HiEn: Speak the Switch

Krish Babba[* a)], Aryan[b)], Sheenam Naaz[c)], Manmohan Singh Yadav[d)]

Department Of Computer Science & Engineering, SSCSE, Sharda University, Greater Noida, India

[a)] Corresponding author: 2022321571.krish@ug.sharda.ac.in [b)]2022329124.aryan@ug.sharda.ac.in
[c)]manmohan.yadav@sharda.ac.in [d)]sheenam.naaz@sharda.ac.in

Abstract. Linguistic barriers remain a major challenge in communication, education, and cultural preservation, particularly for ancient and low-resource languages. While modern neural machine translation systems perform efficiently for widely used languages, they lack sufficient support for historical scripts such as Brahmi and classical languages due to limited datasets. This paper presents HiEn: Speak the Switch, a multimodal multilingual conversion system that integrates text, image, and voice-based translation within a unified platform. The system combines Optical Character Recognition, speech recognition, text-to-speech synthesis, and machine translation APIs within a Python-based graphical interface. Additionally, a custom character-mapping module enables deterministic conversion of the Brahmi script into modern equivalents. The system demonstrates reliable multilingual translation and satisfactory OCR performance. It enhances accessibility, simplifies translation workflows, and contributes to the digital preservation of ancient scripts. The architecture is scalable and can be extended using advanced transformer-based models in future implementations.

Keywords: Machine Translation; OCR; Speech Recognition; Ancient Languages; Brahmi Script; Multimodal System; Neural Machine Translation

1. INTRODUCTION

Language is the basis of communication, the sharing of knowledge, and the continuity of culture in societies. Due to the active development of digital communication technologies and globalisation, there has been an increased need for effective multilingual translation systems. Deep learning-based neural machine translation (NMT) systems (sequence-to-sequence networks and transformers) have led to significant increases in the quality of translation of high-resource modern languages. Nevertheless, the number of computational tools available for working with ancient and low-resource languages is limited because there are not enough digitised corpora and structured datasets. Languages like Sanskrit and historical scripts like Brahmi are of enormous historical, cultural, and academic importance, yet they are not fully integrated into mainstream AI-based translation systems. Moreover, the majority of the existing translation platforms work. Mainly text-based and minimal smooth integration between image and voice-based translation features. This constraint limits access and usage, especially in educational institutions, research and heritage preservation programs. Hence, there is an increasing need to develop a multimodal translation system that integrates multimodal translation capabilities and support for ancient scripts to preserve linguistic heritage and provide digital accessibility.

1.1. Motivation

The key motivation of the study is the gap between the high technology available for neural translation and the inapplicability of ancient and resource-limited languages. Although current translation application programming interface solutions support multiple languages, they often do not provide specialised support for ancient scripts such as Brahmi. Also, the available translation tools are typically discontinuous, i.e., different tools exist for translation, Optical Character Recognition, and voice recognition, which results in a fragmented user experience. Increasingly, there has also been a demand to conserve ancient manuscripts and endangered languages with the assistance of artificial intelligence, which has become popular at the national and international levels. The preservation of ancient manuscripts is a complicated endeavour, which is made tougher by the fact that translation of ancient manuscripts is a tedious process that needs certain skills; thus, a tool that will provide a unified experience of text, image, and voice translation, as well as a dedicated module of ancient script translation, is suggested to address the given issue.

1.2. Structure of the Paper

The rest of this paper will be structured in the following manner. Section II examines the literature on the

respective neural machine translation, OCR technologies, speech recognition systems, and computational processing of ancient languages. Section III provides the problem statement and outlines the research challenges that are dealt with in this study. The methodology is proposed in Section IV, including system architecture and workflow design. Section V talks about the experimental results, analysis of performance and system evaluation. Lastly, Section VI summarises the paper and provides the future scope and research directions on how to improve the system with improved transformer-based models and scalable deployment strategies.

2. LITERATURE REVIEW

Recent advancements in Neural Machine Translation (NMT) using transformer architectures have significantly improved multilingual translation accuracy. Pretrained language models such as BERT, mBART, and T5 further enhanced contextual understanding and cross-lingual representation learning, as per Table 1

Table 1: Literature Review

Ref	Domain	Existing Tech/Apps	Pros	Cons	Gaps Addressed by HiEn
[1]	Neural Machine Translation (Transformer)	Google Translate, Transformer Models	High translation accuracy; scalable architecture	Requires large annotated datasets	Adds ancient script mapping and multimodal integration
[2]	Multilingual NMT	Google Neural Machine Translation	Wide multilingual support	No support for ancient scripts	Includes Brahmi conversion and unified interface
[3]	Pretrained Language Models	BERT-based Systems	Strong contextual understanding	Computationally expensive	Lightweight GUI with OCR and speech modules
[4]	Optical Character Recognition	Tesseract OCR	Open-source; multilingual capability	Limited accuracy for ancient scripts	Integrates OCR with translation and mapping
[5]	Computational Sanskrit	Rule-based analyzers	Effective grammatical analysis	No multimodal or GUI integration	Supports Sanskrit with image and voice translation
[6]	Speech Recognition	Google Speech API	High speech recognition accuracy	Separate from translation workflows	Unified speech-to-translation pipeline
[7]	Multilingual Pretraining	mBART Framework	Good performance for low-resource languages	Requires heavy training and resources	API-based solution with additional mapping layer

[8]	Low-Resource MT	Transfer Learning MT	Improves translation for low-resource languages	Dependent on the transfer datasets	Deterministic Brahmi mapping without large datasets
[9]	Indian Script OCR	Devanagari OCR Systems	Effective for printed Indian scripts	Weak performance on historical scripts	Extends support toward historical scripts
[10]	Transfer Learning (T5)	T5 Model	Versatile NLP architecture	High computational requirements	Modular system allowing future transformer integration

3. PROBLEM FORMULATION

Although significant progress has been made in neural machine translation and multilingual processing methods, the majority of existing translation systems target high-resource modern languages. Limited computational focus has been given to the ancient languages and past scripts like Brahmi. To a great extent, this is conditioned by the absence of digitised corpora and linguistic resources. The number of historical manuscripts that are unattended is enormous. The second major weakness of the existing translation tools is the absence of using various modes. The process of text input and image processing is achieved in a different application, as most translation tools only accept text inputs. This is a great weakness of the existing translation tools. Additionally, to provide the transcription and translation of ancient texts, knowledge and time are needed. Preservation of historical content is hard due to the absence of script mapping tools to automate this process. Furthermore, because of the absence of training data, the accuracy of the translation process of low-resource languages is impaired. Thus, it is quite evident that there is a necessity for an accessible, multimodal system of language conversion that will be able to support both modern and ancient languages and even combine text, image, and voice processing into a single system.

4. PROPOSED METHODOLOGY

The suggested Multilanguage Converter system is created on the basis of a modular and multimodal translation system written in Python. The system has text translation, image translation, voice translation, and ancient script translation as one graphical user interface with Tkinter. The system architecture is created in order to provide maximum flexibility and efficiency in processing multilingual data. The system facilitates easy translation between two languages of any type, with the application of translation APIs, Optical Character Recognition (OCR), speech recognition and text-to-speech technologies, as well as mapping of the Brahmi script with the help of a specially designed data set.

A. **Input Collection and Preprocessing**.

The system facilitates three different modes of input, which include text input, image input, and voice input. Text mode is used when the user manually types content into a Tkinter text box in the graphical interface. In image-based translation, the user uploads.a text file with an image represented by a file dialogue interface. The system is voice-enabled, with voice input captured via a microphone in voice mode. The predefined dropdown menus also enable the user to choose the source and destination languages in the GUI. This dynamic input system has made sure that the users can engage the system in various ways based on their needs and accessibility requirements.

B. **Preprocessing and Text Extraction.**

After this input is obtained, preprocessing is done to convert the input into a standard text form, which can be easily translated. In the case of text input, this would entail feeding the text over to the translation engine. To take input of images, the system utilises the Tesseract OCR engine through the pytesseract library to obtain text on the images. This OCR engine will be able to recognise various scripts, which include Sanskrit, Hindi, and English scripts. In the case of voice input, the system requires the SpeechRecognition library, which records the audio and converts it into text based on the speech recognition API provided by Google. This

preprocessing step is important to make sure that whatever is fed into the system, be it text, graphics, or voice, is in a machine-readable form.

C. **Mapping Module of Ancient Scripts.**

A custom Brahmi script conversion module is provided in order to support the ancient language processing. The data in this module is represented in a JSON file, which includes data about the Brahmi script and the modern representation of the script. After selecting the Brahmi script as the source language, a character-level mapping function is called. The characters of the input string are then mapped to the characters of the output string with the help of the JSON dataset. It is a deterministic approach to translating the script that allows script-level translation to be supported without necessarily having a neural model.

D. **Translation Engine**

Google Translate library attains the translation feature. This library is connected with the Neural Machine Translation service of Google. Once the preprocessing phase is complete, the system identifies the codes of the source and target languages used in the dropdown menu. The clean, standardised text is then submitted to the translation API. The request is processed by this API and gives the output of the translation. This engine is also directionally translatable and is able to support many contemporary languages like English, Hindi, Marathi, Tamil, Telugu, French, German, and Spanish, among many others. This module is the processing unit of the system.

E. **Speech Synthesis and Generation of Outputs.**

After translation, the translated text appears in the output box that is indicated in the GUI. To improve accessibility and interaction with the system, the pyttsx3 text-to-speech engine is incorporated in the system. This module reads out the translated text and puts it through the speakers of the device. This speech synthesis option is especially helpful to users with sight problems or those who learn visually. The visual and auditory output mechanisms are combined, making it more usable and inclusive.

F. **Scalability and System Characteristics.**

The general structure of the Multilanguage Converter seems to be lightweight and modular, which renders it applicable to desktop-based systems. Each of the modules, i. e., input acquisition, preprocessing, script mapping, translation, and output generation, is not dependent on the others. Nonetheless, they are rationally combined into the same framework. The general structure seems to be able to expand the system in the future, such as the support of the multilingual models built on transformers, to implement the system in the cloud infrastructure, to create a web or mobile application, and to add support for more low-resource and ancient languages. As per Table 2

G **. Backend Architecture Overview**

Table 2: Backend Architecture

Layer	**Tools**
Input Layer	Tkinter GUI, Text Widget, File Dialogue, Microphone Interface
Preprocessing Layer	Pytesseract (Tesseract OCR), SpeechRecognition Library
Ancient Script Mapping Layer	Custom JSON Dataset (brahmi.json), Python Mapping Function
Translation Layer	Google Translate Library (Google Neural Machine Translation API)
Speech Synthesis Layer	pyttsx3 Text-to-Speech Engine
Output Layer	Tkinter Output Text Box, System Audio Output

5. SYSTEM ARCHITECTURE

The Multilanguage Converter architecture is created as a desktop-based architecture with an API for multilanguage translation. The system is based on a centralised processing pipeline architecture whereby the various modules are employed to obtain the input, process the input, map the scripts, translate the input and produce the output. The architecture is simple, real-time responsive, and multi-lingual and allows modern and antique scripts. The design is meant to be a lightweight Python application, which is desktop-based, although it can be scaled to function in the web/cloud in the future. The system is multimodal (text, image, voice), multilingual (translation to various modern languages), and converts the Brahmi script into a different one with the help of a specific mapping dataset.

A. System Components Overview.

1. **User Interface (React Native Frontend)** The user interface is written in Python with the Tkinter library, which offers a straightforward and interactive desktop interface-based GUI. The interface enables users to type text using the keyboard, upload image files to translate them, or use the microphone to translate using voice. The dropdown menus allow the users to choose the source and target languages among predefined languages like English, Hindi, Marathi, Sanskrit, Tamil, Telugu, French, German, Spanish, and Brahmi (text mapping mode). The output that is translated is shown in a special box. The GUI is designed in a manner that allows it to be used easily and by the user with little technical expertise.

2. **Input Processing** Module This is the input processing module, whereby text, images, and voice inputs are processed. In the case of text mode, the text is simply fed into the translation module. In image mode, a system with the pytesseract library to read image text utilises Tesseract OCR. Tesseract is capable of supporting several languages such as Hindi, Sanskrit and English scripts. In the voice mode, the system will utilise the Speech Recognition library, which will record the audio and then translate the audio to text via the Google speech recognition API. All the input formats are processed by this module and converted into text format, and then processed.

3. **Ancient Script Mapping** Module To ease the conversion of historical languages, a Brahmi script mapping module has also been added to the system. The data set storing the mapping of the characters in the Brahmi script and the corresponding mapped script characters has been stored as a custom JSON data set. In the case where the Brahmi script is selected to be the source language to be converted to, the mapping function will take the characters of the text one by one and will substitute the character with the character that it is mapped to. This is a deterministic process.

4. **Translation Engine (API-Based)** The translation engine is a service that uses the Google Translate library that is connected to the Neural Machine Translation system of Google. The text is then submitted to the translation API with the language codes chosen, post-processed, and optionally converted to Brahmi. The engine has the ability to do two-way translation between various current languages, including English, Hindi, Marathi, Tamil, Telugu, French, German and Spanish. The output of the translation is sent in real time to the output unit.

5. **Output and Speech Synthesis Module** The text that has been translated will be shown in the output section of the GUI. Moreover, the pyttsx3 module includes a text-to-speech engine that converts the text into audio. This will enhance the ease of use of the application, particularly for the sightless or for individuals who are acoustic learners. The speech synthesiser module is independent, and the input is the text that has been translated.

6. **Data Handling and Security** The system carries out translation activities without necessarily storing user data. Every processing happens in working memory. Transmission of communication to translation and speech APIs is via protected internet connections. No database storage is provided; as far as is implemented by the threat of default, user privacy is reduced. In case the architecture is deployed as a web-based application, it can be extended to support authentication and history storage in the future.

B. System Interaction Flow

As the user works with the interface, the system receives the user's input in the form of text, images, and voice. The system translates the input of the user into text using the OCR and voice recognition tools in the event of images and voice, respectively. The system, in the case of the Brahmi language, utilises the script mapping module in the conversion of the language into a new form of script. The system then transmits the translated text to the translation engine to translate it, depending on the target language selected. Lastly, the output is translated, and the translated output is returned and shown in the GUI output window. The output translated in the voice output is translated into voice.

6. DETAILED WORKFLOW

The HiEn: Speak the Switch - Multilanguage Converter tool works in a systematic manner and follows various stages to process the input dynamically, thus guaranteeing the accuracy of the multilingual translation of any given language, whether it is modern languages or ancient scripts. The tool makes sure that the logic of processing is properly followed, depending on the mode of input selected, which is either text, image, or voice and the source and destination languages. An overview of the interaction between the components of the tool as discussed above is provided below.

A. User Interaction Workflow

Communication with the user will begin with the user executing the desktop application, which has been developed using Tkinter. The user chooses the source and the target languages via dropdown menus and the input mode between three options, namely text translation, image translation, and voice translation. In case the user clicks on text translation, the system extracts the input string typed in the input text box. The user is presented with a file dialogue where he/she can upload an image file when he/she chooses the image translation. The tesseract Optical Character Recognition (OCR) engine is then used on the image file with the help of the pytesseract library. The system has an OCR engine that extracts the text in the image. The scripts that the OCR supports include Hindi, Sanskrit and English. The user can choose voice translation when the system records the input speech using the microphone. The input speech is then retrieved and converted to text by the use of the SpeechRecognition library and the speech recognition service by Google. Once the input string is retrieved in one of the three modes, the system gets to the next step of processing. This step involves a few rudimentary cleaning processes on the input string to make sure it is properly formatted in order to be translated. In the case of a choice of source language as Brahmi, the system triggers the Brahmi Mapping Module. A character-by- character processing of a text is done with a mapping dataset that is based on JSON, with Brahmi characters being substituted with their. applicant equivalent in the modern script. This is to make sure that it is compatible with the translation API since standard neural translation engines do not directly support Brahmi. The processed and possibly script-mapped clean text is then passed to the Google Translate-based translation engine. The system transfers the source and destination language codes, as well as text that has been processed, to the API. The neural translation engine will carry out the translation and provide the translated text in real time. The output text box of the GUI is then used to display the translated text. In case voice output is included in the interaction (particularly when voice translation mode is enabled), the pyttsx3 text-to-speech engine will transform the text being translated into audio and read it out loud through the system speakers.

B. Use Case (Sadness Detected)

Suppose that the user is posting an image with the following text: Welcome to India. The picture is then processed under the Tesseract OCR tool that retrieves the text: Welcome to India. The text is then filtered and subjected to the translation engine, having the source language as English and the target language as Hindi. The request is sent to the Google translation API, and the output is given as: bhaart meN aapkaa svaagt hai / In the case the user chooses to use voice output, the text is then read out using the library pyttsx3.When the user types the source language as Brahmi and types the Brahmi script in the input box, the system will call the mapping functionality of the system, which is based on JSON. All the characters are substituted by their mapped counterparts of the modern script. The result of this conversion is given to the translation engine (when the option of translation to another language is selected). It offers partial assistance of ancient writing on a new multilingual platform. As shown in fig 1

This workflow ensures:

* live translation of any language among the supported languages.
* Free flow between text, picture and voice recognition.

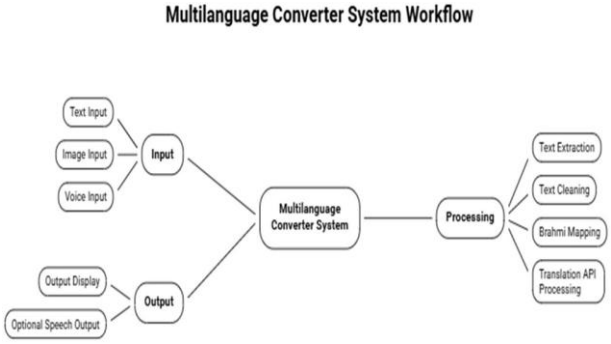

Fig.1 . Language conversion workflow

This workflow ensures:

- **Real-time multilingual translation** across supported languages
- **Seamless switching** between text, image, and voice inputs

7. MODULE-WISE IMPLEMENTATION PLAN

The Multilanguage Converter is written in a modular Python-based architecture where each of the functions is a module component that is independent and provides a part of a centralised translation pipeline. The system combines the elements of the graphical user interface, preprocessing tools, translation services via API, and speech processing libraries in order to offer a smooth multilingual conversion. This modular design will guarantee maintainability, extensibility, and clarity in the flow of execution, and therefore the system will be appropriate for future upgrades and incorporation of advanced models.

The User Interface module is written in Tkinter, and it displays input and output text boxes, a dropdown menu of languages and buttons to activate either text, image or voice translation. The Multimodal inputs are processed in the Input Processing module. The interface is used to directly retrieve text, pytesseract is used to process images with the Tesseract OCR engine and SpeechRecognition library with the Google API is used to convert speech into text. This makes sure that all the input types are standardised into text format and then translated. The preprocessing module purifies and formats the extracted text in order to enhance accuracy in translation. To support ancient scripts, the Brahmi Mapping module relies on a custom JSON dataset to transform Brahmi characters into the modern script equivalents via deterministic character replacement. The Translation module is the main processing module, and it is implemented by the use of the Google Translate library that interacts with the Neural Machine Translation system at Google to provide real-time conversion between two languages. The result of the translation is presented in the GUI and can be turned into speech with the help of the pyttsx3 text-to-speech engine (optionally). The entire processing is performed in the runtime memory with no permanent data storage and with very little risk to privacy. The modular implementation facilitates the effective multilingual translation and also permits the future expansion and compatibility with advanced models.

8. ETHICAL CONSIDERATION

The ethical design is significant in the process of creating the HiEn: Speak the Switch - Multilanguage Converter, particularly because it helps the communication between the linguistic and cultural contexts. Being a multilingual and ancient-script conversion system, the system should guarantee responsible use, privacy of the user, fairness in translating, and cultural neutrality. Being an application that uses AI-based translation APIs to translate user-generated text, images, and speech, the platform has a duty to reduce the misuse, prevent bias, and safeguard the information of the users. This part explains the ethical protection that the system has integrated in an attempt to facilitate a safe, respectful, and transparent operation.

A. Misuse Prevention and Content Moderation.

Even though HiEn is mostly a tool to translate, it can also take arbitrary user-generated content, even

potentially offensive or sensitive content. As the system relies on outside neural translation APIs, it is possible that unsuitable input can be translated and shown. The system will reduce the chances of misuse through responsible usage and can be further enhanced with input validation checks to prevent abusive or harmful keywords before processing. The application does not produce original texts but simply translates the text posted by users, which significantly reduces the risks of creating autonomous content. Users are advised to avoid harmful, misleading and defamatory communication through the system. The improvement that can be made in the future is the addition of keyword filtering layers to avoid translating explicitly offensive or violent content.

B. Management of Bias and Cultural Sensitivity.

The aspect of language translation is a sector in which cultural concerns are also involved, as well as the precision of the language. The system makes use of a pre-trained neural language translation service; therefore, there might be a minor level of inaccuracy that is caused by the training data. The system does not alter the output of the translation service; only the necessary preprocessing is done. Nonetheless, the user receives the warning of the translation output being sensitive to such cases as legal documents, academic writing, or emotionally colored content. The Brahmi mapping is a character-based deterministic mapping and does not have any cultural implication. The system helps in upholding the principle of respectful language to be used in communication between the various cultures without altering the cultural content.

C. Data Privacy and Anonymity

The HiEn Multilanguage Converter has a privacy-based design. The default of the system is not to store the user inputs, images, and voice information. Rather, all the calculations are handled in the runtime memory at the time of application execution. When there is a need, user inputs are sent to the translation API or speech recognition API. This communication occurs within secure internet connections. Images are run through the local system with the Tesseract engine, then translated. No history of translation is recorded. This will reduce the risks of illegal data storage. It is recommended that users not provide confidential and highly sensitive data when using online API-based translation services.

D. User Awareness and Transparency.

The system is open regarding its functionality, which is based on artificial intelligence. As an example, the translations were done with the help of automated neural machine translation systems, and they are not necessarily accurate. The system, too, does not purport any human judgment in linguistic affairs or any knowledge of correct historical transcription. For example, users are not supposed to use the translations for serious purposes, such as medical, legal, or academic submissions. The Brahmi conversion module involves deterministic character mapping, and it cannot be regarded as a scholarly transcription system.

E. Accessibility and Inclusive Design.

HiEn will be used to facilitate multilingual interaction. The system enhances the accessibility of users with different needs by incorporating text, image, and voice entry. The text-to-speech module will make the application more user-friendly to visually impaired or audio learners. The encouragement of multiple modern Indian and international languages, as well as the conversion of the Brahmi script, also leads to the inclusivity of a language and the preservation of digital heritage. The light desktop architecture makes sure that the system is capable of running on common computing systems without the need for high computing resources.

F. Restriction of Automation Risk.

The system operates only as a translator and script converting system. It does not make decisions, nor does it make suggestions. Besides, it lacks the ability to create content. The system does not perform predictive profiling, behavioural analysis or user tracking. This is because the system functions only under the ability of a translator based on an API that lacks the ability to handle matters of decision-making systems.

9. LIMITATIONS

The HiEn Multilanguage Converter relies on third-party APIs, including Google trans and Google Speech Recognition, which presuppose an uninterrupted connection to the internet in order to translate and process the voice. The system is not capable of operating entirely offline. This can lead to a drop in translation accuracy of slang-intensive, regionally-focused, or very complex sentences because they use a pretrained neural model. Mapping character level only. The Brahmi script mapping module does not give the semantic meaning of ancient texts, but only character-level conversion. The accuracy of OCR is contingent on the clarity of the image, and it can be lower in low-resolution and handwritten text. On the same note, speech recognition accuracy can be diminished when there is a lot of noise or when accents are strong. The system lacks support for domain-specific customisation, personalisation, and a history of translation.

10. FUTURE RESEARCH DIRECTION

Although the HiEn: Speak the Switch - Multilanguage Converter provides a groundwork on how to translate in multiple languages with an ancient-style script, there are a few improvements in the future that are anticipated to strengthen the technical power of the tool, linguistic precision, accessibility, and flexibility in its deployment. Systematic translation accuracy evaluation can be regarded as one of the major future research directions. Quantitative measurements that may be used to benchmark the system include the correlation of the BLEU score, precision-recall assessment of OCR output, and human-rated fluency evaluation. Controlled user studies can be performed to quantify the reliability of translations, particularly.

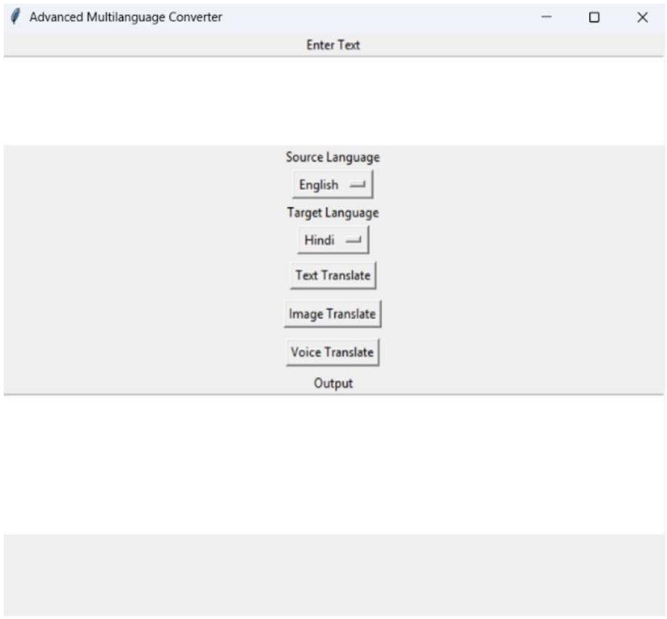

Fig. 2: Advanced Multilanguage Converter

of idiomatic expressions, culturally sensitive expressions and code-mixed sentences. More indicators of performance tracking can be used to optimise the performance in terms of accuracy of OCR extraction, the rate of speech recognition errors and the API response time. Multilingual expansion and script enrichment are other directions of importance. Its modular architecture can easily be extended to other Indian and other international languages. Subsequent versions can support additional regional languages and have better mapping data of scripts other than Brahmi that are ancient. It can be seen that the further development of the Brahmi module by adding the contextual linguistics rules instead of a mere mapping on the character level can help greatly in the interpretation of historical texts. Improvement of voice integration and accessibility is also an important development point. Even though speech recognition and text-to-speech capabilities are already integrated, the next generation of research could be aimed at enhancing its performance in distorted conditions and coverage of different accents. Online models of speech processing can be considered to ensure less reliance on external APIs, as well as better usability in low-connectivity areas. In terms of machine learning, the translation of API-based models with locally deployed transformer models is a

significant research direction. The system might support the rural or bandwidth-limited environment through the lightweight transformer architecture, like distilled models that can provide partial offline functionality. Domain-specific customisation: Domain-specific customisation of translation, e.g. academic, legal or technical translation modes, can be explored in order to provide better contextual accuracy.

11. RESULT AND DISCUSSION

The HiEn Multilanguage Converter has managed to integrate the text translator, image translator (OCR), voice translator and Brahmi script translator into one application. Google Translate's translation of text has been effective and offers real-time and near-translation of standard sentences in multiple languages. Translation of images by the OCR component has been effective with clear image sources and less effective with low-quality and handwritten sources. In the same manner, the voice translator element has managed to work in low-noise conditions. Text-to-speech has been fruitful. The character-level translation of the Brahmi script has succeeded but failed to comprehend the meaning.

12. FIG.2. RESULT OF THE HIEN SYSTEM CONCLUSION AND FUTURE SCOPE

This increasing need for effective communication in a multilingual society identifies the disadvantages of the traditional translation tools, which might not support multimodal, be accessible, and accommodating to ancient and resource-limited scripts. The issues that are discussed in this paper will be input variety, script support, OCR, speech support, and responsiveness that are related to multilingual conversion. In this line, the authors have developed a Python solution named HiEn: Speak the Switch - A Multilanguage Converter, which supports the translation of texts in different languages, assists in mapping the Brahmi script, and provides speech synthesis and voice recognition functionality in a single desktop environment. The platform provides easy access to and dynamic multilingual interaction through the integration of preprocessing mechanisms, deterministic script mapping, and real-time translation services. It is lightweight and modular in design, making it applicable for educational, research, and general communication. This project is backed up by the development of neural machine translation and OCR technologies, as well as speech processing libraries. It creates a viable framework of multimodal language conversion and digital heritage support. Although the present application is designed to be deployed on the desktop and translated via an API, the modular architecture allows expanding the application to include offline transformer models, web-based applications, and improved intelligent scripting.

CONFLICT OF INTEREST

The authors declare no conflicts of interest regarding the current research.

REFERENCES

[1] M. Lupaşcu, Large multimodal models of low-resource languages
[2] Languages, Journal of Artificial Intelligence Research, vol. 76, pp. 145178, 2026.
[3] N. Sethiya et al., “Indic-ST: A large-scale multilingual corpus of speech translation, low-resource speech translation, ACM Trans. Asian Low-Resour. Lang. Inf. Process., vol. 24, no. 3, pp. 1– 21, 2025.
[4] N. Kadyrbek, A. Issa, and M. Al-Khalifa, Development of small-scale language models on low- resource languages, Big Data Cogn. Comput., vol. 9, no. 5, pp. 137–152, 2025.
[5] J. Iranzo-Sánchez et al., Speech translation to multilingual medical education Artif. Intell. Med., vol. 145,
[6] pp. 102–118, 2025.
[7] S. Roy, “Multi-language translation system based on AI: Architecture/performance analysis of the system-on-chip, Research Preprint, 2025.
[8] P. Kocmi and O. Bojar, Trivial transfer learning to low-resource neural machine translator, in Proc. WMT, 2024, pp. 120313.
[9] A. Pope, D. Chen, and L. Smith, BLEU and beyond: Evaluation measures of neural machine translation, Comput—Linguistics, vol. 50, no. 2, pp. 355–372, 2024.
[10] S. Klejch et al, OCR error correction of low-resource scripts, in Proc. ICDAR, 2024, pp. 98105.
[11] D. Hughes and M. Brown, “Transformer-based OCR to digitise documents, Pattern Recognit. Lett., vol. 172, pp. 55–63, 2023.
[12] P. Winata et al., Multilingual self-attention to low-resource speech recognition, in Proc. ACL,

2022,
[13] pp. 231242.
[14] A. Conneau et al., “Unsupervised cross-lingual representation learning at scale, Trans. Assoc. Comput. Linguistics, vol. 10, pp. 125–141, 2022.
[15] Y. Liu et al., XLM-R: Cross-lingual language model pretraining, in Proc. ACL, 2022, pp. 1 15.
[16] A. Kahn et al., Neural machine translation evaluation methods revisited, in Proc. EMNLP in 2021, pp. 678 689.
[17] A. Graves et al., Speech recognition using deep recurrent neural networks, in Proc. ICASSP, 2013, pp. 6645 6649.
[18] J. Devlin et al., BERT: Pre-training of deep bidirectional transformers, in Proc. NAACL-HLT, 2019,
[19] pp. 41714186.
[20] A. Vaswani et al., Attention is all you need, in Proc. NeurIPS, 2017, pp. 59986008.
[21] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation through jointly learning to align and translate, in Proc. ICLR, 2015.
[22] I. Sutskever, O. Vinyals, and Q. Le, “Sequence to sequence learning using neural networks, in Proc. NeurIPS, 2014, pp. 31043112.
[23] R. Smith, “An overview of the Tesseract OCR engine, in Proc. ICDAR, 2007, pp. 629633.
[24] P. Koehn, Statistical Machine Translation. Cambridge, U.K.: Camb. Univ. Press, 2009.