Exploring the Application of Sanskrit in Computer Programming

Madhav Moole[1], Flavia Gonsalves[2]

[1]Student, Institute of Computer Science, Mumbai Educational Trust – MET ICS, Mumbai, India
Email: *mca22_1331ics[at]met.edu*

[2]Professor, Institute of Computer Science, Mumbai Educational Trust – MET ICS, Mumbai, India
Email: *flaviag_ics[at]met.edu*

Abstract: ***The integration of ancient linguistic systems, notably Sanskrit, into modern computer programming has emerged as a promising avenue to enhance the efficiency and expressiveness of code. This manuscript explores the application of Sanskrit in computer programming, delving into its unique linguistic features and their potential benefits for software development. Sanskrit, renowned for its rich morphology and precise grammatical structure, offers a natural framework for expressing complex algorithms and data structures. By leveraging Sanskrit's linguistic principles, programmers can develop code that is not only syntactically elegant but also inherently more understandable and maintainable. Furthermore, the inherent modularity and flexibility of Sanskrit grammar lend themselves well to the principles of modern software engineering, facilitating the creation of reusable components and scalable architectures.***

Keywords: Sanskrit, Language, Logic, Computer Programming, Computer, Programming language

1. Introduction

Programming languages serve as notation systems for writing computer programs, defined by syntax and semantics within a formal language framework. They encompass features such as type systems, variables, and error handling mechanisms. Through compilers or interpreters, programming language implementations enable program execution, either directly or by generating executables. The design of programming languages has been significantly influenced by computer architecture, particularly imperative languages optimized for von Neumann architecture. Over time, languages have evolved to abstract away hardware details for simplicity. Thousands of programming languages, categorized as imperative, functional, logic, or object-oriented, cater to diverse application domains.

In the ongoing quest for more efficient and expressive languages, an unconventional source of linguistic richness emerges: Sanskrit. Sanskrit, an Indo-European language with a rich textual tradition, is renowned for its intricate grammar, precise semantics, and expressive capabilities. With roots dating back to ancient India, Sanskrit's linguistic sophistication predates the earliest known texts, such as the Rigveda, possibly evolving from earlier dialects. Scholars and linguists have intriguingly drawn parallels between Sanskrit and modern programming languages, suggesting potential insights for future language development.

This paper aims to explore the potential integration of Sanskrit's linguistic structures into modern computer programming. Despite their disparate origins and purposes, Sanskrit and programming languages share fundamental principles of syntax, grammar, and structure. Through a systematic analysis of Sanskrit's grammar and its compatibility with computational concepts, this study endeavors to construct a bridge between ancient wisdom and contemporary programming paradigms. By embracing Sanskrit's structural elegance, the paper envisions a future where programming transcends mere technicality, becoming a creative and intellectually enriching pursuit rooted in humanity's linguistic heritage.

The research presented here draws from various scholarly works [1-4] that have investigated the intersections of Sanskrit grammar with computational concepts. This study aims to achieve multiple objectives, including analyzing Sanskrit's linguistic characteristics, identifying parallels with programming languages, and developing a framework for integrating Sanskrit concepts into programming practices.

1) Sanskrit in Computer Programming

Sanskrit has a rich history and was used for early Indian mathematics and science. The grammar of Sanskrit is rule-bound, formula-bound, and logical, which makes it highly appropriate to write algorithms. The grammar also makes Sanskrit suitable for machine learning and even artificial intelligence. The integration of Sanskrit into computer programming is a burgeoning field that holds promise for enhancing code readability, efficiency, and cultural preservation. Existing literature explores various methodologies and tools aimed at leveraging the linguistic richness of Sanskrit for programming purposes [5].

"Sanskrit Programming Language" strives to create programming paradigms grounded in Sanskrit morphology and syntax. The first question that arises is why anyone would use Sanskrit for programming when there are multiple well established programming languages such as Java, C++, Python, etc. that work well with all types of application development.

The characteristics of Sanskrit that attracted the attention of computer scientists are:

- Well-knit syntactic and semantic structure of Sanskrit.
- Positional independence of words in a sentence.
- Low phonetic transcription for audio input and binary

phonetic classification (0,1) of poetic meter.
- Well defined rules of grammar, phrasing and synthesis.

There is no need for a particular sentence structure for Sanskrit. Like, In English: - Subject +Verb + Object
Ex:- I am writing an answer.
But in Sanskrit there is no need for a particular structure. ● अहं उत्तरम ्लिखामि (I am writing an Answer.).
- लिखामि अहं उत्तरम ्(I am writing an Answer.).
- अहंलिखामि उत्तरम ्(I am writing an Answer.).

Magha was a great Sanskrit Poet and Author. He was an expert in writing a whole Sloka with one-two-three-four consonants.
Here is just an example from his book Shishupala Vadha:- In 144th stanza, he writes whole sloka with only one consonant.
● दाददो दद्ुददद्ुदादी दाददो ददू ीददोः ।
● दद्ुदादं दददेदद्ुदेदादाददददोऽददः ॥

(Translation:- Sri Krishna, the giver of every boon, the scourge of the evil-minded, the purifier, the one whose arms can annihilate the wicked who cause suffering to others, shot his pain-causing arrow at the enemy.)
Also, he was an expert in writing palindromes.

He writes in 44th stanza:-
- वारणागगभीरा सा साराभीगगणारवा ।
- कारितारिवधा सेना नासेधा वारितारिका ॥

(Translation: It is very difficult to face this army which is endowed with elephants as big as mountains. This is a very great army and the shouting of frightened people is heard. It has slain its enemies.)

2) Sanskrit in Natural Language Processing (NLP)
Natural language processing (NLP) is a crucial component of Artificial Intelligence, enabling computer programs to comprehend and interact with human languages. It encompasses the intersection of computer science, artificial intelligence, and computational linguistics, focusing on programming computers to effectively process extensive natural language datasets [6].

In the realm of Artificial Intelligence, the comprehension of natural language by machines is paramount. However, human languages, such as English, are often fraught with ambiguities, rendering them challenging for NLP tasks. Nevertheless, as AI permeates various domains, NLP emerges as a key subfield, dedicated to enabling computers to understand and process human language, whether spoken or written.

Natural language processing encompasses two primary facets: natural language understanding (NLU) and natural language generation (NLG). NLU involves converting input in natural language into a meaningful representation by analyzing different linguistic aspects. Conversely, NLG encompasses tasks such as text planning, which involves retrieving relevant content, and sentence planning, which focuses on forming coherent phrases (citation [6]).

NLP enhances human-computer interaction, facilitating more efficient integration of AI systems into contemporary applications. Examples include:
- NLP systems for visually impaired individuals to interact with computers via speech input.
- Assistive devices like Stephen Hawking's chair, which converts text into speech.
- Translation programs facilitating communication across different human languages.
- Grammar checking software to identify and rectify errors in text.

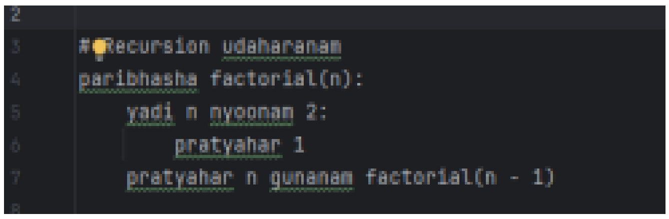
Sanskrit stands out among languages due to its precisely defined grammar. Learning Sanskrit begins with understanding fundamental rules, unlike other languages acquired through continuous communication. The "Ashtadhyayi," a grammatical treatise by ancient scholar Maharshi Panini, is revered for its structured approach, akin to a coded language. Sanskrit's rich declension of nouns, organized into eight predefined cases (vibhaktis), facilitates unambiguous sentence construction and aligns well with semantic net models used in AI systems.

The eight cases of vibhaktis serve specific grammatical functions:
- Nominative: Identifying the subject of a sentence.
- Accusative: Denoting the object of an action.
- Instrumental: Indicating the instrument used in an action.
- Dative: Signifying the recipient of an action.
- Ablative: Expressing the point of separation.
- Genitive: Denoting possession.
- Locative: Indicating the location of an action.
- Vocative: Addressing a person or object directly.

Sanskrit's grammatical structure, with its vibhaktis, mirrors the class-object paradigm of programming languages. The language's segmentation of sentences aligns closely with semantic net models employed in AI systems, facilitating efficient natural language processing.

2. Methodology Adopted

In the present study, the methodology was adopted by reviewing existing technologies, encompassing published research and implementing an interpreter to verify it. For this, the Interpreter was implemented by designing a set of instructions in Sanskrit language and then converting the instructions into machine language using Python. The format of designed sanskrit interpreter is given below:

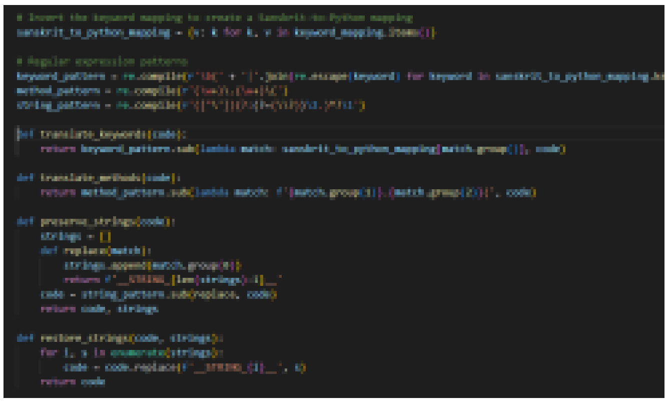


```
#Recursion udaharanam
paribhasha factorial(n):
    yadi n nyoonam 2:
        pratyahar 1
    pratyahar n gunanam factorial(n - 1)
```

Figure 1: A structure created for a sanskrit interpreter

Fig (1) shows a structure created for a sanskrit interpreter , the main takeaway is that there are still a set of instruction defined specifically for this language so that the interpreter can convert it into machine language which would understand those instructions and provide output, but given that sanskrit is already a well structured language with its own well-defined ruleset, this is a scenario that should not have occured, machines can understand a every instruction in the form of large arrays of binaries (0, 1) however in order to use sanskrit as a computer language a different machine must be built from the ground up that would accept it's rules and follow them accordingly.

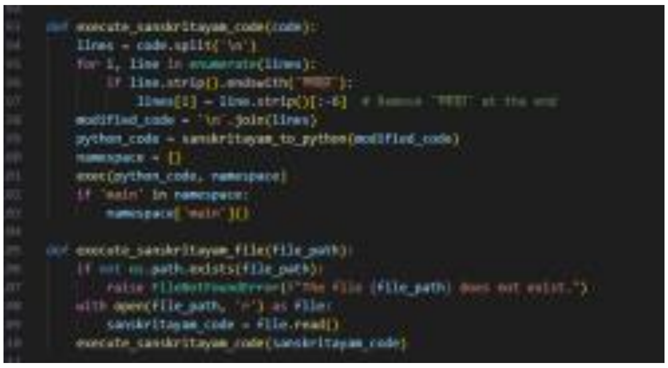
The structure of reading and taking inputs from the instructions is given in Fig 2.



```
# Invert the keyword mapping to create a Sanskrit-to-Python mapping
sanskrit_to_python_mapping = {v: k for k, v in keyword_mapping.items()}

# Regular expression patterns
keyword_pattern = re.compile(r'\b(' + '|'.join(re.escape(keyword) for keyword in sanskrit_to_python_mapping.ke
method_pattern = re.compile(r'(\w+)\.(\w+)\(')
string_pattern = re.compile(r'(["\'])((?:(?!\1).)*?)\1')

def translate_keywords(code):
    return keyword_pattern.sub(lambda match: sanskrit_to_python_mapping[match.group()], code)

def translate_methods(code):
    return method_pattern.sub(lambda match: f'{match.group(1)}.{match.group(2)}(', code)

def preserve_strings(code):
    strings = []
    def replace(match):
        strings.append(match.group(0))
        return f'__STRING_{len(strings)-1}__'
    code = string_pattern.sub(replace, code)
    return code, strings

def restore_strings(code, strings):
    for i, s in enumerate(strings):
        code = code.replace(f'__STRING_{i}__', s)
    return code
```

Figure 2: Structure for reading the inputs

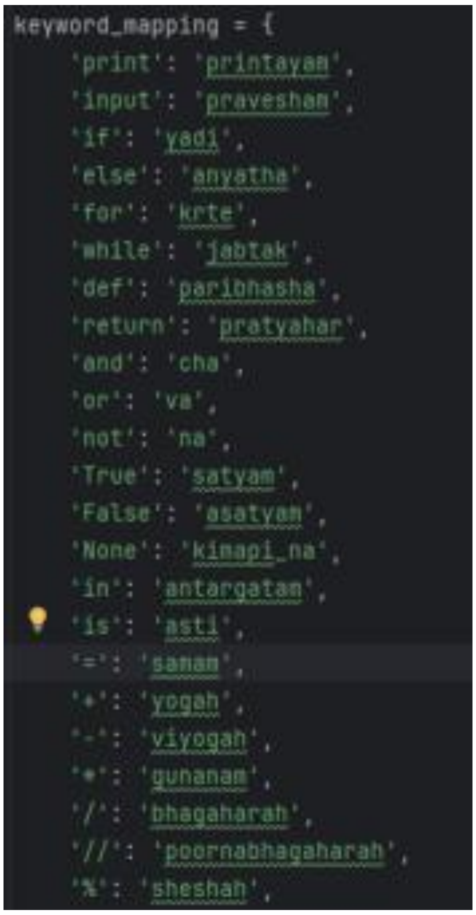
Fig (2) shows the structure for converting these defined instructions into machine code by using the mapping which contains the basic syntax defined for the programming language the meaning for each command used is provided. The execution of all the imputed commands takes place as shown in Fig 3.



```
def execute_sanskritayam_code(code):
    lines = code.split('\n')
    for i, line in enumerate(lines):
        if line.strip().endswith('[illegible]'):
            lines[i] = line.strip()[:-6]  # Remove '[illegible]' at the end
    modified_code = '\n'.join(lines)
    python_code = sanskritayam_to_python(modified_code)
    namespace = {}
    exec(python_code, namespace)
    if 'main' in namespace:
        namespace['main']()

def execute_sanskritayam_file(file_path):
    if not os.path.exists(file_path):
        raise FileNotFoundError(f"The file {file_path} does not exist.")
    with open(file_path, 'r') as file:
        sanskritayam_code = file.read()
    execute_sanskritayam_code(sanskritayam_code)
```

Figure 3: Execution process of each command

Fig (3) shows the execution process of each command by storing it into maps and later executing them based on given conditions in the example as provided in Fig (1), a simple factorial function has been defined and later the result has been printed at the end. The commands used for the same are shown in Fig (4).



```
keyword_mapping = {
    'print': 'printayam',
    'input': 'praveshan',
    'if': 'yadi',
    'else': 'anyatha',
    'for': 'krte',
    'while': 'jabtak',
    'def': 'paribhasha',
    'return': 'pratyahar',
    'and': 'cha',
    'or': 'va',
    'not': 'na',
    'True': 'satyam',
    'False': 'asatyam',
    'None': 'kimapi_na',
    'in': 'antargatam',
    'is': 'asti',
    '=': 'samam',
    '+': 'yogah',
    '-': 'viyogah',
    '*': 'gunanam',
    '/': 'bhagaharah',
    '//': 'poornabhagaharah',
    '%': 'sheshah',
```

Figure 4: Identifiers for each command

1) Challenges for Sanskrit Computer Language Conversion

Challenges in natural language processing frequently involve natural language understanding, natural language generation (frequently from formal, machine-readable logical forms), connecting language and machine perception, dialog systems, or some combination thereof. In Natural Language Processing, Knowledge Representation is the primary step to make machine understandable Natural Language. This is possible with Semantic Nets.

Ambiguity of any natural language which means that the words which form a sentence in a natural language can behold multiple meanings. It should be able to convert the language in such a way that it reduces the ambiguousness, making it more literal so that robots with artificial intelligence could understand it is a challenge. The accuracy of linguistic analysis is a very important factor, but it is difficult for AI [7, 8].

While using Sanskrit as a programming language, there are both possibilities and difficulties. While Sanskrit has the potential to be an extremely powerful and efficient programming language, there are also significant challenges that need to be overcome. One of the biggest difficulties of using Sanskrit as a programming language is its complexity. Sanskrit is an incredibly rich and nuanced language, with a vast array of different words and concepts. This can make it very difficult to create concise and unambiguous code. There is also the challenge of learning Sanskrit in the first place – it is not an easy language to pick up, especially for those who are not already familiar with it. Another difficulty is the lack of resources and support for Sanskrit as a programming language. While there are some individuals and organizations working on developing Sanskrit-based software, there is still relatively little available in terms of tutorials, documentation, or other learning materials. This can make it difficult for people who want to use Sanskrit for programming to get started.

Despite these challenges, however, there are also many reasons why Sanskrit could be an excellent choice for a programming language. Its rich vocabulary and complex grammar offer a great deal of expressive power, making it well suited for tasks such as natural language processing or data mining. Additionally, its unique history and evolution means that it has the potential to be more resistant to hacking and other security threats than other languages. If more resources were available to support its development, Sanskrit could become a very powerful tool for programmers around.

2) Reasons for Sanskrit to be Used in AI

It's essential to recognize that while Sanskrit is valuable for AI, it's not the exclusive focus of research. AI researchers engage with a variety of languages and linguistic datasets to develop inclusive AI systems for a global audience.
The significance of studying Sanskrit for AI stems from several factors:

- Linguistic Roots: Sanskrit, as an ancient Indo-European language, serves as a foundation for many modern languages. Exploring Sanskrit can offer insights into shared linguistic structures, aiding natural language processing (NLP) tasks in AI.
- Richness and Expressiveness: Sanskrit boasts a vast and nuanced vocabulary, enhancing AI systems' ability to understand context and improve language generation.
- Grammar and Structure: Sanskrit grammar is highly structured, offering valuable insights for developing AI language models capable of handling complex sentence structures and capturing text semantics effectively.
- Language Generation and Translation: Leveraging Sanskrit insights can enhance the quality and diversity of text generation in AI models like Generative Pre-trained Transformers (GPT), resulting in more accurate and contextually relevant AI-generated content.
- Knowledge Preservation: Sanskrit encompasses a wealth of ancient texts spanning philosophy, science, and literature. Sanskrit studies enable researchers to access and preserve this knowledge, facilitating its integration into AI systems for knowledge representation and understanding.
- Cognitive Benefits: Learning Sanskrit can enrich the understanding of language structures and cognitive processes among AI researchers and developers, contributing to the refinement of AI models and algorithms.
- Cross-cultural Understanding: Sanskrit's influence on the cultural and religious heritage of South Asia underscores its importance in promoting cross-cultural understanding and diversity in AI research. Integrating Sanskrit studies into AI endeavors fosters more inclusive and culturally aware AI systems.
- This comprehensive approach ensures that AI systems leverage the diverse linguistic landscape to better serve global communities.

3) Artificial Intelligence, Computers, Sanskrit and Semantic Network

The advent of AI and computers has transformed the exploration of ancient languages such as Sanskrit, thanks to semantic networks. Semantic networks, structured graphs that delineate relationships between language elements, play a pivotal role in unraveling Sanskrit's complexity. By employing AI-driven natural language processing (NLP) algorithms, computers can meticulously analyze extensive Sanskrit texts and construct elaborate semantic networks.

These networks not only decipher individual word meanings but also illuminate the intricate web of contextual associations and underlying linguistic structures within Sanskrit.

Understanding the semantic interconnections between words empowers AI systems to accurately interpret and translate Sanskrit texts, thereby enhancing accessibility on a global scale. Semantic networks in Sanskrit also prove invaluable for language learning and comprehension. AI-powered language learning platforms leverage these networks to offer learners profound insights into word meanings and contextual usage. Moreover, computers can generate interactive educational materials like lessons, quizzes, and exercises based on semantic network insights, elevating the effectiveness of Sanskrit language education.

Furthermore, semantic networks contribute significantly to Sanskrit's preservation and rejuvenation. AI systems facilitate the creation of comprehensive digital repositories housing semantic data, thereby safeguarding ancient Sanskrit texts for posterity. As new Sanskrit content emerges, semantic networks aid in maintaining linguistic coherence and consistency with the language's historical context.

Beyond Sanskrit, delving into its semantic network has far-reaching implications for AI research. It serves as a catalyst for the development of advanced NLP models adept at grasping the subtleties of human languages. Insights gleaned from Sanskrit's semantic network analysis can be transposed to enhance AI systems' language comprehension and generation across diverse linguistic landscapes, benefiting various applications ranging from machine translation to content creation.

4) Sanskrit and Machine Translation Systems

Machine translation stands out as a crucial application within the realm of Natural Language Processing (NLP), holding significance as a branch of Artificial Intelligence (AI). Its utility lies in furnishing individuals with a mechanism capable of comprehending diverse languages spoken worldwide. By facilitating translation between languages, machine translation obviates the need for human intermediaries, thereby aiding cross-cultural communication. This process entails converting text from a source language (SL) into its equivalent in a target language (TL), enabling individuals from different linguistic backgrounds to understand content written in unfamiliar languages.

a) DESIKA

According to Ramanujan, a comprehensive morphological analysis forms the foundation for Sanskrit processing. The

DESIKA system, based on Paninian principles, encompasses Vedic processing and shabda-bodha. It comprises three distinct modules: generation, analysis, and reference. In the generation phase, the user specifies the nominal or verbal class, and applicable rules are subsequently generated. The analysis phase involves syntactic identification and role assignment for each word using Karaka-Vibhakti mappings. Morphological analyzers for Sanskrit have been developed by various groups.

b) Sanskrit Karka Analyzer for Machine Translation

In 2007, Sudhir K Mishra designed a Sanskrit Karka analyzer for translation at JNU, Delhi, employing a Rule-Based MTS (RBMTS) methodology. However, the system faced challenges in resolving ambiguities and handling Sandhi and Samasa in Sanskrit.

c) Constrained Based Parser for Sanskrit Language

Anglabharti (1991) introduced a machine-aided translation system for English to other Indian languages, employing a pseudo-interlingua approach. Anglabharti-II (2004) addressed shortcomings by incorporating a generalized example-base and automated pre-editing capabilities.

d) Etrans System

The Etrans software facilitates the translation of English sentences into Sanskrit. It comprises modules for parsing, morphological analysis, and generation, with a user interface developed using the .NET framework and a lexicon based on MS-Access 2007.

e) Google Translate

In 2007, Franz-Josef Ochs implemented a statistical machine translation approach for Google Translate between English and Sanskrit. While Hindi, Urdu, and Sanskrit are the only Indian languages supported, the system demonstrates good accuracy and robustness, adhering to the grammar of the source language in its output.[10]

3. Conclusion

The specific and unambiguous nature and at the same time, the vast literature and vocabulary of the Sanskrit language provides a gateway for implementing this language in a way that a computer can understand. NASA scientists believe that Sanskrit language can provide a huge impetus to the development of the field of artificial intelligence. The Sanskrit language empowers the AI with more randomness, versatility and uncertainty as compared to the AIs programmed to interact in other languages.

Sanskrit, often called the language of the gods, is finding renewed relevance in Artificial Intelligence and Machine Learning. Its unique linguistic attributes and deep-rooted connection to ancient knowledge make it an invaluable asset for enhancing AI capabilities. It has a bright future in the area of computers as well. It has been identified as a computer-friendly language. It has a lot to offer in the areas of artificial, Robotics and Biomechanics.

In conclusion, although the use of Sanskrit as a programming language is still in its early stages and there are many difficulties that need to be addressed, it holds great potential for improving efficiency and reducing development time. By focusing on making this language more accessible, developers can potentially create powerful applications that combine both Indian tradition and modern technology. As such, investing in the further development of Sanskrit as a programming language may be beneficial not only to those with an interest in Indian culture but also computer professionals looking for additional capabilities within their software design.

References

[1] Rick Brigs,Knowledge Representation in Sanskrit and Artificial Intelligence, AI Magazine Volume 6 Number 1, 1985, 32-39.

[2] Neetesh Vashishtha, Implementation of Sanskrit Linguistics in Artificial Intelligence Programming, International Journal of Advances in Computer and Electronics Engineering Volume: 02 Issue: 02, February 2017, pp. 17 – 27.

[3] Chandana Bathulapalli, Drumil Desai and Manasi Kanhere, Use of Sanskrit for natural language processing ,International Journal of Sanskrit Research 2016; 2(6): 78-81.

[4] Jivnesh Sandhan, Anshul Agarwal, Laxmidhar Behera, Tushar Sandhan and Pawan Goyal, SanskritShala: A Neural Sanskrit NLP Toolkit with Web-Based Interface for Pedagogical and Annotation Purposes, https://github.com/Jivnesh/SanskritShala

[5] Deeptanshu Jha , Dr. Rashmi Jha , Varun Varshney, Natural Language Processing and Sanskrit, International Journal of Computer Engineering and Technology (IJCET), Volume 5, Issue 10, October (2014), pp. 57-63.

[6] Shashank Saxena and Raghav Agrawal, Sanskrit as a Programming Language and Natural Language Processing, Global Journal of Management and Business Studies, Volume 3, Number 10 (2013), pp. 1135-1142.

[7] Inderjeet, An Approach to Sanskrit as Computational and Natural Language Processing, IJCSC, Vol 6 • Number 2 April - Sep 2015 pp. 264-268.

[8] Vipin Mishra, Sanskrit as a Programming Language: Possibilities & Difficulties, - International Journal of Innovative Science, Engineering & Technology, Vol. 2 Issue 4, April 2015. pp-1089-1096.

[9] Dr.Arzoo, Significance of Sanskrit in Artificial Intelligence, -International Journal of Computer Science Trends and Technology (IJCST) – Volume 11 Issue 4, Jul-Aug 2023.

[10] Prof. Deepak Mane, Aniket Hirve, Study of Various Approaches in Machine Translationfor Sanskrit Language, -International Journal of Advancements in Research & Technology, Volume 2, Issue4, April-2013.