International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V73I3P139

Volume 73 Issue 3, 554-560, March 2025

Original Article

Must: Machine Learning Based Unsupervised Multi-Lingual Morpho-Semantic Textual Processor for Natural Languages

Anjali Bohra[1], Nemi Chand Barwar[2]

[1,2]*Department of Computer Science & Engineering, MBM University, Jodhpur, Rajasthan, India.*

[1]*Corresponding Author : anjalivb.phdcse@mbm.ac.in*

Received: 18 July 2024 Revised: 21 January 2025 Accepted: 27 January 2025 Published: 28 March 2025

Abstract *- A word is a continuous sequence of alphabetic characters classified and recognized by unique patterns or rules. Morphological structure suffix (affix) of the word with syntactic and semantic representation. Grammatical information of words is marked through inflectional suffixes. Morphological analysis helps perceive a word's semantic and syntactic properties and can be implemented using morpheme-based, lexeme-based, or word-based approaches. Syntactic and semantic analysis is a classification process for placing words in pre-defined groups. Karakas (case) are the classes specifying the relationship of words in a sentence. The paper performs multi-lingual semantic analysis and implements a morphological processor. The multi-lingual semantic analysis of the Sanskrit and the English language is performed, followed by the generation of an unsupervised learning-based morphological processor for English. Word Embedding based approach is used for comparative analysis of Sanskrit and English languages using datasets prepared through available online textual repositories for both languages. The obtained result serves as a motivation for unsupervised morpho-semantic processors. The proposed PFMP algorithm performs morphological processing to extract the root word of the language with its attributes like number, gender, suffix, and karaka(case). The model is trained using the Keras deep learning framework with 15 nouns, 15 unique suffixes and 255 unique inflections of the English language. With limited data and only 20 epochs, the model obtained 52 percent of recall. The system can be used as a generalized platform for extracting linguistic information for a specific language when trained with language-specific grammatical knowledge.*

Keywords *- Deep learning, Karaka Relations, Morphological processing, Natural language processing, Semantic analysis*.

1. Introduction

Some languages are fixed-order languages like English, and others are free word-order languages like Sanskrit. Syntactic and semantic analysis means the classification of each word is one of the pre-defined groups. Syntactic classification methods are rule-based and probability-based statistical models. Rule-based systems require grammar. Statistical models-based systems require a large corpus to find the various probabilities of tag sequences. Classification tasks require a large corpus of data and grammar of a language for training the system with existing categories, followed by testing the performance of the learned system. The Sanskrit language is one of the ancient languages with well-defined grammatical structures, recursive methods, and defined rules, which makes it the centre of research for computational linguistic groups. Artificial Intelligence supports different knowledge structures to capture various categories of meaningful information. Knowledge structures are declarative, procedural, inferential, inheritable, and common sense knowledge. Case Frame is an event-driven declarative knowledge structure that can include Subframes. Some well-known KR structures are predicate logic, semantic net, frames, conceptual dependency, scripts, etc. The complex handling of the semantic net is due to its graphical structure. The network encodes information in the form of a triplet, which contains action and the relationship of words involved in action with the word itself. Triplets and frames are a form of knowledge representation where frames depict information in a structured way. Interrelated frames are used for the inference process. A case frame-based knowledge representation structure was developed for semantic analysis of the Sanskrit language. Identifying semantic roles in the Sanskrit language using Paninian Theory depicts specific knowledge representation structures. An ancient scholar, Panini, defined six classes called karaka (or case) for semantic classification in Sanskrit, known as semantic roles. These roles resemble case-based semantics in event-driven contexts, where entities such as the agent, object, and location are identified and connected to an event or action within a sentence. The verb of a sentence is generally the action or event of the theme; in other words, a

group of words (chunks) are related to it by identifying roles. The six primary roles are karta, karma, karan, sampradaya, and adhikari, which are understood as the agent, object, instrument, cause, source of departure, and location, respectively. The Panini model conducts semantic analysis of language by determining the relationship between the words in a sentence and the action entity. Extraction of these roles and storing them in Case Frame Structure is the overall objective of the system. Frames are the structural way to represent knowledge consisting of slots with values. Acquiring knowledge of Sanskrit requires a deep understanding of its precise grammatical and morphological framework, which outlines how a word's suffixes and affixes relate to its syntactic and semantic functions within a sentence.

A comprehensive morphology analysis is crucial for grasping a word's semantic and syntactic characteristics [1]. Morphological analysis segments words into morphemes and extracts grammatical information through their suffixes. The morpheme-based approach, lexeme-based approach, and word-based approach, also known as word and paradigm approach, are the main approaches of morphological analysis. The process accepts a token as input and generates morphological information, such as gender, number, class, etc., as output [2]. Linguistics deals with two approaches, morphology, namely analysis and synthesis [3]. The analyzer extracts morphemes from a word and constructs the word from its root and grammatical structure. It is used in various NLP applications, such as machine translation, part-of-speech tagging, spell checking, speech recognition, and lemmatization, among others. [4].

2. Research Gap

The morphological analysis helps in understanding the structure of words in a language. Words play a significant role in language understanding; therefore, if any word occurs that is not specified in the lexicon dictionary, then the language processor marks it as out of vocabulary. Character-based morphological processor is suitable to overcome the problem. The system could handle complex morphological structures by detecting ambiguity in word segmentation.

3. Motivation

Karakas are the grammatical functions that specify relations between nominal and verbal roots. According to Paninian theory, Karak forms a unique class when paired with vibhakti and vachan. It is a feature of the Sanskrit language that the words of Sanskrit with specific suffixes classify a definite karak role. The words of Sanskrit and English languages with their specific suffix classes were provided to the machine learning model. The model obtains the embeddings of specific word classes for different languages. Complete English sentences were analyzed based on Paninian karak theory. The comparative study of English and Sanskrit language shows that similar karak classes of two languages share the same domain in embedding space. Figure 1 shows a clear separation of the karak class for both languages. The same karak, when combined with different vachan and vibhakti, appears as a separate class and occupies a separate place in the embedding space. The figure shows the same embedding space for similar features from different languages (English and Sanskrit).

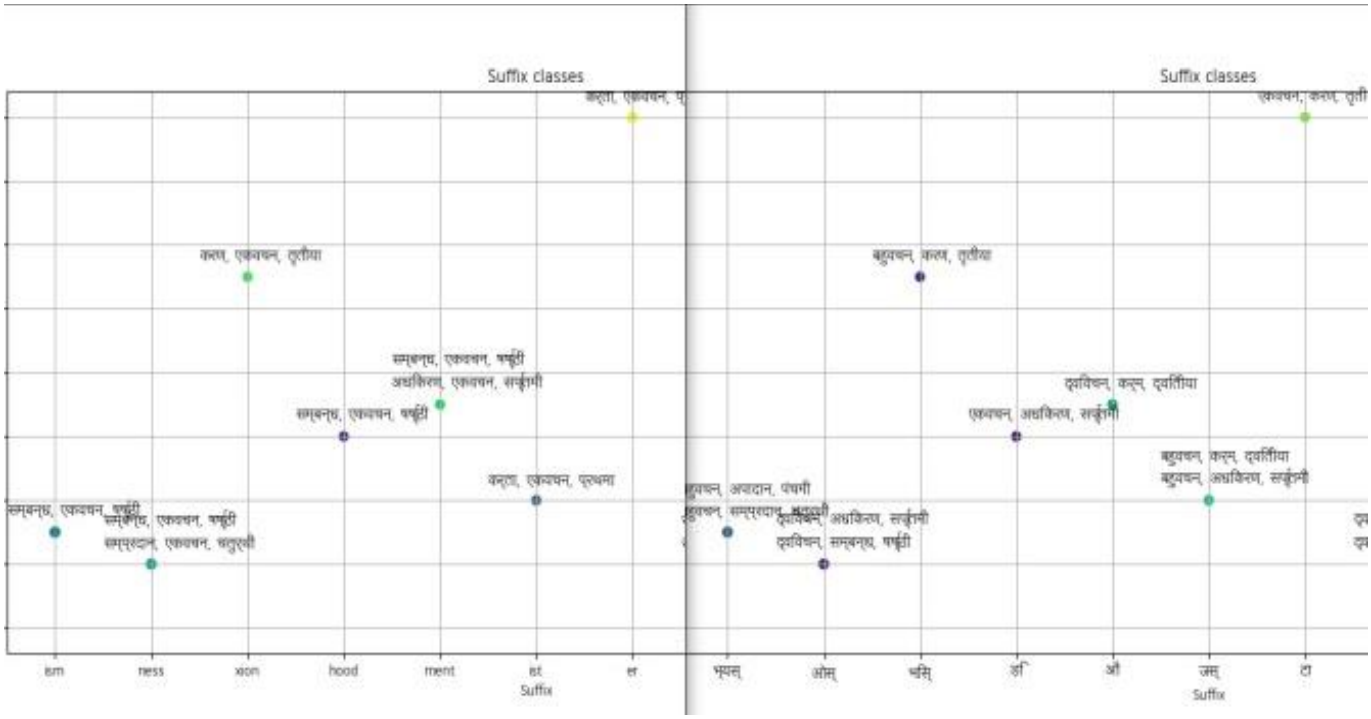

Fig. 1 Embedding space comparison

4. Related Work

Sequence labelling based learning efficiently captures non-linear relationships and other features of natural language [5]. A Morphological analyzer for the Malayalam language using machine learning was developed [6]. The algorithm is based on character-level sequence labelling using RNN, LSTM and GRU deep learning architecture [7]. Multi-Task Deep Morphological analyzer (MT-DMA), a character-level neural morphological analyzer based on multi-task learning of word-level markers for Hindi and Urdu language, was developed which predict a set of six morphological tags for words of Indo-Aryan languages like POS, gender, number, person, case and TAM [8].

The morphological generator utilizes RNN, LSTM, and GRU deep learning architectures to extract nouns and verbs for Malayalam and Tamil languages. A parallel Morphological Analyser for Sanskrit and Malayalam languages using deep learning architectures was designed [9]. Experimentation shows that continuous word embeddings capture multi-degree linguistic similarity with semantics, syntax, or morphology using the Log Bilinear model to predict the morphological tag along with the next word[10]. The two words like working and performing, both share context and 'ing' suffix, therefore character-level features would provide efficient results in meaning extraction. Morphological theory describes words and their potential in a language [11].

Morphological typology describes four categories of languages, namely Isolative languages, Agglutinative languages, polysynthetic languages, and Fusional languages [12]. Isolative languages contain free morphemes, like the Ukrainian language; words of Agglutinative languages are formed through the composition of morphemes, like the Turkish language, and polysynthetic languages contain words with multiple stems. Examples include He is reading; the object book (paper/article) is incorporated within the verb read. In Inflectional languages, words have simpler units, and each unit exhibits a different meaning, like the Latin language [2].

4.1. Types of Morphology

Morphological processes can fall into one of four categories: inflectional, derivational, semi-affixes or combining forms, and cliticization. Inflection involves modifying a word to reflect grammatical attributes such as person, number, tense, gender, case, aspect, and mood. Inflectional morphology combines a word stem with a grammatical morpheme, typically producing a word within the same class, such as 'cat' (cats) or 'play' (played) [13]. Additional grammatical information includes tense, number, person, mood, and aspect. Derivational morphology combines a word stem with a grammatical morpheme, usually creating a word in a different class, such as "compute" (computation, computational).

Table 1. Word formation with suffixes

Word-Formation	Categories
root+suffix	Intensives, causatives
word+suffix	Denominal verbs
root+suffix	Primary (karta) suffixes
word+suffix	Secondary (taddhita) suffixes
word+word	Compounding
root+suffix	Verb inflection
stem+suffix	Noun inflection

Derivational morphology alters the part of speech associated with the root, while inflectional morphology modifies the grammatical form of the stem. Semi affixes are bound morphemes that contribute partial meaning to words, as seen in examples like 'noteworthy', 'antisocial', and 'anticlockwise'. These words draw specific attention from the reader. Stemming involves extracting the stem of a word by removing its affixes. Cliticization refers to the process of attaching a clitic - a reduced form of a word or morpheme to the stem, as in I've or I'd [13]. The following Table 1 shows the seven types of word-formation categories [14].

Morphology divides morphemes into two main categories: stems and affixes, which include prefixes, suffixes, and circumfixes. Inflection is language dependent as the Hindi language has 40 noun inflections, while English has only 7 to 8 inflections for nouns. Nouns, pronouns, and adjectives require gender, number, and case to represent grammatical information, while verb requires only gender, number, and person. Inflected adverbs behave the same as nouns; therefore, no separate paradigms are needed to study their characteristics. The basic unit of written content is a word, a continuous sequence of alphabetic characters classified into word groups recognized by unique patterns or a rule to form a larger word group. One such type of structure is a modifier-modified structure, the head and the modifier. Properties of a headword are inherited by the group and are then modified by the modifier. Words are the lexical categories in which nouns and verbs are the two prominent structures, and adjectives, adverbs, etc., are other categories. The verb denotes an activity (or state), and the noun denotes a participant in the activity (or state), and this structure is known as the karaka relation in paninian grammar.

4.1.1. Approaches to Morphology

Various approaches to morphological analysis include the corpus-based approach, rule-based approach, stemmer-based approach, paradigm-based approach, finite-state automata approach, finite-state transducer-based approach, two-level computational approach, directed acyclic word graph approach, and hybrid approach [3]. Corpus-based methods employ machine learning algorithms along with a large set of annotated words to determine the roots and grammatical structure of an input word [6]. Rule-based systems use specified rules and dictionaries containing root and morphemes to match a root. As the system is dictionary-

dependent, the absence of words in a dictionary leads to complete failure of the system. Stemmer-based approach implements a stemming algorithm to retrieve the stem of a word by reducing inflected and derived words [15]. The paradigm-based approach classifies various word classes like nouns, pronouns, verbs, adverbs, adjectives, pronouns, and prepositions based on inflectional patterns. The root table with all the roots and paradigm numbers is the key component of the morphological analyzer [16]. Finite automata-based systems are 5-tuple systems that include starting and final states, a finite set of states and inputs, and a transition function to identify a word [17]. Finite State Transducers are 6-tuple systems that consist of starting and final states, a finite set of states and inputs, a transition function, and a finite set of output systems to identify a word [17]. The level morphology-based system has two components: one is dictionary-listed roots and their affixes, and the other is finite state automata with a set of rules defined in transition function to recognize a word [18]. A directed acyclic word graph is a data structure representing a string collection and determining whether a given string belongs to a specific set [19]. The hybrid approach uses both paradigm-based approaches with the suffix stripping method [20].

4.1.2. Morphology Word Embedding

An embedding is the process of mapping a high-dimensional vector into a lower-dimensional space. It captures the semantics of the input by placing similar objects closer together in the embedding space. Even different word titles with similar meanings share closeness in embedding vector space. For example, “The squad is ready to win the football match” and “The team is prepared to achieve victory in the soccer game” convey the same meaning but use nearly identical vocabulary. However, these sentences should be positioned near each other in the embedding space due to their similar semantic encoding. [21]. From a mathematical standpoint, an embedding is a mapping from one space or structure to another.

Vector semantics is the common method of representing word meanings in NLP, using a multidimensional semantic space derived from the distribution of neighboring words. These vectors are referred to as embeddings [22]. The vectors are the learned representations that reveal similarities and differences between the words [23]. Vector or distributional models of meaning are based on a co-occurrence matrix, which represents how words appear together in a given context. A vector space is a set of vectors defined by their dimensions. In real term-document matrices, each document is represented by a V-dimensional vector, where V represents the size of the vocabulary [22].

Two classes, namely sparse and dense, are defined for vector semantic models. These sparse models represent words in dimensionality based on vocabulary size related to the co-occurrence count probability function. Dense vectors have dimensionality within the range of 50 to 1000 layers. Word2Vec algorithms like the skip-gram method are popularly used for computing dense embeddings. These algorithms employ a logistic regression classifier to determine the probability of words based on their defined embeddings [22]. These vectors capture both semantic and syntactic information. The semantic dimension encompasses features such as tense (past/present/future), number (singular/plural), gender (masculine/feminine), aspect, and mood associated with the word in the sentence. Word-embedding tools learn word representation, which can be evaluated through specific tasks like POS tagging, semantic analysis, information retrieval, question-answering systems, machine translation, etc. Word embedding models attach a vector for each word in a semantic space [24]. These models learn vectors of words to perform downstream NLP tasks [25, 26].

Encoder decoder or sequence-to-sequence models generate contextual output sequences. The encoder network processes an input sequence and generates a contextualized representation, which is then passed to a decoder to produce a sequence of hidden states of arbitrary length for the corresponding task-specific output. The encoder-decoder architecture can be implemented using RNNs or transformers. LSTMs, GRUs, convolutional networks, and transformers can all serve as encoders. Word representations, i.e. word vectors, reflect poor quality for handling rarely used and out-of-vocabulary words.

Even these models require large training datasets for optimized performance. The best solution for uniform word distribution is to consider the smallest meaning semantic unit, i.e. ‘morpheme’. A morpheme is the smallest meaningful semantic unit composed of a sequence of characters. The word embedding models based on character-level learning can better understand the morphological features and meaning of a word. Character-based word embedding models work for open vocabulary words; therefore, any known or unknown word can be predicted after training.

5. Experimental Results

5.1. Experimental Setup and Tools

The experiment was performed to analyze the suffix-based morphology of Sanskrit and English languages through character embedding of a word. Using Python programming language and keras deep learning framework, a dataset for each word with corresponding vibhakti, suffix, vachan, and karak was prepared. The prepared dataset was classified for training, development, and testing with a corresponding unique vocabulary.

A Keras sequential API model with a dense LSTM layer was compiled to study the suffix-based word characteristics of both languages. In the Keras model, character embeddings are given as input, whereas Karak, vibhakti, vachan, root word, and suffixes are used as output.

5.2. Data Set Preparation

For dataset preparation, multiple words with different suffixes are used for both languages; each word is tagged with its corresponding root, stem, karak, vibhakti and vachan. A complete sentence is analyzed for case frame representation based on the relation of the word with the verb in a sentence. The experiment uses 150 English sentences with a total of 15 nouns with 15 unique suffixes and 225 unique inflections to train the keras model. The noun suffixes were taken from the Cambridge dictionary and are available at 'https://dictionary.cambridge.org/grammar/british-grammar/suffixes'.

6. Methodology

All experiments are conducted using Python 3, along with the Sklearn machine learning toolkit and the Keras and TensorFlow deep learning frameworks. Keras deep learning framework is used to design the model. The model is trained using prepared datasets for defined language using authorized online repositories. The proposed algorithm PFMP explains the Paninian Framework-based Morphological Processor. Lexicon and Morphotactics are the major parts of an algorithm. Lexicon maintains the list of stems and affixes of a language with basic information about them, such as their main categories for parts of speech like nouns, pronouns, adverbs, adjectives, etc. Morphotactics is concerned with ordering morphemes and deals with valid word formation in a language. To scan a word, the embedding of characters used in a word is obtained and stored. The process helps the model in learning the words of a language.

6.1. Algorithm PFPM

- Some nouns are root words in the English language, so for them, fill them as [NA] suffix
- Lexicon module: Read the dataset to create a separate vocabulary for words, Karak, vibhakti, vachan, root word, and suffix, and then label them
- Morphotactics module: With unique word annotations, create character level separations to create a character vocabulary for characters from both languages
- With each vocabulary
  1. Get character-level vectors for each word with 0 padding to max word length.
  2. Get vectors for Karak.
  3. Get vectors for vibhakti
  4. Get vectors for vachan
  5. Get vectors for the root word.
  6. Get vectors for suffixes.

- Create a Keras sequential model.
- Add an Embedding layer with character vocabulary length, number of samples, and maximum word length.
- Add an LSTM layer
- Add a Dropout layer to normalize the LSTM output tensors.
- Add a Desnse layer with output shapes (i.e. 5 for karak, vibhakti, vachan, root word, and suffix) and softmax activation.
- Compile this sequential model using Adam optimizers and binary cross-entropy for the following evaluation measures:
  1. accuracy,
  2. f1 measure,
  3. precision,
  4. and recall

- The model is trained with the above-evaluated vectors for 20 epochs
- The saved trained model is used for further predictions

7. Results and Discussion

- An algorithm performs morphological analysis of a given the word, resulting in a root word with its suffix and vachan (number), vibhakti (case), and karaka (class defined by karaka theory).
- The F-score is obtained for system evaluation. Figures 2(a) and 2(b) show the precision and recall of the system obtained.

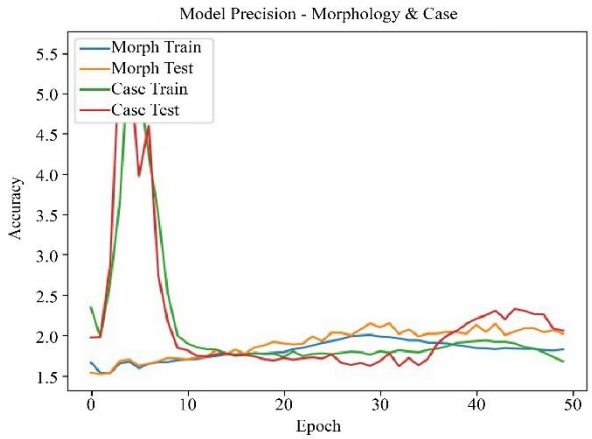

Fig. 2(a) Model precision

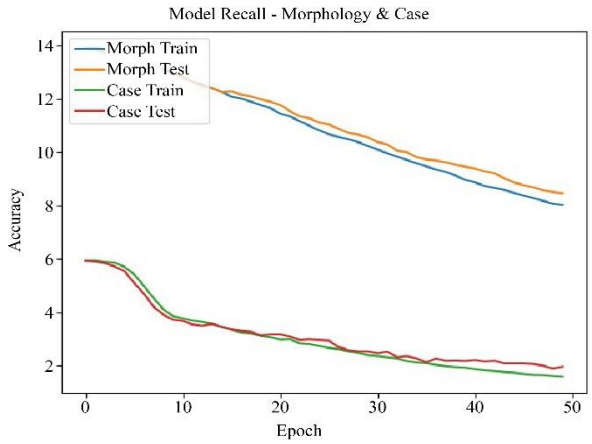

Fig. 2(b) Model recall

- As the experiment is performed using a limited number of nouns and affixes, the results can be improved with the increased dataset.
- The system can be used as a generalized platform for obtaining word-embedding in any language when basic information is provided.
- More efficient results can be obtained with increasing datasets as well as several epochs.

8. Contribution and Limitations

The paper contributes to the design of a multi-lingual morphological language processor based on unsupervised learning. The character-based unsupervised learning would help in understanding languages with less digital content. The processor can handle out-of-vocabulary problems. With more number of epochs with large datasets, the processor can handle ambiguity problems. The working of the processor is limited to dealing with Sanskrit and English language only. Even it is limited to handling ambiguous morphological structures because of the small training dataset. More work is required to handle languages like Hindi and even regional languages.

9. Conclusion

The major challenge in processing natural languages is understanding the meaning of the content. Words in sentences carry information about entities in terms of stem, gender, case, number, etc. Words occurring in the sentence with their semantic classes are stored in a dataset using case-based semantic analysis. Languages are processed to obtain knowledge representation structures known as case frames. Panini defined six karaka classes to specify the semantic roles. Karakas is identified by the vibhakti and inflections of parts of speech of a language. Extraction and annotation of this information help in understanding the language. The proposed system performs morphological processing of a language using an unsupervised learning algorithm. An algorithm performs morphological analysis of a given the word, resulting in a root word with its suffix, vachan (number), vibhakti (case), and karaka (class defined by karaka theory).

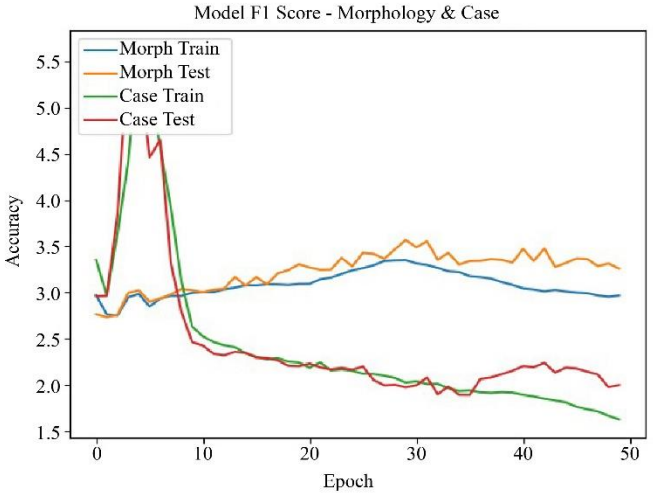

Fig. 3 Accuracy (F1-score)

As shown in Figure 3, the proposed model obtained 52% recall, which can be improved with increased datasets and more epochs for learning.

The system can be used as a generalized platform for obtaining word-embedding of any language when provided with basic information. The system would serve as a tool for linguistic analysis for languages with limited available corpus and annotated data.

10. Conflicts of Interest

The research outcomes reveal the existence of specific word classes based on karak (case) roles. The obtained embedding helped in designing the surface-level interface for the Paninian framework of language processing. The obtained outcomes motivate language processing using machine learning to study the language insights of any language. The research serves a generalized purpose of studying the natural language processing domain using machine learning trends when provided with specifically prepared datasets of a language to be processed.

References

[1] B. Premjith, and K.P. Soman, "Deep Learning Approach for the Morphological Synthesis in Malayalam and Tamil at the Character level," *ACM Transaction on Asian and Low-Resource Language Information Processing*, vol. 20, no. 6, pp. 1-17, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[2] Deepti Chopra, Nisheeth Joshi, and Iti Mathur, *Mastering Natural Language Processing with Python*, Packet Publishing, pp. 1-238, 2016. [Google Scholar] [Publisher Link]

[3] Remya Sivan, "Study on Morphological Analyzer and Generator for Malayalam," *International Journal of Engineering Science Invention*, vol. 8, no. 3, pp. 73-77, 2019. [Publisher Link]

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *Arxiv*, pp. 1-15, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[5] M. Anand Kumar et al., "Morphological Analyzer for Agglutinative Languages Using Machine Learning Approaches," *2009 International Conference on Advances in Recent Technologies in Communication and Computing*, Kottayam, India, pp. 433-435, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[6] V.P. Abeera et al., "Morphological Analyzer for Malayalam Using Machine Learning," *Data Engineering and Management: Second International Conference, ICDEM 2010*, Tiruchirappalli, India, pp. 252-254, 2012. [CrossRef] [Google Scholar] [Publisher Link]

[7] B. Premjith, K.P. Soman, and M. Anand Kumar, "A Deep Learning Approach for Malayalam Morphological Analysis at Character Level," *Procedia Computer Science*, vol. 132, pp. 47-54, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[8] Saurav Jha, Akhilesh Sudhakar, and Anil Kumar Singh, "Multi Task Morphological Analyzer: Context Aware Neural Joint Morphological Tagging and Lemma Prediction," *Arxiv*, pp. 1-28, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[9] C. Rahul, and R. Gopikakumari, "A Character Level Sanskrit-Malayalam Parallel Morphological Analyzer Using Deep Learning," *Design Engineering*, pp. 994-1021, 2021. [Google Scholar]

[10] Ryan Cotterell, and Hinrich Schütze, "Morphological Word Embeddings," *Arxiv*, pp. 1-6, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[11] Jenny Audring, and Francesca Masini, *Introduction: Theory and Theories in Morphology*, The Oxford Handbook of Morphological Theory, pp. 1-16, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[12] Dinesh Ramoo, Psychology of Language, 2021. [Online]. Available: https://opentextbc.ca/psyclanguage/

[13] Mariana Neves, Natural Language Processing, SoSe, 2017. [Online]. Available: https://hpi.de/oldsite/fileadmin/user_upload/fachgebiete/plattner/teaching/NaturalLanguageProcessing/NLP2017/NLP01_IntroNLP.pdf

[14] F. Staal, Pannian Linguistics, 2021. [Online]. Available: https://web.stanford.edu/class/linguist289/encyclopaedia001.pdf

[15] Sherly Elizabeth, N. Rajendran, and R.R. Rajeev, "A Suffix Stripping Based Morph Analyser for Malayalam Language," *Proceedings of 20th Kerala Science Congress*, pp. 482-484, 2007. [Google Scholar]

[16] John A. Goldsmith, Derrick Higgins, and Svetlana Soglasnova, "Automatic Language-Specific Stemming in Information Retrieval," *Cross-Language Information Retrieval and Evaluation: Workshop of the Cross-Language Evaluation Forum for European Languages*, Lisbon, Portugal, pp. 273-283, 2001. [CrossRef] [Google Scholar] [Publisher Link]

[17] Fred Karlsson, "A Paradigm-Based Morphological Analyzer," *Proceedings of the 5th Nordic Conference of Computational Linguistics (NODALIDA 1985)*, Hels Liane Guillouinki, Finland, pp. 95-112, 1986. [Google Scholar] [Publisher Link]

[18] Alexander Fraser, and Liane Guillou, Two Level Morphology, Computational Morphology and Electronic Dictionaries, 2016. [Online]. Available: https://www.cis.uni-muenchen.de/~fraser/morphology_2016/two_level_morph.pdf

[19] Kimmo Koskenniemi, "Two-Level Model for Morphological Analysis," *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pp. 1-3, 2020. [Google Scholar] [Publisher Link]

[20] Kyriakos N. Sgarbas, Nikos D. Fakotakis, and George K. Kokkinakis, "A Straightforward Approach to Morphological Analysis and Synthesis," *Arxiv*, pp. 1-6, 2001. [CrossRef] [Google Scholar] [Publisher Link]

[21] P.M. Vinod, Jayan Vasudevan, and V.K. Bhadran, "Implementation of Malayalam Morphological Analyzer Based on Hybrid Method," *Proceedings of the 24th Conference on Computational Linguistics and Speech Processing (ROCLING 2012)*, Chung-Li, Taiwan, pp. 307-317, 2012. [Google Scholar] [Publisher Link]

[22] Vector Search, Google Cloud. [Online]. Available: https://cloud.google.com/vertex-ai/docs/vector-search/overview

[23] Dan Jurafsky, *Speech and Language Processing*, Pearson Education, pp. 1-908, 2020. [Publisher Link]

[24] Bin Wang et al., "Evaluating Word Embedding Models: Methods and Experimental Results," *APSIPA Transaction on Signal and Information Processing*, vol. 8, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[25] Amir Bakarov, "A Survey of Word Embeddings Evaluation Methods," *Arxiv*, pp. 1-26, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[26] Kris Cao, and Marek Rei, "A Joint Model for Word Embeddinig and Word Morphology," *Proceedings of First Workshop on Representation Learning for NLP*, Berlin, Germany, pp. 18-26, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[27] Yoshua Bengio et al, "A Neural Probabilistic Language Model," *Journal of Machine Learning Research*, vol. 3, pp. 1137-1155, 2003. [Google Scholar] [Publisher Link]

[28] Ronan Collobert et al., "Natural Language Processing (Almost) from Scratch," *Journal of Machine Learning Research*, vol. 12, no. 76, pp. 2493-2537, 2011. [Google Scholar] [Publisher Link]