JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

KANGLISH TO ENGLISH TEXT-TO-TEXT TRANSLATOR

[1] Amina Afaq,[2] Mahek Khan,[3] Syed Jawad
[,4] Prof. Asghar Pasha, [5] Prof. Priyanka K
[1,2,3,4] Students,[4]Associate professors, [5] Assistant Professor
[1,2,3,4,5,6]Department of Information Science and Engineering,
HKBK College Of Engineering, Bengaluru, India

Abstract: **In the present communication-based society, no natural language seems to have been left untouched by the trends of code-mixing. For different communicative purposes, a language uses linguistic codes from other languages. This gives rise to a mixed language which is neither totally the host language nor the foreign language. The mixed language poses a new challenge to the problem of machine translation. It is necessary to identify the "foreign" elements in the source language and process them accordingly.**
The foreign elements may not appear in their original form and may get morphologically transformed as per the host language. Further, in a complex sentence, a clause/utterance may be in the host language while another clause/utterance may be in the foreign language. Code-mixing of Kannada and English where Kannada is the host language is a common phenomenon in day-to-day language usage in the Indian metropolis. The scenario is so common that people have started considering this a different variety altogether and calling it by the name Kanglish. In this project, we present a mechanism for machine translation of Kanglish to pure (standard) Kannada and pure English forms. Our application is suitable for Hindi, Kannada, Tamil and Telugu translation also. Our application gives more accuracy for Kannada and Hindi translation. This work aims to improve the accuracy of Kanglish to English translation up to 98 per cent, to build a novel deep learning architecture to efficiently predict the translation and minimize the translation time in during the entire process.

I. INTRODUCTION

In the present communication-based society, no natural language seems to have been left untouched by the trends of code-mixing. For different communicative purposes, a language uses linguistic codes from other languages. This gives rise to a mixed language which is neither totally the host language nor the foreign language. The mixed language poses a new challenge to the problem of machine translation. It is necessary to identify the "foreign" elements in the source language and process them accordingly. The foreign elements may not appear in their original form and may get morphologically transformed as per the host language. Further, in a complex sentence, a clause/utterance may be in the host language while another clause/utterance may be in the foreign language. Code-mixing of Kannada and English where Kannada is the host language is a common phenomenon in day-to-day language usage in the Indian metropolis. The scenario is so common that people have started considering this a different variety altogether and calling it by the name Kanglish. In this project, we present a mechanism for machine translation of Kanglish to pure (standard) Kannada and pure English forms. Code-mixing is a frequently encountered phenomenon in day-to-day natural language communication in metropolises, especially among educated people. The phenomenon is so common that this is often considered a different (emerging) variety of the language.

Natural language processing (**NLP**) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data. The goal is a computer capable of "understanding" the contents of documents, including t contextual nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves. Natural language processing has its roots in the 1950s. Already in 1950, Alan Turing published an article titled "Computing Machinery and Intelligence" which proposed what is now called the Turing test as a criterion of intelligence, though at the time that was not articulated as a problem separate from artificial intelligence.

Neural networks: A major drawback of statistical methods is that they require elaborate feature engineering. Since 2015, the field has thus largely abandoned statistical methods and shifted to neural networks for machine learning. Popular techniques include the use of word embeddings to capture semantic properties of words, and an increase in end-to-end learning of a higher-level task (e.g., question answering) instead of relying on a pipeline of separate intermediate tasks (e.g., part-of-speech tagging and dependency parsing).

II. IMPLEMENTATION

2.1 Algorithm

Sequence-to-Sequence (Seq2Seq) Modelling

Sequence-to-Sequence (seq2seq) models are used for a variety of NLP tasks, such as text summarization, speech recognition, and DNA sequence modelling, among others. We aim to translate given sentences from one language to another.
Here, both the input and output are sentences. In other words, these sentences are a sequence of words going in and out of a model. This is the basic idea of Sequence-to-Sequence modelling. The figure below tries to explain this method.

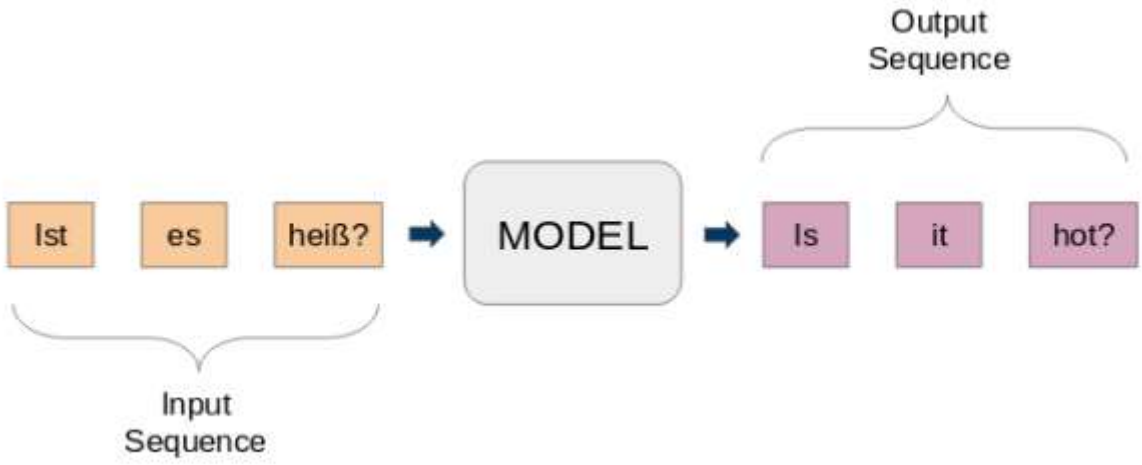


Fig Seq2Seq Model

A typical seq2seq model has 2 major components –

a) an encoder
b) a decoder

Both these parts are essentially two different recurrent neural networks (RNN) models combined into one giant network:

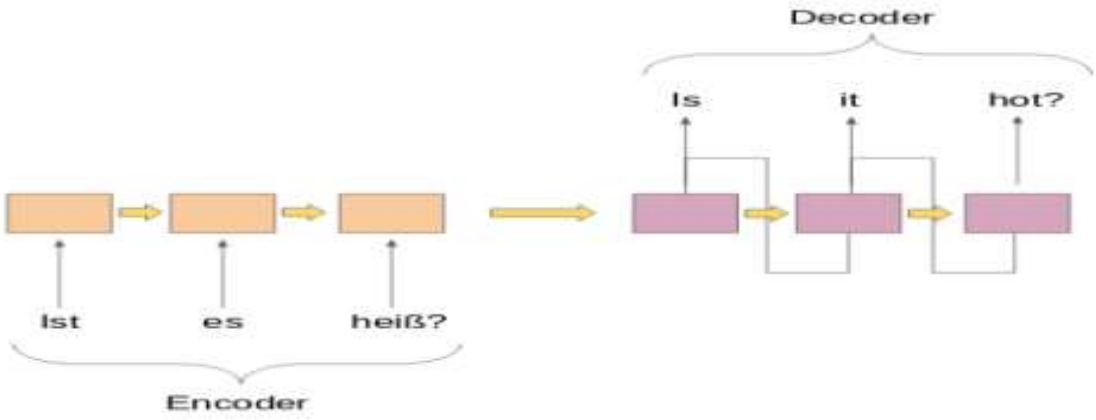


Fig Encoder and Decoder

I've listed a few significant use cases of Sequence-to-Sequence modelling below

- Speech Recognition
- Name Entity/Subject Extraction to identify the main subject from a body of text
- Relation Classification to tag relationships between various entities tagged in the above step
- Chatbot skills to have the conversational ability and engage with customers
- Text Summarization to generate a concise summary of a large amount of text
- Question Answering systems

2.2 Working Steps:

In this section, we go through the flow of the working of the Kanglish To English Text-To-Text Translator

Step 1:Sentence Pre-Process

In this module we are passing Kanglish sentence as input to the system will split the sentence into words and remove the noise in the sentence using NLP.

I. Input the Kanglish sentence.
II. Read the sentence.
III. Remove noise using the stemming process.
IV. Split the sentence into words.

Step 2: Sentence Transcription

In this module, we are using the word as input to the system and will use Indic transliteration. San script class to convert to the given language and merge the words.

I. Input the words.
II. For each word apply the transcription.
III. Merge to sentence.

Step 3: Sentence Translation

In this module, we are passing the transcribed sentence to the system and will apply the google translator to translate Kannada to English.

I. Input the Kannada sentence.
II. Apply Google Translator to convert the Kannada to English.
III. Return English sentence.

Step 4: Design the UI and Integrate it with the saved ML models:

In this model, we are designing the graphical user interface using Tkinter and integrating it with the background code.

I. Design the Desktop UI using Python Tkinter.
II. Use the user-defined functions in each button to integrate UI with the model.
III. Run the project.

III. METHODOLOGY

The methodology is generally a body of methods, rules, and postulates employed by a discipline which generally studies Discipline or the way something is done. The mixed language poses a new challenge to the problem of machine translation. It is necessary to identify the “foreign” elements in the source language and process them accordingly. Thus let us see how to implement Kanglish using NLP for a better understanding of how it works.

3.1 System Architecture

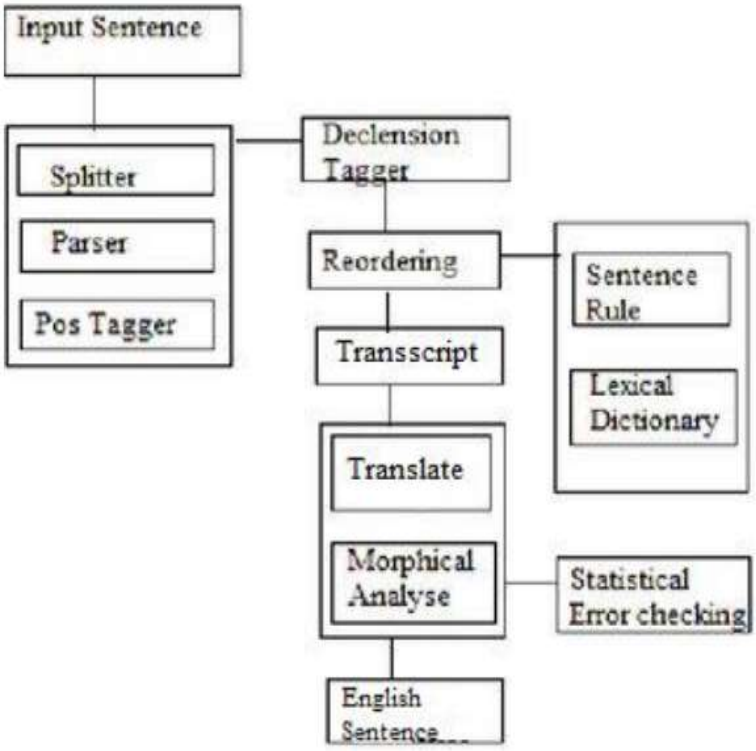


Fig: System Architecture of Kanglish Translator

Existing System:

Code-Mixed language plays a very important role in communication in multilingual societies and with the recent increase in internet users especially in multilingual societies, the usage of such mixed language has also increased. However, the cross translation between the Hinglish Code-Mixed and English and vice-versa has not been explored very extensively. With the recent success of large pre-trained language models, we explore the possibility of using multilingual pre-trained transformers like mBART and mT5 for exploring one such task of code -mixed Hinglish to English machine translation. Further, we compare our approach with the only baseline over the PHINC dataset and report a significant jump from 15.3 to 29.5 in BLEU scores, a 92.8% improvement over the same dataset. Disadvantages of Existing System:

a) It is suitable for only Hinglish to English translation.
b) It is not suitable for our regional languages

Proposed System:

In the present communication-based society, no natural language seems to have been left untouched by the trends of code-mixing. For different communicative purposes, a language uses linguistic codes from other languages. This gives rise to a mixed language which is neither totally the host language nor the foreign language. The mixed language poses a new challenge to the problem of machine translation. It is necessary to identify the "foreign" elements in the source language and process them accordingly. The foreign elements may not appear in their original form and may get morphologically transformed as per the host language. Further, in a complex sentence, a clause/utterance may be in the host language while another clause/utterance may be in the foreign language. Code-mixing of Kannada and English where Kannada is the host language is a common phenomenon in day-to-day language usage in the Indian metropolis. The scenario is so common that people have started considering this a different variety altogether and calling it by the name Kanglish. In this project, we present a mechanism for machine translation of Kanglish to pure (standard) Kannada and pure English forms.

Advantages:
a) Our application is suitable for Hindi, Kannada, Tamil and Telugu translation also.
b) Our application gives more accuracy for Kannada and Hindi translation.

Aim of the project:
- To improve the accuracy up to 98%.
- To build a novel deep learning architecture to efficiently predict the translation.
- To minimize the translation time in during the entire process.

3.2 Data Flow Diagram:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing.

3.2.1 Level 0

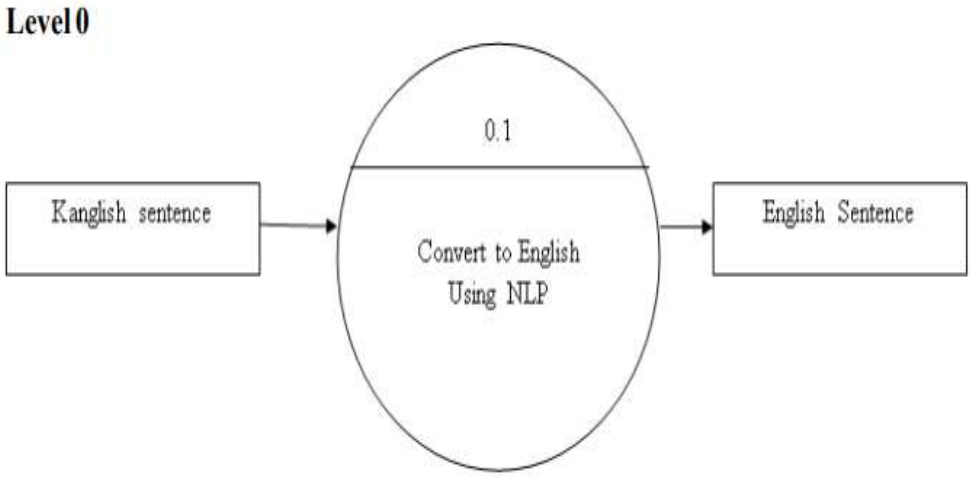


Fig: Overall process of the project

Level 0: Describes the overall process of this project. We are passing the KDD dataset as input the system will efficiently process the imbalanced dataset and convert it to the balanced dataset using the DSSTE algorithm and various machine learning and deep learning algorithm and shows the performance comparison of those algorithms.

3.2.2 Level 1

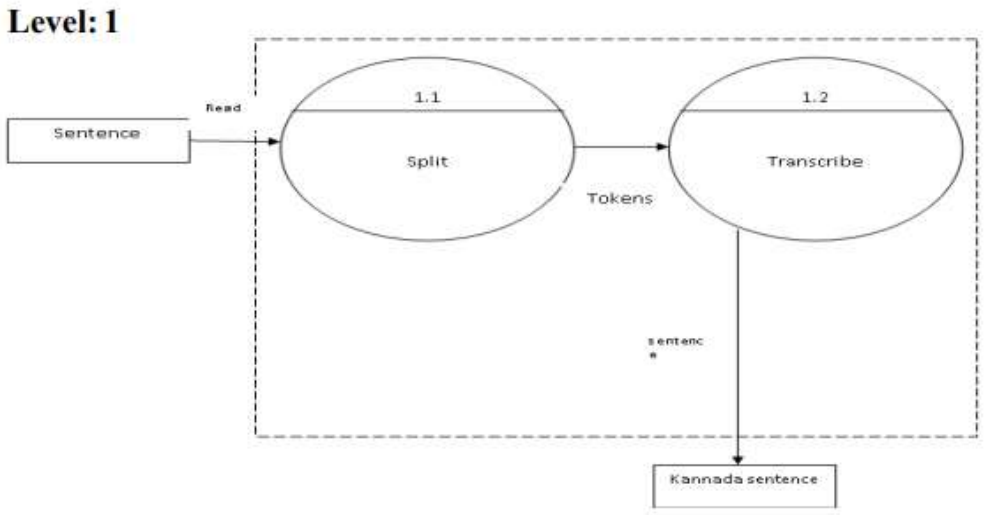


Fig: First Stage Of Project

Level 1: Describes the first stage process of this project. we are passing the KDD dataset as a dataset to the system and will preprocess and extract the important features using DSSTE.

3.2.3 Level 2

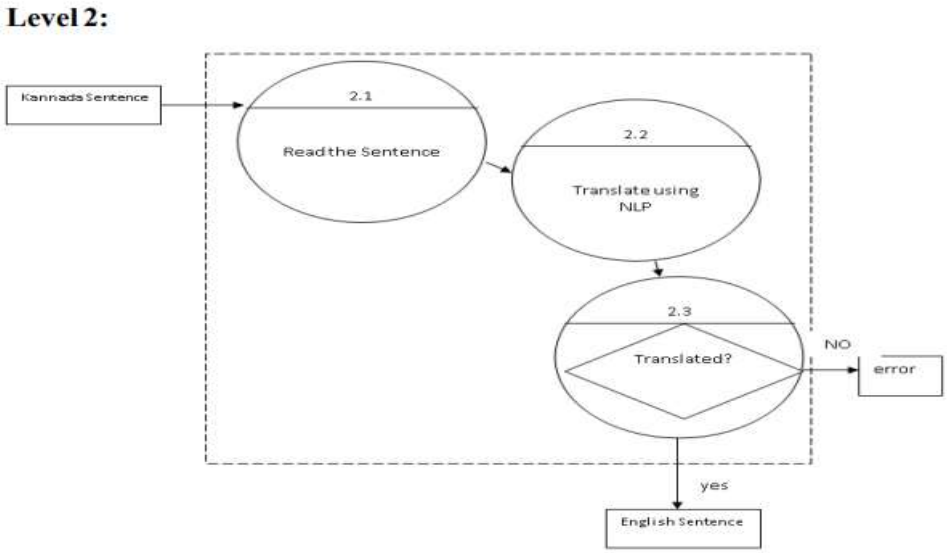


Fig: Final Stage Process of Project

Level 2: Describes the final stage process of this project. We are passing extracted features from level by applying a machine learning model and a deep learning system will detect the intrusion in the imbalanced dataset.

3.3 Sequence Diagram

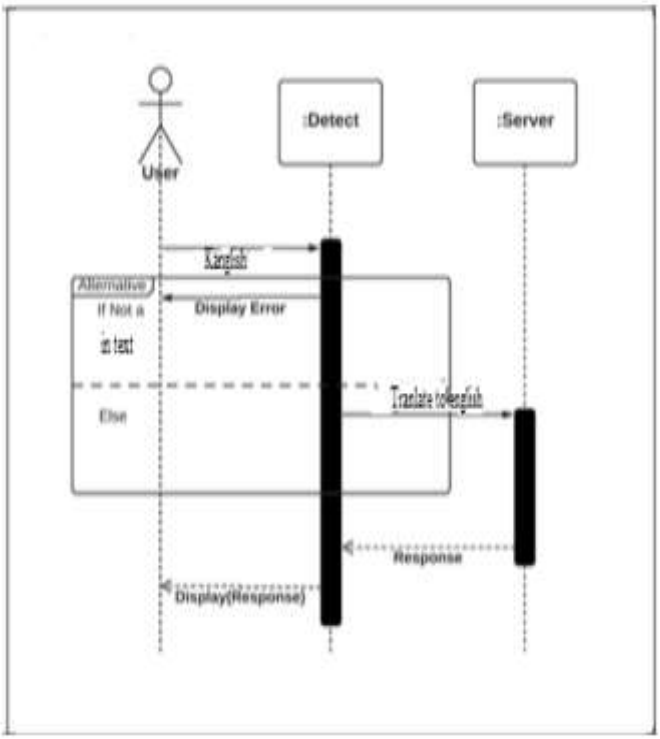


Fig: Sequence Diagram

3.3 Activity Diagram

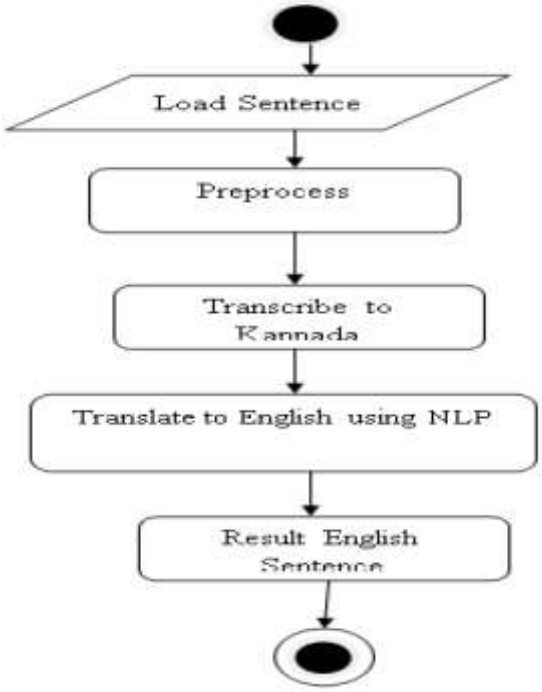


Fig: Activity Diagram

IV. TESTING AND OUTCOMES

Software testing is a method to check whether the actual software product matches expected requirements and to ensure that the software product isDefect-free. It involves the execution ofsoftware/system components using manual orautomated tools to evaluate one or more propertiesof interest. The purpose of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements. Some prefer sayingSoftware testing definition as a White Box andBlack Box testing. In simple terms, Software Testing means the Verification of the Application Under Test.

WHITE BOX TESTING: WHITE BOX TESTING IS A TESTING IN WHICH THE SOFTWARE TESTERKNOWS THE INNER WORKINGS, STRUCTURE AND LANGUAGE OF THE SOFTWARE, OR AT LEAST ITS PURPOSE. IT ISUSED TO TEST AREAS THAT CANNOT BE REACHED FROM A BLACKBOX LEVEL.

BLACK BOX TESTING: Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested.

Test cases for the project:

Test Case 1	TC01
Test Name	User input format
Test Description	To test user input values
Input	Kanglish/Hinglish- letters
Expected Output	The letters should read and display in the console
Actual Output	The letters read and displayed in the console
Test Result	Success

Test Case 2	UTC02
Test Name	User input format
Test Description	To test user input values
Input	Kanglish/Hinglish- letters as null
Expected Output	Show alert messages enter the character
Actual Output	Shown alert messages enter the character
Test Result	Success

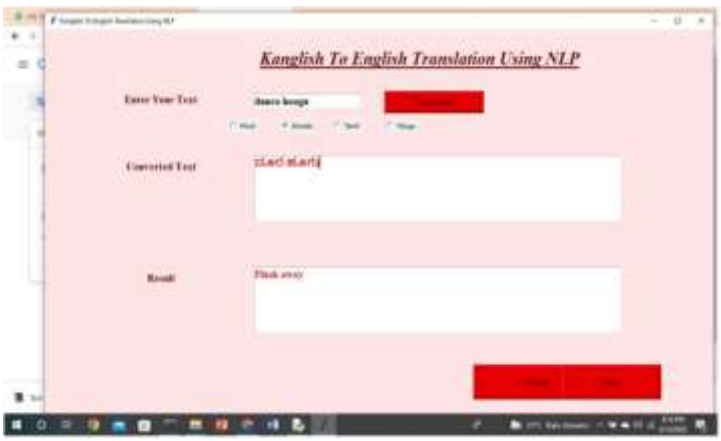
Test Case 3	UTC03
Test Name	Transcribe
Test Description	Transcribing the kanglish to Kannada
Input	Kanglish text
Expected Output	It Should convert to Kannada
Actual Output	It showed Kannada word
Test Result	Success

TABLE: TEST CASES

THE EXPECTED OUTCOME OF THE PROJECT:

Once we input the Kanglish sentence on the website through a keyboard, the application translates the sentence into pure Kannada form and its literal and meaningful English translation

ACTUAL OUTCOMES OF THE PROJECT:



V CONCLUSION

In this project, we have examined the Kannada-English code-mixed text to obtain machine translation for the mixed texts. We have reviewed different constraints on code-mixing that have been proposed in the literature on codemixing/-switching. However, we have not gone into detail about the theoretical implications of the different constraints. We have examined them from the point of view of their usefulness in identifying/detecting the categorical status of the English words in predominantly Kannada texts. Based on the discussion of the patterns of occurrence of the different English words in the Kannada texts, we have devised strategies for their identification and translation to pure Hindi form and pure English form. As the mixing of Kannada and English usually takes place in verbal communication and online chatting on the internet, such a mixed corpus is not available for elaborate testing. The testing has been performed by recording transliterated versions of narrations from a few subjects using existing components of Kannada to English and English to Kannada translation systems.

VI. REFERENCES

1. Mikel Artetxe, Gorka Labaka, and Eneko Agirre. 2018. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 789–798.

2. Suman Banerjee, Nikita Moghe, Siddhartha Arora, and Mitesh M Khapra. 2018. A dataset for building code-mixed goal-oriented conversation systems. In Proceedings of the 27th International Conference on Computational Linguistics, pages 3766–3780.

3. Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching

4. word vectors with subword information. Transactions of the Association for Computational Linguistics, 5:135–146.

5. Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. Information Sciences, 509:257–289.

6. George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In Proceedings of the second international conference on Human Language Technology Research, pages 138–145.

7. Yingying Gao, Junlan Feng, Ying Liu, Leijing Hou, Xin Pan, and Yong Ma. 2019. Code-switching sentence generation by Bert and generative adversarial networks. In Proceedings of the 20th Annual Conference of the International Speech Communication Association, INTERSPEECH 2019, pages 3525–3529.

8. Deepak Gupta, Asif Ekbal, and Pushpak Bhattacharyya. 2020. A semi-supervised approach to generate the code-mixed text using the pre-trained encoder and transfer learning. In Findings of the Association for Computational