released. The model achieved top performance on ten classic cross-modal benchmarks.

In this work, we developed a mobile app to make these technologies accessible for the visually impaired or blind. This app facilitates hassle-free interaction with ChatGPT through natural speech by utilizing automatic speech recognition (ASR), text-to-speech (TTS), keyword spotting (KWS), and voice activity detection (VAD), running these models locally on a smartphone to safeguard data privacy. Given the recent progress in open-source LLMs, it's foreseeable that ChatGPT could be substituted soon with an optimized LLM that operates locally. Our source code facilitates easy replacement of ChatGPT with a lightweight, locally running LLM model for future developments.

ASR is a technology that transforms spoken language into readable text. We use ASR to convert verbal user requests into text format. KWS is a subcategory of speech recognition that specializes in identifying specific voice commands within a limited vocabulary and context. This technology allows efficient processing, lower complexity, and quicker response times. KWS is utilized in voice assistants such as Amazon's Alexa and Apple's Siri to detect initiation commands. We employ a lightweight and energy-saving KWS model that activates the system only when required, thereby conserving device power. In the case of using cloud-based ASR and TTS systems, KWS assists in preventing the transmission of unintended inputs, making it a cost-effective tool for chargeable ASR and TTS models. VAD is another technology we utilize, which identifies human speech within an audio signal and separates it from the background noise. It detects when a voice request ends and initiates the processing of the request with ASR. TTS is technology that transforms written text into audible language. We apply TTS to convert ChatGPT's text responses into speech. Our main contributions are as follows:

- We developed a mobile app for the visually impaired and blind to interact with ChatGPT via natural conversation.
- We presented an effective method of creating a synthetic speech dataset to train a custom KWS model.
- We measured the real-time performance of the employed speech models by deploying them on a real mobile phone.
- We open-sourced our code, models, and UI to stimulate the development of assistive mobile applications.

This paper is structured as follows: In Section II, we delve into the architectural design of our mobile app. Section III provides the details of the KWS model development. In Section IV, we discuss the results and limitations of our work. Finally, we conclude our work in Section V.

[1]https://openai.com/blog/chatgpt-can-now-see-hear-and-speak

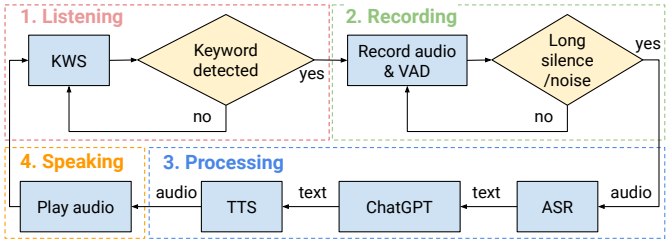

Fig. 1. System architecture of the app: 1) Persistent audio monitoring to discern a triggering keyword, 2) Capturing user dialogue and identifying its conclusion using VAD, 3) Handling requests through ASR, ChatGPT, and TTS conversion, and 4) Vocalizing ChatGPT's generated reply.

II. Methodology

The architecture of our system is illustrated in Figure 1. It comprises four main modules: Listening, Recording, Processing, and Speaking. The system activates the components sequentially. Initially, the system operates in the Listening module, where it continuously runs a tiny KWS model to detect a specific activation keyword uttered by a user. Until the keyword is recognized, the other components remain inactive. Once the KWS model detects the keyword, the Recording module is triggered, and the KWS model is deactivated. The Recording module starts writing the user's spoken words. The system utilizes VAD to determine when the user has finished speaking based on a predefined duration of silence or noise. Then, it ceases recording and switches off the microphone and VAD. Subsequently, the Processing module is engaged, where an ASR model transcribes the user's verbal request into text. Then, the text is processed by ChatGPT, which formulates a response that is transformed back into audio via a TTS model. Finally, the system's last component, Speaking, takes over by enabling the device's speaker to output the ChatGPT response audibly. After that, the system turns off the speaker and reverts to the Listening mode, turning on the microphone and starting the KWS model for the next iteration.

A. Keyword Spotting

KWS focuses solely on detecting a designated activation word. Usually, the keyword is selected deliberately to initiate the system. For example, Apple's activation word is "Hey, Siri," Google uses "Ok, Google," and Amazon uses "Alexa" as their wake-up word. In these systems, the input audio stream is divided into segments or "windows," and the KWS model classifies each window to one of two categories: 1) the specified keyword or 2) other sounds. It is important to include a wide variety of spoken words and noises within the "other sounds" class to enhance the robustness of the model's ability to detect the keyword.

For our system, we have selected the term "ISSAI" as the wake-up word. This abbreviation represents the name of our institute - the Institute of Smart Systems and Artificial Intelligence (ISSAI). To streamline the process of collecting speech data for the targeted word, we leveraged the Piper text-to-speech system [17] to generate synthetic data. This approach showed efficacy in training a KWS model [18], as manual collection would have been time-consuming and labor-intensive. In this way, it becomes feasible to collect speech data for any desired word, enabling the development of a personalized wake-up word detector. Piper provides TTS models for about thirty languages. We utilized three TTS models with the highest number of speakers. The models were trained using English datasets representing American and British accents. This made it possible to generate speech data with variations in pronunciation and intonation, resulting in a more diverse and representative dataset for our wake-up word detection system.

The first TTS model includes the voices of 904 speakers trained on the LibriTTS corpus [19]. The second model, trained on the LibriTTS-R corpus [20], also offers voices from 904 speakers. The last model was trained on the VCTK corpus [21], consisting of voices from 109 speakers with British accents. As a result, we generated 1,917 synthetic utterances, with a sampling rate of 16 kHz, for the word "ISSAI". Additionally, to generate data for other sounds, we employed 1,000 most frequently used English words[2]. For each speaker, we randomly selected ten words from the list and generated ten utterances by applying the corresponding model. This methodology enabled us to generate a substantial total of 19,170 utterances with a 16 kHz sampling rate for the "other sounds" class. We deliberately produced ten times more samples for the "other sounds" class compared to the "ISSAI" class. This strategy challenges the model's ability to detect the specific keyword "ISSAI" amidst a vast array of other sounds.

B. Voice Activity Detection

VAD, also known as speech activity detection or speech detection, is a technique utilized in speech processing to identify and detect the presence or absence of human speech in an audio signal. It is commonly employed to segment audio into the regions of speech activity and silence, enabling more effective analysis and processing of the speech. We utilized VAD to detect the absence of human speech to stop audio recording and then start processing the recorded audio with ASR. VAD could alternatively be used to prompt the system rather than using KWS. However, this approach could produce unintended system triggers as VAD would activate not just for the desired keyword but whenever it detects conversational background noise or any non-target speech from the user.

In this project, we utilized a pre-trained, enterprise-grade voice activity detector named Silero-VAD [22]. This model is popular in VAD applications, especially for IoT, edge, and mobile use cases. This popularity is due to its lightweight design with just 250,000 parameters and its small size, approximately two megabytes. Further enhancing its usability, the model supports over a hundred languages and delivers impressive accuracy on established VAD benchmark datasets. It also integrates smoothly with major ecosystems like PyTorch and

[2]https://gist.github.com/deekayen/4148741

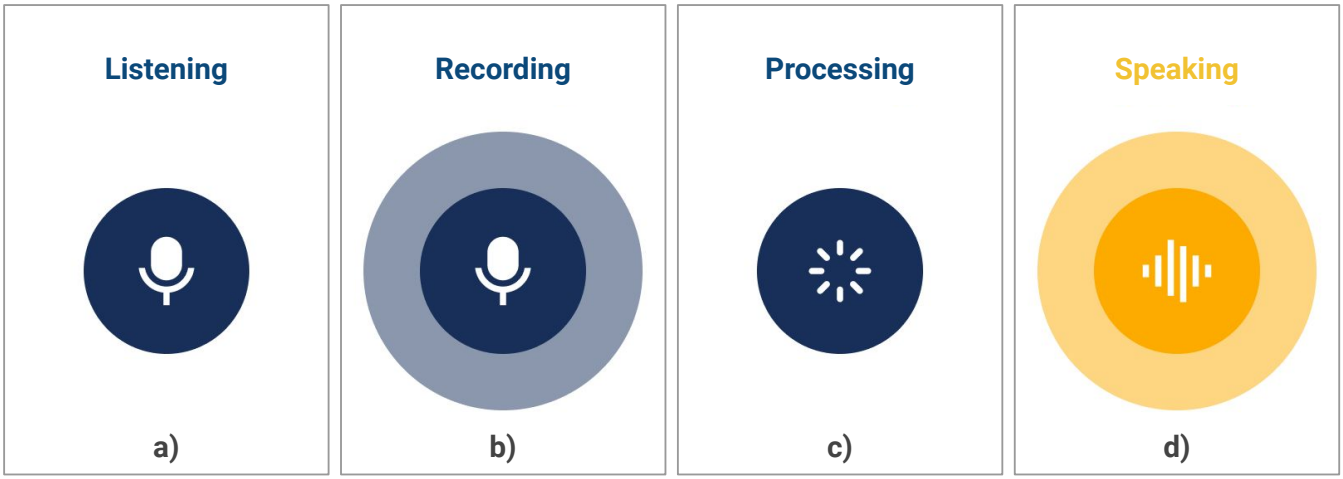

Fig. 2. UI consists of four main parts: a) Real-time detection of a wake-up word, b) Recording user speech either upon wake-word detection or pressing the mic button, c) Processing user's request via ASR, ChatGPT, and TTS, and d) Vocalizing the ChatGPT response.

ONNX, allowing operations on platforms that support these runtimes. Moreover, its distribution under the MIT license ensures unrestricted use.

C. Automatic Speech Recognition

In this work, we used Whisper [11] for a speech recognition task similar to ChatGPT's original app. Whisper offers models with varying parameters, including tiny (39M), base (74M), small (244M), medium (769M), and large (1550M). Whisper offers multilingual models and models trained solely in English. Nonetheless, operating these models, even the smallest one, demands substantial computational resources. Considering the deployment on mobile phones, we used the Whisper.cpp library [23]. It is an optimized version of Whisper designed for high-performance inference. It executes the fundamental tensor operations in C to achieve efficient inference. Also, the library offers quantized models with 4-bit and 5-bit integer quantization. These quantized models occupy less memory and disk space. As a result, they can be processed more efficiently. We used a tiny Whisper model (75MB) to get the fastest inference on mobile phones. The model was available in the BIN format and under the MIT license. All models can be found here[3].

D. Text-To-Speech

We utilized Piper text-to-speech system to convert the responses of ChatGPT into speech. Piper provides TTS models for about thirty languages. The models were trained using VITS [24] and optimized by exporting to the ONNX format. Also, Piper presents TTS models in four distinct quality tiers: x-low, low, medium, and high. The x-low and low models trained on audio data with a 16 kHz sampling rate. The x-low and low models have 5-7M and 15-20M parameters, respectively. The medium and high models were trained on audio data with a 22.05 kHz sampling rate. The medium models have 15-20M parameters, while the high models comprise 28-32M parameters. We utilized a low model featuring an English female speaker's voice. The model was in the ONNX format and had a size of 63MB[4]. The model was available under the Creative Commons Attribution Share Alike 4.0 International license. All available models can be found here[5].

E. User Interface

We designed the UI of the app in Figma. The main pages are illustrated in Fig. 2. In addition to facilitating natural conversation, the UI empowers users who are visually impaired or blind to operate the app through tactile feedback and audible responses. The "Listening" page, seen in Fig. 2a, includes a sizable microphone button at the center. A user clicks and holds this button to start audio recording. Once activated, the "Recording" page becomes visible, as shown in Fig. 2b. The system provides tactile and sound feedback when the button gets pressed. Once the microphone button is released, the system provides sound feedback to notify the user that the recording has been captured and is being processed, as depicted in Fig. 2c. The user can halt this process by double-clicking on the processing button. Upon completion of processing, the "Speaking" page, which includes a central player button, is displayed. The user can start or halt the playback by clicking this button. To return to the "Listening" page, the user must swipe upward on the "Speaking" page.

The transition between the pages happens automatically during the voice-only interaction. On the "Listening" page, the KWS model operates in the background. When the user says the keyword "ISSAI", the system automatically starts recording audio without requiring a press on the microphone button and consequently switches to the "Recording" page. Once the VAD senses a silence, recording is halted by the system and it transitions to the "Processing" page. After the

[3]https://huggingface.co/ggerganov/whisper.cpp

[4]https://huggingface.co/rhasspy/piper-voices/tree/main/en/en_US/amy/low

[5]https://huggingface.co/rhasspy/piper-voices/tree/v1.0.0

processing phase, the system showcases the "Speaking" page and audibly plays back the created audio to the user. After that the system returns to the "Listening" page.

III. Experiments

A. Splitting and Augmenting the KWS Dataset

We split the synthetic KWS dataset into training, validation, and testing sets, ensuring no overlap of speakers across these sets. Specifically, we allocated 1609 speakers for the training set, 100 speakers for the validation set, and the remaining 208 speakers to the testing set. The synthetic dataset created by the TTS models doesn't contain any form of noise. To make our KWS model more robust and suitable for real-world applications, we augmented the dataset by incorporating various types of background noises. We utilized the ESC-50 dataset [25], a collection of 2,000 environmental audio clips divided into 50 categories. We created five augmented versions by incorporating randomly selected background noises into the original utterance. In addition, we included all the samples present in the ESC-50 dataset into the "other sounds" class. This further enriched the diversity of the sound samples in that category. After augmentations, the training set comprised 9,654 utterances for the "ISSAI" class and 98,240 utterances for the "other sounds" class. The validation set held 600 utterances dedicated to "ISSAI" and 6,700 utterances for the "other sounds" class. Furthermore, the test set contained 1,248 utterances for the "ISSAI" class and 12,680 utterances for the "Other sounds" class. All utterances were saved in the WAV format at a 16 kHz sampling rate.

B. KWS Model Selection and Training

We used the Keyword-MLP model [26] for the KWS task. Our choice was motivated by its high accuracy (97.56%) on the test set of the benchmark Google Speech Commands dataset v2 [27]. Its accuracy is comparable with state-of-the-art models such as KWT [28] and AST [29]. The main advantage of the Keyword-MLP model is its superior parameter efficiency. Boasting a relatively small footprint with only 424,811 parameters, it is an excellent choice for deployment on compute-constrained edge devices. The original model was designed to classify the 35 commands in the Google Speech Commands Dataset v2. However, our task is easier because we have only two classes to differentiate. So, we reduced the model size to 33,922 parameters without compromising its high accuracy on our testing set. It enhanced the model's efficiency further, particularly in environments with limited resources, enabling it to function effectively in real-time applications. We trained the Keyword-MLP model on a single NVIDIA A100 graphics processing unit (GPU) for 100 epochs with a batch size of 256. We applied spectrogram augmentations such as time and frequency masking to mitigate overfitting during training.

C. Deploying the Models on a Mobile Phone

To assess real-time performance of the models on a real device, we deployed them on a Samsung Galaxy S22+ equipped with Android 14 OS and 8GB RAM. We used inference time (IT) and real-time factor (RTF) as a metric. IT is the time taken by a model with a constant input size to predict the input data. On the other hand, RTF is a latency measurement used in audio processing systems such as ASR and TTS with varying input data sizes. We used IT as a metric for Keyword-MLP and Silero-VAD because their input sizes are constant. On the other hand, we employed RTF as a metric for Whisper and Piper due to their varying input sizes. We ran each model ten times and computed the mean and standard deviation.

TABLE I
Results of the Keyword-MLP model on the test set

Class	Precision (%)	Recall (%)	F1-score (%)
ISSAI	99.04	98.88	98.96
Other sounds	99.89	99.91	99.90

IV. Results & Discussion

The accuracy of the Keyword-MLP model was evaluated using key metrics such as precision, recall, and the F1-score. Precision is a metric that calculates the percentage of correct positive predictions out of all positive predictions made by the model. Recall, also known as sensitivity, measures the percentage of actual positives correctly identified by the model. The F1-score is the harmonized mean of precision and recall. This single metric combines both precision and recall to provide a more unified view of model performance.

The results of the Keyword-MLP model on the KWS test set are shown in Table I. The model delivers high performance for the "ISSAI" class, securing a precision rate of 99.04% and a recall score of 98.88%. The F1-score, a measure of the model's overall accuracy considering precision and recall, for the "ISSAI" class is 98.96%. For the "Other sounds" class, the model displays exceptionally high performance with precision, recall, and F1-score exceeding 99.89%, demonstrating its robust capability to accurately differentiate the target keyword from a vast range of other sounds. The original checkpoint of Keyword-MLP in the Pytorch format had a size of 1.2MB. However, by transitioning it to the ONNX format, we successfully reduced its size to merely 0.45MB. This substantial reduction improves its suitability for deployment in resource-constrained environments.

The IT and RTF results are shown in Table II. Keyword-MLP, on average, exhibited an inference time of 2.8 ms with a standard deviation of 0.6 ms. On the other hand, the Silero-VAD model had an average inference time of 3.8 ms and a

TABLE II
Summary and inference results of the deployed models

Model	Keyword-MLP	Silero	Whisper	Piper
Format	ONNX	ONNX	BIN	ONNX
Deployment	Offline	Offline	Offline	Offline
Size (MB)	0.45	1.8	75.0	63.1
IT (ms)	2.8±0.6	3.8±0.3	-	-
RTF	-	-	0.12±0.01	0.19±0.01

standard deviation of 0.3 ms. Whisper exhibited an RTF of 0.12 with a standard deviation of 0.01 for an 11-second audio clip. Piper showed an RTF of 0.19 and a standard deviation of 0.01 for a two-second audio synthesized from a 50-character text. The results demonstrate that these models can operate efficiently on modern mobile phones.

Our current app has certain limitations. One such limitation is the dependency on the OpenAI API to interact with ChatGPT, which necessitates creating an account on the OpenAI Platform and obtaining an API Key. This process can be challenging for users who are visually impaired or blind. To make authentication more user-friendly, we plan to implement a wrapper around ChatGPT to enable users to authenticate using their Google or Microsoft accounts. The next limitation is that our app does not process images because GPT4-V was not available in our region at the time of writing this paper. The next important point is the lack of feedback from blind or visually impaired individuals. We plan to collect feedback to enhance the user experience. Finally, the app only supports the English language. We will incorporate more languages in our future work.

V. Conclusion

In this work, we have developed a mobile app with an intuitive user interface specifically designed for visually impaired and blind individuals to engage in natural conversations with ChatGPT. The app incorporates various speech technologies, such as automatic speech recognition, text-to-speech, keyword spotting, and voice activity detection. We employed offline speech models to preserve user privacy. Also, we presented an efficient method of generating synthetic speech data to train a custom KWS model. We deployed the models on a mobile phone. The inference time and real-time factor results showed that the speech models can run efficiently on the edge device. We have made the source code, pre-trained KWS model, and UI publicly accessible to foster the advancement of assistive mobile applications.

References

[1] R. R. A. Bourne, S. R. Flaxman, T. Braithwaite, M. V. Cicinelli, A. Das, J. B. Jonas *et al.*, "Magnitude, temporal trends, and projections of the global prevalence of blindness and distance and near vision impairment: A systematic review and meta-analysis," *The Lancet Global Health*, vol. 5, no. 9, pp. e888–e897, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214109X17302930

[2] "The Braille Literacy Crisis in America," National Federation of the Blind Jernigan Institute, Tech. Rep., 2009.

[3] M. C. McDonnall and Z. Sui, "Employment and unemployment rates of people who are blind or visually impaired: Estimates from multiple sources," *Journal of Visual Impairment & Blindness*, vol. 113, no. 6, pp. 481–492, 2019. [Online]. Available: https://doi.org/10.1177/0145482X19887620

[4] M. C. McDonnall, J. L. Cmar, and Z. S. McKnight, "Beyond employment rates: Earnings of people with visual impairments," *Journal of Visual Impairment & Blindness*, vol. 116, no. 4, pp. 526–532, 2022. [Online]. Available: https://doi.org/10.1177/0145482X221121830

[5] L. Marshall and J.-L. Moys, "Readers' experiences of Braille in an evolving technological world," *Visible Language*, vol. 54, no. 1-2, pp. 9–29, 2020.

[6] R. C. Berwick, A. D. Friederici, N. Chomsky, and J. J. Bolhuis, "Evolution, brain, and the nature of language," *Trends in Cognitive Sciences*, vol. 17, no. 2, pp. 89–98, 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364661312002823

[7] T. Brown, B. Mann, N. Ryder, M. Subbiah *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf

[8] OpenAI. (2022) ChatGPT: Optimizing language models for dialogue. [Online]. Available: https://openai.com/blog/chatgpt/

[9] ——, "Gpt-4 technical report," 2023.

[10] ——, "Gpt-4v(ision) system card," 2023. [Online]. Available: https://openai.com/research/gpt-4v-system-card

[11] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," 2022.

[12] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix *et al.*, "LLaMA: Open and efficient foundation language models," 2023.

[13] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei *et al.*, "Llama 2: Open foundation and fine-tuned chat models," 2023.

[14] G. Gerganov, "llama.cpp," 2023. [Online]. Available: https://github.com/ggerganov/llama.cpp

[15] H. Liu, C. Li, Y. Li, and Y. J. Lee, "Improved baselines with visual instruction tuning," *ArXiv*, vol. abs/2310.03744, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:263672058

[16] W. Wang, Q. Lv, W. Yu, W. Hong, J. Qi, Y. Wang *et al.*, "CogVLM: Visual expert for pretrained language models," *ArXiv*, vol. abs/2311.03079, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:265034288

[17] M. Hansen, "Piper: A fast, local neural text to speech system," 2023. [Online]. Available: https://github.com/rhasspy/piper

[18] A. Kuzdeuov, S. Nurgaliyev, D. Turmakhan, N. Laiyk, and H. A. Varol, "Speech command recognition: Text-to-speech and speech corpus scraping are all you need," *TechRxiv*, 2023.

[19] H. Zen, V.-T. Dang, R. A. J. Clark, Y. Zhang, R. J. Weiss, Y. Jia, Z. Chen, and Y. Wu, "LibriTTS: A corpus derived from librispeech for text-to-speech," in *Proc. Interspeech*, 2019. [Online]. Available: https://api.semanticscholar.org/CorpusID:102352475

[20] Y. Koizumi, H. Zen, S. Karita, Y. Ding, K. Yatabe *et al.*, "LibriTTS-R: A restored multi-speaker text-to-speech corpus," 2023.

[21] J. Yamagishi, C. Veaux, and K. MacDonald, "CSTR VCTK Corpus: English multi-speaker corpus for CSTR voice cloning toolkit," 2019.

[22] S. Team, "Silero VAD: pre-trained enterprise-grade voice activity detector (vad), number detector and language classifier," https://github.com/snakers4/silero-vad, 2021.

[23] G. Gerganov, "whisper.cpp," 2023. [Online]. Available: https://github.com/ggerganov/whisper.cpp

[24] J. Kim, J. Kong, and J. Son, "Conditional variational autoencoder with adversarial learning for end-to-end text-to-speech," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. Meila and T. Zhang, Eds., vol. 139. PMLR, 18–24 Jul 2021, pp. 5530–5540. [Online]. Available: https://proceedings.mlr.press/v139/kim21f.html

[25] K. J. Piczak, "ESC: Dataset for Environmental Sound Classification," in *Proc. of the Annual ACM Conference on Multimedia*. ACM Press, 2015, pp. 1015–1018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2733373.2806390

[26] M. Morshed and A. Ahsan, "Attention-free keyword spotting," 2022.

[27] P. Warden, "Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition," *ArXiv e-prints*, Apr. 2018. [Online]. Available: https://arxiv.org/abs/1804.03209

[28] A. Berg, M. O'Connor, and M. T. Cruz, "Keyword Transformer: A Self-Attention Model for Keyword Spotting," in *Proc. Interspeech*, 2021, pp. 4249–4253.

[29] Y. Gong, Y.-A. Chung, and J. Glass, "AST: Audio Spectrogram Transformer," in *Proc. Interspeech*, 2021, pp. 571–575.