A Project report

on

Multilingual Machine Translation

A Dissertation submitted to JNTU Hyderabad in partial fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology
in
Computer Science and Engineering

Submitted by

Gouher Naaz
(17H51A0524)

K Sai Kiran
(17H51A05E6)

P Dhanrajnath
(17H51A05G4)

Under the esteemed guidance of
Dr. N. Swapna
(Assistant Professor,
Dept. of CSE)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING & TECHNOLOGY
(An Autonomous Institution, Approved by AICTE, Affiliated to JNTUH, NAAC 'A+')
Kandlakoya, Hyderabad - 501401.

2017- 2021

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Major Project report entitled **" Multilingual Machine Translation "** being submitted by **Gouher Naaz *(17H51A0524)*, Kondi Sai Kiran *(17H51A05E6)*, Porika Dhanrajnath *(17H51A05G4),*** in partial fulfillment for the award of **Bachelor of Technology in Computer Science and Engineering** is a record of bonafide work carried out his/her under my guidance and supervision.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree.

Dr. N. Swapna
Assistant Professor
Dept. of CSE

Dr. K. Vijaya Kumar
Professor and HOD
Dept. of CSE

Acknowledgment

With great pleasure we want to take this opportunity to express our heartfelt gratitude to all the people who helped in making this project work a grand success.

We are grateful to **Dr. N. Swapna,** Assistant Professor, Department of Computer Science and Engineering for her valuable suggestions and guidance during the execution of this project work.

We would like to thank **Dr. K. Vijaya Kumar**, Head of the Department of Computer Science and Engineering, for his moral support throughout the period of my study in CMRCET.

We are highly indebted to **Major Dr. V.A. Narayana**, Principal CMRCET for giving permission to carry out this project in a successful and fruitful way.

We would like to thank the Teaching & Non- teaching staff of Department of Computer Science and Engineering for their co-operation

Finally, we express my sincere thanks to **Mr. Ch. Gopal Reddy**, Secretary, CMR Group of Institutions, for his continuous care. We sincerely acknowledge and thank all those who gave support directly and indirectly in completion of this project work.

GOUHER NAAZ (17H51A0524)

KONDI SAI KIRAN (17H51A05E6)

PORIKA DHANRAJNATH (17H51A05G4)

ABSTRACT

This project Multilingual Machine Translation is a computerized system that is designed to translate source text from various natural languages into target text of another natural languages. Neural machine translation is the application that is being studied from decades back to overcome the communication barriers that are mainly there due to the diversity in the regional languages. Machine translation (MT) system is available mostly for a specific language pair. In this project the translation of languages is done using Google trans libraries. Due to increased need of global communication, multilingual machine translation is the propel for researchers. With the advancement in technology, now we have computerized systems that can replace the human experts in particular domains like artificial intelligence (AI).

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1

INTRODUCTION

1. INTRODUCTION

1.1 INTRODUCTION

Translation enables communication between people from different regions. It provides meaningful communication from one language to another language. Languages embody many indigenous values and concepts and contain indigenous people's histories and development.

INDIA as a multi-lingual country is faced with challenges such as local language revitalization, language preservation and sharing knowledge and information in pursuit of development goals in rural areas. The cost of human interpretation and translation is high. However, the improvement in computer-aided software engineering tools coupled with availability of cheap technologies such as mobile phones and personal computers has reduced the cost for machine translation. Computer technology can be a powerful tool for providing materials in local languages to foster participation and inclusion of minorities in national development. Technologies that offer speech-to-speech or text-to-text communication from one language to another are one of the many ways that residents in multi-lingual societies can bridge communication gaps. For INDIA to achieve its national goals, there needs to be effective communication among the diverse people, whilst different local languages and cultures are preserved.

This project reports the development of a Multilingual Machine translation, with the aim of providing solutions to language barriers and improving the understanding of how technology can be used to bridge the communication gap among residents in India.

1.2 PROJECT DEFINITION

Translation is an activity, a product, and a process. As an activity, translation is a complex act that requires close reading of a text in the source language, understanding its meaning, and creating an equivalent text in the target language. The word "translation" also refers to the product of this activity: the final target language text that will be published or distributed

Almost any organization can benefit from translation. People learn better and retain information more effectively in their own language, and providing materials in the reader's native tongue will probably result in greater understanding and better learning outcomes. Translation can help organizations that work in languages that are not widely spoken around the world, that serve a public that speaks a second language, or that do not have the financial resources to develop their own materials to meet their training needs.

1.3 EXISTING SYSTEM

Machine translation is an automatic translation from one language to another. The benefit of machine translation is that it is possible to translate large swathes of text in a very short time. There are many machine translation systems are available with different translation approaches. Theses translation systems gives different performance based on the language pair and the approach used to translate. These Machine Translation approaches are normally categorized with considering the level of attention made on morphology, syntax, and semantics of the source and target.

There are many translators online, which provide functionality to translate the language into target language. Some of the translators like ‘Bablyon Online translator’ and ‘Reverso’ in the market just provide a single text input function to translate the language and doesn’t accept image file and voice as input.

Babylon Translator provides comprehensive translation results and full text translations between languages. - Pasteboard Integration – just copy a word from an email, browser or any application and get immediate translations! But it is provided with only one copying a text/sentence or typing it to translate, no features like accepting images or voice as input to translate into target language.

Reverso is an AI-based language tools translation aids and language services. These include online translation based on NMT (Neural Machine Translation), here in this tool the translation of language is done when the user select the target language and types some word/sentence in source language. This existing Reverso translator tool has a feature to accept

typing text as well as accepting documents for converting them, but it cannot perform actions to take input in the form of images or voice.

Disadvantage

All these existing systems have similar feature like, a text box to type or an option to attach a document file to translate and do not have feature likes accepting images and voice as input for translating text from one language to target language.

So to overcome this type of issues “Multilingual machine translation” is introduced.

CHAPTER 2

BACKGROUND WORK

2. BACKGROUND WORK

2.1 LITERATURE SURVEY

The first public Russian to English machine translation system was presented at the University of Georgetown in 1954 with a vocabulary of around 250 words. METEO is a fully automatic system for translating weather forecast in French and English for the whole of Canada. Starting at 7,500 words a day in 1977, the system today translates close to 80,000 words a day or more than 30 million words a year. It now performs 91% of the workload of Environment Canada's translation team in Ville Saint-Laurent, Quebec. SYSTRAN is French to English machine translation system based on hybrid 'direct-transfer' system in a rule-based paradigm. It was developed for technical texts at first, but with the huge dictionaries, it became able to translate any text from global domain. SYSTRAN versions were mostly developed for European languages. It now also supports Chinese, Russian, Japanese, Korean and Arabic languages. SYSTRAN was one of the first and most successful commercial MT systems. EUROTRA was an ambitious machine translation project established and funded by the European Commission from the late 1970s until 1994.The other rule based machine translation systems are ARIANE, and SUSY.

The dominant framework of MT research until the end of the 1980s was based on linguistic rules of various kinds: rules for syntactic analysis, lexical rules, and rules for lexical transfer, rules for syntactic generation, rules for morphology, etc. The rule-based approach was most obvious in the dominant transfer systems (Ariane, SUSY and Eurotra), but it was at the basis of all the various Interlingua systems -both those which were essentially linguistics-oriented and those which were knowledge-based.

Since 1989, however, the dominance of the rule-based approach has been broken by the emergence of new methods and strategies which are now loosely called `corpus-based methods. Firstly, a group from IBM published in 1988 the results of experiments on a system based purely on statistical methods. The effectiveness of the method was a considerable surprise to many researchers and has inspired others to experiment with statistical methods of various kinds in subsequent years. Secondly, at the very same time certain Japanese groups

began to publish preliminary results using methods based on corpora of translation examples, i.e. using the approach now generally called `example based translation. For both approaches the principal feature is that no syntactic or semantic rules are used in the analysis of texts or in the selection of lexical equivalents.

Most comprehensive Statistical Machine Translation systems are CMU, IBM, ISI and Google. These systems use phrase based approach and came out with good results. Google Translate is a free statistical machine translation service provided by Google Inc. to translate a section of text, document or webpage, into another language. The service was introduced in April 28, 2006 for the Arabic language, now it supports a large array of languages from all over the world. Prior to October 2007, for languages other than Arabic, Chinese and Russian, Google used a SYSTRAN based translator which is used by other translation services such as Babel Fish, AOL, and Yahoo. Adam Lopez at University of Edinburgh published a comprehensive survey of Statistical Machine Translation in August 2008. According to this survey report, statistical systems can be word based and phrase based. The major contribution to word-based system is by IBM [Brown et al. 1993, Berger et al. 1994]. IBM model 4 is described as target-to-source model. It produces the source sentence f from the target sentence . The model entails three steps:

1. Each target word e, select a fertility n and copies itself n times.
2. Each copy of each target word is translated into a single source word.
3. The source words are reordered into their final positions.

Model 4 alignment is asymmetric. Each source word can align to exactly one target word or the null word. However, a target word can link to arbitrary source words, as defined by fertility. The major problem with this model is of word reordering. If we ignore the word-reordering problem and use monotonic alignment as suggested by Zen and Ney 2004 will enable faster decoding. IBM Model for alignment is implemented in the open source toolkit GIZA and GIZA++. In phrase based model [Koehn et al. 2003] the unit of translation is phrase. Null transitions and fertility are not used. Each source phrase is non-empty and corresponds to exactly one target phrase. The translation process takes three steps:

1. The sentences are first split into phrases.
2. Each phrase is translated.
3. The translated phrases are permuted into their final order.

2.1.1 SMT for Indian Languages

Sanjay Kumar et.al. (2010) published a comprehensive survey of machine translation system developed for Indian Languages. Most of the systems are developed for translation from English to Hindi (Mantra machine translation system, MaTra system, AnglaBharti Technologies, Anuvaadak machine translation).

Statistical approach is used by a few Machine Translation Systems. Most of the systems exploit the close relationship among Indian languages and use direct approach or rule based approach. Goyal V. discussed a large number of Machine Translation Systems developed for nonIndian and Indian Languages. A few Indian systems are found to use Statistical Approach partially. Anglabharti-II(2004) uses statistical language model for automatic post editing. Shakti(2004) combines rule based approach with statistical approach for translation from English to Hindi, Marathi and Telugu. IBM has started work on English-Hindi translation system using Statistical approach. Unnikrishhanan et. al. (2010) explained the development of machine translation system using statistical approach for translating English to South Dravidian languages like Malayalam and Kannada. The various tools used for the development of the said system are: SRILM for creating language model, GIZA++ for training translation model and MOSES decoder for translating English to Malayalam (or Kannada). Other tools used at various levels of translation process are: The Stanford statistical parser, Roman to Unicode and Unicode to Roman converter, Morphological analyzer and generators, English morphological analyzer, Malayalam and Kannada morphological analyzers, Malayalam and Kannada morphological generators and Transfer rule file.

The main ideas implemented and proven very effective for English to south Dravidian languages SMT system are: i) reordering the English source sentence according to Dravidian syntax, ii) using the root suffix separation on both English and Dravidian words and iii) use of

morphological information for improving the quality of translation. The BLEU score using baseline system for English to Malayalam is 15.9 (and for English to Kannada is 15.4). When the syntax and morphological information was incorporated to the baseline system, BLEU scores show considerable improvement i.e. 24.9 and 24.5 respectively for above systems. The low BLEU scores are due to small training corpus size.

The performance evaluation shows that when reordering and morphological information was applied, the blue score increased approximately by 9.0. Akshar Bharti proposes an algorithm for aligning sentences in parallel corpus using Bilingual dictionary, Numerical matching, and Phonetic matching. The algorithm divides the sentences in noun chunks and verbs chunks and then aligns the sentences based on chunks using above dictionaries. The results of the algorithm were compared with Gale & Church Algorithm (1993) that is based on sentence length in terms of character and Brown's Algorithm (1991) that is based on sentences length in terms of words. The algorithm was tested on English Hindi parallel corpus. Solberg described a method for automatically creating a free lexicon for a part of speech tagger from a proprietary lexicon. The method uses annotated training data to extract word frequency, word-tag pair frequency, word tag lemma triple frequency, Tag-n-gram frequency and suffix information. Freely available taggers tested on Swedish corpus are fnTBL, Granska, MxPost, Stamp, TnT and Tree Tagger. Pradipta Ranjan Ray et al. discuss the rule-based algorithm for POS tagging and word grouping and suggest developing a Statistical Model of POS tagging using larger tag set. Vishal Goyal discusses the machine translation systems for non-Indian languages and second part discusses the machine translation systems for Indian languages. Philipp Koehn provides a gentle and accessible introduction to the latest methods and enables the reader to build machine translation systems for any language pair. It provides the necessary grounding in linguistics and probabilities, and covers the major models for machine translation: word-based, phrase based, and tree-based, as well as machine translation evaluation, language modeling, discriminative training and advanced methods to integrate linguistic annotation.

Richu Mittal provides overview of different approaches of Machine Translation. Due to Globalization Language translation is becoming a necessary part of human being's life.

Various machine translation techniques are there such as direct translation, example based, statistical machine translation, rule based translation. Sumita Rani presents a new approach to improve Punjabi to Hindi transliteration by combining a basic character to character mapping approach with Statistical Approach. Adam Lopez presents a tutorial overview of the state of the art. the context of the current research and then move to a formal problem description and an overview of the main sub problems: translation modeling, parameter estimation, and decoding. Along the way, we present a taxonomy of some different approaches within these areas. They conclude with an overview of evaluation and a discussion of future directions. Douglas Arnold gives detailed study of MT system which includes popular conceptions and their misconceptions, document preparation, authoring and Pre-Editing, Representing Linguistic Knowledge and the resource of Machine Translation. Gurpreet Singh Josan implemented Punjabi to Hindi translation with various research techniques based on Direct MT architecture and language corpus. The output is evaluated by already prescribed methods in order to get the suitability of the system for the Punjabi Hindi language pair. The existing Hindi to Dogri machine translation system has been developed by Dr.Preeti Dubey, Dr.Devanand and Dr.Shashi Pathania at University of Jammu, Jammu. The system is based on the direct approach of translation.

2.2 OBJECTIVES

Due to increased need of global communication, multilingual machine translation is the propel for researchers. With the advancement in technology, now we have computerized systems that can replace the human experts in particular domains like artificial intelligence (AI).

- The objective of this project is to translate text from one language to any other language in real-time with a button click. This project will be built using the Tkinter, googletrans libraries.
- In this project, the user enters text in any language and get it translated in any other language by selecting the output language.
- To develop a model which is used to translate the source language into target language.

- The ultimate goal of MMT project is to develop one model for translation between as many languages as possible by effective use of available linguistic resources.

2.3 SCOPE OF THE PROJECT

This project Multilingual Machine Translation is a computerized system that is designed to translate source text from various natural languages into target text of another natural languages. Machine translation (MT) system is available mostly for a specific language pair by providing additional features.

2.4 SYSTEM REQUIREMENTS

A) Software requirements :-

Operating system : Windows or Mac OS.

GOOGLE COLAB or Jupyter Notebook (ANACONDA) for developing the system

B) Hardware requirements :-

Hard disk: minimum 20GB, recommended 100GB or more

RAM: Minimum 4GB or more

Processor: intel i5 or more latest versions

2.5 WORKING

A graphical user interface is developed for better use, where any user can interact with the application. The GUI represents four kinds of services for particular user, where the user needs to select required one from among them. The services that are provided here are ‘word or sentence translation’, ‘documentation translation’, ‘Image text input translation’ and ‘voice input translation’.

If the user wants to select word or sentence service then a single line of sentence entered and target language code is selected then that language is translated to the targeted language. Similarly for documentation translation, image translation, voice translation the same process is followed.

CHAPTER 3

PROPOSED SYSTEM

3. PROPOSED SYSTEM

3.1 PROPOSED SYSTEM

The system which proposed is used to translate source language into target language. Source language is detected automatically and the target language code should be given by the user which is provided.

This Multilingual machine translator is a tool to translate text, words and phrases from one language to any other language. It is like a dictionary where we can translate the text. It accepts four types of inputs as show in fig 3.1

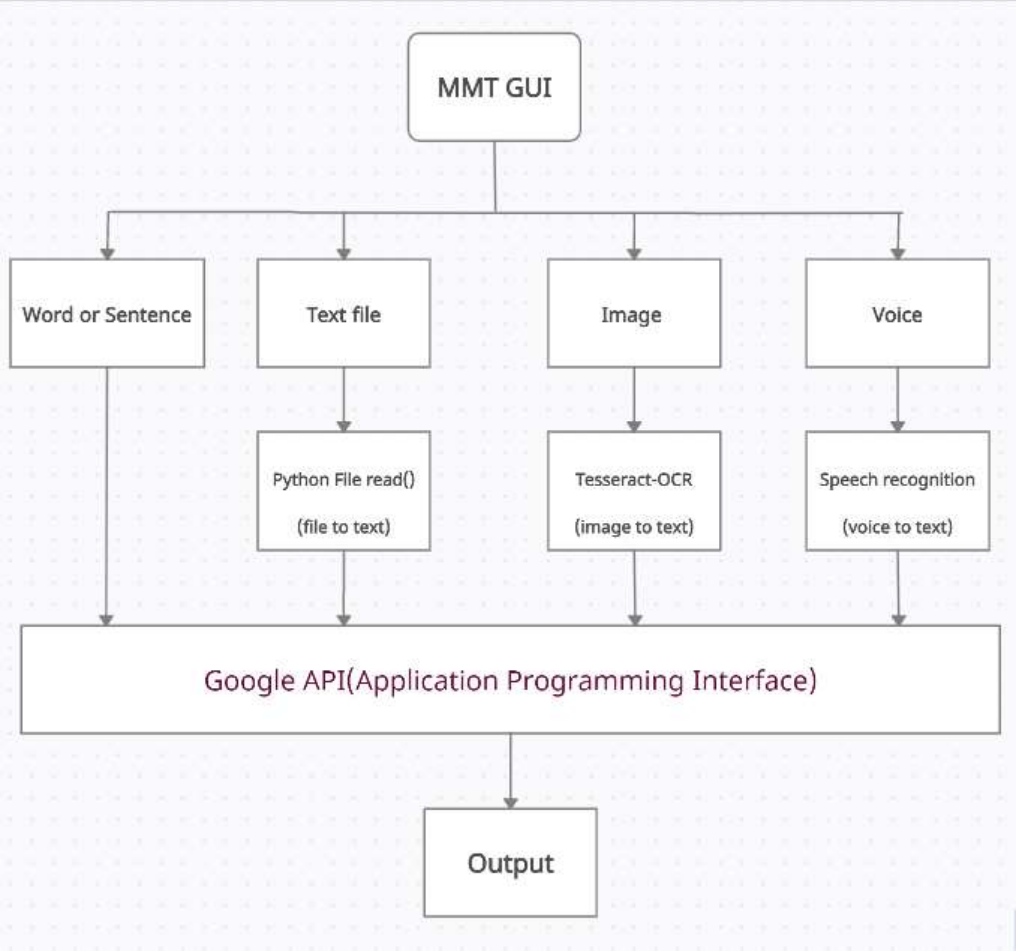


Fig 3.1 Proposed system workflow

The four types of inputs are:

1) Word or sentence
2) Text file
3) Image
4) Voice

Word or sentence

This feature accepts word or sentence as an input, if any user wants to type some words or sentence in source language and convert it into target language. Here in the GUI when the user selects “Word or sentence” a textbox is provided to enter the text for translating it into another language with a click by entering target language code.

Text file

This feature accepts text file as an input, if any user wants to translate a document containing text into another language. Here in the GUI the user has to select “Text file” option, then an option is provided to choose files for translating it into another language with a click by entering target language code.

Image

This feature accepts Image as an input, if any user wants to translate text that is present on an image into another language. Here in the GUI the user has to select “Image” option, then an option is provided to choose image for translating text present on it into another language with a click by entering target language code.

Voice

This feature accepts Voice as an input, if any user wants to translate whatever they speak. Here in the GUI the user has to select “Voice” option, then an option is provided to record voice by holding a button for translating voice of source language it into another language with a click by entering target language code.

All natural languages have different grammatical structures, although the magnitude of difference in grammatical structure between two languages is relative, that is, some language

pairs might have a low difference magnitude of grammatical structure while others have higher. Translating languages with low difference magnitude is little challenge, but the reverse poses a great challenge.

- This system supports all Indian regional languages.
- This proposed system has additional features like Image text translation and voice input translation when compared to existing systems.

3.2 PYTHON

Python is a general-purpose coding language—which means that, unlike HTML, CSS, and JavaScript, it can be used for other types of programming and software development besides web development.

That includes back end development, software development, data science and writing system scripts among other things. Technically, machine learning falls under data science, but bear with me here. Using Python for machine learning is pretty cool, so it felt like it warranted an additional line item.

Using python programing language the developement of project code is done.

3.3 JUPYTER NOTEBOOK

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.

The Jupyter Notebook is based on a set of open standards for interactive computing. Think HTML and CSS for interactive computing on the web. These open standards can be leveraged by third party developers to build customized applications with embedded interactive computing.

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

3.4 GRAPHICAL USER INTERFACE

The graphical user interface is a form of user interface that allows users to interact with electronic devices through graphical icons and audio indicator such as primary notation, instead of text-based user interfaces, typed command labels or text navigation. GUIs were introduced in reaction to the perceived steep learning curve of command-line interfaces which require commands to be typed on a computer keyboard.

The actions in a GUI are usually performed through direct manipulation of the graphical elements. Beyond computers, GUIs are used in many handheld mobile devices such as MP3 players, portable media players, gaming devices, smartphones and smaller household, office and industrial controls. The term GUI tends not to be applied to other lower-display resolution types of interfaces, such as video games, or not including flat screens, like volumetric displays because the term is restricted to the scope of two-dimensional display screens able to describe generic information, in the tradition of the computer science research at the Xerox Palo Alto Research Center.

3.5 TKINTER

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with standard Linux, Microsoft Windows and Mac OS X installs of Python.

There are several popular GUI library alternatives available, such as wxPython, PyQt, PySide, Pygame, Pyglet, and PyGTK. This term has different meanings in different contexts, but in general it refers to a rectangular area somewhere on the user's display screen.

A window which acts as a child of the primary window. It will be decorated with the standard frame and controls for the desktop manager. It can be moved around the desktop and can usually be resized.

3.5.1 Tkinter filedialog

Python Tkinter offer a set of dialogs that you can use when working with files. By using these you don't have to design standard dialogs your self. Example dialogs include an open file dialog, a save file dialog and many others. Besides file dialogs there are other standard dialogs, but in this article we will focus on file dialogs.

File dialogs help you open, save files or directories. This is the type of dialog you get when you click file,open. This dialog comes out of the module, there's no need to write all the code manually. Tkinter does not have a native looking file dialog, instead it has the customer tk style.

The appearance of the dialog is different on every operating system. It will look different on Windows, Mac and Linux (gnome).

3.6 NEURAL MACHINE TRANSLATION

Neural Machine Translation (NMT) has arisen as the most powerful algorithm to perform this task. While Google Translate is the leading industry example of NMT, tech companies all over the globe are going all in on NMT. This state-of-the-art algorithm is an application of deep learning in which massive datasets of translated sentences are used to train a model capable of translating between any two languages. With the vast amount of research in recent years, there are several variations of NMT currently being investigated and deployed in the industry. One of the older and more established versions of NMT is the Encoder Decoder

structure. This architecture is composed of two recurrent neural networks (RNNs) used together in tandem to create a translation model as shown in fig 3.5

Neural machine translation is a recently proposed approach to machine translation. Unlike the traditional statistical machine translation, the neural machine translation aims at building a single neural network that can be jointly tuned to maximize the translation performance. The models proposed recently for neural machine translation often belong to a family of encoder–decoders and encode a source sentence into a fixed-length vector from which a decoder generates a translation. In this project, we conjecture that the use of a fixed-length vector is a bottleneck in improving the performance of this basic encoder–decoder architecture as show in fig 3.2, and propose to extend this by allowing a model to automatically search for parts of a source sentence that are relevant to predicting a target word, without having to form these parts as a hard segment explicitly.

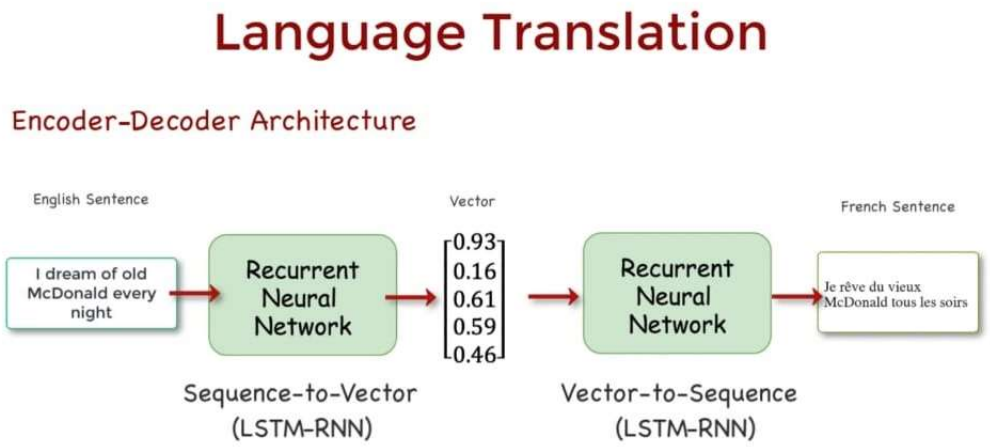


Fig 3.2 Encoder – Decoder Architecture

With this new approach, we achieve a translation performance comparable to the existing state-of-the-art phrase-based system on the task of language translation.

Most of the proposed neural machine translation models belong to a family of encoder–decoders, with an encoder and a decoder for each language, or involve a language-specific encoder applied to each sentence whose outputs are then compared. An encoder neural network

reads and encodes a source sentence into a fixed-length vector. A decoder then outputs a translation from the encoded vector. The whole encoder–decoder system, which consists of the encoder and the decoder for a language pair, is jointly trained to maximize the probability of a correct translation given a source sentence.

3.6.1 Google Neural Machine Translation

(GNMT) is a neural machine translation (NMT) system developed by Google and GNMT improves on the quality of translation by applying an example-based (EBMT) machine translation method in which the system "learns from millions of examples". GNMT's proposed architecture of system learning was first tested on over a hundred languages supported by Google Translate. With the large end-to-end framework, the system learns over time to create better, more natural translations. GNMT attempts to translate whole sentences at a time, rather than just piece by piece. The GNMT network can undertake interlingual machine translation by encoding the semantics of the sentence, rather than by memorizing phrase-to-phrase translations.

3.6.2 Long Short Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks are a modified version of recurrent neural networks, which makes it easier to remember past data in memory.

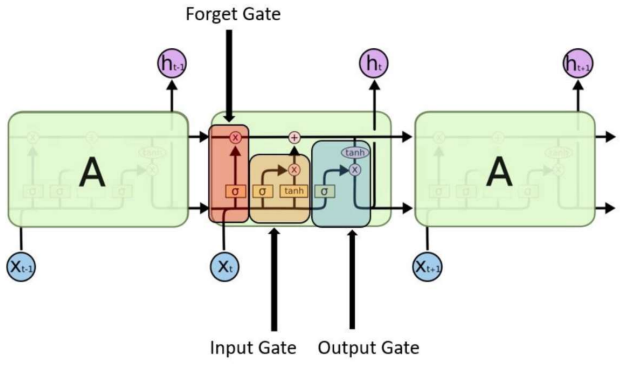


Fig 3.3 Long Short Term Memory

The vanishing gradient problem of RNN is resolved here. LSTM is well-suited to classify, process and predict time series given time lags of unknown duration. It trains the model by using back-propagation.

3.6.3 Recurrent Neural Network (RNN)

Recurrent Neural Network is a generalization of feed forward neural network that has an internal memory. RNN is recurrent in nature as it performs the same function for every input of data while the output of the current input depends on the past one computation as depicted in fig 3.4. After producing the output, it is copied and sent back into the recurrent network. For making a decision, it considers the current input and the output that it has learned from the previous input.

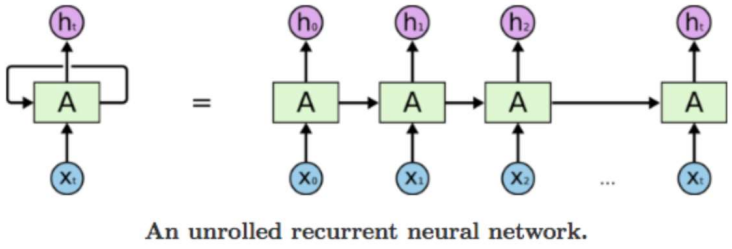


Fig 3.4 Recurrent Neural Network

Unlike feedforward neural networks, RNNs can use their internal state (memory) to process sequences of inputs. This makes them applicable to tasks such as unsegmented, connected handwriting recognition or speech recognition. In other neural networks, all the inputs are independent of each other. But in RNN, all the inputs are related to each other.

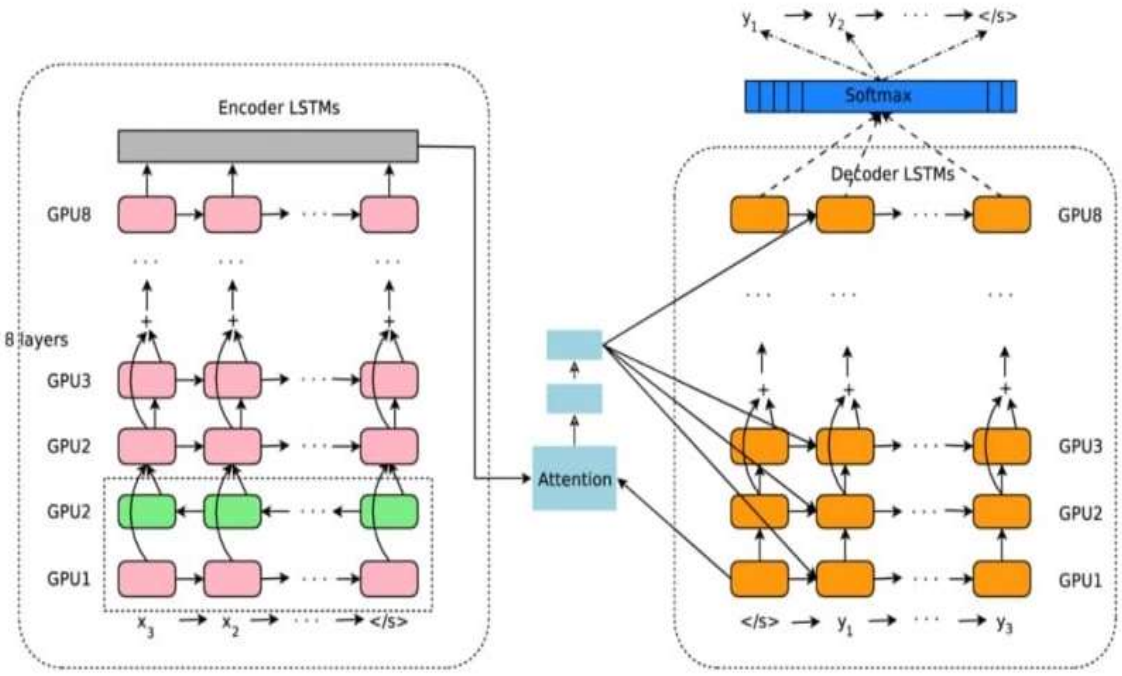


Fig 3.5 LSTM-RNN Architecture

3.7 TESSERACT OCR

Tesseract is an open source text recognition (OCR) Engine, available under the Apache 2.0 license. It can be used directly, or (for programmers) using an API to extract printed text from images. It supports a wide variety of languages. Tesseract doesn't have a built-in GUI, but there are several available from the 3rdParty page. Tesseract is compatible with many programming languages and frameworks through wrappers that can be found here. It can be used with the existing layout analysis to recognize text within a large document, or it can be used in conjunction with an external text detector to recognize text from an image of a single text line.

Tesseract 4.00 includes a new neural network subsystem configured as a text line recognizer. It has its origins in OCRopus' Python-based LSTM implementation but has been redesigned for Tesseract in C++. The neural network system in Tesseract pre-dates TensorFlow but is compatible with it, as there is a network description language called

Variable Graph Specification Language (VGSL), that is also available for TensorFlow. The Tesseract OCR process flow is shown in fig-3.6

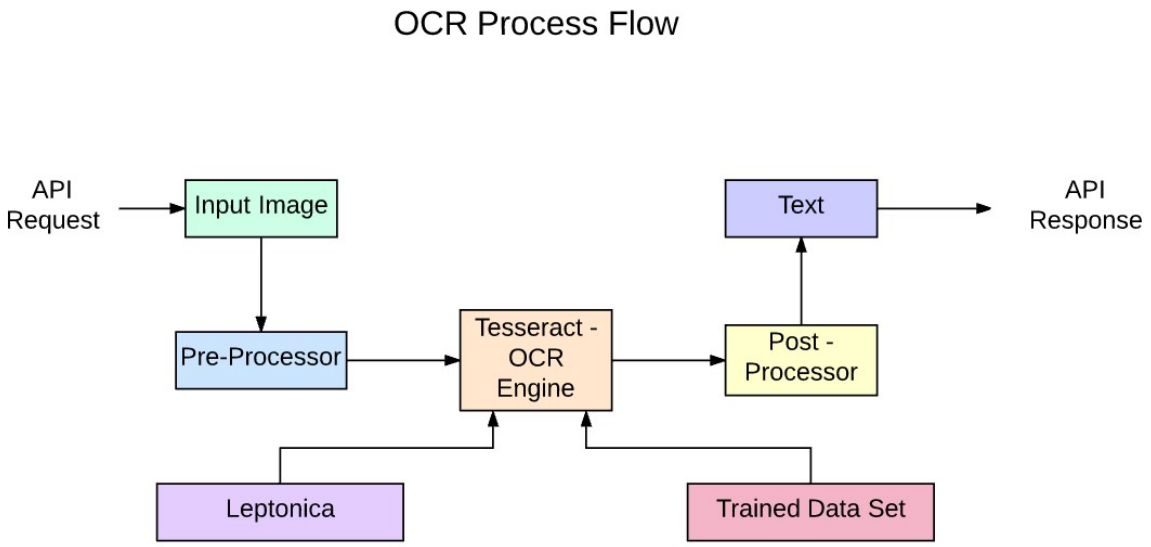


Fig 3.6 OCR Process Flow

3.7.1 Pytesseract

Python-tesseract is an optical character recognition (OCR) tool for python. That is, it will recognize and "read" the text embedded in images.

Python-tesseract is a wrapper for Google's Tesseract-OCR Engine. It is also useful as a stand-alone invocation script to tesseract, as it can read all image types supported by the Pillow and Leptonica imaging libraries, including jpeg, png, gif, bmp, tiff, and others. Additionally, if used as a script, Python-tesseract will print the recognized text instead of writing it to a file.

image_to_string Returns unmodified output as string from Tesseract OCR processing as shown in fig 3.7

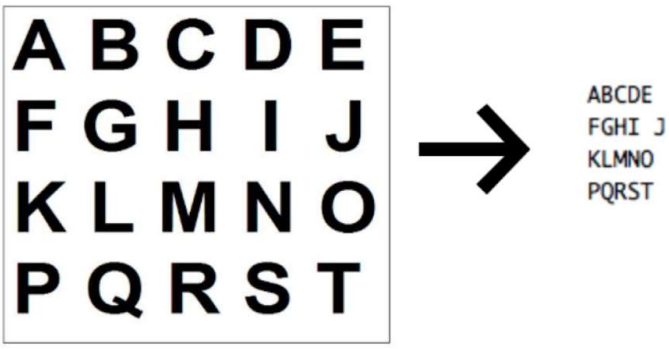


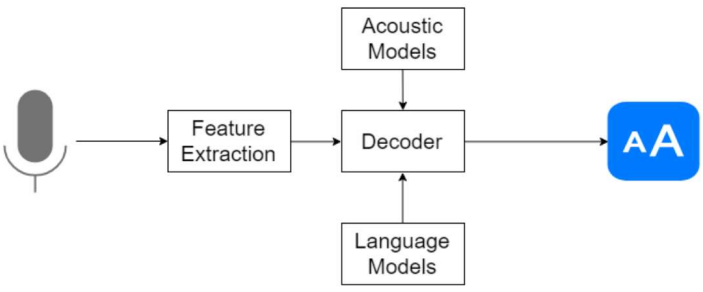
Fig 3.7 image to string

3.8 SPEECH RECOGNITION

Speech Recognition is an important feature in several applications used such as home automation, artificial intelligence, etc. This article aims to provide an introduction on how to make use of the SpeechRecognition library of Python. This is useful as it can be used on microcontrollers such as Raspberri Pis with the help of an external microphone.

3.8.1 Speech to text translation:

This is done with the help of Google Speech Recognition. Speech to text translation is based on the Acoustic and Language modeling as depicted in fig 3.8. Acoustic modeling represents the relationship between linguistic units of speech and audio signals. Language modeling matches sounds with word sequences to help distinguish between words that sound similar.



3.8 Speech to text

CHAPTER 4
DESING & IMPLEMENTATION

4. DESIGN & IMPLEMENTATION

4.1 USE CASE DIAGRAM

Use case diagram is a behavioral UML diagram type and frequently used to analyze various systems. They enable you to visualize the different types of roles in a system and how

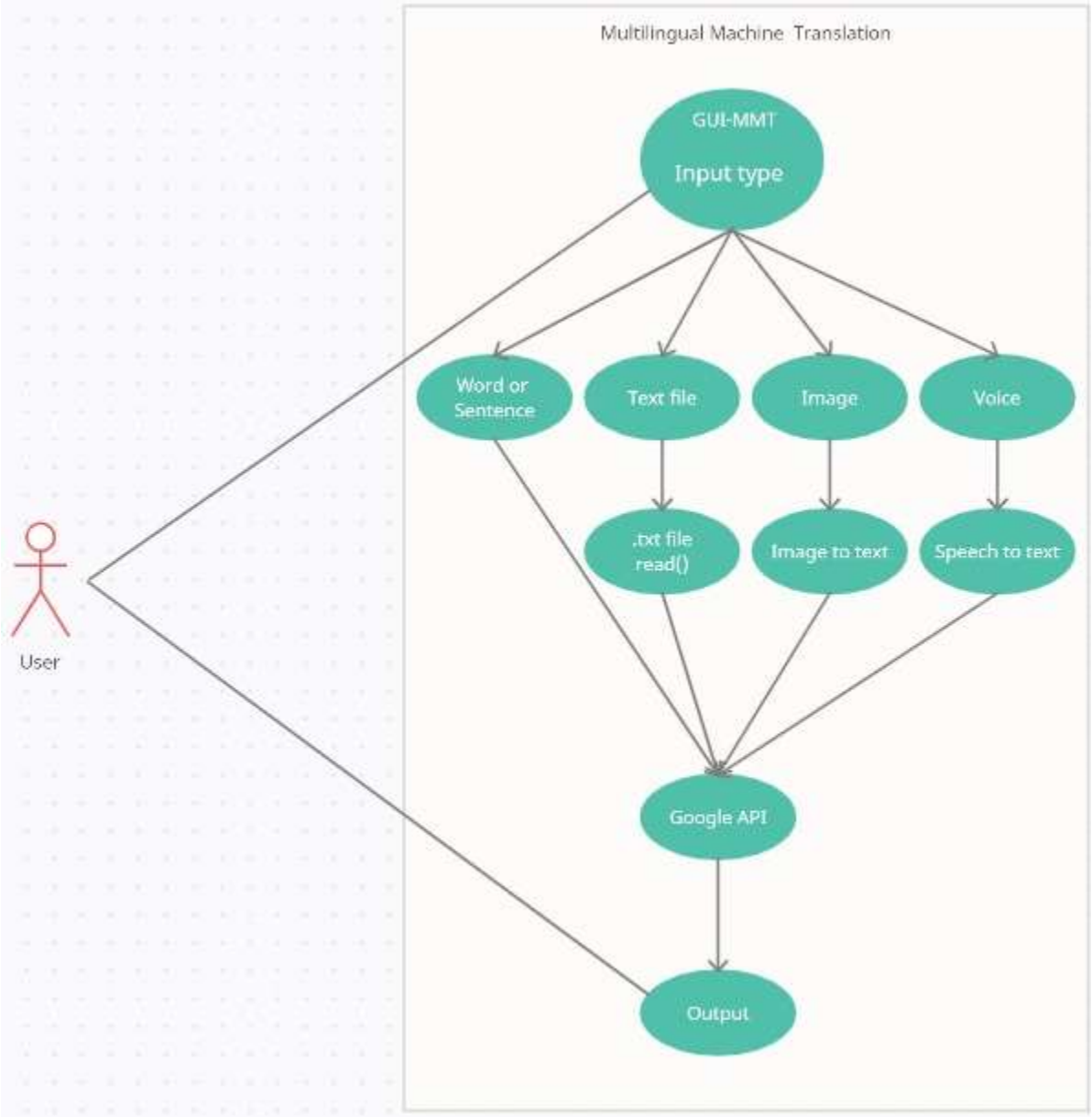


Fig 4.1 Use case Diagram

those roles interact with the system. This use case diagram tutorial will cover the following topics and help you create use cases better.

To implement this project, we will use the concepts of Python, Tkinter, and google_trans_new libraries. Google_trans_new is a module to translate text. We import the Translator from google_trans_new, which is used to do translations. We also import LANGUAGES from google_trans_new which support various languages but we only involved Indian regional languages.

4.2 GOOGLE_TRANS

- Google_trans_new is a module to translate text. We import the Translator from google_trans_new, which is used to do translations.

We also import LANGUAGES from google_trans_new which support various languages but we only involved Indian regional languages.

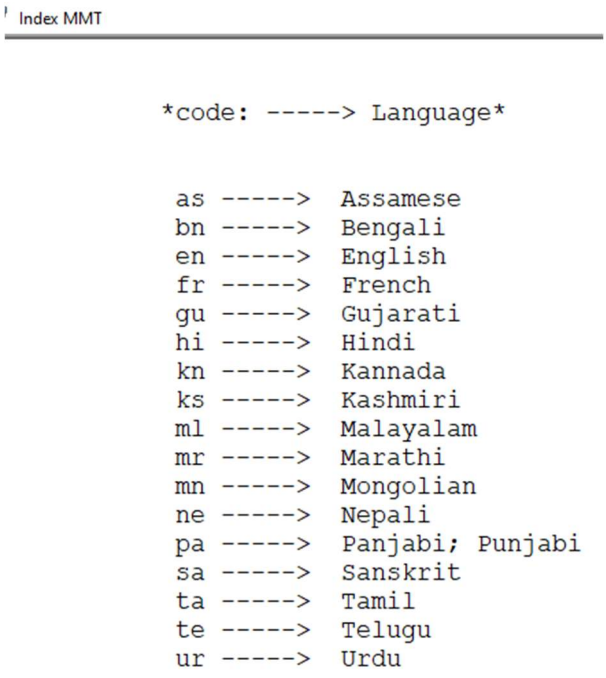


Fig 4.2 languages Codes

4.3 Improting libraries:

We import python packages like langdetect for detecting the input language or text automatically, google_trans_new for calling google api which translates the text , pytesseract for converting image to text and speech_recognition for voice to text.

Tkinter python package is used to develop the Graphical User Interface (GUI) for this model. Filedialog.askopenfile() method is used to collect the files from the user like .txt, .png, .jpg etc

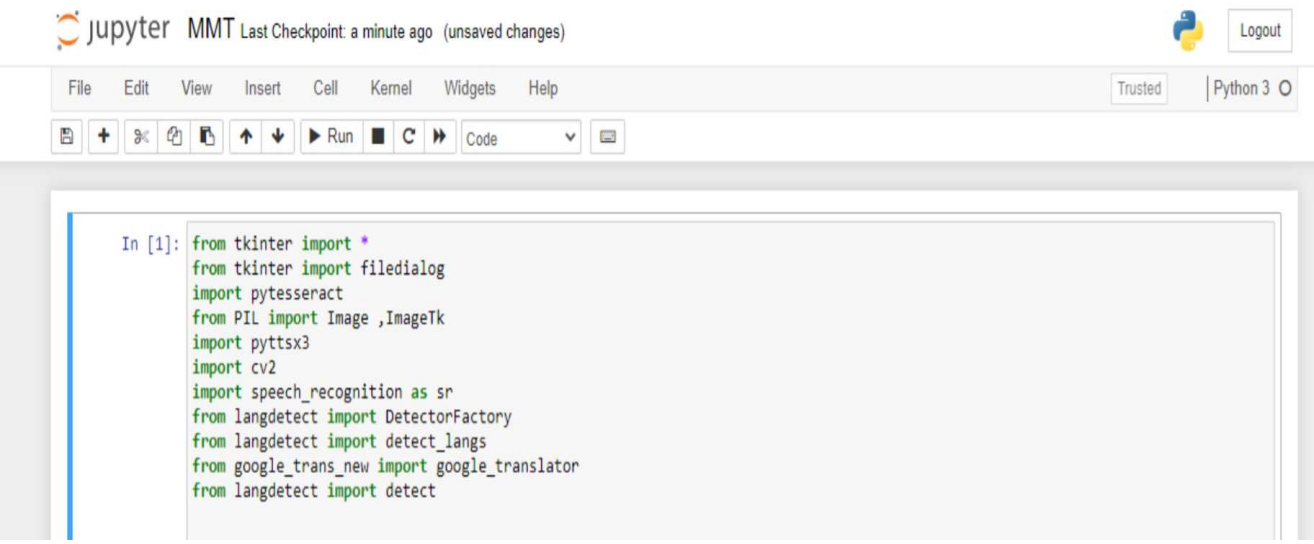


Fig 4.3 libraries

4.4 TYPES OF INPUTS

There are 4 different types of inputs present in this project

1. Word or Sentence
2. Text document
3. Image
4. Voice

4.4.1 Word or Sentence

In this type of input the input text can be a word or a sentence with 5000 characters limit.

This input is passed to google api, then google_translator() and translate() methods will perform actions like sending input value to google translator and returns the translated text or sentence.

```
#input type-1
def word_sentence():

    def printData(res):
#         print(res)
#         result = res
#         resultLabel.config(padx = 10, justify = CENTER, font =("Courier", 14), text=result)
        text.delete("1.0", "end")
        text.insert(INSERT, res)
        text.pack()


    def get_input():
        sentence = entry1.get()
        code = entry2.get()
        r = detect_and_translate(sentence,target_lang=code)
        printData(r)

    def delete():
        entry1.delete(0,'end')
        entry2.delete(0,'end')




    root1 = Tk()
    root1.title("Multilingual Machine Translation")

    # root.geometry("500x500")
```

Fig 4.4 Word or sentence code

4.4.2 Text document.

In this type of input the input has to be document with 5000 characters limit.

This input is taken by filedialog.askopenfile() method and python files concept is used to read the content of file and then passed to google api then google_translator() and translate() methods will perform actions like sending input value to google translator and returns the translated sentence.

```
#input type-2
def txtfile():

    root2 = Tk()
    root2.title("Multilingual Machine Translation")

    # root.geometry("500x500")

    #getting screen width and height of display
    width= root2.winfo_screenwidth()
    height= root2.winfo_screenheight()

    #setting tkinter window size
    root2.geometry("%dx%d" % (width, height))
    root2.configure(bg='#856ff8')

    text = Text(root2)

    def printData(res):
#         print(res)
#         result = res
#         res_text.config(padx = 10, justify = CENTER, font =("Courier", 14), text=result)
        text.delete("1.0", "end")
        text.insert(INSERT, res)
        text.pack()

    # Function for opening the file
    def openFile():
        filepath = filedialog.askopenfilename(initialdir=r"C:\Users\Porika Dhanrajnath\Desktop\NLP",
                                              title="Open file",
                                              filetypes= (("text files","*.txt"),
                                              ))
#         file = open(filepath,'r')
        text_data = open(filepath, 'r').read()
        sentence = text_data
        code = entry1.get()
        r = detect_and_translate(sentence,target_lang=code)
        printData(r)
```

Fig 4.5 Text document code

4.4.3 Image

In this type of input the input will be an image.

This input is taken by filedialog.askopenfile() method and Tesseract-OCR is used to convert image to string and then string is passed to google api then google_translator() and translate() methods will perform actions like sending input value to google translator and returns the translated sentence or string.

```
#input type-3
def img_file():
    root3 = Tk()
    root3.title("Multilingual Machine Translation")
    # root.geometry("500x500")
    #getting screen width and height of display
    width= root3.winfo_screenwidth()
    height= root3.winfo_screenheight()
    #setting tkinter window size
    root3.geometry("%dx%d" % (width, height))
    root3.configure(bg='#856ff8')

    text = Text(root3)
    def printData(res):
        text.delete("1.0", "end")
        text.insert(INSERT, res)
        text.pack()
    def select_image():
        path = filedialog.askopenfilename()
        code = entry1.get()
        img = Image.open(path)
        printData(img)
        pytesseract.pytesseract.tesseract_cmd ='C:/Program Files (x86)/Tesseract-OCR/t
        result_text = pytesseract.image_to_string(img)
#         result_text = pytesseract.image_to_string(img, lang='tel')
#         print("Text detected!")
#         print("\n"+result_text)
        text4=detect_and_translate(result_text,target_lang=code)
        printData(text4)
```

Fig 4.6 Pytesseract code

4.4.4 Voice

In this type of input the input is a voice that is given by user.

This input is taken by user voice which undergoes speech_recognition() and then it is converted into text then passed to google api then google_translator() and translate() methods will perform actions like sending input value to google translator and returns the translated sentence.

```
#input type-4
def voice():

    root4 = Tk()
    root4.title("Multilingual Machine Translation")

    # root.geometry("500x500")

    #getting screen width and height of display
    width= root4.winfo_screenwidth()
    height= root4.winfo_screenheight()

    #setting tkinter window size
    root4.geometry("%dx%d" % (width, height))
    root4.configure(bg='#856ff8')

    text = Text(root4)
#     dtext= Text(root4)

    def printData(res):
#         print(res)
#         result = res
#         res_text.config(padx = 10, justify = CENTER, font =("Courier", 14), text=result)
        text.delete("1.0", "end")
        text.insert(INSERT, res)
        text.pack()

    def detect_text(s):
        label2 = Label(root4,text=s)
        label2.pack()
        label2.config(padx = 10, justify = CENTER, font =("Courier", 14))


    def record():

        recorder=sr.Recognizer()
        with sr.Microphone() as source:
#             print("Speak Now! ")
            printData("Speak Now!")
            audio=recorder.listen(source)

    # txt=recorder.recognize_google(audio)
        try:
        # Auto detect the language
            txt=recorder.recognize_google(audio)
        except sr.UnknownValueError:
            printData("Google Speech Recognition could not understand audio")
```

Fig 4.7 speech_recognition code

CHAPTER 5

RESULTS

5. RESULTS

5.1 Dataset

A data set (or dataset) is a collection of data. The data set lists values for each of the variables, for each member of the data set. Each value is known as a datum. Data sets can also consist of a collection of documents or files.

The proposed project dataset contains 350 documents related to telugu language, 250 documents related to Hindi language, 450 documents related to English language and 4797 words related to Urdu.

Table 5.1 Dataset length

Language	**Dataset length**
Telugu	350 documents
Hindi	250 documents
English	450 documents
Urdu	4797 words

5.2 Accuracy

The accuracy of our project is calculated using the below formula

$$Accuracy = \frac{\text{Correctly identified words with root}}{\text{Total no. of words}}$$

Accuracy calculated for four languages is show in fig 5.1, the languages are english, telugu, hindi and urdu. The highest accuracy among these four languages is for english and the lowest accuracy is for urdu.

Table 5.2 Accuracy percentage

Language	**Accuracy**
Telugu	83%
English	84%
Hindi	82%
Urdu	80%

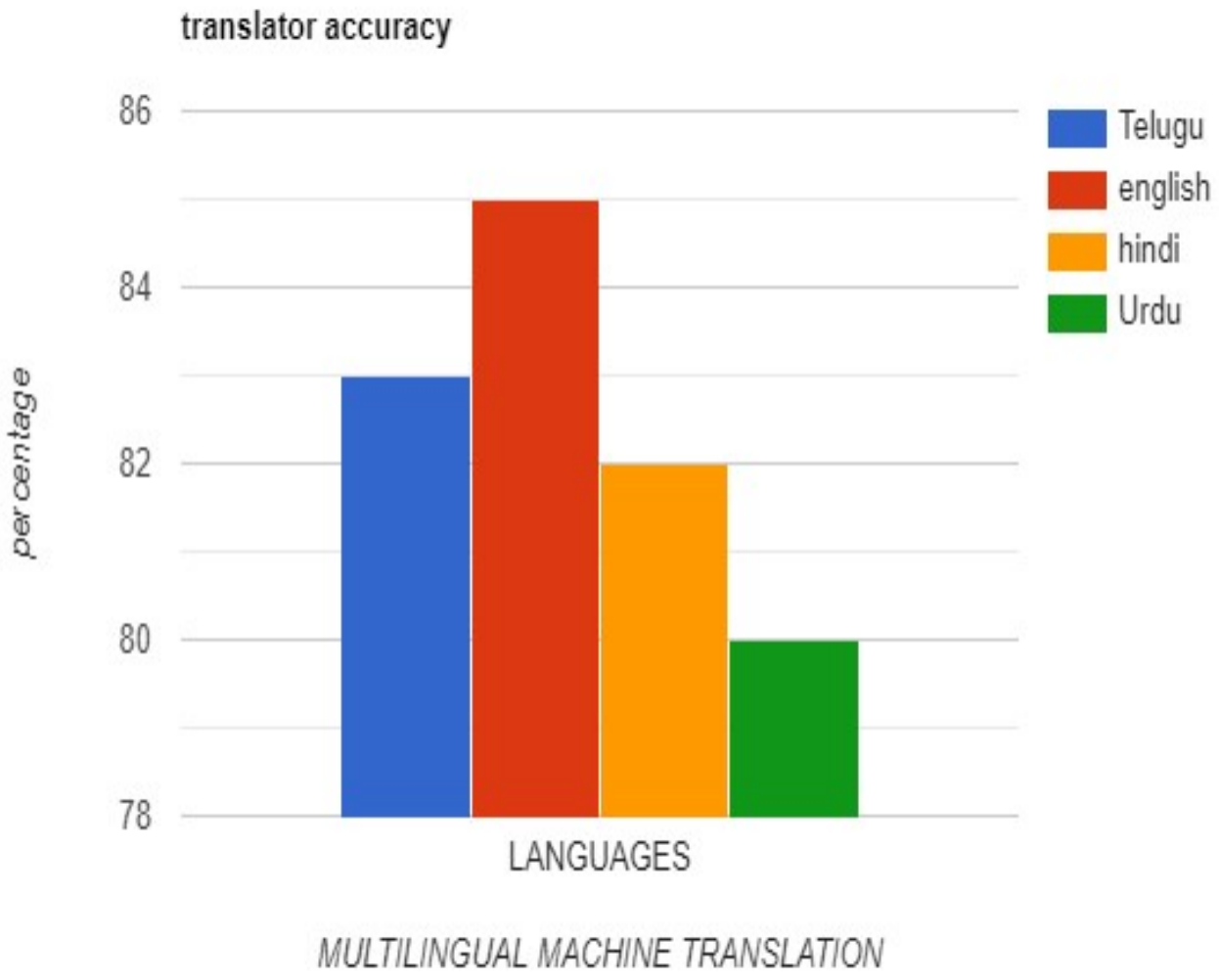


Fig 5.1 language translation accuracy

5.3 Results

This is the Main screen interface of our project as dipicted in the fig 5.2 . When any user open Multilingual machine translator this interface interface appears where the user will be able to select anyone among four types of options avaliable.

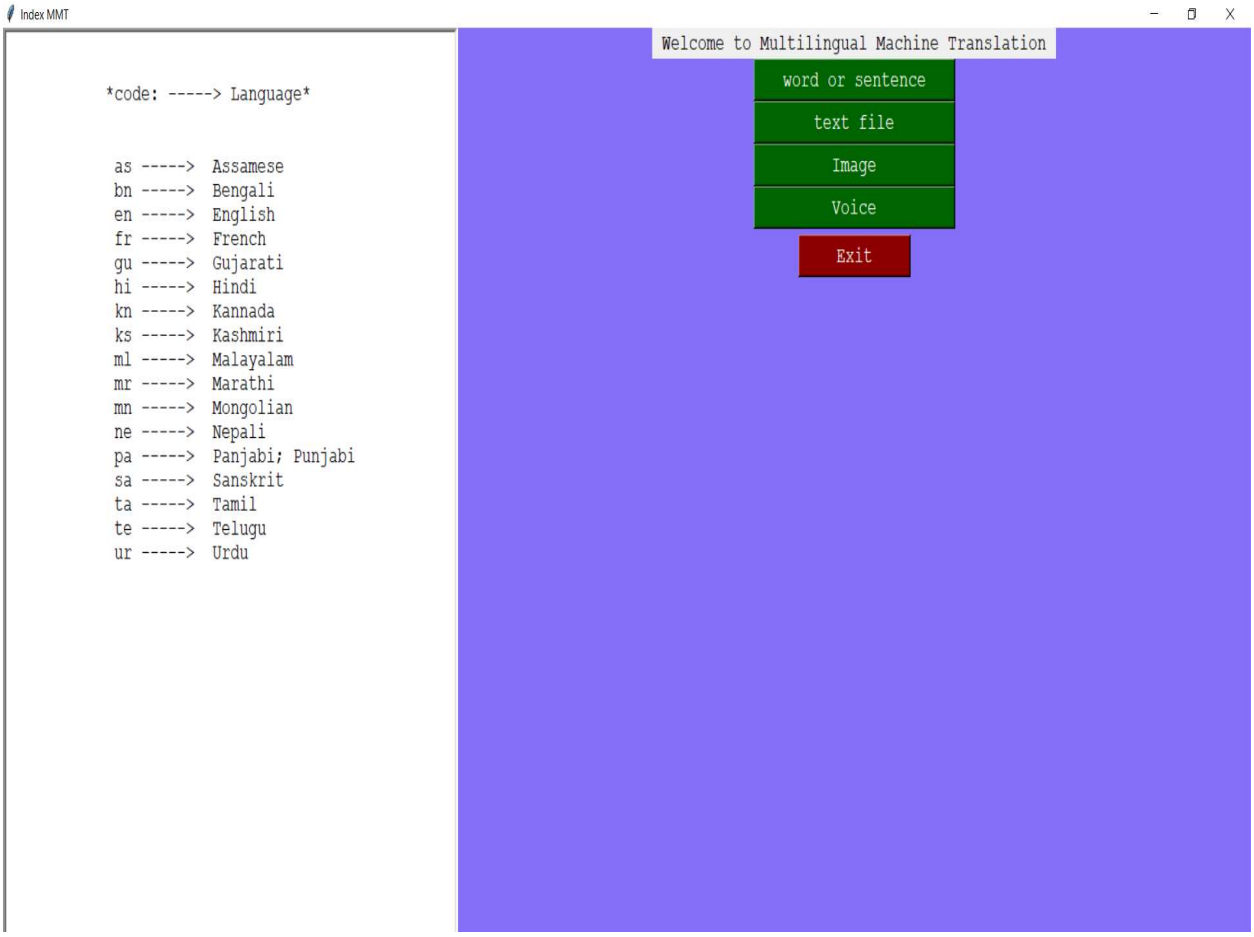


Fig 5.2 MMT Main screen interface

Word or Sentence interface

When the user select the option ‘Word or sentence’ the resulting interface displays as show below in fig 5.3 appears.

Here the user will be able to type word or sentence for translating it/them to the target language by enter the respective language code available.

Fig 5.3 MMT word or sentence interface

The output for word or sentence the is as shown in Fig 5.4

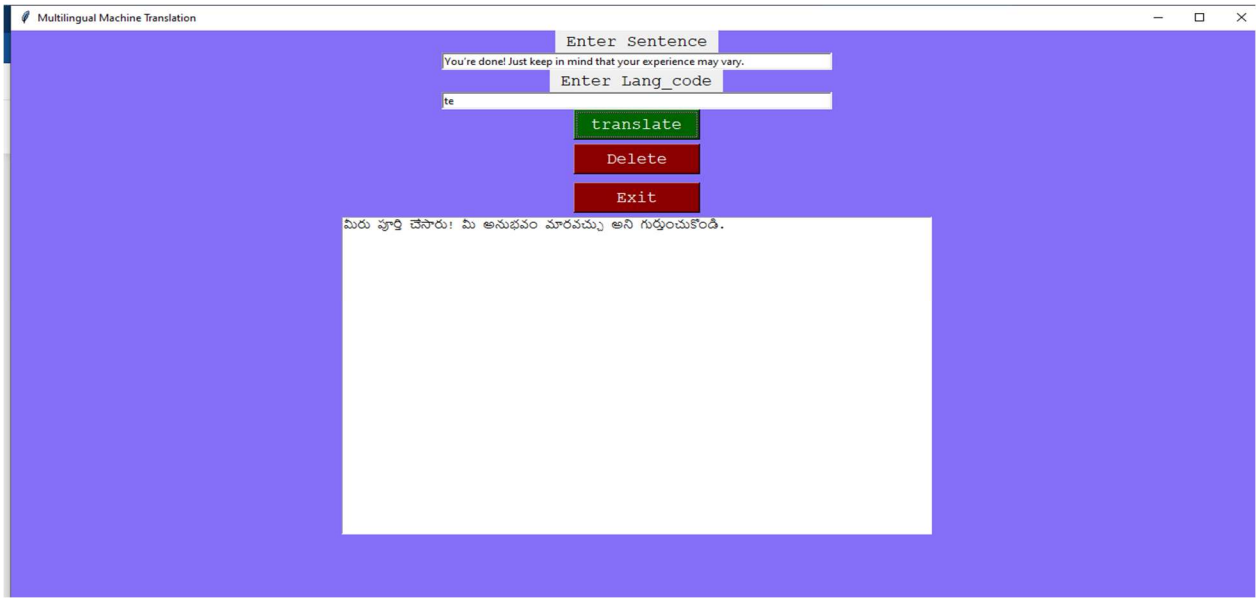


Fig 5.4 MMT word or sentence output

Text document interface

When the user select the option ‘Text document’ the resulting interface displays as show below in fig 5.5

Here the user has to select the target language by enter the respective language code available as shown in fig 4.1 at first, then will be able to choose text document for translating the text document language into target language.

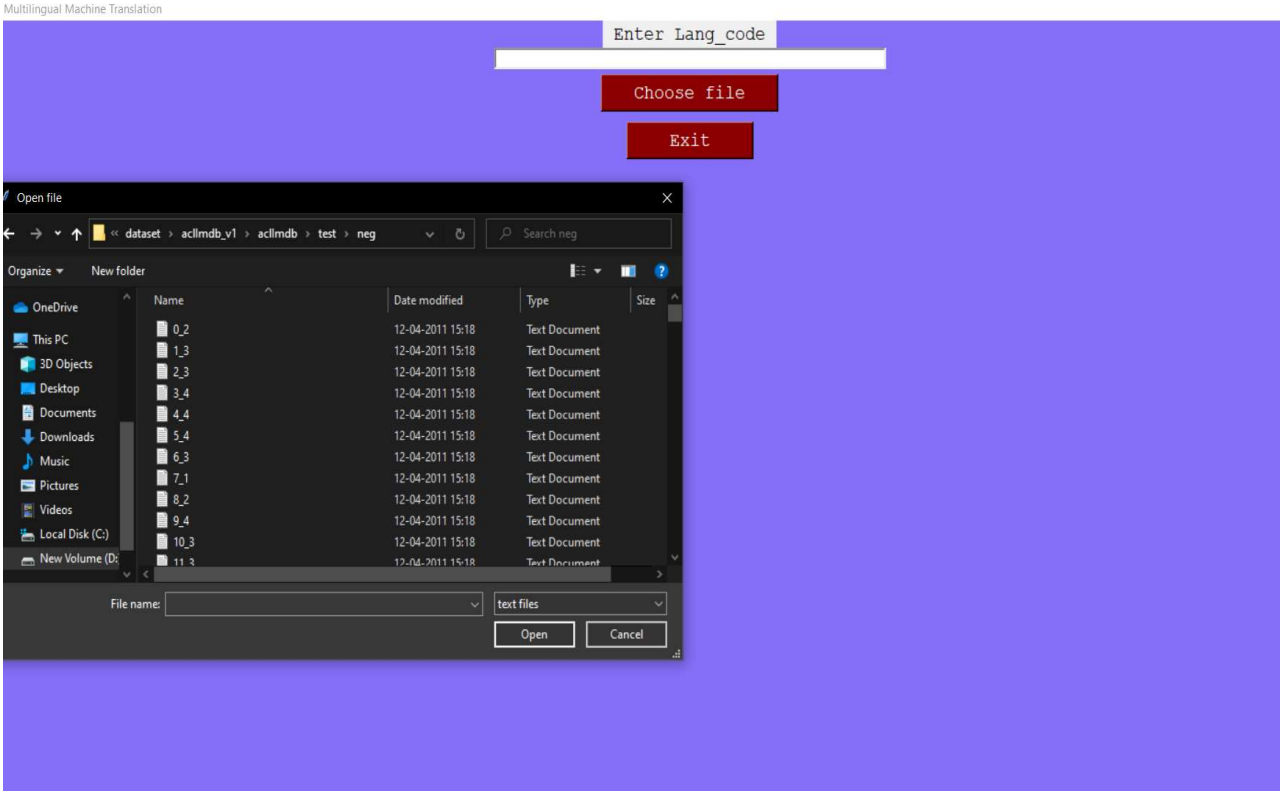


Fig 5.5 MMT Text document interface

The output for the language present in Text document after translation will result as show in Fig 5.6

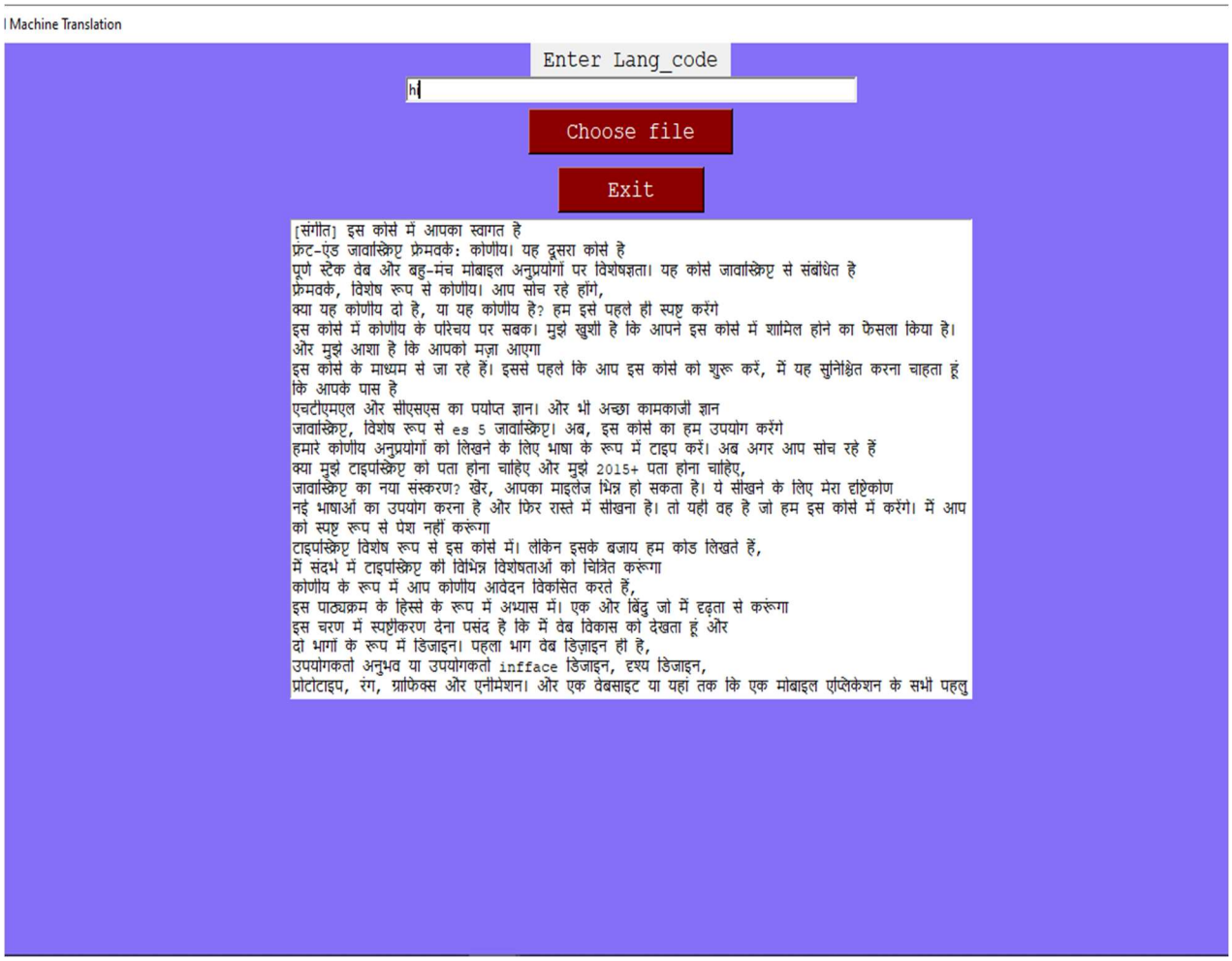


Fig 5.6 MMT Text document output

Image interface

When the user select the option ‘Image’ the resulting interface displays as show below in fig 5.7

Here the user has to select the target language by enter the respective language code available as shown in fig 4.1 at first, then user will be able to choose image containing text for translating the language that is present on the image in to the target language

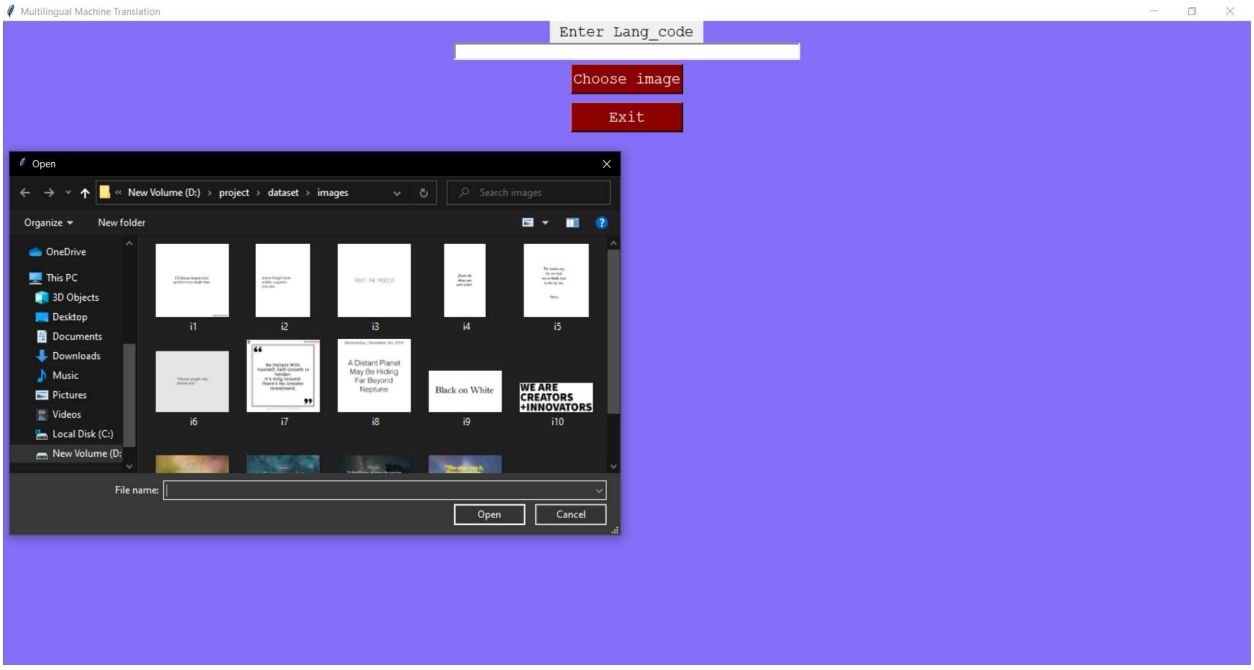


Fig 5.7 MMT Image interface

The ouput for the text present in image after translation will results as show in fig 5.8

Fig 5.8 MMT image output

Voice interface

When the user select the option “Voice” the resulting interface will display as show below in Fig 5.9

Here the user has to select the target language by enter the respective language code available as shown in fig 4.1 at first, then user has to press the ‘record voice’ and then start speaking. After recording, the language is translated in to the target language

Fig 5.9 MMT voice interface

The output for recorded voice after translating into target language is show along with the record (you said) as show in Fig 5.10

Fig 5.10 MMT voice output

CHAPTER 6

CONCLUSION
&
FUTURE WORK

6. CONCLUSION & FUTURE WORK

6.1 Conclusion

We have developed the Language Translator python project. We used the popular tkinter library for rendering graphics on a display window, google_trans_new library to translate text from one language to another.

- Machine translation is challenging given the inherent ambiguity and flexibility of human language.
- The work presented in this project is also promising, because it demonstrates an effective translation.

6.2 Future Work

In the future this project can be evolved by implementing the text to speech of target language and can increase accuracy.

And also a feature to work with, where users can download the translated text.

REFERENCES

- https://machinelearningmastery.com/introduction-neural-machine-translation/
- https://machinelearningmastery.com/encoder-decoder-recurrent-neural-network-models-neural-machine translation/#:~:text=The%20encoder%2Ddecoder%20recurrent%20neural,units%20and%20an%20attention%20mechanism.
- https://pypi.org/project/langdetect/
- https://pypi.org/project/google-trans-new/
- https://pypi.org/project/SpeechRecognition/
- https://docs.python.org/3/library/tkinter.html
- https://www.tutorialspoint.com/python/python_gui_programming.htm
- https://realpython.com/python-gui-tkinter/
- https://opensource.google/docs/
- https://docs.python.org/3/library/dialog.html
- https://towardsdatascience.com/neural-machine-translation-15ecf6b0b
- https://research.google/pubs/pub45610/
- https://translate.google.co.in/