International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Comprehensive Analysis of Sanskrit Text Lexical Processing

Golla Saikumar[1], K.M. Ramesh[2], M. Mary Sujatha[3]

[1,2] MSc Students, Department of Computer Science, National Sanskrit University Tirupati.
[3]Assistant Professor, Department of Computer Science, National Sanskrit University Tirupati.

Abstract:
Sanskrit is one of the oldest languages in human civilization, holds immense historical, literary, and philosophical significance. However, due to its complexity and limited accessibility in the modern digital landscape, there is a growing need for tools that facilitate its understanding and utilization. This article aims to bridge this gap by developing a comprehensive Sanskrit text summarization and translation system using Natural Language Processing (NLP) techniques. The system follows a structured three-phase approach to process Sanskrit text efficiently. Initially, the input Sanskrit text is translated into English text using the Google Translate API, ensuring that the content is accessible to a broader audience. Once translated, the English text undergoes an advanced summarization process utilizing the Latent Semantic Analysis (LSA) algorithm from the Sumy library, which extracts the most relevant sentences while retaining the original meaning. This summarized content is then translated back into Sanskrit, providing a concise and refined version of the original text while preserving its essence. The entire process is seamlessly integrated into a user-friendly graphical interface developed using Tkinter, which allows users to input, process, and retrieve results efficiently. The system leverages asynchronous programming to enhance performance, ensuring that translations and summarizations are executed swiftly. By incorporating state-of-the-art NLP methodologies, it significantly contributes to the field of computational linguistics, particularly in Sanskrit language processing. The proposed methodology is not only aids researchers, students, and linguists in analysing Sanskrit texts but also serves as a foundational tool for future advancements in automated Sanskrit text comprehension. The proposed ideology has the potential to facilitate the preservation and dissemination of Sanskrit literature by making it more accessible in the digital era, thereby fostering continued interest and scholarly exploration in the domain of classical languages.

Keywords: Natural Language Processing (NLP) Machine Translation, Google Translate API, Latent Semantic Analysis (LSA), Singular Value Decomposition (SVD).

1. INTRODUCTION

Sanskrit is an ancient Indo-Aryan language, is regarded as one of the most significant classical languages in human history. It serves as the foundation for a vast collection of literary, religious, and philosophical texts that have shaped the intellectual and cultural heritage of India and the world. Despite its historical importance, the use of Sanskrit in contemporary studies and digital applications remains limited due to the complexity of its grammar, vocabulary, and script. Unlike modern languages that have been extensively digitized, Sanskrit lacks robust computational tools for efficient processing,

summarization, and translation. Traditional methods of studying Sanskrit texts require significant linguistic expertise and manual effort, making the comprehension and dissemination of knowledge a challenging task. In recent years, advancements in Natural Language Processing (NLP) and Artificial Intelligence (AI) have made it possible to automate text translation, summarization, and analysis. Leveraging these technologies, this project focuses on the development of a Sanskrit text summarization and translation system that enables seamless processing of Sanskrit content. The system follows a structured pipeline, beginning with the translation of Sanskrit text into English using the Google Translate API.[1] Once translated, the English content undergoes a summarization process using the Latent Semantic Analysis (LSA) algorithm, which identifies and extracts the most meaningful sentences while maintaining coherence. The summarized English text is then translated back into Sanskrit, ensuring that the final output retains the core message of the original content in a more concise form.

To provide a user-friendly experience, this system is implemented with a graphical user interface (GUI) using Tkinter, allowing users to input Sanskrit text and receive processed output efficiently.[2] The application also incorporates asynchronous programming techniques to enhance performance and responsiveness, ensuring smooth execution of translation and summarization tasks. By integrating modern NLP methodologies, this project addresses a critical gap in Sanskrit language processing and provides an effective tool for scholars, researchers, and students who seek to engage with Sanskrit literature in a structured and efficient manner [3]. The significance of this project extends beyond simple translation and summarization. It contributes to the preservation and digital accessibility of Sanskrit texts, making ancient manuscripts and scriptures more understandable to a global audience. Additionally, it fosters further research in computational linguistics, particularly in the domain of underrepresented languages like Sanskrit.[4] By automating key aspects of text processing, this initiative not only promotes the study of Sanskrit but also opens doors for future enhancements in machine translation, text analytics, and artificial intelligence-driven language models.

2. Methodology

This project follows a structured methodology that integrates Natural Language Processing (NLP), machine translation, and text summarization techniques to process Sanskrit text efficiently. The system operates in a sequential manner, ensuring accuracy and coherence throughout the translation and summarization process. Initially, the user inputs Sanskrit text into a Graphical User Interface (GUI) built using Tkinter[5]. The system then translates this text into English using the Google Translate API, leveraging asynchronous processing to enhance speed and performance. Once translated, the Latent Semantic Analysis (LSA) method from the Sumy library is applied to generate a concise summary of the English text while retaining its key information. After summarization, the system retranslates the summarized English text back into Sanskrit using the Google Translate API, ensuring that the final output remains in the original language but in a more condensed and meaningful form. The results, including the original Sanskrit text, translated English text, summarized English content, and the final summarized Sanskrit text, are displayed in the GUI for user convenience. Asynchronous programming techniques are employed to optimize performance and responsiveness. The integration of NLP techniques, machine translation, and text summarization makes this system an effective tool for computational Sanskrit language.[6]

The system facilitates research and digital preservation of Sanskrit texts through a user-friendly Graphical User Interface (GUI) built with Tkinter, allowing users to input Sanskrit text, including long

passages via a scrolling text field. It employs the Google Translate API for Sanskrit-to-English translation, ensuring asynchronous execution for better performance[7]. The translated English text undergoes summarization using the Latent Semantic Analysis (LSA) technique from the Sumy library, which extracts key sentences to retain the core meaning, generating a concise two-sentence summary. The summarized English text is then translated back into Sanskrit using the Google Translate API, ensuring a more meaningful and concise final output. The GUI displays all stages of processing, including the original Sanskrit text, translated English text, summarized English text, and final summarized Sanskrit text, with clear formatting for differentiation[8]. To improve efficiency, the system leverages asynchronous programming using the asyncio library, allowing real-time responsiveness. Built entirely in Python, the system integrates Google trans for translation, Sumy for summarization, and Tkinter for the GUI**,** ensuring a lightweight and efficient desktop application.[9]

Latent Semantic Analysis (LSA) Algorithm for Text Summarization

Latent Semantic Analysis (LSA) is an unsupervised machine learning algorithm used for extractive text summarization. It identifies the most significant sentences in a document based on word co-occurrence patterns and semantic relationships [10]. The core idea is to reduce the dimensionality of a term-document matrix using Singular Value Decomposition (SVD) to identify the most important concepts in the text.

Importance of SVD in LSA:

Given a term-sentence matrix **A**, the SVD decomposition is performed as:

A=U·S·VTA = U \cdot S \cdot V^TA=U·S·VT

- **U:** Contains word-topic relationships.
- **S:** Contains singular values representing the importance of each topic.
- **V^T:** Contains sentence-topic relationships.

By selecting the top k singular values, LSA extracts the most meaningful sentences representing the core meaning of the text.

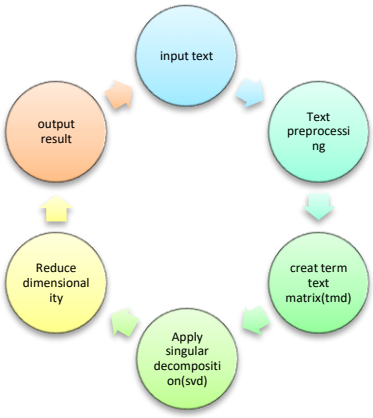

Fig.1 Latent Semantic Analysis (LSA) Process

Figure1 explores Latent Semantic Analysis (LSA) is a dimensionality reduction technique used in Natural Language Processing (NLP) to identify hidden relationships between words in a document. It applies Singular Value Decomposition (SVD) to transform a high-dimensional term-document matrix into a lower-dimensional representation. The process begins with text pre-processing, where the text is tokenized, stop words are removed, and words are lemmatized or stemmed to their root forms. The text is then converted into a Term-Document Matrix (TDM), where rows represent unique words, columns

represent documents, and the values indicate the frequency of words (TF) or their weighted frequency (TF-IDF)

LSA Summarization Algorithm

Step 1: Start
Step 2: Pre-process the Text
Step 3: Create Term-Sentence Matrix
Step 4: Apply Singular Value Decomposition (SVD)
Step 5: Rank Sentences by Importance
Step 6: Select Top Sentences for Summary
Step 7: Return the summarized text
Step 8: End

In the following grounds Fig.2 explains the process of translating Sanskrit text to English, summarizing it in English, and then translating the summary back into Sanskrit involves multiple Natural Language Processing (NLP) techniques. Initially, the Sanskrit text undergoes pre-processing, which includes tokenization, stop word removal, and morphological analysis to identify root words[11]. This ensures a clean and structured input for translation. The next step is translating Sanskrit to English**,** which can be achieved using APIs like Google Translate or Neural Machine Translation (NMT) models trained on parallel Sanskrit-English corpora. Once translated, the English text is summarized using either extractive or abstractive summarization techniques.

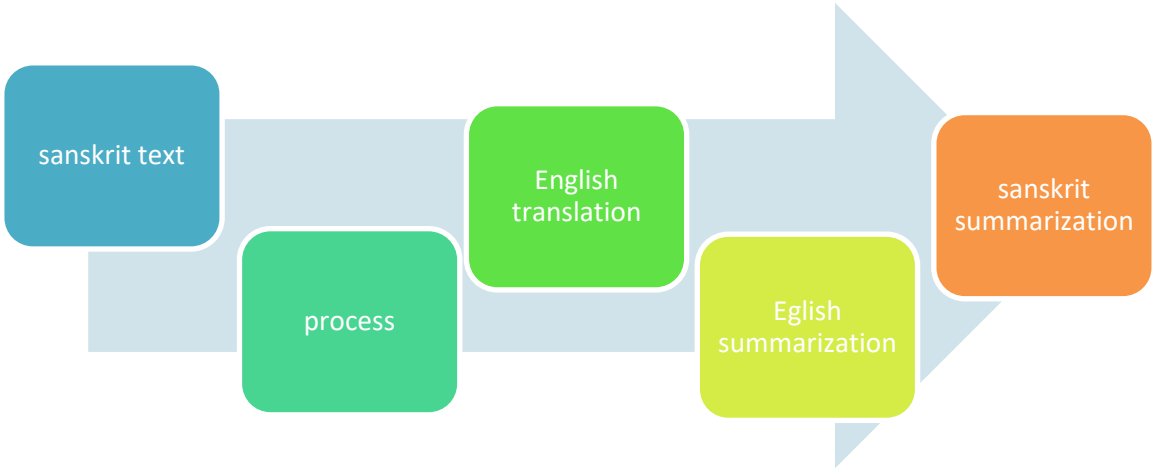

Fig.2 Process Summarization

Fig.3 flow chat explains the user inputs Sanskrit text, which is translated into English using the Google Translate API**.** The translated English text is then summarized using NLP techniques (like NLTK, SpaCy, or Hugging Face). The summarized English text is translated back into Sanskrit using the same API, and the final output is displayed in the GUI.

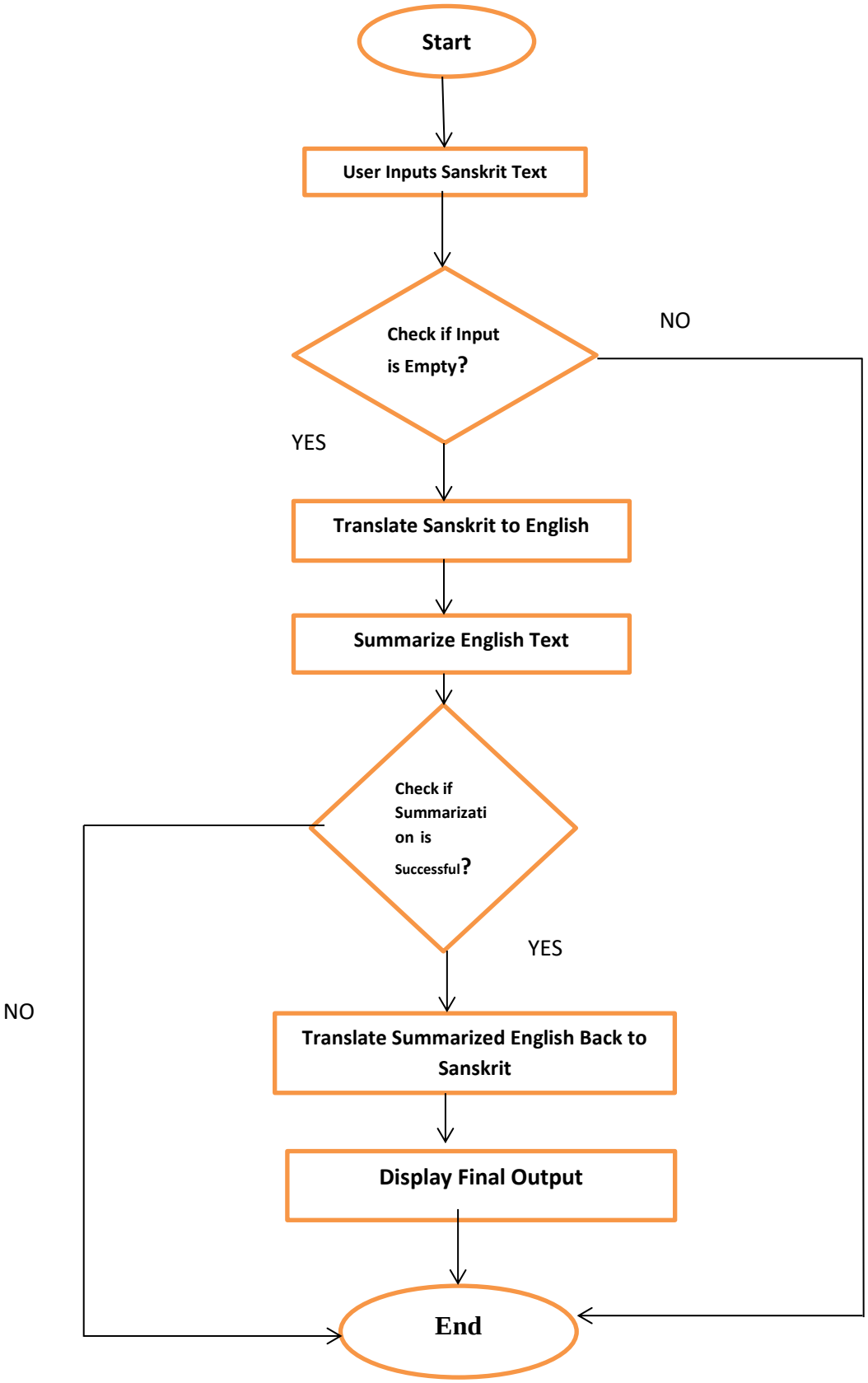

Fig.3 Flow of summarization

Here's the Python pseudocode for implementing Latent Semantic Analysis (LSA) summarization using the Sumy library:

```
from sumy.parsers.plaintext import PlaintextParser
from sumy.nlp.tokenizers import Tokenizer
from sumy.summarizers.lsa import LsaSummarizer
def summarize_text(text, num_sentences=2):
"""Summarizes input text using LSA algorithm."""
parser = PlaintextParser.from_string(text, Tokenizer("english"))  # Parse text
summarizer = LsaSummarizer()  # Initialize LSA Summarizer
summary = summarizer(parser.document, num_sentences)  # Generate summary
return " ".join(str(sentence) for sentence in summary)  # Return summarized text
# Example Usage
text = """Sanskrit is an ancient language of India. It is known for its rich literary tradition.
Many Hindu scriptures, including the Vedas and Upanishads, are written in Sanskrit.Despite its historical significance, the language is not widely spoken today.
However, computational linguistics is making Sanskrit more accessible through NLP techniques.
summary = summarize_text(text, num_sentences=2)
print("Summarized Text:", summary)
```

Drawbacks of the Sanskrit Text Summarization and Translation System

Dependence on Google Translate API – The system relies on the Google Translate API for translation, which may not always provide accurate or contextually appropriate Sanskrit translations due to limited training data for Sanskrit.

Loss of Context in Summarization – The Latent Semantic Analysis (LSA) method focuses on statistical relationships between words rather than true linguistic meaning, which can sometimes lead to loss of important context in the summarized output.

Future Scope

Integration of AI-driven Sanskrit translation models for better accuracy, Incorporation of deep learning techniques for improved summarization, Offline functionality using local NLP models, enhancing grammar rules and context understanding for more accurate Sanskrit processing.

Conclusion

The Sanskrit Text Summarization and Translation System successfully integrates Natural Language Processing (NLP), machine translation, and text summarization techniques to enhance the accessibility and comprehension of Sanskrit texts. By leveraging the Google Translate API for translation and the Latent Semantic Analysis (LSA) algorithm for summarization, ensures an efficient and automated workflow for processing Sanskrit texts. The user-friendly Tkinter-based GUI simplifies interaction, allowing users to input Sanskrit text and receive a summarized version with minimal effort. Additionally, the implementation of asynchronous processing improves the system's performance, making it responsive and efficient. It contributes to the digital preservation of Sanskrit literature, aiding researchers, students, and language enthusiasts in analysing Sanskrit texts more effectively. By automating text translation, summarization, and retranslation, it addresses key challenges in Sanskrit

language processing. Future enhancements could include improving translation accuracy with AI-driven models**,** supporting additional NLP techniques, and expanding the system to process larger texts more effectively**.** Overall, the given article serves as a foundational step toward making Sanskrit more accessible in the modern digital landscape, bridging the gap between ancient knowledge and contemporary technology.

References:

1. Google Translate API Documentation – Used for Sanskrit-to-English and English-to-Sanskrit translation.Google Cloud Translation API
2. Sumy Library for Text Summarization – Used for extractive text summarization with the Latent Semantic Analysis (LSA) algorithm. https://github.com/miso-belica/sumy
3. D., & Martin, J. H. (2021). *Speech and Language Processing* Natural Language Processing (NLP) Techniques – Techniques used in text summarization and translation. Jurafsky,
4. Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990) Latent Semantic Analysis (LSA) in NLP – Mathematical foundations of LSA for summarization.. "Indexing by Latent Semantic Analysis." Journal of the American Society for Information Science, 41(6), 391-407.
5. Tkinter GUI for Python Applications – Used for building the graphical user interface.*Python,Documentation,Tkinter* https://docs.python.org/3/library/tkinter.html
6. Mary Sujatha "A Study On Artificial Intelligence behind Natural Language Processing" published in 2021 IJCRT | Volume 9, Issue 1 January 2021 | ISSN: 2320-2882 https://ijcrt.org/papers/IJCRT2101141.pdf
7. Asynchronous Processing in Python – Used for handling translation and summarization efficiently. https://docs.python.org/3/library/asyncio.html
8. Chiranjeevi Joshi, Balaji K, Sai Saketh B,Abhishek R,Dr.C N Shariff " Summarization and Translation Using NLP" DOI Link: https://doi.org/10.22214/ijraset.2024.61391
9. Model Awais ShaikhSheth L.U.J and Sir M.V. College "Sanskrit to English Translation: A Comprehensive Survey and Implementation using Transformer Based", MumbaiUniversity, Mumbai, Maharashtra,India https://cspubijcisim.org/index.php/ijcisim/article/view/537
10. ShrabantiMandal, GirishKumar Singh "LSA Based Text Summarization"
11. https://www.ijrte.org/wp-content/uploads/papers/v9i2/B3288079220.pdf
12. 11. PranaliBombale , AditiAuti ,AmrutaBandewar , Prof. Vina M. Lomte "SANSKRIT-ENGLISH TRANSLATOR WITH NLP TECHNIQUE" https://www.jetir.org/papers/JETIR1905F17.pdf