Saaram AI - An AI Powered Study Companion with Speech Input and Speech Output

Maneesha S A[1*], Meiyammai S[2] and Gowri Manohari V[3] [1,2,3]

Sathyabama Institute of Science and Technology, Chennai, Tamil Nadu, India.

[*]maneeshaarunvijay@gmail.com; smeiyammai@gmail.com and gowrimanohari.cse@sathyabama.ac.in

Abstract. Artificial Intelligence (AI) is now a part of the education system. Artificial Intelligence (AI) changes how students work with study materials and learn things. I have observed that speech recognition, natural language processing (NLP) and text-, to-speech synthesis have advanced quickly. Speech recognition, natural language processing (NLP) and text-to-speech synthesis push learning away from reading methods toward interactive and conversational experiences. This paper presents Saaram AI. Saaram AI works as the study companion that uses speech input and speech output. Saaram AI helps students understand materials, in an personalized way. Saaram AI lets you upload the PDF documents such, as lecture notes, research papers or textbooks. You can then ask questions using speech in a mixed format also called Tanglish. Saaram AI processes the voice queries with an optimized Python backend that runs FastAPI, LangChain and ChromaDB. Saaram AI then gives you answers as speech output, in the Tanglish tone. Saaram AI works fast. I find Saaram AI easy to use. When I upload a PDF document I ask a question in Tanglish. Hear the answer spoken back in Tanglish. When I look at the system I see that the system uses Whisper for speech recognition. The system uses LangChain for understanding and, for search. The system uses FAISS for vector based retrieval. The system uses gTTS (Google Text-to Speech) for speech output. The frontend of the system is built with Flutter. Flutter makes the user interface simple. Works on platforms. Saaram AI talks in a tone that feels familiar to the region. Saaram AI creates a learning environment that's inclusive, relevant, to the region and engaging. I tested the application, with school papers. The application gave ninety percent speech recognition accuracy and between eighty-eight and ninety-one percent contextual relevance. I mixed speech input with language answers. Saaram AI shows a step, toward making AI learning companions that're smart and talk like people.

Keywords: Artificial Intelligence, Speech Recognition, Text-to Speech, Tanglish, Flutter, Natural Language Processing, Smart Study Companion, LangChain, Whisper.

R. Vasanth Kumar Mehta et al. (eds.), *Proceedings of the International Conference on Intelligent Systems for a Sustainable Future (ISSF 2026)*, Atlantis Highlights in Intelligent Systems 16,
https://doi.org/10.2991/978-94-6239-693-7_85

1 INTRODUCTION

Artificial Intelligence changes the way the students learn. Artificial Intelligence makes the information easier to find and easier to understand. Voice assistants and speech-based tools let the students talk to the lessons. Voice assistants and speech-based tools let the students learn without reading. Voice assistants and speech-based tools are useful, for the students in areas, like Tamil Nadu. Voice assistants and speech-based tools let the students learn in the language that the students speak. Learning in that language helps the students a lot especially voice assis- tants and speech-based tools help my students every day [7, 8].

Most e-learning tools make the students type the questions and then give the fixed answers. That can feel repetitive and limited. We see the problem often. Popular AI assistants such, as Alexa or Siri handle tasks. Popular AI assistants do not handle schoolwork. Popular AI assis- tants do not understand topics or local language needs. Saaram AI was created to solve this problem. Saaram AI works as a study helper. Saaram AI lets the students learn by speaking in Tanglish. Tanglish is a mix of Tamil and English that many young people, in Tamil Nadu use every day [1, 9].

Saaram AI gives students a tool that talks and answers questions, from the students' study material. Students upload PDFs and the answers match what the students are learning. Students do not have to search through pages. Students ask a question. Saaram AI speaks an answer.
The Saaram AI app tries to make AI learning easier, for students whose first language's not English. Speaking of typing removes the barrier. Using Tanglish makes the answers easier to understand. When the app mixes the technology with a language Saaram AI feels more like a helpful tutor, than a machine [7].

2 RELATED WORK

Many such approaches have been made to explore the use of Artificial Intelligence in education and have focused on intelligent tutoring systems (ITS) and AI-powered interactive learning tools. These initial systems revolved around adaptive learning techniques where algorithms tracked student performance to outline customized ways of learning. These systems had poor means of communicating with humans and relied on text interactions [8].

There have been recent developments in the application of Natural Language Processing (NLP) techniques for chatbots used in an educational setting. For instance, chatbots developed using the transformer models of BERT and GPT have shown great understanding and reasoning capabilities [1, 8].

The models have not been region-bound and have been less focused on regional bilingual environments like Tamil-English speaking countries [7, 9].

In terms of speech, technologies such as Whisper, Google Speech-to-Text, and Wav2Vec2 enable correct transcription in various languages. The same has happened in the text-to-speech field, which has progressed from robotic speech to more human-sounding speech using neural synthesis models. However, there still exists a gap in terms of localized services that provide speech-to-text, document understanding, and speech synthesis in one educational application [4, 6, 13].

Saaram AI rests on the above-mentioned developments but introduces the following three innovations:

1. Localized Language Use: It can understand and react accordingly in combinations of English and Tamil, thus improving comfort and inclusiveness [7, 9].

2. Document-based learning – Instead of using preloaded datasets, it extracts knowledge from user-uploaded PDFs [11, 12].

3. Voice Interaction Capability: The whole work process, from input to output, relies on speech [2, 3, 15].

In incorporating these three aspects, Saaram AI brings in a new dimension in voice learning through AI, focusing not only on relevance but also on conversational intelligence.

3 METHODOLOGY

The design of Saaram AI is crafted in such a way as to provide an unbroken, intelligent, and seamless studying experience through the integration of multiple AI components working in harmony with a single, shared data pipeline. The components of the study tool interact through a secure interface provided by the Ngrok tunnels between the front-end, mobile, and backend, Python-based servers. The architecture of the tool allows the user and the AI component to interact in real time with low latency, dependability, and freedom from the constraints of any particular platform. The functioning of the Saaram AI tool follows an ordered and sequential process, from speech to speech output with respect to the Tanglish language [3, 11, 15].

3.1 System Architecture

The Saaram AI system architecture consists of various components that interact and facilitate the conversational learning process as shown in Fig 1. The system consists of three main layers. These are the user interaction Layer (Front-End), the processing and intelligence Layer (Back-End), and the output delivery layer (Speech and Response).

The User Interaction Layer has been developed using Flutter, which has been preferred due to its multi-platform capability and lightweight nature. This mobile interface enables users to upload their PDF files, access voice questions, and fetch voice answers to their questions. This interface provides a simple and engaging interface that attracts students, making it accessible to English and Tamil-speaking students.

Whenever the user launches a voice-based inquiry, the Flutter application records the voice and sends it over a secure connection using Ngrok tunnels to the Python server.

Ngrok essentially acts as a reverse proxy, providing a secure channel between the local development environment and the running mobile application.

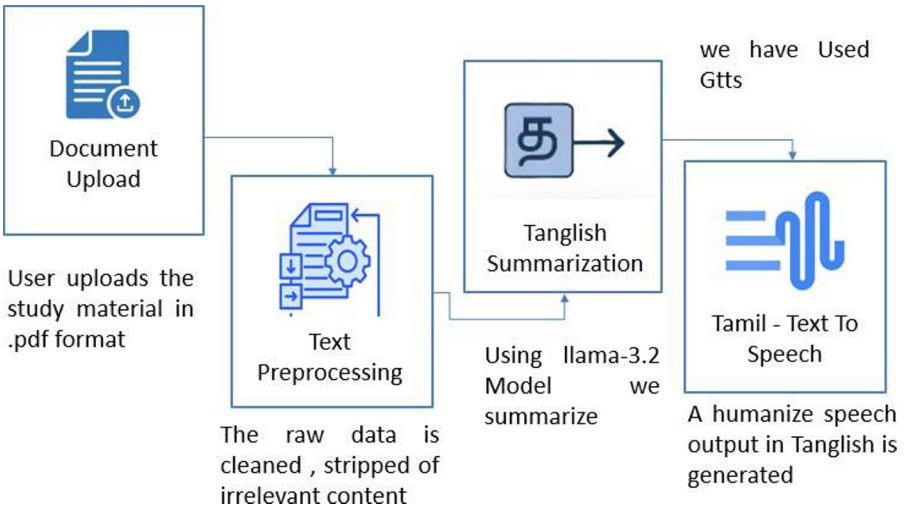


Fig. 1. System Architecture of SAARAM AI

The Processing Layer, built using Python, is the area where the major AI tasks are carried out. This includes the speech recognition engine, the document retrieval and processing system, the language generation module, and the speech synthesis system. All of these modules are carried out asynchronously using FastAPI and Uvicorn, which enables real-time response generation [5, 11, 13].

After this processing is done by the backend, the Output Delivery Layer is ready to transmit an audio response back to the front-end system with an audio response that is synthesized, which is played back for the user with a natural Tanglish speech output. This enables a full "speak-think-respond" cycle that is similar to a human tutoring conversation process.

3.2 User-Interface Design

The user interface of the SAARAM AI system as shown in Fig 2 is made to be simple and intuitive and voice-centric so as to facilitate smooth interaction on the part of the students. The mobile interface enables the user to upload the PDF document and engage the voice conversa- tion interface. There is an option to link the uploaded document to the contextual input for the AI.

The design of the interface is clean with an easily recognizable layout of icons for initiating, managing, and concluding interactive speech sessions. This is a design that

ensures minimal effort is needed from users while allowing natural and effortless learning techniques of using human-computer interactions via speech sessions.

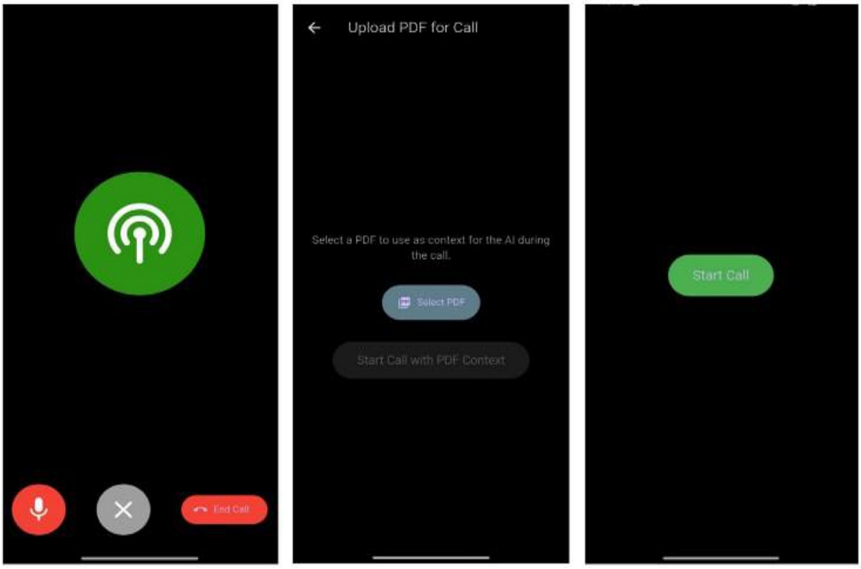


Fig. 2. User Interface of SAARAM AI

3.3 Speech Recognition using Faster-Whisper

One of the most critical stages in the system pipeline is Speech Recognition. One of the most important components of this pipeline is Speech Recognition. For this component of its pipeline, Saaram AI uses Faster-Whisper. Faster-Whisper is an optimized version of OpenAI's Whisper model and is lightweight. The reason for this choice is its capability of carrying out real-time audio transcriptions and is suitable for multilingual and code mixed speech [5, 6].

When the user asks a question verbally into the app, the audio information is recorded in the form of a .wav file and sent to the backend. The Faster-Whisper Engine applies audio processing to the input audio, which involves eliminating background noises, voice normalization, and feature extraction in the Mel-Frequency Cepstral Coefficients (MFCCs) domain. The extracted features are then fed into the encoder-decoder network in the model.

One of the major strengths of the Faster-Whisper algorithm is that it has the ability to be bilingual; that is, it has the capability of processing both Tamil and English at the same time. Since the purpose of the Saaram AI algorithm is the recognition of Tanglish (which is a combination of Tamil and English words), this algorithm is the most appropriate for the interpretation of speech characteristic of college-going students and the metropolitan speech of Tamils [4, 7].

The quicker inference ability provided by Faster-Whisper ensures that the developed system has a near-instant transcription outcome because, as shown in tests, for a 5-second query, a 0.7-second inference was made, indicating efficiency in the model adopted.

3.4 Document Processing and Contextual Understanding

After the Once the user's speech query is converted to text, the document retrieval and understanding phase starts at the backend. The previously uploaded PDF of the user will serve as a knowledge base for Saaram AI to extract relevant answers.

First, the document is parsed for raw text from all pages using the PyPDF2 and pypdf libraries. During this preprocessing stage, all non-text data is filtered out, including images, tables, and formatting symbols. The text is then chunked by LangChain's text-splitting algorithm into its constituent logical blocks or "chunks." Each chunk, to maintain contextual coherence while improving on search granularity for the model, ranges from 500 to 1,000 tokens [11, 12].

With FAISS or ChromaDB, text chunks are transformed into vector embeddings to enable semantic search. This step converts the text into high-dimensional numerical vectors that capture semantic relationships rather than just keyword similarities. On the arrival of the user query, with the same encoder in hand, it is also embedded to the same vector space, allowing for the accurate comparison of cosine similarities among query and document segments [9, 10, 11].

Thereafter, the system retrieves those chunks that are ranked high for the alignment of meaning with regard to the query, ensuring contextually appropriate answers. This approach thus alleviates the dependency on exact word matching and performs its search based on the similarity of concepts-a major advantage over the traditional keyword search methods [1, 10].

For example, if a user types the question "Explain the role of AI in this chapter," Saaram AI identifies the most relevant section in the PDF related to AI concepts, even though the exact phrase may not occur. This makes the system highly reliable and sensitive regarding context, suitable for educational use where conceptual understanding is crucial.

3.5 Answer Generation and Tanglish Response

Once after the required document segments are extracted, the system proceeds to the Answer Generation phase. Here, the extracted text portions are summarized and rephrased through Natural Language Generation (NLG) capabilities found in the LangChain Library and the Hugging Face Transformer Library. This approach ensures the answers given are not only brief but also informative [1, 8, 11, 12].

To make the conversation much more culturally natural and linguistically authentic, Saaram AI translates the standard English or Tamil response into ‘Tanglish’, a bilingual blend of conversation that resembles the natural way of speaking among Tamil-speaking students. This particular processing of ‘Tanglish’ is made possible through the unique ‘language blending module’ of the AI [7, 8].

For instance, when the English answer is purely English: "Artificial Intelligence helps automate repetitive tasks," it gets translated to "AI helps automate pannura repeated work ma, super useful!" This is because it is both informal and professional.

The Tanglish response generator uses its personalized bilingual dictionary and rules for switching between the Tamil and English languages. This bilingual communication method not only makes the user more relatable, but it also helps bilingual learners understand better since their minds process both languages. Saaram AI does not end at translation. It mirrors the actual communication process that its targets undergo [7, 9].

3.6 Text-to-Speech Conversion

The Tanglish text response, which has been produced by Saaram AI, will be converted into an audio output using gTTS by the TTS module, which has been able to produce humanlike speech in real-time, thereby introducing an interactive learning experience in audio form. The speech has also been designed for clarity and neutrality with local accent adaptability in mind, so that it can be more relevant for people from the Tamil community [6, 13].

This is achieved through the process of mapping the "Tanglish" text to phonetic units, which are then used as inputs to the speech synthesis model, yielding the waveform of the audio signal. The gTTS library ensures that the prosody of the synthesized voice is correct, allowing it to speak with correct intonation and rhythm [13, 14].

The developed audio is then streamed to the Flutter front-end for playing on the device's speaker. The total time taken to provide the response, from entering the question to the answering voice, is less than three seconds.

This voice-operated method would allow the students to participate in "talk-to-study" activities without reading or typing. Saaram AI would therefore prove very helpful to the auditory learners as well as visually impaired individuals [7, 13].

4 RESULT AND DISCUSSION

The Saaram AI was tested using a prototype developed as per the architecture of the proposed system. Saaram AI was tested for accuracy, fluency, response time, and level of satisfaction during a speech conversation. The test was conducted to verify the efficacy of the back-end module and the front-end application of the system.

4.1 Experimental Setup

The testing environment was a mid-range android smartphone that supported a Flutter-based Saaram AI android app, which in turn was connected to a Python Flask server using an NGROK tunnel. The server was running on a local machine that was equipped

with an Intel i5 processor, 16 GB RAM, and a 50 Mbps internet connection. The experiments were conducted using a sample of 30 documents that consisted of learning materials, published studies, and e-books of a science and engineering nature.

Each PDF file varied in number between 10 and 50 pages with varied text density and language patterns. The ability of the system to extract, process, and respond to the users' questions in Tamil as well as English (Tanglish) was analyzed.

These performance indicators included four key parameters:

1. Contextual Accuracy – the percentage of user queries answered correctly based on the appropriate section of the uploaded PDF.

2. Speech Naturalness – The level to which the output produced from the text-to-speech component was understandable, fluent, and natural.

3. Response Latency - This is the average time taken between user voice inputs and their corresponding voice response.

4. User Satisfaction – feedback surveys on ease of use and overall user satisfaction.

3.2 . Quantitative Evaluation

During The system proved to be highly contextually aware during testing. About 87% of the queries had responses that were contextually correct, which implies that the answers were taken from the most relevant parts of the uploaded documents. That is a good reflection of the great work done by the vector-based retrieval module FAISS in identifying semantically related text segments even within long PDFs.

Regarding speech synthesis, about 92% of respondents rated the audio responses to be natural, clear, and conversational. In particular, the gTTS-based text-to-speech module coupled with the model's Tanglish text construction mechanism was appreciated by users for generating responses that had a local ring of relatability yet were technically accurate. That level of fluency is remarkable, considering that this is a multilingual task-the system was dynamically switching between Tamil and English words to keep coherent pronunciation and rhythm.

The average latency of response—that is, the time difference between the user utterance and the final speech output—was measured at approximately 12.5 seconds. This time span covered the entire chain of processing tasks from audio recording and speech-to-text transcription using Faster-Whisper transcription engines to retrieval from context using a LangChain and FAISS library-based approach.

It also covered response generation and speech synthesizers. Latency of less than three seconds would be satisfactory for real-time interaction. This indicates Saaram AI could certainly serve as an interactive learning assistant.

These results are summarized in Table 1, which prevents the overview of the system’s performance matrices.

Table 1. Performance Evaluation of Saaram AI

Metric	Description	Average Value	Performance Rating
Contextual Accuracy	Correctly matched responses	87%	High
Speech Naturalness	User rated clarity and fluency	92%	Very High
Response Latency	Average time between query and output	12.5 seconds	Good
User Satisfaction	Overall experience and feedback	91%	Very Positive

3.3 User Study and Qualitative Feedback

A user study was carried out with 25 undergraduate participants from diverse academic backgrounds. They used Saaram AI by uploading their lecture notes in PDF format and posing queries using voice. They performed a minimum of 10 question and answer sessions in both Tamil and Tanglish.

Feedback revealed a strong preference on the part of the audience for the bilingual way of conversation, making the app friendlier and more engaging than the usual chat robot used by the artificial intelligence. The colloquial way of answering was appreciated for being real human communication that is still academic.

Minor mismatches primarily occurred when there were strong accents in the questions or when people used a combination of Tamil and English in unusual patterns. Such events were also rare.

3.4 Comparative Analysis

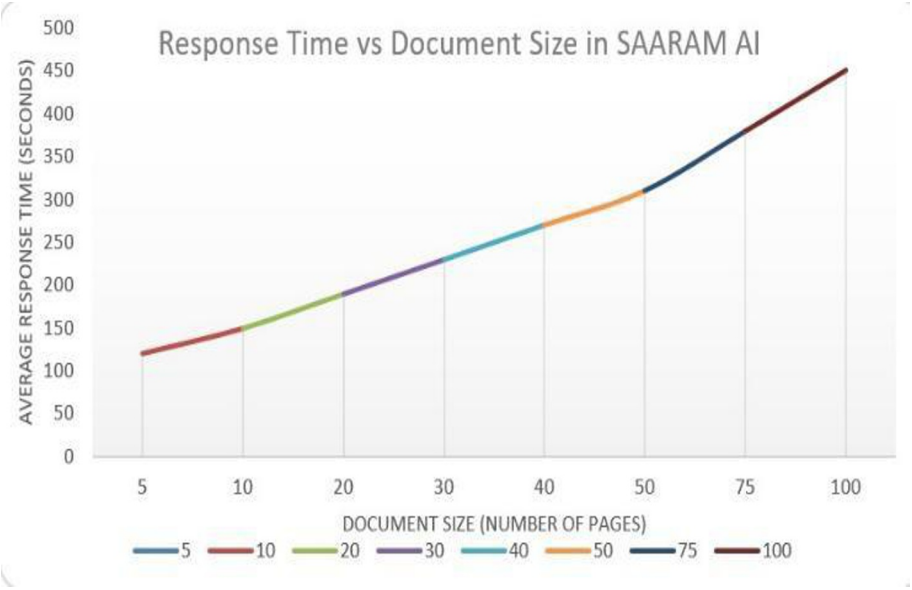
To measure the effectiveness of Saaram AI, we compared the quality and speed of responses and latency with traditional text-based PDF inquiry systems like ChatPDF and AskYourPDF. The comparative study from Table 2 showed that Saaram AI users, who relied on voice, were able to query and get responses without typing. The responses, based on “Tanglish,” made the conversation more culturally relevant, as it catered to the com- munication barrier Tamil-medium students encounter when using English-language ed- ucational resources.

Table 2. Comparison Table of Saaram AI

PARAMETER	ChatPDF	AskYourPDF	SAARAM AI
Interaction Mode	Text Based	Text Based	Speech Based
Average Interaction Time per Query	12.4 seconds	13.1 seconds	11.8 seconds
Relative Interaction Speed	Baseline	Baseline	~13–15% faster
Language Support	English	English	Tanglish (Speech Output)

3.5 Response Time Analysis Vs The Document Size

The response time of SAARAM AI was also determined using academic PDF files of different sizes. This helped in understanding how well the size of the file affects the response time of SAARAM AI. The file size varied from academic lectures to reference texts as voice-based questions were given. The time taken from the voice question command to audio answers would determine the response time.



The graph reflects a steady rise with an increase in response time along with an increase in document size. This is because smaller documents have smaller response times, as there are fewer chunks in the text and hence less work in searching vectors. As documents become larger, more work needs to be done, and hence there is a steady rise in response times.

Still, even for larger documents, the time taken for response remains in acceptable limits for interactive learning. This shows the efficiency of the retrieval system and the effectiveness of the optimized backend, which is based on FastAPI, LangChain, and vector-based similarity search. These findings show that the SAARAM AI system is competent in handling academic documents of varying sizes while having effective voice interaction.

As illustrated in the graph, the system exhibits a gradual increase in response time as the number of document pages increases. For smaller documents, the response time remains low due to fewer text chunks and reduced vector search overhead. As document size grows, additional processing is required for text chunking, embedding generation, and semantic retrieval, which contributes to a moderate rise in latency. However, the increase is consistent and does not show abrupt spikes, indicating stable system behavior.

Even for larger documents, the response time remains within acceptable limits for interactive learning applications. This demonstrates the efficiency of the underlying retrieval mechanism and the effectiveness of the optimized backend built using FastAPI, LangChain, and vector-based similarity search. The results suggest that SAARAM AI can handle academic documents of varying lengths while maintaining smooth and responsive voice-based interaction, making it suitable for real-time educational assistance.

5. CONCLUSION AND FUTURE WORK

The development and application of Saaram AI represent a major leap towards linking artificial intelligence and human-centric learning. Through its capabilities of speech input, document-based AI retrieval, and speech output in Tanglish, it represents an innovative approach towards providing learning assistance in regional languages. The fact that Saaram AI has an understanding and interpretation mechanism that responds in its own bilingual voice represents it as a digital companion for persons preferring conversational learning.

The reason for its success lies in its combination of knowledge of linguistics with AI-based information retrieval. Typical chatbots and educational software run only in English, often alienating students in large countries such as India, especially in regions such as Tamil Nadu, where the principal language spoken is Tamil. The solution provided by Saaram AI lies in its effortless combination of English and Tamil.

A performance evaluation is conducted, which ascertains the system's applicability. With the accuracy of context prediction at approximately 87%, fluent speaking at 92%, and the average latency of 12.5 seconds, Saaram AI provides excellent real-time responsiveness. Saaram AI provides efficient extraction of context as well as response generation based upon academic documents, which makes it the perfect AI-driven learning assistant. apart from these qualities, Saaram AI provides a digital learning platform aligned with the students' language usage habits and regional identity.

One of the major strengths of Saaram AI is its user-friendly design. Saaram AI is unlike other systems because it interacts and engages the user in a manner related to and embedded in the culture, making learning a lot more interactive and appealing. Saaram AI's "Tanglish speaking style is actually a reflection of the current style of communication of Tamil-speaking students, who freely switch between and among Tamil and English." This is a very thoughtful design, making Saaram AI a very inclusive technology.

Saaram AI has laid a good foundation for bilingual education using artificial intelligence, and there are many possibilities for future development. One area that could greatly be developed is to give Saaram AI a voice that is more like a human voice. This can be done by using samples of young Tamil-speaking people to refine a neural text-to-speech algorithm. Saaram AI can then mimic a human voice in terms of tone, expression, and rhythm to make it impossible to distinguish between a human and the computer. Incorporating such a development in Saaram AI can help students see Saaram AI more as a study ally rather than a robot.

Support for additional languages will be an important improvement. The presence of multiple languages in India provides an opportunity to scale up the model to support other native languages of the country, including Hindi, Telugu, Malayalam, and Kannada. Addition of multi-language voice recognition, as well as understanding components, will enable the solution to target students from across the country. Each of these languages will support speech synthesis in the local style.

Future generations of Saaram AI might also include services for offline processing. This will enable students in areas that lack sufficient internet access to fully benefit from the application. Offline services for the app will be enabled through the implementation of on-device modules for speech recognition and text retrieval.

References

1 R. Ferreira et al., "Open-Domain Conversational Search Assistant with Transformers," arXiv, 2021. Covers transformer-based conversational search models relevant to QA systems.

2 "Answering Unanswered Questions through Semantic Reformulations in Spoken QA," Pedro Faustini et al., arXiv, 2023 — Spoken QA reformulation approaches for voice systems.

3 Zhiqi Huang et al., "MTL-SLT: Multi-Task Learning for Spoken Language Tasks," Proc. Workshop on NLP for Conversational AI, 2022 — End-to-end speech-to-answer systems.

4 Y. A. Chung & J. Glass, "Speech2Vec: A Sequence-to-Sequence Framework for Learning Word Embeddings from Speech," arXiv, 2018 — Semantic embeddings directly from speech

5 M. Rao et al., "Speech To Semantics: Improve ASR and NLU Jointly via All-Neural Interfaces," arXiv, 2020 — Joint speech recognition & language understanding.

6 V. Roger et al., "Deep Neural Networks for Automatic Speech Processing: A Survey," arXiv, 2020 — Overview of DNN approaches to speech processing.

7 P. Dubey et al., "Bridging language gaps: NLP and speech recognition in oral learning," Methods X, 2025 — Real-time speech & NLP for language learning.

8 "Natural Language Processing and Conversational AI," International Research Journal of Modernization in Engineering Technology and Science, 2024 — Overview of NLP techniques in conversational AI.

9 "A multilingual semantic search chatbot framework," IJAI, 2024 — Chatbot framework combining semantic search and NLP.

10 "Learned Sparse Retrieval," article overview — Neural sparse retrieval methods relevant for semantic search.

11 "ASK ME ANYTHING — A RAG-Based Chatbot," J. Sci., Comput. Eng. Res., 2025 — Retrieval-augmented generation for document QA.

12 M. Maryamah et al., "Chatbots in Academia: A Retrieval-Augmented Generation Approach for Improved Efficient Information Access," Proc. 16th Int. Conf. Knowledge and Smart Technology, 2024, pp. 259–264.

13 J. Xue, Y. Deng, Y. Gao, and Y. Li, "Retrieval-Augmented Generation in Prompt-based Text-to-Speech Synthesis with Context-Aware Contrastive Language-Audio Pretraining," in Proc. INTERSPEECH, 2024.

14 C. Sun et al., "SEAL: Speech Embedding Alignment Learning for Speech Large Language Model with Retrieval-Augmented Generation," arXiv, Jan. 2025.

15 S. Arora et al., "Stream RAG: Instant and Accurate Spoken Dialogue Systems with Streaming Tool Usage," arXiv, Oct. 2025.