Research Article

A. P. Ajees*, K. J. Abrar, Mary Idicula Sumam, and M. Sreenathan

A Deep Level Tagger for Malayalam, a Morphologically Rich Language

https://doi.org/10.1515/jisys-2019-0070
Received Mar 14, 2019; accepted Aug 15, 2019

Abstract: In recent years, there has been tremendous growth in the amount of natural language text through various sources. Computational analysis of this text has got considerable attention among the NLP researchers. Automatic analysis and representation of natural language text is a step by step procedure. Deep level tagging is one of such steps applied over the text. In this paper, we demonstrate a methodology for deep level tagging of Malayalam text. Deep level tagging is the process of assigning deeper level information to every noun and verb in the text along with normal POS tags. In this study, we move towards a direction that is not much explored in the case of Malayalam language. Malayalam is a morphologically rich and agglutinative language. The morphological features of the language are effectively utilized for the computational analysis of Malayalam text. The language level details required for the study are provided by Thunjath Ezhuthachan Malayalam University, Tirur.

Keywords: Malayalam, Natural Language Processing, Word embedding, Support Vector Machine, Multi-Layer Perceptron, Deep level tagger

2010 Mathematics Subject Classification: 68T50

1 Introduction

Internet is the fastest growing resource in this world. A lot of texts and images are added to the web day by day. Whenever a person wants to get some information from this data, he must go through all the documents and search for the required content. This is a laborious task which requires an enormous amount of time. Expanding availability of data has demanded extensive analysis of it through automated mechanisms. Automatic text summarization, semantic graph construction, anaphora resolution, etc. are some such mechanisms that bring this notion to reality. Deep level tagger is a middle way technology towards the automatic analysis of natural language text. It is the process of assigning deeper level information to each and every noun and verb in a text document.

Malayalam is a resource-constrained morphologically rich language. It is the native language of Kerala and is also spoken in different parts of India such as Lakshadweep, Pondicherry, Mahi, etc. It is one of the scheduled languages in India with a speaking population count of 38 million [22]. Malayalam belongs to the family of Dravidian languages with the inherited characteristics of Sanskrit, the language of Vedas. The highly

***Corresponding Author: A. P. Ajees:** Research Scholar,Department of Computer Science, Cochin University of Science and Technology, Cochin, India; Email: ajeesap87@gmail.com
K. J. Abrar: Research Scholar,Department of Linguistics, Thunjath Ezhuthachan Malayalam University, Tirur, India; Email: abrarkj@gmail.com
Mary Idicula Sumam: Professor, Department of Computer Science, Cochin University of Science and Technology, Cochin, India; Email: sumam@cusat.ac.in
M. Sreenathan: Professor, Department of Linguistics, Thunjath Ezhuthachan Malayalam University, Tirur, India; Email: msreenathan@gmail.com

productive morphology of Malayalam results in the generation of highly ambiguous and compound words. It is also a free word order language with a common format of SOV (subject, object, verb) [11].

Automatic analysis of verbs and nouns in sentences is an essential task for the computational understanding of the natural language text. Different studies are conducted to analyze the morphology of Malayalam words [4, 10, 15, 18, 20, 21]. Morphological analyzers take one word at a time and analyze its structure, syntax, and morphological properties [5]. Identifying the morphological properties of agglutinative words is a challenging task. However, it does not contribute much to the semantic understanding of the document. Here comes the advantage of deep level taggers. Deep level taggers are tools that help to process the text in a semantically meaningful manner. It considers all the nouns and verbs in a document and generates an in-depth analysis, which can be effectively utilized for higher end tasks such as anaphora resolution, text summarization, sentiment analysis, etc. The in-depth analysis of nouns includes capturing the number, gender and case information associated with them. Whereas, the in-depth analysis of verbs includes capturing the tense, aspect and modality information associated with them.

The number and gender information associated with nouns is essential for building applications like anaphora resolutions systems. Since anaphors used in a particular discourse refers to an antecedent which in turn agree with the number and gender of the anaphor, finding the gender and number information associated with the nouns in a document is of utmost importance in anaphora resolution. Similarly, identifying the subject and object in a sentence is very important when we are dealing with a machine translation system. Case information associated with the nouns can provide linguistic cues about the subject and object in a sentence [6]. A proper understanding of the natural language text is possible only after the identification of tense, aspect, and modality features associated with the verbs in a document. Tense indicates the location of the verb with respect to time and aspect indicate how the verb extends over time. Aspect applies equally to the present and future tense. Whereas, the mood of a verb indicates the degree of necessity or obligation.

Proper analysis and classification of verbs are of prime importance in applications like sentiment analysis, abstractive text summarization, machine translation, etc. Many grammarians and poets have classified verbs in Malayalam to a number of different classes. Prof. A R Rajarajavarma, who also known as 'Kerala panini' for his contributions to Malayalam grammar, classified Malayalam verbs to 38 different classes [2]. Later his classification has been analyzed in detail and found not suitable for computational purposes. Suranad kunjanpillai, a historian and scholar of Malayalam language, also classified verbs into different categories [14]. He considered tense suffixes for the classification of verbs into 16 categories. The morphemic changes occurring in the root verbs during the addition of tense suffixes to root verbs are taken care for the classification. Similarly, Wickremasinghe and Menon classified verbs into eight classes and Sekhar classified them to 12 classes [7]. However, none of these classifications considered 'TAM' information for their analysis. Hence, we decided to go for a verb classifier based on the 'TAM' information associated with it. Table 1 shows the different classes of verbs in Malayalam according to the 'TAM' information associated with it. Each class indicate the 'TAM' details associated with a particular verb.

There are numerous challenges associated with the deep level tagging of Malayalam words. The primary issue is the lack of a proper morphological analyzer for Malayalam. Even though different works are reported in the morphological analysis of Malayalam text, lack of a full-fledged system for morphological analysis is still a dilemma. The highly productive morphology of Malayalam results in the generation of words which are often ambiguous. Thus, many word forms can be generated from a single root word with the addition of suffixes. Moreover, the suffixes, in turn, carry a lot of information regarding the meaning of the text. Hence effective utilization of morphological features is necessary to handle all these problems from the perspective of an NLP researcher.

In this paper, we propose a machine learning based deep level tagger for Malayalam. Tagging is performed using the power of word embeddings and suffix stripping based classification methodologies. In comparison with the reported works in the field, the main contributions of this work are

- A machine learning based methodology for the in-depth analysis of nouns and verbs
- The power of word embedding is effectively utilized for the analysis of nouns

- Morphological features of the language are exploited in such a way that it could be utilized for machine learning algorithms
- Linguistic knowledge is incorporated in the study with the help of language researchers from Malayalam university
- Provides better results and can be incorporated into futuristic systems.

The structure of this article is as follows. Section 2 briefly reviews the related works. Section 3 describes the proposed method. Section 4 discusses the experiments and results. And section 5 concludes the article along with some directions for future works.

Table 1: Different classes of verbs according to 'TAM' analysis

Verb Label	Class name
Verb_PAST	Past Tense
Verb_PRES	Present Tense
Verb_FUT	Future Tense
Verb_mood_IMPR	Imperative Mood
Verb_mood_CMPL	Compulsive Mood
Verb_mood_CMPL_NEG	Negative Compulsive Mood
Verb_mood_PROS	Promissive Mood
Verb_mood_PRMS	Permissive Mood
Verb_mood_OPT	Optative Mood
Verb_mood_PRCT	Precative Mood
Verb_mood_PRCT_NEG	Negative Precative Mood
Verb_mood_DSRV	Desiderative Mood
Verb_mood_ABLT	Abilitative Mood
Verb_mood_IRLS	Irrealis Mood
Verb_mood_PURP	Purposive Mood
Verb_mood_COND	Conditional Mood
Verb_mood_STSF	Satisfactive Mood
Verb_mood_MONI	Monitory Mood
Verb_aspect_PROG_SAT	Progressive Stative Aspect
Verb_aspect_PROG_INS	Progressive Instantaneous Aspect
Verb_aspect_PROG_ITR	Progressive Iterative Aspect
Verb_aspect_PRF_SMPL	Simple Perfect Aspect
Verb_aspect_PRF_CTP	Contemporaneous Perfect Aspect
Verb_aspect_PRF_RMT	Remote Perfect Aspect
Verb_aspect_HBTL	Habitual Aspect

2 Related work

Numerous works are reported for the morphological analysis of Malayalam language [4, 10, 15, 18, 20, 21]. However, none of them deals with deep level tagging. Most of the reported works in morphological analysis use rule-based or stochastic methods for morphological analysis. Based on our knowledge, only very few works are reported in machine learning based morphological analysis [6, 19]. Rajeev et al. [16] reported a suffix stripping based morph-analyzer for Malayalam. Sandhi rules are used to identify the root forms of words through suffix stripping methodologies. According to them, suffix stripping is the simplest method that achieves morphological analysis rather than brute force and other approaches. A morphological analyzer, as

well as a morphological generator for Malayalam-Tamil machine translation, is reported by Jisha et al. in 2011 [4]. They made use of a lexicon and a bilingual dictionary to perform both the operations. Through their work, they have proved suffix separation as an efficient method for morphological analysis.

Latha et al. reported a system for splitting compound words in Malayalam language [10]. They used a rule-based system for compound word splitting with an accuracy of 90%. Lexicon tries are utilized in their study, which is not reported by any other similar systems. A hybrid approach for the morphological analysis of Malayalam is reported in 2012 [21]. A combination of paradigm and suffix stripping approaches experimented in that study. 'Lttoolbox', an essential module in the appertium package is the backbone of the proposed hybrid system. It reported an average accuracy of 83.67% on test data. According to the authors, the performance of the system can be improved by refining the morphological dictionary and suffix list employed in the study. Recently, a machine learning based approach to suffix separation in Malayalam was reported by Marypriya et al. [19]. They discussed a method to generate a sandhi rule annotated dataset for Malayalam words. The prepared dataset was used to develop a machine learning model which could automatically predict the sandhi rules associated with Malayalam words. The issues encountered in developing a compound word splitting tool for the Malayalam language is also incorporated in their study.

One work that considered the morphological analysis of verbs in Malayalam was reported by Sunil et al. in 2012 [20]. They proposed a methodology for the morphological analysis and synthesis of verbs using a paradigm approach. A paradigm defines all the word forms of a given stem and also provides a feature structure with every word form. Another work in verb analysis was reported by R Ravindrakumar et al. in 2011 [7]. They classified verbs based on the past tense forms and the morphogenic changes in the verb roots. This classification is applicable to rule-based machine translation systems and other similar NLP applications. Based on our knowledge, no work is reported in the deep level tagging of Malayalam words.

3 Proposed Method

The proposed methodology is illustrated in figure 1. It shows the general block diagram of deep level tagger for verbs and nouns in Malayalam. The general architecture contains three modules. The first module is the

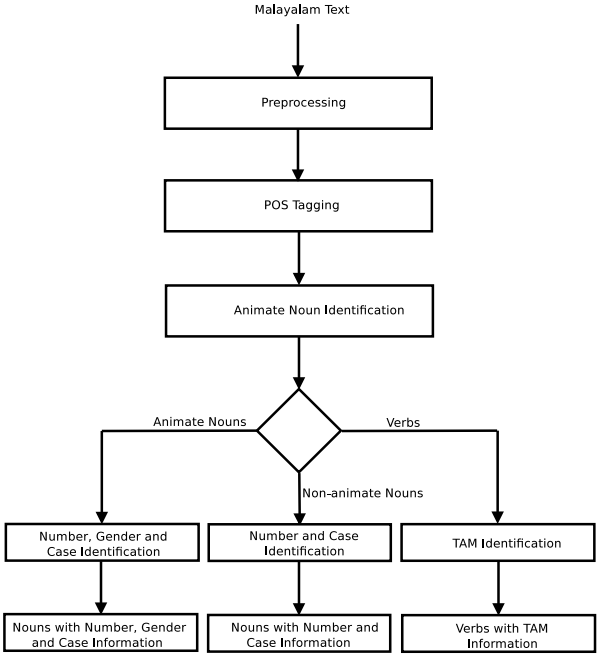

Figure 1: Architecture of the proposed system

POS tagging module, where the words from the preprocessed input text are tagged with POS information. The tags used for this study belong to the BIS (Bureau of Indian Standards) tag set. BIS tagset is a hierarchical tag set that exploits the linguistic hierarchy among different categories. The second module is the animate noun identification module, which identifies the animate nouns (nouns that refer to humans and animals) from the set of noun words. And the final module is the deep level tagging module which unveils the in-depth information associated with nouns and verbs.

3.1 POS Tagging

In the first phase, the preprocessed Malayalam text is provided to the POS tagging module. POS tagging is the preliminary step in most of the NLP applications. It identifies the grammatical category of words in a natural language text. In our study, we have used a CRF based POS tagger developed in our department [1]. CRFs are exceptionally powerful tools for sequence labelling tasks. A piece of text tagged with the above-mentioned tagger is shown in figure 2. These tags are from a limited set of tags (36) developed by BIS (Bureau of Indian Standards). Different tags and their descriptions are given in table 2.

Table 2: BIS Tagset and its Description

Tag	Description	Tag	Description
N_NN	Common noun	RB	Adverb
N_NNP	Proper noun	PSP	Postposition
N_NST	Locative noun	CC_CCD	Co-ordinator
V_VM	Main verb	QT_QTC	Cardinals
V_VM_VF	Finite verb	QT_QTO	Ordinals
V_VM_VNF	Non-finite verb	RD_RDF	Foreign words
V_VM_VINF	Infinite verb	RD_SYM	Symbol
V_VN	Verbal noun	RD_PUNC	Punctuation
V_VAUX	Auxiliary verb	RD_UNK	Unknown
JJ	Adjective	RD_ECH	Echo words
DM_DMD	Deictic demonstrative	RP_INTF	Intensifier particle
DM_DMR	Relative demonstrative	RP_NEG	Negation particle
DM_DMQ	Wh-word(Demonstrative)	QT_QTF	General quantifier
PR_PRP	Personal pronoun	CC_CCS	Subordinator
PR_PRF	Reflexive pronoun	CC_CCS_UT	Quotative
PR_PRL	Relative pronoun	RP_RPD	Default particle
PR_PRC	Reciprocal pronoun	RP_CL	Classifier particle
PR_PRQ	Wh-word(Pronoun)	RP_INJ	Interjection particle

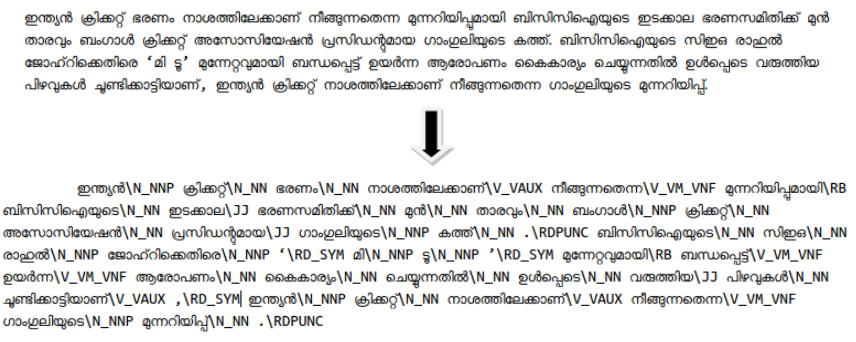

Figure 2: Sample text showing the input and output of the POS Tagging module

3.2 Animate Noun Identification

The second module of the architecture is the animate noun identification module which identifies the animate information associated with nouns. The case information associated with the nouns is also identified in this module. A rule-based module is employed for this purpose (case identification). A set of suffixes corresponding to each case are stored in a look-up table which returns the case information associated with the nouns. The case identified nouns from the tagged text is provided to an animate noun classifier. The animate noun classifier is developed with the help of a set of nouns belonging to both animate and non-animate category. Table 3 shows the class-wise statistics of the training data for this classifier.

Table 3: Class-wise statistics of the training data (animate noun identifier)

Noun type	Count
Animate nouns	43016
Non-animate nouns	66414

In our study, we have used five families of classification algorithms, including Naive Bayes, kNN, SVM, Random Forest and MLP, as a basis for finding the best classifier on our data. Among the different classification techniques, the SVM algorithm- a popular technique for pattern recognition and classification, gave the best performance in comparison with the other classification algorithms. Given a set of instance-label pairs, the SVM algorithm maps the training instances into a higher dimensional space by applying a kernel function and then discovers a linear separable hyperplane with maximal margin. Provided a set of training samples (x_i, y_i), i=1,2,....n, the SVM algorithm tries to optimize the following equation:

$$\min_{w,b,\epsilon} 1/2 W^T W + C \sum_{n=1}^{\infty} \varepsilon_i \quad subject\ to\ y_i(W^T \Phi(x_i) + b) \geq 1 - \varepsilon_i \tag{1}$$

where $x_i \in R^N$ are training instances belonging to different classes, $y \in R^n$ is a vector such that $y_i \in \{1, -1\}^n$, i are slack variables and C is the penalty parameter of the error term.

Figure 3 shows the detailed architecture of module 2. Since words are symbolic constituents, it can't be directly fed into neural networks. Hence words are converted into numeric values using Word2vec [9]. Word2vec is one of the easiest ways to produce the vector representation of words in any language. The dimension of the word embeddings also has a considerable impact on the performance of the classifier. Figure 4 shows the output of the second module on the sample text.

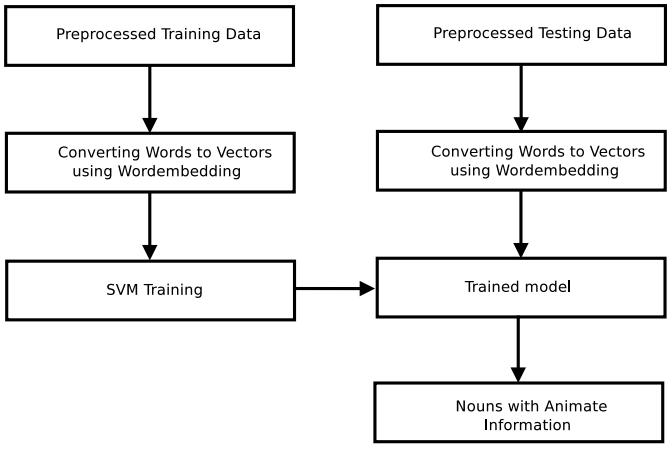

Figure 3: Architecture of the animate noun identification module

[['നരേന്ദ്രമോദി\N_NNP', 'സർക്കാരിനെ\N_NN', 'പിടിച്ചുലയ്ക്കുന്ന\V_VM_VNF', 'വിവാദബോംബാണ്\V_VAUX', 'വിജയ്\N_NNP', 'മല്യ\N_NNP', 'ഇന്നലെ\N_NN', 'ലണ്ടനിൽ\N_NNP', 'പൊട്ടിച്ചത്\V_VM_VNF', '.\RDPUNC'], ['ഇന്ത്യയിൽ\N_NNP', 'സാമ്പത്തിക\JJ', 'തട്ടിപ്പുകൾ\N_NN', 'നടത്തിയവർക്ക്\N_NN', 'ബിജെപിയുമായും\N_NN', 'കേന്ദ്ര\JJ', 'സർക്കാരിലെ \N_NN', 'ഉന്നതരുമായും\N_NN', 'അടുത്ത\N_NST', 'ബന്ധമുണ്ടെന്ന\V_VM_VNF', 'ആരോപണം\N_NN', 'പ്രതിപക്ഷ\N_NN', 'കക്ഷികൾ\N_NN', 'നിരന്തരം\RB', 'ഉയർത്തുന്ന\V_VM_VNF', 'ഘട്ടത്തിലാണ്\V_VAUX', 'അവർക്ക്\PR_PRP', 'വെടിക്കോപ്പ \N_NN', 'പകരുന്ന\V_VM_VNF', 'മല്യയുടെ\N_NNP', 'വെളിപ്പെടുത്തൽ-\N_NN', '.\RDPUNC'], ['ഇന്ത്യ\N_NNP', 'വിട്ടം \N_NN', 'മൻപ്\N_NN', 'ധനമന്ത്രി\N_NN', 'അരുൺ\N_NNP', 'ജയ്റ്റ്ലിയെ\N_NNP', 'കണ്ടിരുന്ന\V_VM_VF', '.\RDPUNC'], ['ചർച്ച\V_VM_VNF', 'നടത്തിയിരുന്ന\V_VM_VF', '.\RDPUNC']

[['നരേന്ദ്രമോദി\N_NNP\Nom\Ani', 'സർക്കാരിനെ\N_NN\Acc\Nani', 'പിടിച്ചുലയ്ക്കുന്ന\V_VM_VNF', 'വിവാദബോംബാണ് \V_VAUX', 'വിജയ്\N_NNP\Nom\Ani', 'മല്യ\N_NNP\Nom\Ani', 'ഇന്നലെ\N_NN'\Acc\Nani, 'ലണ്ടനിൽ\N_NN\Loc\Nani', 'പൊട്ടിച്ചത്\V_VM_VNF', '.\RDPUNC'], ['ഇന്ത്യയിൽ\N_NNP\Loc\Nani', 'സാമ്പത്തിക\JJ', 'തട്ടിപ്പുകൾ\N_NN\Nom\Nani', 'നടത്തിയവർക്ക്\N_NN\Dat\Nani', 'ബിജെപിയുമായും\N_NN\Nom\Nani', 'കേന്ദ്ര\JJ', 'സർക്കാരിലെ\N_NN\Acc\Nani', 'ഉന്നതരുമായും\N_NN\Nom\Nani', 'അടുത്ത\N_NST', 'ബന്ധമുണ്ടെന്ന\V_VM_VNF', 'ആരോപണം\N_NN\Nom\Nani', 'പ്രതിപക്ഷ\N_NN\Nom\Nani', 'കക്ഷികൾ\N_NN\Nom\Nani', 'നിരന്തരം\RB', 'ഉയർത്തുന്ന\V_VM_VNF', 'ഘട്ടത്തിലാണ് \V_VAUX', 'അവർക്ക്\PR_PRP', 'വെടിക്കോപ്പ\N_NN\Nom\Nani', 'പകരുന്ന\V_VM_VNF', 'മല്യയുടെ\N_NN\Ins\Ani', 'വെളിപ്പെടുത്തൽ-\N_NN\Nom\Nani', '.\RD_PUNC'], ['ഇന്ത്യ\N_NNP\Nom\Nani', 'വിട്ടം\N_NN\Nom\Nani', 'മൻപ്\PSP', 'ധനമന്ത്രി\N_NN\Nom\Nani', 'അരുൺ\N_NNP\Nom\Ani', 'ജയ്റ്റ്ലിയെ\N_NNP\Nom\Ani', 'കണ്ടിരുന്ന\V_VM_VF', '. \RDPUNC'], ['ചർച്ച\V_VN', 'നടത്തിയിരുന്ന\V_VM_VF', '.\RDPUNC']]

Figure 4: Sample text showing the input and output of the Animate Noun Identification module

3.3 Deep Level Tagging

The final module of the architecture is the deep level tagging module, which performs the in-depth analysis of nouns and verbs in the text document. The verbs and animate nouns from the previous modules are fed to the deep level tagging module. The deep level information includes the number and gender details associated with the animate nouns. Whereas the deep level information associated with verbs include tense, aspect and modality details. Since Malayalam is a morphologically rich language, the morphological richness of the language is utilized to capture the in-depth information associated with nouns and verbs. The morphological features are captured with the help of a suffix stripper which can strip the suffixes of different length. Figure 6

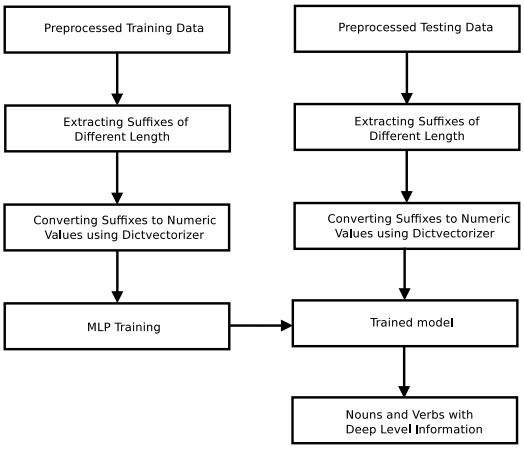

Figure 5: General architecture of the Deep level tagging module

Table 4: Class-wise statistics of the training data (number-gender classifier)

Class	Label	Count
Singular male	M/S	5000
Singular female	F/S	5000
Plural male	M/P	1400
Plural female	F/P	1200

[['നരേന്ദ്രമോദി\N_NNP\Nom\Ani\M/S', 'സർക്കാരിനെ\N_NN\Acc\Nani', 'പിടിച്ചുലയ്ക്കുന്ന\V_VM_VNF\Verb_FUT',
'വിവാദബോംബാണ്\V_VAUX', 'വിജയ്\N_NNP\Nom\Ani\M/S', 'മല്യ\N_NNP\Nom\Ani\M/S', 'ഇന്നലെ\N_NN'\Acc\Nani,
'ലണ്ടനിൽ\N_NN\Loc\Nani', 'പൊട്ടിച്ചത്\V_VM_VNF\Verb_PAST', '.\RDPUNC'], ['ഇന്ത്യയിൽ\N_NNP\Loc\Nani',
'സാമ്പത്തിക\JJ', 'തട്ടിപ്പുകൾ\N_NN\Nom\Nani', 'നടത്തിയവർക്ക്\N_NN\Dat\Nani', 'ബിജെപിയുമായും\N_NN\Nom\Nani',
'കേന്ദ്ര\JJ', 'സർക്കാരിലെ\N_NN\Acc\Nani', 'ഉന്നതരുമായും\N_NN\Nom\Nani', 'അടുത്ത\N_NST', 'ബന്ധമുണ്ടെന്ന
\V_VM_VNF\Verb_FUT', 'ആരോപണം\N_NN\Nom\Nani', 'പ്രതിപക്ഷ\N_NN\Nom\Nani', 'കക്ഷികൾ\N_NN\Nom\Nani',
'നിരന്തരം\RB', 'ഉയർത്തുന്ന\V_VM_VNF\Verb_FUT', 'ഘട്ടത്തിലാണ്\V_VAUX', 'അവർക്ക്\PR_PRP', 'വെടിക്കോപ്പ\N_NN\Nom
\Nani', 'പകരുന്ന\V_VM_VNF\Verb_FUT', 'മല്യയുടെ\N_NN\Ins\Ani\M/S', 'വെളിപ്പെടുത്തൽ-\N_NN\Nom\Nani', '.
\RD_PUNC'], ['ഇന്ത്യ\N_NNP\Nom\Nani', 'വിട്ടം\N_NN\Nom\Nani', 'മൻപ\PSP', 'ധനമന്ത്രി\N_NN\Nom\Nani', 'അരുൺ
\N_NNP\Nom\Ani\M/S', 'ജയ്റ്റ്ലിയെ\N_NNP\Nom\Ani\M/S', 'കണ്ടിരുന്നു\V_VM_VF\Verb_aspect_PRF_SMPL', '.
\RDPUNC'], ['ചർച്ച\V_VN', 'നടത്തിയിരുന്നു\V_VM_VF\Verb_aspect_PRF_SMPL', '.\RDPUNC']]

Figure 6: Sample text showing the output of the deep level tagging module

shows the output of the deep level tagger on sample text. Table 4 shows the class-wise statistics of the training data for the number and gender identification classifier.

Similar to the second module, a bunch of classifiers were attempted to build an in-depth analyzer for verbs and nouns. MLP-a feed-forward artificial neural network classifier showed the best performance in this context and chosen as the in-depth analyzer for verbs and nouns. MLP employs a supervised learning approach called backpropagation for training. Multiple layers and non-linear activation function differentiate MLP from a linear perceptron. Relu-the most frequently used activation in neural networks is utilized in our network. It is analogous to a half-wave rectifier in electrical circuits and is described by the equation:

$$F(x) = max(0, x) \tag{2}$$

From the animate/non-animate tagged text, words with the animate tag are provided to the number and gender identification module. A suffix stripper is used to extract suffixes of different lengths, which in turn acts as the feature set for the number and gender identification module. Similarly, words with verb tag are provided to the 'TAM' information identification module. Here also, the suffixes of different length are extracted by a suffix stripper, which provides the feature set for 'TAM' identification module. MLP classifier with 25 labelled classes is prepared for this purpose. Each class indicates the 'TAM' information associated with that particular verb. Figure 5 shows the general architecture of module 3. In our experiments, we have used different number of hidden layers with various sizes for MLP. The number of hidden layers and their size determines the accuracy and speed of the classifier.

4 Experiments and Results

In this section, the experiments performed on each phase of the architecture are discussed in detail. The first step is the preprocessing of raw Malayalam text, where sentence segmentation and word tokenization operations are carried out. We have used NLTK implementation of sentence tokenizer and word tokenizer for this purpose [8]. The preprocessed raw text is supplied to POS tagging module, where the tagged text is generated. The reported accuracy of the POS tagger employed for this purpose is 91.2%, which appears to be a comparable performance in low resource language such as Malayalam. The second module of the architecture deals with case labelling and animate noun identification. Suffixes corresponding to different cases are stored in the look-up table. For each noun word, suffixes of length 2,3,4 and 5 are extracted and sent to the look-up table. If any match is found, the corresponding case is triggered. Otherwise, the nominative case is returned by the rule-based module. Different cases and their corresponding suffix list is shown in table 5.

Animate noun identification is performed using a machine learning approach. The dataset used for this purpose contains a set of nouns belonging to both the animate and non-animate category. Each noun word from the dataset is labelled with the corresponding class information (animate or non-animate). A set of 109430 nouns are prepared in this way and used for building the machine learning model. Word2vec is utilized to convert nouns into vectors of numeric values. Word2vec [17] model is created using a corpus of 27 lakhs words from different domains. Skip-gram configuration is employed to build the Word2vec model.

Table 5: Case markers and corresponding suffixes

CASE	SUFFIXES
Accusative	e
Sociative	odu
Dative	kku , nu
Genitive	aal
Instrumental	ude, nte
Locative	il
Nominative	All remaining suffixes

Different classification algorithms are used to build the classifier model [13]. The performance of different classification algorithms on our dataset is shown in figure 7. It is clear from the figure that SVM outperforms all the remaining algorithms on discriminating the animate nouns from non-animate nouns. The accuracy achieved by the SVM classifier is 95.1%, beating the second best model by a margin of 1.01%. Parameter tuning of SVM classifier is carried out using GridsearchCV from Scikit learn. The best performance is obtained for a gamma value of '0.01' and C value of '10'. Size of the word vectors also has a considerable impact on the performance of the classifier. In our experiments, we have considered different word vector sizes on different classifier configurations. The performance of the best-functioned model on different word vector sizes is shown in figure 8. As shown in the figure, the best performance is given by a vector size of 200 and above. Hence, we have finalized our word vector size to be 200.

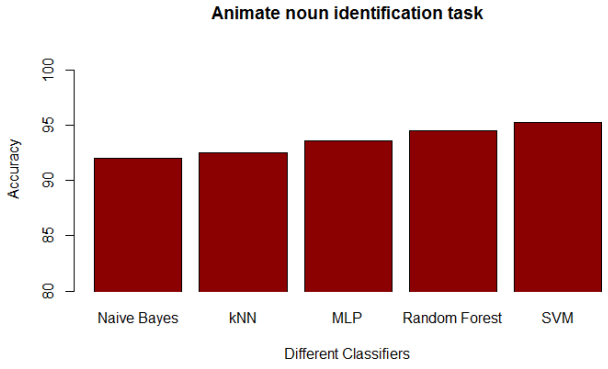

Figure 7: Performance of different classifiers on animate identification task

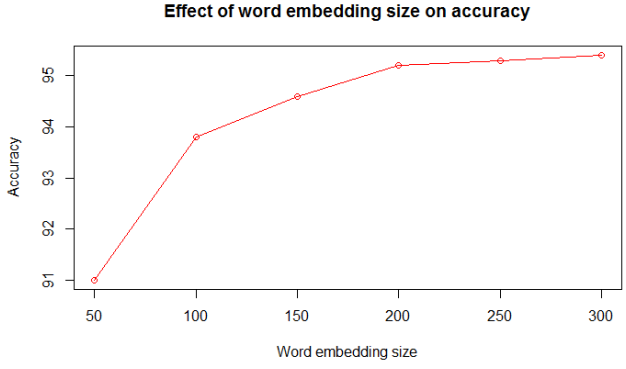

Figure 8: Performance of the animate noun identification network for different word vector sizes

The third module of the architecture deals with the in-depth analysis of verbs and animate nouns. Here too, different classifiers are attempted to distinguish the best performing classifier on our dataset. Unlike in the earlier scenario, MLP outperformed the remaining classifiers on both the tasks ('TAM' analysis and number-gender analysis). Figures 9 and 10 illustrate this point. First, we consider the case of number and gender identification classifier. Training data required to build the first classifier is a list of names belonging to different classes. A list of 12600 names belonging to the different categories is prepared for this purpose. Suffixes of length 1 to 8 are used as features for each name. Since machine learning algorithms require features as numeric values, we have converted our feature set (suffixes) to numeric values using Dictvectorizer, a python functionality [3]. We have chosen Dictvectorizer over Word2vec in this phase, since they are well suited in encoding categorical features with multiple possible values. Moreover, Word2vec is appropriate in situations where the syntactic and semantic roles of words are necessary. Dictvectorizer is employed in situations where the feature set is a list of dictionaries rather than a list of categorical items. In our case, the feature set is a list of dictionaries where each dictionary refers to a set of suffixes corresponding to a single word. Thus, the total feature size is 7468.

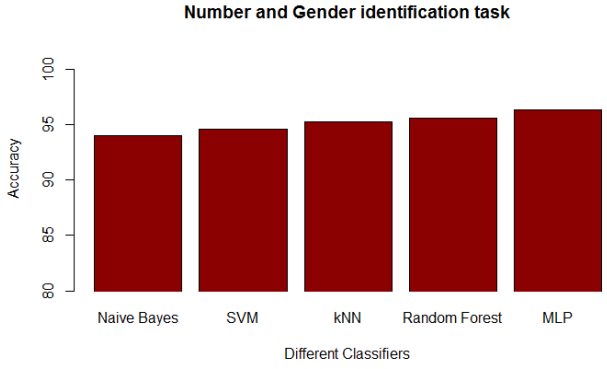

Figure 9: Performance of different classifiers on number and gender identification task

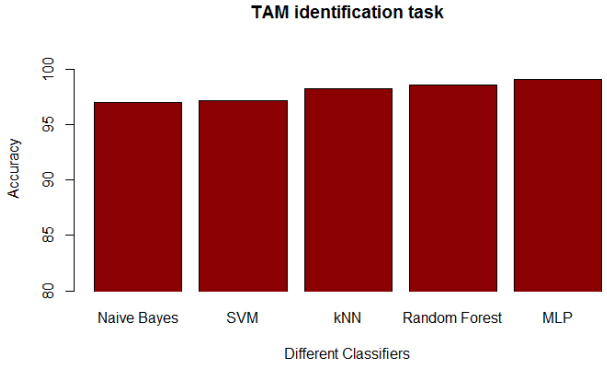

Figure 10: Performance of different classifiers on 'TAM' analysis task

Different configurations of the MLP classifier were attempted in our study. A smaller network was not able to represent the data efficiently and increasing the number of layers did not improve the accuracy significantly. Hence, we have experimentally finalized our hidden layer configuration as (2,100), where 2 is the number of hidden layers, and 100 is the size of each hidden layer. 'Relu' is used as the activation function

and 'Adam' as the optimizer. The performance of the number and gender classifier with the different number of features is shown in figure 11. From the figure, it is clear that the accuracy of the system increases with the increase in suffix length and the maximum accuracy is achieved when the number of features is 10. The maximum accuracy obtained by the classifier is 96.21%.

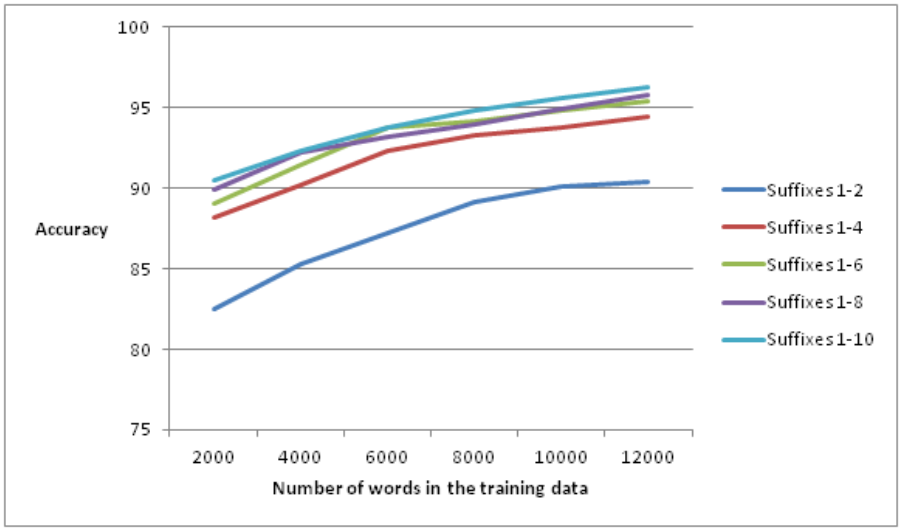

Figure 11: Effect of different features on the performance of the number and gender identification classifier

The configuration of the 'TAM' analysis neural network is not completely different from the number and gender neural network. The only difference is in the number of classes and training data. The training data contains 1205 verbs belonging to 25 classes. Similar to the number and gender network, suffixes of different length are used as features. Here also, the feature vector is converted to a numeric representation using Dictvectorizer, a python functionality. A neural network with two hidden layers constitutes the developed model. The parameters of the network are exactly the same as the number and gender identification neural network. The maximum performance obtained by the classifier is 99.17%. Performance of the 'TAM' analysis model on different sets of features is shown in figure 12. From the figure, it can be inferred that the accuracy of the model increases with the increase in suffix features.

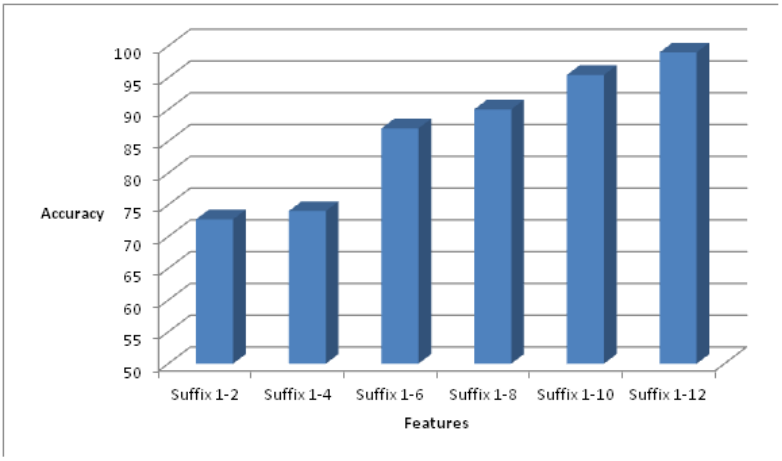

Figure 12: Effect of different features on the performance of the 'TAM' identification classifier

The training data required for all our experiments are prepared with the help of the students from Malayalam University, Tirur. All the datasets used in this study are made publicly available through our department website 'www.cs.cusat.ac.in'. The final accuracy of the complete system is 90.2%. The detailed information regarding the overall performance of the proposed system is shown in table 6.

Table 6: Overall performance of the proposed system

Module-1	Training	Testing
Number of words	230371	57517
Accuracy	91.2%	
Module-2	Training	Testing
Number of words	87544	21886
Accuracy	95.1%	
Module-3-A (number and gender)	Training	Testing
Number of words	10080	2520
Accuracy	96.21%	
Module-3-B (TAM)	Training	Testing
Number of words	964	241
Accuracy	99.17%	
Overall accuracy	90.2%	

4.1 Analysis

To better understand the performance of our models on the constructed datasets, a detailed analysis is also performed. ROC curve-the best metric for evaluating the performance of any classifier is employed to evaluate the performance of each model. The area under the ROC curve represents the degree or measure of separability between different classes predicted by the classifier. Figure 13 tells us, how much the animate noun identification model is capable of distinguishing between the two classes-animate noun and non-animate noun. Similarly, figures 14 and 15 show the ROC curves for the number-gender identification model and 'TAM' identification model respectively. All the models are relatively good in demonstrating the tradeoff between different classes across various settings of the classifiers. Nevertheless, we still can observe that the area un-

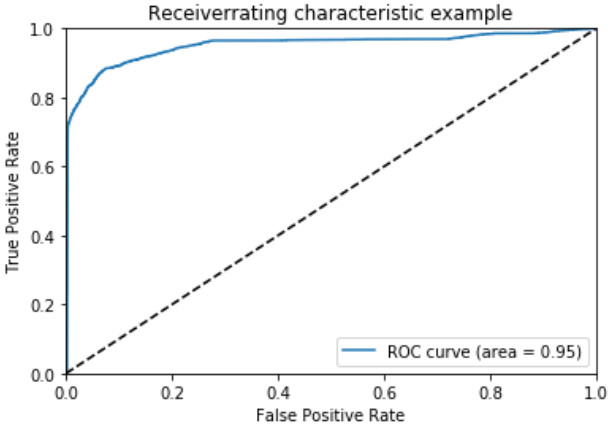

Figure 13: ROC curve of the animate noun classifier

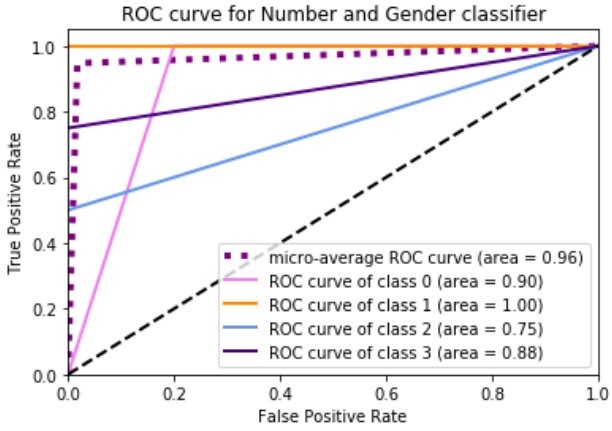

Figure 14: ROC curve of the number and gender classifier

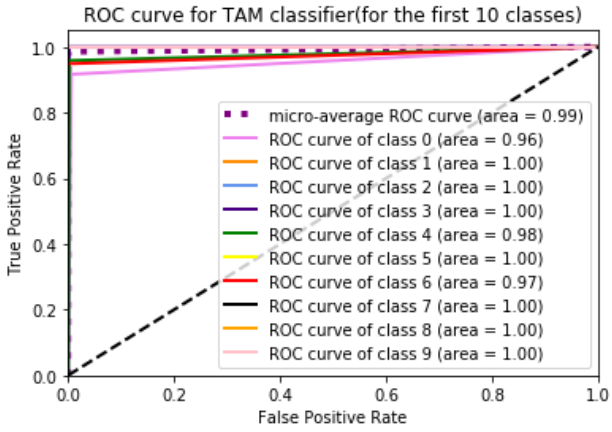

Figure 15: ROC curve of the 'TAM' analysis classifier

der 'TAM' curve is mostly higher than the area under the curves of the other two models. This is contributed by the expressive nature of suffix endings in Malayalam verbs.

We have observed several instances where the Word2vec based word embedding vectors helped in identifying the animate information of nouns. We have also observed some instances where the animate nouns were tagged as non-animate nouns. The major reason for this misclassification is the lack of word embedding vectors (Out of Vocabulary problem) for such nouns. Hence, an effective word representation method (free from OOV problem) capturing the syntactic and semantic properties of words should be formulated to avoid such errors. To determine the contribution of suffixes of different length (as compared to simply using a fixed length suffix) towards classification accuracy, we ran our model with suffixes of different length (ranging from 1 to 12). The performance of the models (in-depth analysis) without using the suffixes of different lengths were even lower than the proposed combined feature systems. This shows us the impact of suffix level features on computational processing of Malayalam text.

Further, the effect of different classifiers on the training data is also verified. It is found that SVM-one of the state-of-the-art tools for binary classification outperformed the other classifiers on animate noun identification task. This is due to the explicit determination of decision boundaries (by the SVM algorithm) from the training data for binary classification problems. However, this is not true in the case of other classification tasks (TAM and number-gender classification), since they are multiclass classification problems. MLP is found to be the best choice for such tasks. Theoretically, MLP can estimate any function or equivalently able to find any mappings [12].

5 Conclusion

In this paper, we have discussed a deep level tagger for Malayalam, a morphologically rich and resource-poor language. The exclusive feature of the proposed system is its in-depth analysis of verbs and nouns. The deeper level analysis of nouns and verbs helps in the semantic understanding of the natural language text and can be used for various language processing applications. The main reason we preferred a machine learning based approach rather than traditional rule-based approaches is its convenience, scalability and low operational cost. We have used word embeddings to recognize the animate nouns from non-animate nouns, which is a fresh thought that is not proposed by any other researchers in this domain. A study on 'TAM' analysis of Malayalam verbs is also presented in this paper. The morphological richness of Malayalam language is utilized in this study with the help of suffix stripping algorithms. In our experiments, we have observed that the increase in morphological features increases the accuracy of the system. Hence incorporating morphological features in the analysis of natural language text appears to be promising for languages such as Malayalam. In future, we would like to study the effect of deep level tagged text on different semantic processing applications of the natural language text such as anaphora resolution, sentiment analysis, text summarization, machine translation, etc.

Acknowledgement: We want to thank the students of Malayalam University, Tirur for their seamless cooperation to this project.

References

[1] AP Ajees and Sumam Mary Idicula, A POS Tagger for Malayalam using Conditional Random Fields.
[2] A.R.Rajarajavarma, Kerala panineeyam, (2000).
[3] Pedregosa Fabian, Varoquaux Gaël, Gramfort Alexandre, Michel Vincent, Thirion Bertrand, Grisel Olivier, Blondel Mathieu, Prettenhofer Peter, Dubourg Vincent, Vanderplas Jake et al., Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* **12** (2011), 2825–2830.
[4] Jisha P Jayan, RR Rajeev, S Rajendran et al., Morphological analyser and morphological generator for malayalam-tamil machine translation, *International Journal of Computer Applications* **13** (2011), 0975–8887.
[5] Jurafsky and Martin, Speech and language processsing, (2002).
[6] K Krishnakumar, S Rajendran and N Rajendran, SUBJECT AND OBJECT IDENTIFICATION IN MALAYALAM USING MACHINE LEARNING APPROACH, *International Journal of Dravidian Linguistics: IJDL*. **42** (2013).
[7] R Ravindra Kumar, KG Sulochana and V Jayan, *Computational aspect of verb classification in malayalam*, Information Systems for Indian Languages, Springer, 2011, pp. 15–22.
[8] Edward Loper and Steven Bird, NLTK: The natural language toolkit, in: *Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, Association for Computational Linguistics, pp. 63–70, 2002.
[9] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, L Sutskever and G Zweig, word2vec, *URL https://code. google. com/p/word2vec* (2013).
[10] Latha R Nair and S David Peter, Development of a rule based learning system for splitting compound words in malayalam language, in: *Recent Advances in Intelligent Computational Systems (RAICS), 2011 IEEE*, IEEE, pp. 751–755, 2011.
[11] Ravisankar S Nair, A grammar of malayalam, (2012).
[12] Peyman Passban, Qun Liu and Andy Way, Boosting Neural POS Tagger for Farsi Using Morphological Information, *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)* **16** (2016), 4.
[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research* **12** (2011), 2825–2830.
[14] Suranad Kunjan Pillai, *Malayalam Lexicon (Volume 1)*, The University of Kerala, 2000.
[15] RR Rajeev and Elizabeth Sherly, Morph Analyser for Malayalam Language: A suffix stripping approach, Proceedings of 20th Kerala Science Congress, 2007.
[16] RR Rajeev and Elizabeth Sherly, A suffix Stripping based Morph Analyser for Malayalam Language, in: *Proceedings of 20th Kerala Science Congress*, pp. 482–484, 2007.

[17] Radim Řehůřek and Petr Sojka, Software Framework for Topic Modelling with Large Corpora, in: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pp. 45–50, ELRA, Valletta, Malta, May 2010, http://is.muni.cz/publication/884893/en (English).
[18] SK Saranya, Morphological analyzer for malayalam verbs, *Unpublished M. Tech Thesis, Amrita School of Engineering, Coimbatore* (2008).
[19] Mary Priya Sebastian and G Santhosh Kumar, Machine Learning Approach to Suffix Separation on a Sandhi Rule Annotated Malayalam Data Set.
[20] R Sunil, Nimtha Manohar, V Jayan and KG Sulochana, Morphological analysis and synthesis of verbs in Malayalam, *ICTAM-2012* (2012).
[21] PM Vinod, V Jayan and VK Bhadran, Implementation of Malayalam morphological analyzer based on hybrid approach, in: *Proceedings of the 24th Conference on Computational Linguistics and Speech Processing (ROCLING 2012)*, pp. 307–317, 2012.
[22] Wikipedia, Malayalam, (2010 (accessed December 7, 2018)).