International Journal of Engineering Trends and Technology
ISSN: 2231 – 5381 / https://doi.org/10.14445/22315381/IJETT-V70I12P217

Volume 70 Issue 12, 160-169, December 2022

Original Article

Evaluating the Grammatical Correctness of Malayalam Text using improved Text GCN

Merin Cherian[1], Kannan Balakrishnan[2]

[1,2]Artificial Intelligence Research Lab, Department of Computer Applications, Cochin University of Science and Technology, Ernakulam, Kerala, India

[1]Corresponding Author : merincherian123@gmail.com

Received: 25 August 2022 Revised: 13 November 2022 Accepted: 24 November 2022 Published: 24 December 2022

Abstract - *Extensive research has been conducted in the domain of automatic grammatical error correction and detection in English and other high-resource languages. However, research in the expanse of Grammatical Error Detection and Correction (GEC) tasks has been very limited in Indian languages. This research uses enhanced TextGCN to perform a grammatical error detection task in Malayalam. It is the first-ever such work in the Malayalam language. This task is evaluated by comparing the results of improved text graph convolutional networks (Text GCN) with TextGCN, LSTM, BiLSTM and CNNLSTM. The results of cross-validation data and unseen sample test data are presented. A training dataset of 200k sentences was created, and 20% of the data was taken as the validation set. Improved Text GCN achieved an accuracy of 90.41% on unseen test data compared to other architectures. This is the first attempt to create a Malayalam grammar checker. Preliminary results from this work show that a graphical representation of text data can be used to check the grammatical correctness of Malayalam text.*

Keywords - *Error detection, Malayalam grammar, Malayalam corpus, Malayalam natural language processing, Text graph convolutional networks.*

1. Introduction

A language's syntactic rules and morphology are governed by its grammar [1]. The incorrect usage of prepositions, articles, conjunctions, tenses etc., commonly causes syntactic errors in English. On the other hand, mistakes in affixation, compound words, and using the plural in noun phrases result in morphological errors. Typographical errors, misuse of punctuation, and syntactic and morphological errors also contribute to grammatical and syntactic errors.

A grammar checker is defined as a program that tries to verify the grammatical correctness of a given text's morphological, syntax and semantic correctness. Creating a complete grammar checker is daunting since creating a complete formal grammar for natural language is complex. A formal grammar constructed for natural language may not be able to represent the entire language because there will be exceptions regarding the usage of grammar in real life scenarios.

Automated grammar checkers are considered writing aid for language learners. The primary function of a grammar checker is to identify incorrect sentences from a text and propose corrections along with a possible linguistic explanation [2]. A grammar checker should deal with various kinds of errors, including context-independent errors, context-dependent errors, punctuation mistakes, style problems, graphical problems [3] etc. Most grammar checkers designed to date address only a subset of these errors. As explained by Uszkoreit (quoted by Hein [4]), the development of a grammar checker is a four-step process.

- The first step is the detection phase, which involves identifying possible ungrammatical segments.
- The second step involves a recognition phase, where the nature of the error is identified based on localization and constraint violation (e.g., subject-verb disagreement).
- Next is the diagnosis step, which identifies the possible sources of errors to form a basis for correction.
- The final step is grammar correction by finding, constructing, ordering, or substituting alternatives.

Making rules for Malayalam is challenging because of the language's open word order. A data-driven approach is more suitable for performing language processing tasks in Malayalam. The lack of Malayalam corpora for tasks hampers the development of language processing for the Malayalam language. Another issue that Malayalam language processing researchers go against is a lack of standardized test data. This work is a pioneering effort in Malayalam grammar checking. Here, a data-driven method is applied, and the training and test data sets were built especially for this task. By building a corpus and a test set, this study attempts to serve as a foundation for the Malayalam GEC tasks.

A grammar checker for the Malayalam language using improved Text Graph Convolutional Networks is presented in this research. A training dataset with 200,000 sentences labeled as grammatical or ungrammatical was created. The trained model was tested on an unseen data set of 500 sentences. The test data was obtained by manually collecting the sentences from language learners, translating some of the sentences in the Corpus of Linguistic Acceptability (CoLA) [5], and collecting various competitive exam questions. In this paper, each Malayalam word or sentence is followed by its pronunciation in English as well as its English meaning.

2. Related Work

In this section, the various approaches used for grammatical error detection and correction (GEC), different grammar checkers available for Indian languages and a bird's eye view of Malayalam grammar and the common errors that occur in the Malayalam language are discussed.

2.1. GEC Approaches

There are many existing approaches for developing a grammar checker. They are broadly classified into rule-based and data-driven approaches [6]. The earliest grammar-checking tools, like Writer's Work Bench, were based on string matching [7]. Later systems developed in the early 1990s involved linguistic analysis and used rule-based parsers. The advent of the new millennium saw the emergence of data-driven approaches for grammar checking. Data-driven techniques use methods like classification, language models (LM), statistical machine translation (SMT) and Web-based techniques for error checking.

LM [1] methods model the data from well-formed text and detect errors based on this model. Classification [1,8] and SMT [1,9] methods introduce artificial errors and use error-annotated data and well-formed text to construct a grammar checker. Automatically generated ungrammatical data or error corpora are used for the training and evaluation of the system. The availability of corpora like Cambridge Learner Corpus (CLC), Chinese Learner English Corpus (CLEC) and similar facilitated the development of these machine learning-based grammatical error checkers. Data-driven approaches gained further momentum after introducing the GEC shared task at the Conference on Computational Natural Language Learning (CoNLL). GEC-shared tasks aim to correct grammatical errors instead of just detecting the grammatical errors. With the advent of deep learning, neural machine translation (NMT) [10,11] based GEC systems have achieved state-of-the-art grammatical error detection and correction results. Machine translation-based approaches need massive parallel corpora to train the model.

Deep learning techniques have assisted in developing generic end-to-end systems for various natural language processing tasks. State-of-the-art results are being produced for NLP tasks in English [12] because of the massive availability of English data. Grammar checkers are available for various languages like Chinese [13,14], French [15], Arabic [16] etc. Many of the Indian languages are free word order languages and are morphologically rich. However, Dravidian languages like Malayalam are highly agglutinative. The unavailability of large datasets in Indian languages also poses a barrier to creating tools for various NLP tasks.

2.2. Grammar Checkers in Indian Languages

A few grammar checkers have been developed for Indian languages like Hindi, Punjabi, and Bangla. CDAC has developed a grammar checker for Hindi that handles Noun Phrase Concord, Verb Phrase Concord, NP - VP Concord. A rule-based Hindi grammar checker was developed by Bopche and Dhopavakar [36], which performs POS tagging using morphological analysis on a Hindi text. It compares the tagged sentence against a set of predefined grammatical patterns. Punjabi grammar checker [18] is the first system developed for an Indian language. This system uses rule-based methods for part-of-speech tagging, phrase chunking, and a whole form lexicon for morphological analysis. Using the grammatical data displayed by POS tags as feature value pairs, agreement checks are carried out at the phrase and clause levels. In literary style Punjabi writings, the system can identify and recommend corrections for various grammatical problems that may be brought on by a lack of agreement, the wrong word order in different phrases, etc. A hybrid grammar checker for Punjabi, based on rules and Machine learning, is implemented in [19].

A spell and grammar checker for Tamil is explained in [20]. It is developed by creating a dictionary, morphological analyser and syntactic analyser. The morphological analyser is built using finite state automata created after a detailed analysis of Tamil grammar. This work resulted from the UGC-sponsored project entitled" Spell and grammar checker for Tamil".

A Natural Language generation approach for grammar correction has been proposed by Bibekananada Kundu [21] for Bangla. This method uses a morphological analyser to break down an input sentence into a series of root words, which are then over-generated to build a trellis by a morphological synthesiser. The search space is then reduced using a linguistic fitness function, and the best repair is chosen using a language model. To ensure that the correct sentence is not too far from the ungrammatical input sentence, word error rate and BLEU score are employed. The burdensome linguistic restrictions are designed using an HMM-based semi-supervised POS tagger and a rule-based mal-rule filter. These hard constraints help in avoiding inappropriate paths in the trellis. Statistical methods involving n-gram analysis of words and POS tags were used to develop the Bangla grammar checker by Alam et al. [22].

An LSTM-based grammar checker was proposed in [37]. Here a Word2Vec embedding of the Kannada language is generated and then trained using the LSTM layer.

The lack of large human-labeled annotated corpora for Indian languages hinders the development of NLP applications using machine learning techniques. As a result, efficient and generalized solutions for NLP tasks like POS taggers, morphological analysers, and grammar checkers are not available for Indian languages.

2.3. Malayalam Grammar

Malayalam is a Dravidian language spoken in the southern state of India, Kerala. It is a highly agglutinative language with 'free word order' and has the following flat clause structure [24], shown in Fig. 1.

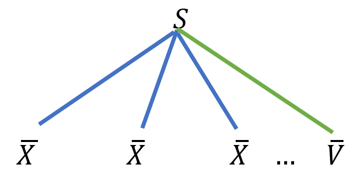

Fig. 1 Malayalam Sentence Structure

In Malayalam, a simple sentence comprising a subject, an object, and a verb has six possible permutations. Thus, the potential word orders in Malayalam [24] are subject-object-verb, subject-verb-object, verb-subject-object, verb-object-subject, object-subject-verb, and object-verb-subject. The verb, object, or subject may be absent from some sentences. Examples of sentences with various word orders in Malayalam are provided in Table 1.

Table 1. Different Word Orders in Malayalam

Word Order	Malayalam Sentence	English Translation
Subject- Object- Verb	അവൻ എതിരാളിയെ ചവിട്ടിയാണ് വീഴ്ത്തിയത്. [avan etirāḷiye caviṭṭiyāṇ vīḻttiyat.]	He kicked his opponent down.
Subject- Verb- Object	അവൻ ചവിട്ടിയാണ് എതിരാളിയെ വീഴ്ത്തിയത്. [avan caviṭṭiyāṇ etirāḷiye vīḻttiyat]	He kicked his opponent down.
Verb- Subject- Object	ചവിട്ടിയാണ് അവൻ എതിരാളിയെ വീഴ്ത്തിയത്. [caviṭṭiyāṇ avan etirāḷiye vīḻttiyat.]	He kicked his opponent down.
Verb- Object- Subject	ചവിട്ടിയാണ് എതിരാളിയെ അവൻ വീഴ്ത്തിയത്. [caviṭṭiyāṇ etirāḷiye avan vīḻttiyat.]	He kicked his opponent down.
Object- Subject- Verb	എതിരാളിയെ അവൻ ചവിട്ടിയാണ് വീഴ്ത്തിയത്. [etirāḷiye avan caviṭṭiyāṇ vīḻttiyat.]	He kicked his opponent down.
Object- Verb- Subject	എതിരാളിയെ ചവിട്ടിയാണ് അവൻ വീഴ്ത്തിയത്. [etirāḷiye caviṭṭiyāṇ avan vīḻttiyat.]	His opponent was kicked down by him.
Subject- Object	രാധയുടെ പണം. [rādhayuṭe paṇaṁ.]	Radha's money.
Subject- Verb	രാമു ഓടി. [rāmu ōṭi.]	Raamu ran.
Object- Verb	രാധയെ ഓടിച്ചു. [rādhaye ōṭiccu.]	(They)chased away Radha.

Subject and predicate can be created by compounding multiple words. The subject can be a pronoun, a nominative noun, a gerund, or a noun phrase. An in-depth discussion on Malayalam grammar is given in [25]. Due to the absence of a fixed word order, sentence components can be moved to the beginning of the phrase or the end of the sentence. Adverbs are usually placed before the verb and after the subject. Sentence connectors like പിന്നെ [pinne] (and then), എന്നിട്ടു [enniṭṭu] (and then), അപ്പോൾ [appēāḷ] (then), അതിനാൽ [atināl] (so), നേരെ മറിച്ച് [nēre maṟicc] (on the other hand), എന്നിട്ടും [enniṭṭuṁ] (still), അതിനു പുറമെ [atinu puṟame] (apart from that), എന്നാൽ [ennāl] (if so), അങ്ങനെ ഇരിക്കെ [aṅṅane irikke] (meanwhile), which take up the first position in a sentence are exceptions to this rule.

If an adverb is placed before the subject, it implies emphasis for the adverb. In the sentence അവൻ നാളെ പോകും [avan nāḷe pēākuṁ] (He will leave tomorrow), നാളെ [nāḷe] (tomorrow) is moved to the left to obtain നാളെ അവൻ പോകും [nāḷe avan pēākuṁ] (Tomorrow he will leave). The second sentence emphasizes നാളെ [nāḷe] (tomorrow). The emphasis does not change when a noun or adverbial phrase is moved to the right.

2.3.1. Malayalam Word Classes and Inflections

Malayalam has six-word classes - Nouns, Verbs, Adjectives, Adverbs, Postpositions and Conjunctions. Nouns are inflected for numbers. A singular noun is unmarked, while a plural noun is marked using the suffix കൾ [-kaL] (plural suffix 's') or special plural marker മാരെ [-maare] (plural marker denoting belongs to) [26]. Nouns inflect for six different cases - Nominative, Accusative, Dative, Sociative, Locative, Instrumental and Genitive - using different bound suffixes. Nouns do not inflect for gender except for third person singular human pronouns and some human nouns that may refer to male or female.

Verb morphology in Malayalam is complex. Malayalam verbs inflect for tense, aspect, and mode. The number of finite and non-finite grammatical word forms of a verb in Malayalam is very large [25]. Most adverbs are morphologically complex and derived from nouns or adjectives.

Adjectives in Malayalam are of 5 types [27] and do not undergo inflection. Postpositions in Malayalam are not inflected, but their etymology is diverse. Conjunctions which are also invariant, join a whole clause to the main clause.

2.3.2. Grammatical Errors in Malayalam

Word order has no bearing on a Malayalam sentence's grammatical mistakes.

In Malayalam, mistakes are frequently brought on by extraneous words, incorrect suffixes, etc. Conjugational errors are the most common type of errors made by learners of Malayalam. Similar terms must be conjugated in a sentence.

Another common error is the repetition of similar words like ഏതാണ്ട് [ētāṇṭ] (about), -ഓളം [ōḷaṁ] (about), മൂലം [mūlaṁ] (due to), -കാരണം [kāraṇaṁ] (due to) in the same sentence. Such analogous words should be used only once in a sentence. While using numerals as adjectives, plurals of collective nouns should not be used. Collective nouns like വെള്ളം [veḷḷaṁ] (water) should not be inflected with plural suffixes.

Unnecessary usage of some terms like എന്നാൽ[ennāl] (but), എന്നിട്ട് [enniṭṭ] (and then), പക്ഷെ [pakṣe] (but), കൂടി [kūṭi] (also), ഒരു [oru] (one), തന്നെ [tanne] (same), കൊണ്ട് [keāṇṭ](with) etc. causes grammatical errors. Adjectives should not be used before an adjective-noun compound word. Table 2. lists the various types of errors and their examples.

Table 2. Various types of grammatical errors in Malayalam

Type of Error	Erroneous Sentence	Corrected Sentence	English translation
Conjugational Error	അമ്മ രാവിലെയും രാത്രിയിൽ അച്ഛനും വന്നു. [am'ma rāvileyuṁ rātriyil acchanuṁ vannu.]	അമ്മ രാവിലെയും അച്ഛൻ രാത്രിയിലും വന്നു. [am'ma rāvileyuṁ acchan rātriyiluṁ vannu.]	Mother came in the morning and father at night.
Analogous word	ഏതാണ്ട് മുന്നൂറോളം ആളുകൾ എത്തിയിരുന്നു. [ētāṇṭ munnūṟēāḷaṁ āḷukaḷ ettiyirunnu]	ഏതാണ്ട് മുന്നൂറ് ആളുകൾ എത്തിയിരുന്നു. [ētāṇṭ munnūṟ āḷukaḷ ettiyirunnu.]	About three hundred people had arrived.
Numerals and Plural	അവൾക്ക് അഞ്ച് മാങ്ങകൾ വേണം. [avaḷkk añc māṅṅakaḷ vēṇaṁ.]	അവൾക്ക് അഞ്ച് മാങ്ങ വേണം. [avaḷkk añc māṅṅa vēṇaṁ.]	She wants five mangoes.
Unnecessary words	പാടുന്നത് അവൾക്കും കൂടി കേൾക്കാം. [pāṭunnat avaḷkkuṁ kūṭi kēḷkkāṁ.]	പാടുന്നത് അവൾക്കും കേൾക്കാം. [pāṭunnat avaḷkkuṁ kēḷkkāṁ.]	She can hear the singing too.
Adjective-noun compound	ചെറിയ ചെറുകഥ [ceṟiya ceṟukatha]	ചെറുകഥ [ceṟukatha]	Short story.

3. Materials and Methods

The methodologies utilised and the implementation details are described in this section. First, the process for creating both the test data and the corpus is outlined. Next, the specifics of TextGCN and improved TextGCN used from training the Malayalam grammar checker are discussed. The experimental setup and the training parameters employed by the different models and cross-validation are described in detail towards the end.

3.1. Malayalam Corpus

Developing a rule-based grammatical structure for the language is challenging because there are no strict constraints for word order in Malayalam. Hence, a data-driven approach for Malayalam grammar checking is used. For Malayalam grammar checking, a training corpus of 200k sentences was created. Grammatically correct sentences were extracted from Malayalam school textbooks, Wikipedia dump and internet archive.

Ungrammatical sentences were collected from the study materials for students. Since the number of erroneous sentences obtained through the manual collection was less, a synthetic data set was generated by introducing errors to the grammatically correct sentences. A round-trip mechanism [28] was used to create errors in the corpus. This technique selected 69k grammatically correct sentences from the Wikipedia dump and translated them into Portuguese using Google translate. Then this Portuguese text was translated into English and finally translated from English to Malayalam. This mechanism created an erroneous corpus which was further manipulated by substituting wrong suffixes to words. Commonly occurring errors among language learners while adding suffixes to words and making compound words were used to create the synthetic dataset. Thirty different substitutions were made for various suffixes to introduce suffix errors to the corpus. The most commonly occurring suffix errors for creating synthesized datasets are summarized in Table 3. The maximum length of each sentence is set to 100, and the minimum number of words in the sentence is two. The final dataset consists of 200k sentences with 70k erroneous sentences and a vocabulary size of 247097 words. The average document length is 5.3.

Table 3. A few of the suffix errors used for creating the synthesized dataset

Original suffix	Replacement Suffix	Correct usage → incorrect usage
-മാരെ [-māre]	-കളെ [-kaLe]	കൂട്ടുകാരന്മാരെ [kūṭṭukāranmāre] (friends) → കൂട്ടുകാരങ്കളെ [kūṭṭukāraṅkaḷe]
-യുടെ [-yuTe]	-ന്റെ [-inte]	കൂട്ടുകാരിയുടെ [kūṭṭukāriyuṭe] (friend’s) → കൂട്ടുകാരിന്റെ [kūṭṭukārinte]
-ന്റെ [-inte]	-നുടെ [nuTe]	കൂട്ടുകാരന്റെ [kūṭṭukārante] (friend’s) → കൂട്ടുകാരനുടെ [kūṭṭukāranuṭe]
-ഇൽ [-il]	-കിൽ [-kil]	കൂട്ടുകാരനിൽ [kūṭṭukāranil] (in friend) → കൂട്ടുകാരങ്കിൽ [kūṭṭukāraṅkil]
-രുടെ [-RuTe]	-ന്റെ [-inte]	കൂട്ടുകാരുടെ [kūṭṭukāruṭe] (of friends) → കൂട്ടുകാരിന്റെ [kūṭṭukārinte]

3.2. Text Graph Convolutional Network (TextGCN)

A convolutional graph network generates embedding vectors based on the properties of neighborhood nodes on a graph [38]. Seq2Seq models and CNN models used for language processing tasks better represent the semantic and syntactic information of local consecutive word sequences. TextGCN [30] models a heterogenous graph from the corpus and uses graph convolutional networks to train the classifier. The graph generated uses words and documents as nodes, and the word co-occurrence matrix creates edges between two-word nodes. The word frequency and the word's document frequency are used to build an edge between a word node and a document node. The number of words gives the total number of nodes in the vocabulary and the number of documents in the corpus.

The word-word edge weights are determined using the pointwise mutual information (PMI) of words. The term frequency-inverse document frequency (TF-IDF) between words and documents forms the word-document edge weights. These global word co-occurrence statistics is collected using a fixed-size sliding window.

Thus, the adjacency matrix of the graph is defined as

$$A_{ij} = \begin{cases} PMI(i,j) & i,j \text{ are words}. PMI(i,j) > 0 \\ TF-IDF_{ij} & i \text{ is document}, j \text{ is word} \\ 1 & i = j \\ 0 & otherwise \end{cases} \quad (1)$$

The PMI value is given by

$$PMI(i,j) = \frac{p(i,j)}{p(i)p(j)} \quad (2)$$

The probability *p(i,j)* is the probability of a word pair *(i,j)* occurring in a sliding window, and *p(i)* is the probability of a word *i* occurring in a sliding window. A positive PMI indicates a high semantic correlation between words as opposed to a negative PMI. So positive PMI is used for obtaining the features, which can be seen in Equation 1. The text classification problem can now be modelled as a node classification problem. The graph generated is given as an input to a two-layered GCN [38], and the convoluted output is given by Equation 3,

$$Z = softmax(\tilde{A}\, ReLu(\tilde{A}\, XW_0W)W_1) \quad (3)$$

where

$$\tilde{A} = D^{-\frac{1}{2}}AD^{-\frac{1}{2}} \quad (4)$$

$\tilde{A}$ is the normalized symmetric adjacency matrix, D is the degree matrix of the graph, W_0 is the weight of the first layer of GCN and W_1 is the weight of the second GCN layer. The input feature matrix given by X is a one-hot encoding of each graph node. The output is obtained using a SoftMax classifier with a cross-entropy loss function.

TextGCN records document-word and global word-word relationships. New features are calculated as the weighted sum of itself and its second-order neighbors. In all the evaluated datasets, Text GCN performs better than all baseline models [31].

3.3. Improved TextGCN

In this work, an improved TextGCN is used to train the Malayalam grammar checker. The adjacency matrix for constructing the graph is obtained by calculating the PMI, BM25 (Best Match 25) and cosine similarity measure of word vectors.

BM25 [39] is an upgrade of TF-IDF where term frequency (TF) and inverse document frequency (IDF) components are refined. TF is refined to become responsive to term saturation and document length. Term frequency in BM25 is calculated using Equation 5.

$$\boldsymbol{TF^{BM}} = \frac{\boldsymbol{TF}}{\boldsymbol{TF+(k*(1-b+b*\frac{dl}{avdl})}} \tag{5}$$

Where k is the parameter controlling the term saturation curve, and b controls the importance of document length. The values of k and b are set to the default values of 1.2 and 0.75, respectively. The term *dl* is the document length, and *avdl* is the average document length.

The probabilistic IDF drops sharply for highly frequent terms. The IDF value is negative for words appearing in more than half of the corpus. In order to prevent negative values, BM25 adds a 1 to the IDF calculation. Thus, in BM25 IDF value is given by Equation 6.

$$IDF^{BM} = \log\left(\frac{N-DF+.5}{DF+.5} + 1\right) \tag{6}$$

where N is the length of the document and DF is the word document frequency. BM25 takes term frequency saturation and document length into account and removes negative values for words which occur in more than half the documents in the corpus.

Cosine similarity [33] between word vectors is also taken as a feature while constructing the adjacency matrix. Cosine similarity expresses the similarity between two different texts. For calculating the cosine similarity, construct a word vector map for every word in the corpus. Cosine similarity between two vectors, A and B, is then calculated as

$$Similarity = \frac{A.B}{\|A\|\|B\|} \tag{7}$$

If the similarity measure is more than 0.95, add the similarity value to the adjacency matrix resulting in an edge between most similar terms. Thus, the adjacency matrix for improved TextGCN is given by

$$A_{ij} = \begin{cases} PMI(i,j) & i,j\ are\ words.\ PMI(i,j) > 0 \\ TF_{ij}^{BM} * IDF_{ij}^{BM} & i\ is\ document, j\ is\ word \\ Similarity & If\ \ Similarity > 0.95 \\ 1 & i = j \\ 0 & otherwise \end{cases} \tag{8}$$

In the improved version of TextGCN, the mish activation function [34] is used instead of ReLu. The Mish activation function is given by

$$f(x) = xtanh\big(softplus(x)\big) = xtanh(\ln(1+e^x)) \tag{9}$$

Mish activation function is continuously differentiable with infinite order, self-regularized, non-monotonic and self-gated. It is unbounded above and bounded below. Compared to ReLU, Mish [34] offers significantly higher accuracy, overall lower loss, and a smoother and easy-to-optimize loss landscape.

3.4. Experimental Setup

A labeled training set of 200k sentences of Malayalam text was used for performing the grammar-checking task. The input sentences were pre-processed by removing unwanted symbols and punctuation. The corpus was then tokenized and padded. The maximum number of words in a sentence was set to 25, and the text length was restricted to 100.

The dataset was split into training and validation sets with a validation split of 0.2. A comparison of the Malayalam grammar checker using improved TextGCN and the baseline models is made. The baseline models used were TextGCN, LSTM, Stacked LSTM, BiLSTM, CNNLSTM and CNNBiLSTM. The parameters used for training these models are given in Table 4.

Table 4. Parameters used for training the baseline models and improved TextGCN

Model	Loss function	Activation function
LSTM, BiLSTM, Stacked LSTM	Binary Cross Entropy	ReLu
CNNLSTM, CNNBiLSTM	Binary Cross Entropy	ReLu
TextGCN	Categorical Cross Entropy	ReLu
Improved TextGCN	Categorical Cross Entropy	Mish

TextGCN consists of two layers of graph convolutional network and uses a sliding window of size 20 while calculating the adjacency matrix. The embedding dimension of 300 is used for TextGCN and improved TextGCN. LSTM was also trained using pretrained FastText embeddings of dimension 300. The FastText pretrained Malayalam embedding was used because an evaluation of various word embeddings for the Malayalam corpus gave better results for FastText [35].

The pretrained embeddings were obtained from a Malayalam corpus of 3.8 million unique words. LSTM, BiLSTM and Stacked LSTM were also trained with a dropout value of 0.2 and without dropout. CNNLSTM and CNNBiLSTM were trained using a kernel size 3, filter size 128 and a max-pooling layer.

The test dataset comprised 500 unseen sentences collected from language learners and the CoLA [5] corpus. The evaluation metrics used for the classification task are accuracy, the weighted average of precision, recall and F-score.

4. Results and Discussion

The trained model is tested on unseen test data of 500 sentences. The result obtained for each model is given in Table 5. Grammar checkers using improved TextGCN gave the best results for the Malayalam grammar-checking task. An accuracy of 90.41%was obtained using improved TextGCN, while TextGCN gave an accuracy of 87.67%. The model's training and validation accuracies are 96.67 and 96.32%, respectively.

Table 5. Summary of the results obtained for various models

Model	Testing Accuracy	Precision	Recall	F1- Score	Training Accuracy	Validation Accuracy
LSTM	55.87%	28.00%	50.00%	36.00%	96.65%	90.69%
Stacked LSTM	56.00%	28.00%	50.00%	36.00%	96.50%	90.97%
$LSTM_{Droput0.2}$	56.00%	28.10%	50.30%	36.05%	96.68%	90.60%
Stacked $LSTM_{Droput0.2}$	56.00%	28.67%	50.87%	36.67%	96.86%	95.39%
LSTM with pre-trained embeddings	56.50%	29.00%	50.9%	36.94%	95.20%	93.10%
BiLSTM	57.12%	29.10%	51.20%	37.10%	98.51%	94.64%
Stacked BiLSTM	57.82%	29.80%	51.90%	37.86%	98.90%	95.10%
$BiLSTM_{Droput0.2}$	57.60%	29.50%	51.70%	37.56%	96.88%	93.97%
BiLSTM with pre-trained embeddings	57.80%	29.80%	52.10%	37.91%	95.6%	93.40%
CNNLSTM	56.00%	28.00%	50.00%	36.00%	98.6%	93.94%
CNNBiLSTM	56.00%	28.00%	50.00%	36.00%	95.04%	94.90%
TextGCN	87.67%	99.30%	87.67%	92.91%	97.03%	96.20%
Improved TextGCN	90.41%	99.28%	90.42%	94.45%	96.67%	95.49%

Table 6. Test Sentences and classification outcomes

Sentence	Classification outcome
പനി തുടങ്ങിയിട്ട് ഏതാണ്ട് രണ്ടാഴ്ച്ചയായി. [pani tuṭaṅṅiyiṭṭ ētāṇṭ raṇṭāḻccayāyi.] (It's been almost two weeks since the fever started.)	True Positive
എല്ലാ ശനിയാഴ്ചതോറും ക്ലാസുണ്ട്. [ellā śaniyāḻcatēāṟuṁ klāsuṇṭ.] (There is a class every Saturday.)	True Negative: Analogous word error
ബഹിരാകാശവാഹനം ഭൂമിയിനെ ചുറ്റുന്നു [bahirākāśavāhanaṁ bhūmiyine cuṟṟunnu] (A spacecraft orbits the Earth)	False Positive: Incorrect suffix used for ഭൂമി [bhūmi] (earth)
ഞാൻ അദ്ദേഹത്തിന്റെ മൂന്ന് പുസ്തകങ്ങൾ വായിച്ചു. [ñān addēhattinṟe mūnn pustakaṅṅaḷ vāyiccu] (I have read three of his books)	False Negative: പുസ്തകം [pustakaṁ] (book) is not a collective noun. So, it is not a numeral-plural error.

Sequence to Sequence networks like LSTM and BiLSTM gave poor results. These sequence-to-sequence networks could not correctly model the relation between words. As a result, when unseen data was received, it could not perform the classification accuracy. A larger dataset for training the sequence-to-sequence network might improve the accuracy, as this will add more terms to the vocabulary. Including pretrained embeddings obtained by training a larger corpus did not improve the results. It is because the pretrained embeddings were generated using grammatically correct sentences. The testing accuracy was only about 55%, even though all the sequence-to-sequence networks displayed a validation accuracy of about 90%.

It was seen that the true negative values were less than that of true positives and false positive values were less than that of false negatives. Table 6 lists some of the test sentences along with the classification outcomes. Conjugational errors were a substantial contributor to the false positives. Although sentences with conjugation errors appear grammatically correct, the placement of related words must be conjugated. False negatives were primarily the result of numeral-plural errors and conjugational errors. The training dataset was unable to represent all the collective nouns. As a result, even though the numeral does not modify a collective noun, sentences with numerals and plurals are regarded as erroneous statements. Precision values for TextGCN and Improved

TextGCN were quite similar. However, upgraded TextGCN demonstrated a recall improvement of around 3% over conventional TextGCN. A comparison of the evaluation metrics for conventional TextGCN and improved TextGCN is given in Figure 3.

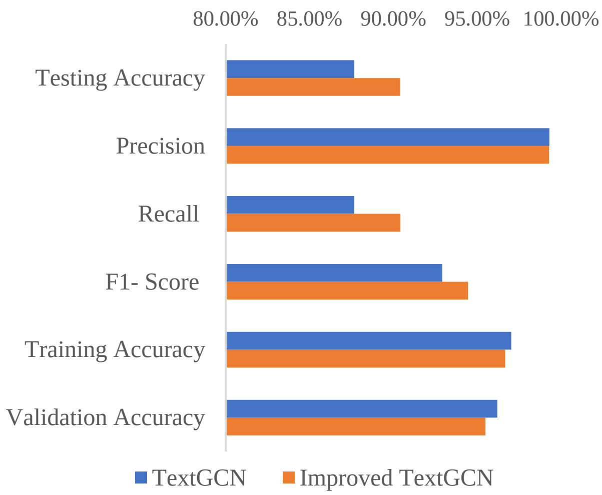

Fig. 3 Comparison of conventional TextGCN and Improved TextGCN

The accuracy and overall loss have been lowered by the adoption of BM25, cosine similarity measure, and mish activation function in enhanced TextGCN as opposed to TF-IDF and ReLu activation function in conventional TextGCN. During testing, the proposed model was able to lower the number of false negatives.

Test data contained sentences with conjugational errors, analogous word errors, numerals and plural errors. The training data set did not adequately represent these errors because the corpus was primarily collected from books and public archives. A major chunk of these sentences was grammatically correct. Errors artificially introduced could not emulate conjugational and analogous word errors. So, a dataset with a larger representation of various error categories will further improve the accuracy of this model.

5. Conclusion

In this paper, we evaluated the grammatical correctness of Malayalam text. This work is the first work done in Malayalam for creating a grammar checker. Being a low-resource language, the progress made in various NLP tasks in Malayalam is very low. We used modified TextGCN for creating the classification model. This grammar checker achieved an accuracy of 90.41% on unseen test data. This is a significant improvement over the state-of-the-art TextGCN technique in terms of accuracy. TextGCN achieved an accuracy of only 87.67% for the test dataset. Given the lack of resources and the fact that this is the first study in this field, this is an impressive outcome. Expanding the dataset size by including more incorrect sentences of various error categories can further generalize the proposed model. Using hybrid models for classification tasks will improve the accuracy of the Malayalam grammar checker. The Malayalam grammar checker can be further developed into a grammar correction system by creating a multiclass classifier for each error category.

References

[1] C. Leacock et al., *Automated Grammatical Error Detection for Language Learners*, Second Edition, Synthesis Lectures in Human Language Technologies, vol. 7, pp. 1-185, 2014. *Crossref*, https://doi.org/10.2200/S00562ED1V01Y201401HLT025

[2] Lionel Clément, Kim Gerdes, and Renaud Marlet, "A Grammar Correction Algorithm: Deep Parsing And Minimal Corrections for a Grammar Checker," *Series Lecture Notes Computer Science*, pp. 47-63, 2011. *Crossref*, https://doi.org/10.1007/978-3-642-20169-1_4

[3] G. E. Heidorn et al., "Epistle Text-Critiquing System," *IBM Systems Journal*, vol. 21, no. 3, pp. 305-327, 1982. *Crossref*, https://doi.org/10.1147/sj.213.0305

[4] Anna Sågvall Hein, "A Chart-Based Framework for Grammar Checking, Initial Studies," *Proceedings of the 23rd Nordic Conference on Computational Linguistics*, 1998.

[5] Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman, "Neural Network Acceptability Judgments," *arxiv prepr. arxiv1805.12471*, 2018. *Crossref*, https://doi.org/10.48550/arXiv.1805.12471

[6] Madhvi Soni, and Jitendra Singh Thakur, "A Systematic Review of Automated Grammar Checking in English Language," *arxiv1804.00540*, 2018. *Crossref*, https://doi.org/10.48550/arXiv.1804.00540

[7] N. Macdonald et al., "The Writer's Workbench: Computer Aids for Text Analysis," *IEEE Transactions on Communications*, vol. 30, no. 1, pp. 105-110, 1982. *Crossref*, https://doi.org/10.1109/TCOM.1982.1095380

[8] Daniel Dahlmeier, and Hwee Tou Ng, "A Beam-Search Decoder for Grammatical Error Correction," *Proceedings of Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2012*, pp. 568-578, 2012.

[9] Zheng Yuan, and Mariano Felice, "Constrained Grammatical Error Correction using Statistical Machine Translation," *in Conference on Computational Natural Language Learning 2013*, pp. 52-61, 2013.

[10] Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme, "Grammatical Error Correction with Neural Reinforcement Learning," *Proceedings of IJCNLP'17*, vol. 2, pp. 366-372, 2017.

[11] Zhu Kaili et al., "A Simple but Effective Classification Model for Grammatical Error Correction," *arxiv.1807.00488*, 2018. *Crossref*, https://doi.org/10.48550/arXiv.1807.00488

[12] Tom Young et al., “Recent Trends in Deep Learning Based Natural Language Processing,” *IEEE Computational Intelligence Magazine,* vol. 13, no. 3, pp. 55-75, 2018. *Crossref,* https://doi.org/10.1109/MCI.2018.2840738

[13] Hailan Kuang et al., “A Chinese Grammatical Error Correction Method Based on Iterative Training and Sequence Tagging,” *Applied Sciences*, vol. 12, no. 9, 2022. *Crossref,* https://doi.org/10.3390/app12094364

[14] Nawei Zhong, Xiaoge Li, and Long Qin, “Hybrid Chinese Grammar Error Checking Model Based on Transformer,” *Proceedings of AIPR 2021*, pp. 574-579, 2021. *Crossref,* https://doi.org/10.1145/3488933.3489034

[15] Fabrizio Gotti et al., “Reducing Overdetections in a French Symbolic Grammar Checker by Classification,” *Computational Linguistics and Intelligent Text Processing*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 390-401, 2011. *Crossref,* https://doi.org/10.1007/978-3-642-19437-5_32

[16] Nora Madi, and Hend Al-Khalifa, “Error Detection for Arabic Text Using Neural Sequence Labeling,” *Applied Sciences,* vol. 10, no. 15, p. 5279, 2020. *Crossref,* https://doi.org/10.3390/app10155279

[17] Sanjay Kumar, Sandhya Umrao, "Extraction of Syntactically Similar Sentences from Huge Corpus for Language Research," *SSRG International Journal of Computer Science and Engineering,* vol. 5, no. 8, pp. 1-5, 2018. *Crossref,* https://doi.org/10.14445/23488387/IJCSE-V5I8P101

[18] Mandeep Singh Gill, and Gurpreet Singh Lehal, “A Grammar Checking System for Punjabi,” *Proceedings of Coling 2008,* pp. 149-152, 2008.

[19] Vikas Verma, and S. K. Sharma, “Critical Analysis of Existing Punjabi Grammar Checker and a Proposed Hybrid Framework Involving Machine Learning and Rule-Base Criteria,” *ACM Transactions on Asian and Low-Resource Language Information Processing,* vol. 21, 2022. *Crossref,* https://doi.org/10.1145/3514237

[20] R. Sankaravelayuthan, “Spell and Grammar Checker for Tamil,” 2015. *Crossref,* https://doi.org/10.13140/RG.2.1.3700.6803

[21] B. Kundu, S. Chakraborti, and S. Choudhury, “NLG Approach for Bangla Grammatical Error Correction,” *International Conference on Natural Language Processing - 2011,* 2011.

[22] Md. Jahangir Alam, Naushad UzZaman, and Mumit Khan, “N-gram based Statistical Grammar Checker for Bangla and English,” *International Conference on Convergence Information Technology,* pp. 3-6, 2006.

[23] Ankita Nohria, and Harkiran Kaur, "Evaluation of Parsing Techniques in Natural Language Processing," *International Journal of Computer Trends and Technology*, vol. 60, no. 1, pp. 31-34, 2018. *Crossref,* https://doi.org/10.14445/22312803/IJCTT-V60P104

[24] K. Mohanan, “*Grammatical Relations and Anaphora in Malayalam*,” MIT Working Papers in Linguistics, vol. 4, 1981.

[25] T.C.Kumari, R.E Asher, “*Language in Society*,” Malayalam (Descriptive Grammars) London and New York: Routledge, 1997. *Crossref,* https://doi.org/10.1017/s004740459922307x

[26] Haowen Jiang, “*Malayalam: A Grammatical Sketch and A Text*,” 2010.

[27] Joseph Peet, “*A Grammar of the Malayalam Language*,” 2008. *Crossref,* https://doi.org/10.31826/9781463214937

[28] Jared Lichtarge et al., “Corpora Generation for Grammatical Error Correction,” *Proceedings of NAACL HLT 2019,* pp. 3291-3301, 2019.

[29] Uthkarsha sagar, "A Broad Survey of Natural Language Processing," *SSRG International Journal of Computer Science and Engineering,* vol. 6, no. 12, pp. 15-18, 2019. *Crossref,* https://doi.org/10.14445/23488387/IJCSE-V6I12P103

[30] Liang Yao, Chengsheng Mao, and Yuan Luo, “Graph Convolutional Networks for Text Classification,” *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 905, pp. 7370-7377, 2019. *Crossref,* https://doi.org/10.1609/aaai.v33i01.33017370

[31] Masoud Malekzadeh et al., “Review of Graph Neural Network in Text Classification,” *IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON),* pp. 84-91, 2021. *Crossref,* https://doi.org/10.1109/UEMCON53757.2021.9666633

[32] Bindhu J S, and Pramod K V, "A Novel Approach for Satellite Image Classification using Optimized Deep Convolutional Neural Network," *International Journal of Engineering Trends and Technology,* vol. 70, no. 6, pp. 349-365, 2022. *Crossref,* https://doi.org/10.14445/22315381/IJETT-V70I6P236

[33] Jiawei Han, Micheline Kamber, and Jian Pei, *2 - Getting to Know Your Data,* Third Edition Data Mining, Morgan Kaufmann, Boston, pp. 39-82, 2012. *Crossref,* https://doi.org/https://doi.org/10.1016/B978-0-12-381479-1.00002-2

[34] Diganta Misra, “Mish: A Self-Regularized Non-Monotonic Neural Activation Function,” *arxiv prepr. arxiv1908.08681,* 2019. *Crossref,* https://doi.org/10.48550/arXiv.1908.08681

[35] Merin Cherian, and Kannan Balakrishnan, “Evaluating Word Embedding Models for Malayalam,” *Proceedings of International Conference on Advances in Security and Computing,* vol. 11, no. 11, pp. 3769-3783, 2021.

[36] Lata Bopche, Gauri Dhopavkar, and Manali Kshirsagar, “Grammar Checking System Using Rule-Based Morphological Process for an Indian Language,” *Communications in Computer and Information Science,* vol. 270, no. 2, pp. 524-531, 2012. *Crossref,* https://doi.org/10.1007/978-3-642-29216-3_57

[37] Caryappa B C, Vishwanath R Hulipalled, and J B Simha, “Kannada Grammar Checker Using LSTM Neural Network,” *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics,* pp. 332-337, 2020. *Crossref,* https://doi.org/10.1109/ICSTCEE49637.2020.9277479

[38] Thomas N. Kipf, and Max Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” *International Conference on Learning Representations*, 2017.

[39] Stephen Robertson, and Hugo Zaragoza, “The Probabilistic Relevance Framework: BM25 and Beyond,” *Foundations and Trends in Information Retrieval*, vol. 3, pp. 333-389, 2009. *Crossref,* https://doi.org/10.1561/1500000019