Exploring Recent NLP Advances for Tamil: Word Vectors and Hybrid Deep Learning Architectures

Archchitha Aravinthan, Charles Eugene

***Abstract*—The advancements of deep learning methods and the availability of large corpora and data sets have led to an exponential increase in the performance of Natural Language Processing (NLP) methods resulting in successful NLP applications for various day-to-day tasks such as Language translation, Voice to text, Grammar checking, Sentiment analysis, etc. These advancements enabled the well-resourced languages to adapt themselves to the digital era while the gap for the low-resource languages widened. This research work explores the suitability of the recent advancements in NLP for Tamil, a low-resource language spoken mainly in South India, Sri Lanka, and Malaysia. From the literature survey, it has been found that there is a lack of comprehensive study on the effect of the recent advancements of NLP for the Tamil text. To fill this gap, this research work analysed the performance of deep learning based text representation and classification approaches namely word embedding, Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN) for Tamil text classification tasks. Different dimensional pre-trained word2Vec and FastText word vectors were built for Tamil and their effectiveness on Text classification was evaluated. The study found that the pre-trained 300-dimensional FastText word vector showed better performance than other pre-trained word vectors for Tamil text classification. Further, in this study, four simple hybrid CNN and Bi-GRU models were proposed for Tamil text classification and their performances were evaluated. The study found that hybrid CNN and Bi-GRU models perform better compared to the classical machine learning models, individual CNN and RNN models, and the Multilingual BERT model. These results confirm that the jointly learned embeddings with different deep learning architectures like CNN and RNN can achieve remarkable results for Tamil text classification, thus ensuring that the deep learning approaches can be successful for NLP on Tamil text.**

***Keywords*— Natural Language Processing, Text classification, Tamil, Word2vec, FastText, CNN, RNN.**

I. Introduction

Recent years have witnessed significant advancements in the field of Natural Language Processing (NLP) owing to the emergence of deep learning methodologies. These advancements have ushered in a new era of NLP, where cutting-edge methodologies now rival human capabilities and find extensive applications across various domains. Among the most prevalent NLP tasks are text classification, machine translation, and question answering. Text classification involves categorizing text into predefined groups, with tasks like sentiment analysis, spam detection, and news categorization being prominent examples.

Correspondence: Archchitha Aravinthan (E-mail: archchithak@gmail.com)
Received: 10-03-2023 Revised: 26-01-2024 Accepted: 22-02-2024

Archchitha Aravinthan and Charles Eugene are from University of Jaffna, Sri Lanka. (archchithak@gmail.com, charles.ey@univ.jfn.ac.lk)

DOI: https://doi.org/10.4038/icter.v17i2.7279

The integration of deep learning approaches has significantly advanced NLP, particularly in applications for highly-resourced languages like English. However, many regional languages face challenges as they still lack developments in language technology. NLP applications in the Tamil language are still in the initial stages compared to other high-resource languages. Tamil and English are two hugely different languages, and they differ in their structure, writing system, vocabulary, pronunciation, and cultural context. Tamil, characterized by its morphological richness and flexible word order, typically adheres to the Subject-Object-Verb (SOV) sequence. In contrast, English follows a fixed Subject–Verb-Object (SVO) word order. In Tamil language different affixes are added to words to convey meaning. In contrast, English uses prepositions and auxiliary verbs to convey meaning. Tamil has a rich vocabulary but many of the words are not used frequently. Most deep learning approaches are developed and tested based on English text, and it is necessary to determine their suitability for Tamil.

The effectiveness of NLP highly depends on the quality of text representation. Contrary to the classical text representation approaches, word embedding can provide precise and well-crafted representation especially suitable for deep learning models to discern patterns and derive meaningful insights from textual data. A large numbers of research studies have reported the effect of word-embedding approaches and deep-learning models for Tamil NLP tasks. However, these studies failed to provide a comprehensive analysis comparing the effectiveness of different word embedding model, in particular their dimensions for Tamil text representation.

Further on, different deep learning models poses various capabilities for text processing. Numerous research works have individually explored the effectiveness of deep learning models, including Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and hybrid models, in the context of Tamil NLP tasks. These analyses were often conducted independently for each model, and in many instances, different datasets were utilized, making it challenging to compare their performances directly. Therefore, this study aims to provide a unified and comprehensive analysis of deep learning and hybrid deep learning models for Tamil NLP tasks, with the goal of identifying suitable approaches for effective text classification.

To address the aforementioned research gaps, the primary objectives of this research work are:

- Provide a comprehensive summary of the advancements achieved in deep learning-based NLP with a specific

focus on research pertaining to NLP applications for the Tamil language.
- Explore the effectiveness of deep learned word embedding-based text representation and the impact of the dimensions of word embedding vectors for Tamil text classification.
- Proposing and analysing the efficacy of hybrid CNN and RNN deep learning models for Tamil text classification.

This research work analyses the performance of word embedding models in terms of its dimension for Tamil text classification and tries to identify a highly performing embedding model. For this purpose, a Tamil text corpus was created with 254633 Tamil articles collected from Wikipedia and other web sources. Further, word embedding models for Word2Vec (cbow and skipgram) and FastText with varying dimensions (100 to 500) were created and evaluated. The study concentrated on the performance of CNN and RNN models for NLP applications by proposing and analysing four hybrid deep learning models for Tamil text classification.

The remaining sections of this paper are organized as follows: Section II summarises the existing text classification methods and related work that achieved remarkable results in similar tasks for Tamil and other languages. Section III provides the proposed methodologies and Section IV presents the experiments and evaluations conducted in the study and the obtained results. Finally, the conclusion and future work are presented in Section V.

II. Background and Related Work

Compared to other computing research domains, Natural Language Processing (NLP) has undergone extensive investigation and proven its successful integration into numerous real-world applications. The most important part of NLP research is the text representation and the classical approaches used methods such as the bag-of-words, n-gram, continuous bag-of-words, and Term Frequency-Inverse Document Frequency (TF-IDF). The classical text representation models such as TF-IDF, Bag of Words, and n-grams struggled to preserve the syntax, semantics, and contextual meaning of text in a numerical representation. The introduction of statistical language models has made it possible to overcome this limitation. These models can predict the next word in a sequence, given the preceding words, and thus capture the context of the text. After Bengio et al. [1, 2] proposed the first deep learning-based distributed representation of words, NLP focused on reducing the computational difficulties with the large vocabulary size.

The advancements in word embedding have facilitated the integration of neural networks into NLP, resulting in notable enhancements. Word embedding involves mapping words or phrases from a vocabulary to vectors of real numbers, effectively capturing semantic relationships and contextual information in natural language processing. Mikolov et al. [3, 4] introduced the word2vec algorithms, namely CBOW and Skip-gram, for learning word embedding. Additionally, FastText, an extension of word2Vec, has emerged, utilizing the technique of breaking words into several n-grams instead of processing individual words [5]. The big tech companies such as Google and Facebook have introduced pre-trained word embedding models, trained on extensive corpora, which can be leveraged for various NLP tasks. This approach is especially advantageous for low-resource languages, facilitating their analysis and processing. The following section (*A.*) gives a comprehensive review of deep-learning approaches for text representation.

Classical approaches of NLP used machine learning models such as Naïve Bayes, hidden Markov models, random forests, conditional random fields, decision trees, k-nearest neighbour, and support vector machines. If we consider the literature of deep learning-based classification, the RNN [6] were dominant in the field of NLP since other classical and CNN models had difficulty in handling the temporal information from the input data. Hochreiter and Schmidhuber [7] introduced the Long Short-Term Memory (LSTM) neural network model to address the challenge of connecting relevant information in RNNs when distant words are considered in a large input text. Schuster et al. [8] proposed bidirectional RNNs to enhance the network's access to input information from both directions. Additionally, Cho et al. [9] introduced the Gated Recurrent Unit (GRU) as a variant of LSTM to mitigate the vanishing gradient problem encountered in traditional RNNs.

The GRU is characterized by its simpler structure compared to LSTM, resulting in faster computation time and lower resource consumption. CNN was initially formulated for computer vision applications. After getting impressive results in computer vision, it is adopted for NLP applications to lower the complexity and to reduce the processing time [10-12].Following that, in recent years, deep learning approaches have gained widespread acceptance in various NLP applications. Section (*B.*) discusses the above significant advancements in detail, particularly in the context of Tamil NLP.

A. Deep learning-based text representation

Research on language models includes a rich history about statistical approaches which have achieved satisfactory performance. After the widespread use of deep learning, NLP approaches turned towards deep learning-based models which can automatically generate and learn multilevel features concurrently. Deep learning-based NLP focuses on modelling as well as engineering features while the traditional machine learning algorithms mainly used time consuming hand-crafted features. Statistical methods that involve learning the joint probability functions of language models have shown promising performances but encountered the curse of dimensionality as a drawback for building efficient models. The curse of dimensionality refers to the challenges and issues that arise when working with high-dimensional data in machine learning. In high-dimensional spaces, data points become increasingly sparse, and the distance between points may lose meaning, making it difficult for algorithms to effectively learn patterns and make accurate predictions. As more features or dimensions are added, the data required for accurate generalization grows exponentially.

Moreover, the computational demands for processing and analysing data in high-dimensional spaces become more intensive. To address these challenge, Bengio et al. [2] introduced a deep learning-based language model emphasizing the distributed representation of words in a

low-dimensional space, a technique commonly known as word embedding. This approach has demonstrated state-of-the-art performance across various NLP tasks [13-16].

Collobert and West [17] introduced the concept of pre-trained word embeddings that made the word embedding method as a convenient tool for many NLP tasks. The first popular word embedding model known as Word2vec [3] is considered as a big turning point in text representation. There are two approaches to build a Word2vec embedding, namely, continuous bag-of-words (CBOW) and skip-gram. Both approaches can represent the text as a high-quality distributed vector. The CBOW model predicts the conditional probability of a word using the distributed representations of surrounding words. On the other hand, the Skip-gram model predicts the surrounding context using the distributed representation of the input word.

GloVE, is an extension of the word2vec model, developed by researchers at Stanford University [18]. It generates word embedding slightly differently than word2vec by means of their co-occurrence statistics. It represents the words using a classical vector space model that contains matrix factorization techniques such as Latent Semantic Analysis.

FastTextis another popular word embedding model, which was released by Facebook in 2016. It generates embedding vectors by breaking down the words into multiple character n-grams and feed them as an input to build the model [5, 19]. This contrasts with word2Vec model, which feeds individual words as input. Compared with other word embedding methods such as GloVe and Word2Vec, FastText has an advantage of producing sub-word embeddings and can be able to generate an embedding for out-of-vocabulary words, i.e.,the words which are not in the corpus used for building the word embeddings. Hence, FastText is a suitable methodwhen the available text corpus is small and encountering out-of-vocabulary words during NLP applications [5].

Building a word embedding model using a large corpus and using these pre-trained word vectors in the NLP tasks is an efficient technique that resulted in remarkable outcomes. Google's Word2Vec, Stanford's GloVe (Common Crawl, Twitter) and Facebook's FastText are some notable pre-trained word vectors. The research works of Kim [20], Camacho-Collados et al. [21], Zhang and Wallace [22] and Liu et al. [23] are some examples that got remarkable results with pre-trained Word2Vec word vectors.

Recent pre-trained language models such as ULMfit [24], ELMo [25], OpenAI[26], BERT [27] and XLM [28], have obtained a large advancement over the state-of-the-art models across many different tasks as they are context sensitive and have transfer learning capabilities.Transfer learning is a machine learning approach where a model trained on one task is leveraged to improve the performance on a different, but related, task, by transferring knowledge gained during the initial training. This helps to enhance the learning efficiency and generalization of the model to new tasks with limited data.The context sensitive models can capture contextual information from the surrounding words, allowing them to understand and generate language based on the context in which a word or phrase appears.

B. Neural Networks for text classification

To deal with text, RNN is preferred in NLP since it can deal with sequential information better than other neural network models. RNNs have demonstrated superior performance in tasks such as unsegmented, connected handwriting recognition, or speech recognition, primarily due to their ability to process variable-length sequences of inputs using internal memory [29]. Even though, if the words considered in a large input text are far apart, it is difficult to connect the relevant information using RNN. To address this issue, Long Short-Term Memory (LSTM) is employed, as it can effectively handle long-term dependencies. Hochreiter and Schmidhuber[7] proposed LSTM in 1997 which has achieved almost all the results that were achieved by RNNs. Models for diverse types of tasks like speech recognition, sentence embedding, trajectory prediction and correlation analysis have adapted LSTM as it has a powerful learning ability and works remarkably well than other neural networks. However, they have a limitation—they can only use input information up to a pre-set future frame.

To increase the availability of the context information, bidirectional RNN was introduced by Schuster et.al.[8]. These networks incorporate two LSTMs, with one handling input in the forward direction and the other in the backward direction. By training both directions simultaneously, BRNNs effectively increase the context available to the algorithm. For example, they can consider not only what words immediately follow a given word but also what precedes it in a sentence. Graves et al. [30] applied bidirectional LSTM to capture long-range context and consequently it was found to be appropriate for generating sequences.

In comparison to a standard recurrent cell, the learning capacity of the LSTM cell is high. It is worth mentioning that the extra parameters in LSTM contribute to its computational complexity. Therefore, Cho et al.[9] introduced the GRU as a variation of the LSTM that intends to solve the vanishing gradient problem, which is a significant issue in the standard RNNs.

Initially, computer vision applications introduced the models based on CNN and achieved impressive results. CNNs consist of layers equipped with convolving filters, which are employed to extract local features. Following the success of CNN in computer vision, NLP adopted CNN for text classification [10, 11]. Yoon Kim [12] applied the CNN for text classification tasks in English language with pre-trained Word2Vec embeddings. Generally, the NLP applications used one or two dimensions with multiple filter widths that slide along the temporal dimension.

After the successes of individual CNN and RNN learning models, hybrid CNN and RNN models were applied to NLP tasks and improved the performance further. Guo et al. [31] proposed a hybrid CNN-RNN attention-based model for text classification. Alharbi et al. [32] applied CNN and Bidirectional LSTM (Bi-LSTM) based model for Arabic sentiment classification task. Zhang et al. [33] considered CNN and LSTM for emotion classification. Ombabi et al. [34] proposed a CNN and LSTM ensemble model with SVM classifier. Jin Liu et al. [35] proposed an Attention-Based BiGRU-CNN network for classifying Chinese questions. Yang Fan et al. [36] introduced a clustering algorithm using neural feedback, which merges Bi-LSTM and CNN with the k-means method.

Senarath Yasaset al. [37] developed a hybrid model for aspect extraction from consumer reviews. Their study

introduced an enhanced CNN architecture specifically designed for aspect extraction, achieving results comparable to existing systems. Moreover, they introduced a fusion of classifiers for aspect extraction, integrating the refined CNN with an SVM leveraging existing manually engineered features. This combined system surpasses standalone systems and exhibits a notable enhancement over prevailing existing systems employing intricate neural architectures.

Goonathilake et al. [38] proposed a hybrid model centered on the CNN and RNN-LSTM models for fake news detection. The study utilizes pre-trained GloVe word embeddings to represent text features. A Conv1D layer and MaxPooling layers are employed in the CNN model to extract higher-level text features. Following this, LSTM units are integrated into the RNN model to detect longer contexts in the text and to capture the relationships or dependencies among word sequences. The resulting hybrid model achieves a remarkable accuracy rate of 92%, surpassing many conventional models currently in use. It maintains an average precision, recall, and F1-score of 91%, indicating its robust performance in identifying fake news.

In recent years, the field of NLP has seen widespread adoption of advanced techniques such as GPT (Generative Pre-trained Transformer) [39], XLNet [40], and Zero-shot Learning[41], all of which have applications in text classification.GPT for text classification entails fine-tuning pre-trained GPT models on labelled data to categorize text into various classes. Leveraging the model's deep understanding of natural language and contextual cues, GPT proves effective across a range of classification tasks. Nonetheless, this process can be computationally intensive, and performance may vary depending on task specifics and available data.

XLNet, another powerful technique for text classification, involves fine-tuning pre-trained XLNet models on labelled data to classify text into different categories. By capturing bidirectional context and long-range dependencies in language, XLNet exhibits effectiveness across various classification tasks. However, fine-tuning XLNet may demand significant computational resources and labelled data.

Zero-shot learning for text classification represents an innovative approach where a model is trained on known categories but evaluated on unseen ones. This method relies on auxiliary information like semantic embeddings or class descriptions to generalize to new categories. It offers a valuable solution for scenarios where labelled data for all categories is lacking, enabling adaptation to new categories without additional data.

C. Tamil text classification

According to a literature survey, there have been few machine learning-based research attempts on Tamil language NLP, particularly in deep learning-based solutions. Below is a summary of the classical machine learning and deep learning-based approaches utilized for feature extraction and text classification in Tamil.

M. NarayanaSwamy et al. [42] conducted a study on the representation and categorization of diverse Indian regional language text documents using text mining methodologies. Their research explored various techniques including the naive Bayes, k-Nearest-Neighbour, and decision tree for text categorization, establishing a corpus in three South Indian languages, including Tamil.

Reshma et al. [43] applied supervised methods for domain classification of Tamil documents. This work used the Dinakaran newspaper dataset from EMILLE/CIIL Corpus [44], which contains Tamil news and articles about cinema, politics, and sports from Dinakaran newspaper. In this study Max-Ent, CRF and SVM algorithms are utilised for domain classification. The research work trained the classifier in the document level and sentence level and achieved more than 90% average accuracy.

Arunselvan et al. [45] developed a sentiment analysis system for movie reviews using machine learning-based classifiers such as Naive Bayes (NB), Logistic Regression (LR), and Support Vector Machine (SVM). This work considered the term frequency of unigrams and bigrams as features. The study found that the SVM classifier with a Radial Basis Kernel achieved the best accuracy of 64.69% using bigram features.

Seshadiri et al. [46] reported a work on predicting sentiments for Tamil movie reviews using SVM, Maximum Entropy classifier (Maxent), Decision tree and NB classifiers. The study found that the SVM performed better than the other classifiers and obtained accuracy of 75.96%.

Phani et al. [47]analysed the sentiment analysis on tweets in three Indian languages using the SAIL dataset, and obtained better results for all three languages. The research work used word n-grams, character n-grams, surface feature and SentiWordNet as features. Here Multinomial Naive Bayes (NB), Logistic Regression (LR), Decision Tree, Random Forest and SVM are used as classifiers. The research reported that the word unigrams feature with NB classifier achievedhighest accuracy for two-classes and character unigram features with RF classification obtainedbest accuracy for three-classes.

Nivedhitha et al. [48] proposed a model to classify the Tamil tweets using a decision-making algorithm. The study used Word2vec to calculate the centroid vectors for positive, negative, and neutral classes. To classify the tweets, the decision-making algorithm computed a distance vector using the sentence vector and centroid vectors. This work reported an accuracy of 70.62%.

Seshadri et al. [49] introduced RNN for the Tamil text classification. The study built a model to classify the tweets into positive, negative, and neutral using the SAIL dataset for Tamil, Hindi, and Bengali languages. The model was developed using RNN that iterated 1000 times to get the results. The research obtained a classification accuracy of 88.23% for Tamil, 72.01% for Hindi and 65.16% for Bengali.

S. Ramraj et al. [50] proposed deep learning-based techniques using pre-trained Word2Vec embeddings with CNN to categorize Tamil news articles. The research applied TF-IDF and pre-trained embeddings for feature representation and CNN, SVM and NB for classification. The study found that CNN with pre-trained embeddings gave better F1 scores compared to TF-IDF features learned with SVM or NB algorithm.

N. Rajkumar et al. [51] presented a deep learning-based model for Tamil document classification. The study used the TF-IDF feature and employed Chi-Squared (CS) and Extra Tree (ET) feature selection methods. For classification, CNN and ANN models were utilized. The study obtained effective classification outcomes using the ET feature selection

method and the CNN model with a maximum accuracy of 90%, precision of 90.57%, recall of 90%, and F-score of 89.89%.

Abhinav Kumara et al. [52] proposed a hate speech detection method for tweets on YouTube comments written in code-mix script. The study explored deep learning and traditional machine learning-based models for classifying offensive and non-offensive sentences. To identify hate speech in code-mixed and script-mixed posts, the research applied support vector machine, Logistic regression, Random Forest, Naive Bayes, LongShort-Term Memory (LSTM) and Convolution Neural Network (CNN) models. The study found that the use of the character N-gram TF-IDF feature is a key factor in identifying offensive social media posts. To Tamil code-mixed text, the character N-gram TF-IDF based Naive Bayes classifier has shown the best result.

Sreelakshmi K et al. [53] proposed three different deep learning architectures for offensive language identification, namely, a hybrid network with a convolution layer, a bidirectional LSTM network layer and a hidden layer, a network containing a Bi-LSTM and another with a Bi-RNN. To overcome the class imbalance problem in the training data, the study applied a cost-sensitive learning approach. The Bi-LSTM model got the best performance for Tamil-English text with 89% of accuracy.

Recently some classification-based tasks in Tamil NLP also applied the transformer-based pre-trained models like multilingual BERT and got better results [54, 55]. Instead of just considering the context word and n-grams like FastText, BERT understands the context. It can be pre-trained for unsupervised tasks such as masked language, and next-sentence prediction. Multilingual BERT is a multilingual form of BERT that is trained on and usable in 102 languages.

[56] employed CNN and BERT alongside a word2vector model for sentiment analysis in Tamil-English mixed text classification. The manuscript delves into the classification models designed for an annotated Tamil-English mixed dataset, highlighting improved performance through preprocessing with the NLTK library for mBERT-based classification. Specifically, data processing for the Positive class is computed for mBERT, Logistic Regression (LR), and CNN models, whereas prediction for all classes is conducted for the H2O library-based solution. Notably, LR demonstrates superior accuracy and F1 score compared to CNN and BERT models.

Recently, some research work on Tamil text have also adopted hybrid approaches. [57] considered the application of diverse models, encompassing traditional machine learning, deep learning, transfer learning, and hybrid deep learning approaches, for the task of Tamil–English code-mixed sentiment analysis. The research developed hybrid deep learning models by combining CNN with LSTM and CNN with Bi-LSTM to effectively capture both local and global features implicit in code-mixed data for sentiment analysis. The performance of these hybrid deep learning models was assessed by comparing them with previous methods. The results indicate that, among the diverse models employed, the hybrid deep learning model, particularly the CNN+BiLSTM model, exhibited superior performance for code-mixed sentiment analysis.

[58] utilized hybrid deep learning methods, including CNN-BiLSTM, CNN-LSTM, and CNN-BiGRU to conduct sentiment analysis in the Tamil language. The outcomes indicate that CNN-BiLSTM exhibited superior performance, achieving the highest accuracy at 80.2%, whereas CNN with LSTM and BiGRU achieved an overall accuracy of 77%.

[59] introduces a novel ensemble-based deep learning framework designed for the detection of Tamil Code-Mixed hate speech in media postings. This framework combines a CNN and a Dense Neural Network (DNN). The CNN incorporates both word-level and character-level features, while the DNN utilizes character-level TF-IDF features. The incorporation of character-level features in this ensemble framework has yielded state-of-the-art performance in the realm of Tamil hate speech detection.

[60] investigated "Tanglish" data, a blend of "Tamil" and "English" text. The proposed Sentiment Analysis technique, based on Ensemble Learning, employs Generative Adversarial Network (GAN) and Self Attention Network (SAN) models with XLM-R multilingual embeddings. The study scrutinizes the performance of different models across various tweet categories, highlighting the effectiveness of the proposed approach in achieving more accurate classification of neutral tweets compared to existing methods.

[61] employed a range of machine learning and deep learning algorithms for hate speech identification, and their performances were systematically compared, culminating in the creation and evaluation of ensemble models. The first ensemble model was a fusion of SVM, LR, and NB, while the second ensemble hinged on SVM and LR. The study considered six selected algorithms: SVM, LR, NB, Bi-LSTM, Multi-layer Perceptron (MLP), and Multilingual BERT. Despite four algorithms, including both ensemble models, yielding identical accuracy, a nuanced evaluation based on F1 score demonstrated the superior performance of ensemble model 2 over other classifiers.

[62] proposed a Multi-Stage Deep Learning Architecture for cross-lingual sentiment analysis on Tamil text. Additionally, an ANOVA statistical comparison showed enhancements over various models, such as mT5, XLM, mBERT, ULMFiT, BiLSTM, LSTM with Attention, and ALBERT. The Multi-Stage Deep Learning Architecture exhibits robust performance across different domains, as evidenced by achieving an accuracy of 0.8551 on the Tamil News Classification dataset and 0.8624 on the Thirukkural dataset, surpassing baseline models.

[63] introduced a novel hybrid deep learning model architecture aimed at detecting fake news in low-resourced Dravidian languages. This study integrates dilated temporal convolutional neural networks (DTCN), BiLSTM, and a contextualized attention mechanism (CAM) to tackle the challenge. Moreover, MuRIL contextualized word embeddings are utilized for text representation. The experimental results showcase that the proposed model outperforms existing works on the Dravidian_Fake dataset.

D. Findings from related research

In the recent past, the deep learning approaches has made a significant improvement in NLP. Both deep learning-based feature extraction and classification are found to improve the performance of the models for a range of NLP tasks. In the near past Word2Vec and FastText word embedding models contributed to excellent results in many NLP applications. The performance of these word embedding models depends on the size of the text corpus. Using the pre-trained word

embedding vectors is a preferable technique for low resource languages like Tamil due to the absence and or limitations of large corpora. For classification, the method of jointly learning pre-trained embeddings with deep learning models showed better results in NLP tasks [12].

In the existing work, CNN and RNN showed state-of-the-art results in many text classification tasks, both having their own advantages. CNN is computationally less complex than RNN because it learns by batch while RNNs train sequentially. RNN can process temporal information. However, it cannot be parallelizable like CNN. The effectiveness of parallelization can be considered as the main benefit of CNN for NLP and CNN can be modelled with widened receptive fields, which can capture a wider context using dilated convolutions [64].

CNN can be applied to accelerate the RNN model [65] since it is most compatible with RNN [66]. In recurrent networks, GRU utilized less optimized parameters than LSTM even though it showed comparable performance with LSTM [67]. Compared with GRU, the Bi-GRU models showed improved performance for the NLP tasks [68-70].

For Tamil NLP, the classical classification approaches like LR, SVM and NB got impressive results with BOW and TF-IDF features. After the widespread use of deep learning methods, Tamil NLP adapts the CNN and RNN based learning method with word embedding representation [49-53]. To date, research in Tamil NLP has predominantly focused on hybrid models comprising a recurrent unit followed by CNN. However, none of these studies has explored alternative hybrid model structures for Tamil NLP.

Hybrid deep learning approaches have demonstrated improved results in text classification tasks across various languages [31-38].The deep learning-based approaches gave better performance than existing methods in Tamil NLP [71-75]. Recently, there has been a trend in Tamil research to adopt hybrid approaches as well [57-59].

III. Methodology

This research work focuses on the following three:

- **Performance analysis of pre-trained word vectors for Tamil NLP:** Analysing the performance of different pre-trained word embeddings for Tamil, namely Word2vec (skip-gram), Word2vec (CBOW) and FastText for different dimensions.
- **Developing Deep learning-based classification models:** Analysing the performance of CNN and RNN models and developing models with improved performance for Tamil text classification.
- **Analysing the performance of the proposed models:** the performance of the proposed models using various Tamil text classification datasets.

The operation flow of text classification is shown in Fig.1: Process diagram for Tamil text classification and the intermediate tasks are described below:

- **Data pre-processing:** The unnecessary punctuation marks, symbols and web links are removed from the text to reduce the noise. After this, the text data is tokenized. Tokenization splits an input text as a sequence of tokens which will be used in the subsequent stages of text classification. Here white space is used to split the tokens in a text.
- **Feature extraction:** Feature extraction focuses on converting high-dimensional text data into low-dimensional numerical representation. The study considered the conventional features to find the baseline performance and the word embedding features for deep learning models.
- **Classification:** The task selected to evaluate the performance of the proposed models is the classification of Tamil text. Traditional classification algorithms such as NB, LR and SVM are used to find the baseline performance. CNN and RNN are used to develop deep learning models.

A. Conventional approaches

The study considered classical machine learning approaches for comparing the performance of the proposed deep learning models. Therefore, the classical feature extraction and classification techniques that showed better performance were identified through the literature study and applied to Tamil text classification.

1) Feature extraction:

BOW is a general technique to represent text features in NLP. To build a baseline model, the research work applied the BOW approach for word representation with the conventional classifiers for Tamil NLP tasks. BOW is a technique that develops a dictionary of unique words and counts the occurrence of words in each document. It represents particular text content as a high-dimensional sparse vector where each dimension holds the term frequency of a word [76].

In addition, the study considered the TF-IDF [77] as

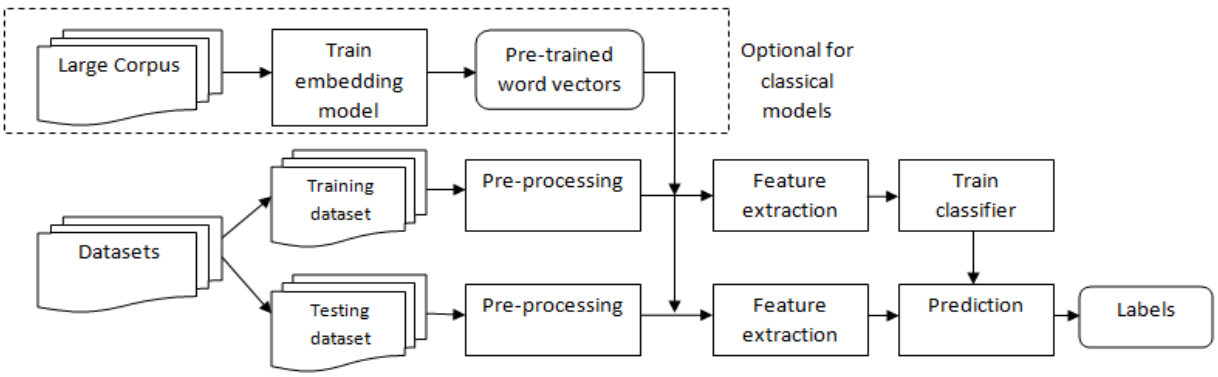

Fig.1: Process diagram for Tamil text classification

another representation method that can be considered as an extension of BOW representation. This approach captures the importance of the words in a document compared to frequently-used words in all the documents. This importance metric is known as weight. For a word *i* in document *j*, the weight of the word $W_{i,j}$ can be calculated using the equation below.

$$W_{i,j} = tf_{i,j} * log\left(\frac{N}{df_i}\right)$$

Where, $tf_{i,j}$is the frequency of the word *i* in document *j*,df_i is the number of documents containing the word *I* and N is the number of documents in the corpus.

2) Classification:

SVM is a successful machine learning model for classification, introduced by [78]. It got state-of-art results for English text classification tasks [79, 80] and remarkable results for Tamil NLP tasks [45-47]. LR is another common machine learning-based probabilistic and statistical model which has performed well in text classification [81-83]. J. Zhang et al. [83] and P. Komarek et al. [84] showed that LR can perform nearly equal to SVM.NB classifier is a simple probabilistic machine learning model that is commonly used for classification tasks including text classification[85-88]. For a given unseen document, NB is capable to compute the probabilities of classes using the joint probabilities and it assumes that the features are independent. NB classifier has been applied for various Tamil NLP tasks [42, 45-47].

B. Deep learning-based feature extraction and generation

This study adopted the Word2vec and FastText word embedding models which can be considered as milestones in text representation for machine learning. The Word2vec model can be built using two approaches namely CBOW and skip-gram. This study used both approaches and analysed their performances.

The objective of word embedding is to encapsulate each word from a text corpus in a compact, low-dimensional space, preserving both syntactic and semantic nuances, while also capturing the word's contextual relationships within the corpus. The Word2vec embedding using skip-gram approach is built by training the model to find the probability of the context$\mathrm{w(i)}$corresponding to the probability of the current word $\mathrm{w(t)}$. It can be represented as $\mathrm{P\big(w(i)|w(t)\big)}$where, $\mathrm{t - c \le i \le t + c}$is the range of i.

After training the neural network, each word is represented as a vector that maps its contextual position. Subsequently, the word vector v for each word in the text can be derived. If we consider a sequence of words $\mathrm{w(1), w(2), w(3), \dots\dots, w(n)}$, the ultimate goal of the model is to maximize the function given below.

$$\mathrm{F} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i=1}}^{\mathrm{n}} \sum_{\mathrm{-c \le i \le c, i \ne 0}} \log \mathrm{P\big(w(t+i)|w(t)\big)}$$

Where, *c* is the window length which considers the first *c* words before the current word $\mathrm{w(t)}$and next *c* words after the current word $\mathrm{w(t)}$.

The CBOW approach is almost like skip-gram approach. In this approach, the embedding can be built by learning to predict the word $\mathrm{w(t)}$ for a givencontext$w(t-2), w(t-1), w(t+1), w(t+2)$. Likewise for a given word $\mathrm{w(t)}$the skip-gram model can predict the context $w(t-2), w(t-1), w(t+1), w(t+2)$.

The Word2vec embedding can be created by training the model using the text from the data set of a downstream task or by using an independent collection of text. The pre-trained word embedding, which is built using an independent corpus is a better option for the low-resourced Tamil language since it has a limited amount of text in the data sets for downstream applications such as sentiment analysis, question answering, spam detection, author identification etc.

To build pre-trained word vectors, a Tamil corpus was created containing 254633 Tamil articles collected from Wikipedia and other web sources. This corpus consists of 3018902 unique tokens. Thus, the study used the Skip-Gram and CBOW approaches in Word2vec to build the pre-trained word embedding for the Tamil language. In addition, the FastText model was also applied to create pre-trained word embedding. Further word embedding with dimensions 50, 100, 200, 300 and 400 were created for Skip-gram, CBOW and FastText. The performance of these 15 pre-trained word vectors were compared by task-based performance analysis to identify the suitable dimension for Tamil text. The best performing model was selected to build the jointly learning neural network model for classification.

C. Hybrid CNN and RNN models

Through the analysis of existing research work, it can be stated that the neural network models which can learn the embedding as well as do the downstream task together provides better performance instead of using a word embedding as a separate feature extraction and representation for a machine learning model [20]. Hybrid CNN and RNN models have shown better performance for NLP tasks in other languages. Further, it can be concluded from the existing work that Bi-GRU is an efficient recurrent neural network than others. This study, proposes and compares four hybrid models using CNN and Bi-GRU.

1) Parallel CNN and Bi-GRU model:

CNNs perform very well for extracting local and position-invariant features while RNNs are good to keep the long-term semantic dependencies instead of local features [89]. To merge both sets of features, this study proposes a parallel CNN and Bi-GRU model, drawing inspiration from the success of the parallel CNN and RNN approach observed in other tasks [90].The local and position-invariant features are extracted using CNN layer with max-pooling layer while the sequential features are extracted using Bi-GRU layer. Finally, the output feature vector of the max-pooling layer and the output feature vector of the Bi-GRU layer are concatenated and passed through the dense layer. This hybrid model can learn the embedding features as well as learn to classify.

2) CNN + Bi-GRU model:

In prior research, studies referenced as [31-34, 57, 58] employed a sequential approach with RNN followed by CNN, achieving successful outcomes. Therefore, we proposed a Bi-GRU followed by CNN model as our second approach. In this model, the output text feature vector of the convolution layer with max-pooling is given as input to the Bi-GRU. Instead of using individual CNN model, the model

can capture sequential information while extracting local features. CNN model is known to be much faster than the Bi-GRU model. Compared with the parallel CNN and Bi-GRU model, proposed CNN+Bi-GRU model has a simplified architecture and expected to have reduced time complexity.

proposed models are illustratedinFig.3: Architecture of all the proposed models. The detailed functionalities of each layer of the models are as follows:

D. Functionality of the layers in the proposed models:

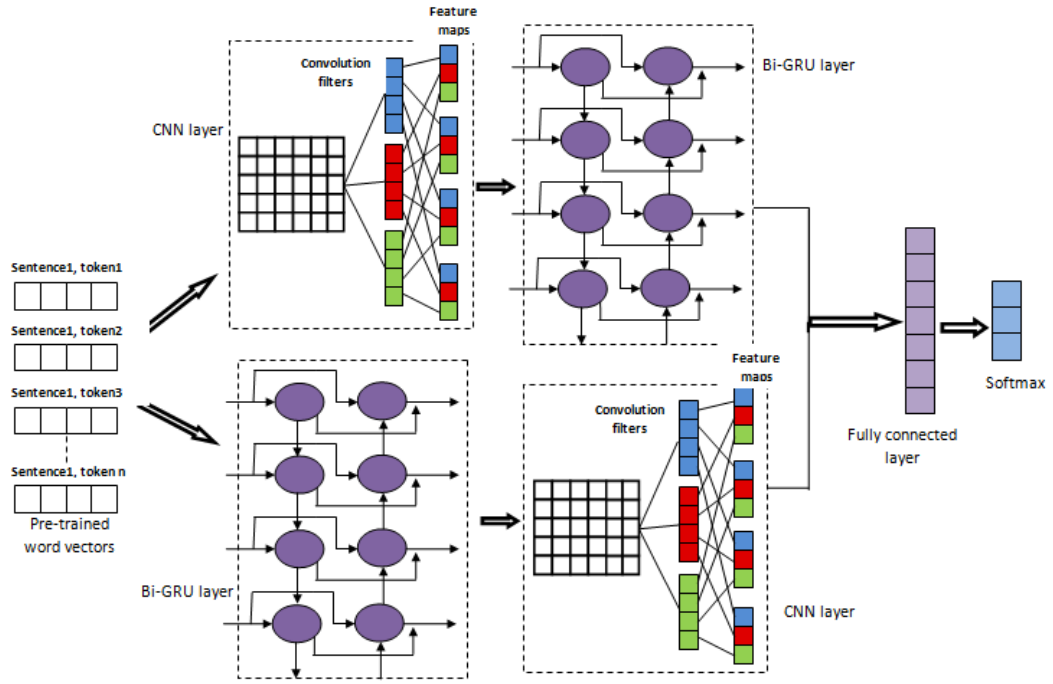

Fig.2:Proposed ensemble model

3) Bi-GRU + CNN model:

Studies referenced as [35, 36] applied a CNN followed by RNN approach for tasks in other languages and achieved remarkable results. The third model considered in this study applied the approach, which is similar as CNN + Bi-GRU, but in reverse order. For this model, the output text feature vector of the Bi-GRU model is fed as input to CNN. CNN can capture the semantics of texts efficiently when an appropriate window size of the filters is used[11-12]. CNN may fail to capture essential information when an inappropriate small window size used.

It faces an explosion of parameter space issue when a large window size used. This difficulty to decide the appropriate window size leads to an increased resource requirement in model training [13].

The proposed model tries to reduce this difficulty by adding Bi-GRU before CNN since it can learn long-term dependencies. Also, the proposed model ensures text integrity and reduces time complexity.

4) Proposed ensemble CNN and Bi-GRU model:

CNN and Bi-GRU models can capture distinctive features. This research proposes an ensemble of CNN and Bi-GRU model as the fourth model. This model makes the decision based on the output from CNN+Bi-GRU and Bi-GRU+CNN models. The architecture of this ensemble model is given in Fig.2:Proposed ensemble model. The architectures of all the

- Embedding layer

The functionality of the embedding layer is to convert input from a sparse representation into a distributed or dense representation. The i^{th} word in a text can be represented as a d dimensional vector $w_i \in \hat{A}^d$, where$d$ is the dimension of the word embedding. A sentence or text portion having n words can be represented by an embedding vector$W \in \hat{A}^{n\times d}$. Since each text may have different number of words, they are zero padded to have a fixed number of tokens. The embedding vector for a text is the concatenation of embedding vectors for the tokens in that text. Hence $W \in \hat{A}^{n\times d} = w1 \oplus w2 \oplus \dots\dots \oplus wn$. Here $\oplus$ is the concatenation operator. Let us use the notation $w_{i:i+j}$ to represent the concatenation of vectors$w_i, w_{i+1}, \dots\dots, w_{i+j}$.

- Convolution layer

Convolution layer performs convolution operation on the input received from the embedding layer and produces a feature map. Here a convolution operation is performed on a window of h word embedding $w_{i:i+h-1}$ with a filter $k \in \hat{A}^{h\times d}$using the function$f(W.w_{i:i+h-1} + b)$. In this function h is the window size for performing convolution operation. The convolution operation is performed for all the tokens by sliding the window. f is a non-linear activation function and $b \in \hat{A}$is a bias term. The feature map c is created by applying filter k to all possible windows$c = [c_1, c_2, \dots\dots, c_{n-h+1}]$. A CNN extracts different specific patterns of n-gram by applying different widths of kernels which is a number of

convolutional filters that slide over the entire word embedding matrix.

as W_x and U_x. The term b_x represents the bias vector which adaptively selects and discards historical information.

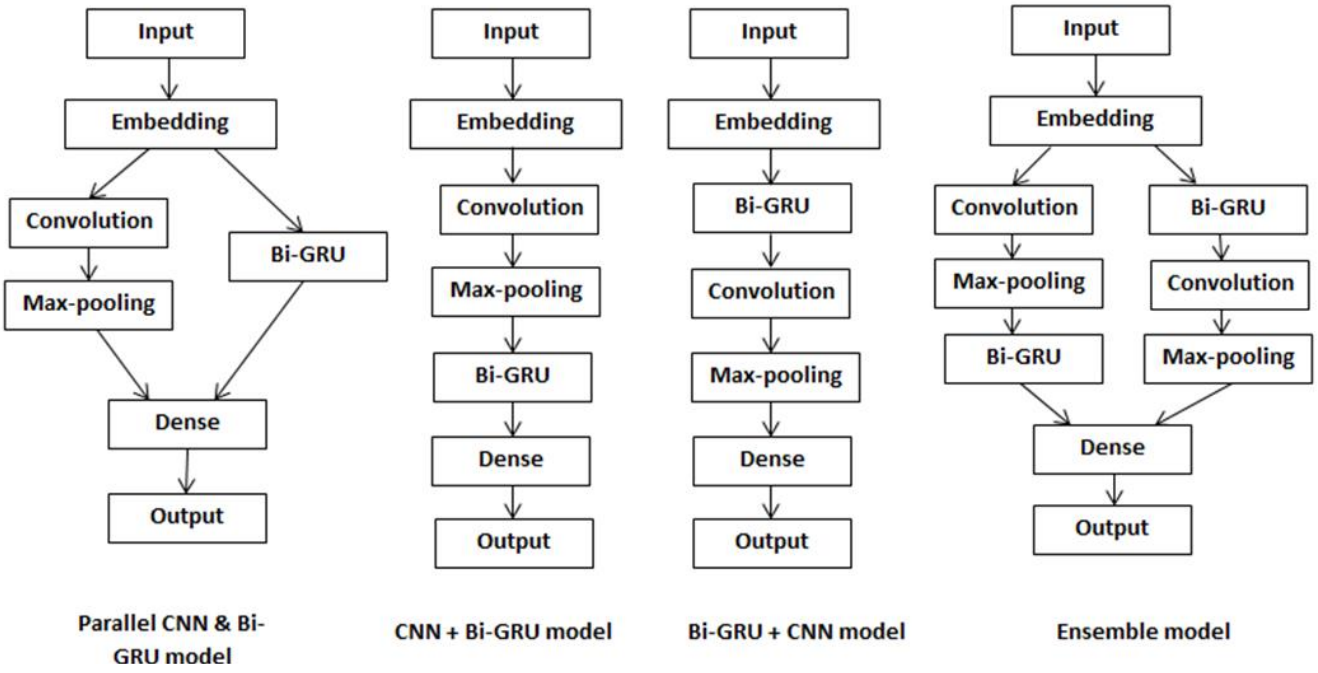

Fig.3: Architecture of all the proposed models

- Max-poolinglayer

To decrease the dimensionality in representation, the convolved output features undergo pooling, which helps identify prominent features. The max-pooling technique selects the maximum x values from the feature map c, denoted as $\hat{c} = \max(c)$. Also, max pooling can map the input to a fixed dimension of output. Therefore, max pooling is employed to maintain a fixed-length output, typically necessary for classification purposes. It can capture the most useful local features extracted through convolution by using the max operation on each filter.

- Bi-GRU layer

The Bi-GRU model incorporates both forward and backward GRU components. While the forward GRU processes the text from the beginning to the end, the backward GRU processes it in the opposite direction, from the end to the beginning. Gated recurrent neural network accepts a sentence as input vector $S_t$at each time point and combines the output vector h_{t-1}at the previous time point to update its hidden layer node state h_t. Output vector h_t can be computed using the following equations:

$$z_t = \sigma(W_Z S_t + U_Z h_{t-1} + b_Z)$$

$$r_t = \sigma(W_r S_t + U_r h_{t-1} + b_r)$$

$$\hat{h}_t = tanh(W_h S_t + r_t * U_h h_{t-1} + b_h)$$

$$h_t = (1 - z_t) * h_{t-1} + z_t * \hat{h}_t)$$

here the symbol * specifies the cross product. In each hidden layer, reset threshold r_t and update threshold z_t are used to update the information. Coefficient matrices are represented

- Fully connected layer

The output from hybrid CNN and Bi-GRU layers is finally passed to the fully connected layer which uses the softmax function for classification. Softmax function is used to map the output of neurons in the interval (0,1) and the class with the largest probability value can be selected as the predicted class. The calculation of softmax value for the input vector $\vec{z_i}$ can be performed using the Equation.

$$\sigma(\vec{z_i}) = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}}$$

Where e^{z_i} is the standard exponential function for an input vector and e^{z_j} is the standard exponential function for an output vector and k represent the number of classes in the multi-class classifier.

IV. EXPERMENTAL SETUP FOR THE ANALYSIS

A. Datasets for the study

The performances of the proposed models were analyzed using four publicly available Tamil data sets: Tamilmurasu news classification data set [91], Tamil short text classification data set [92], Tamil movie review data set [92] and IndicNLP News Article Classification Dataset [93].

TABLE I: DATASET DESCRIPTION the details about data sets,

Data set name	Number of target classes	Number of samples
Tamilmurasu news article classification dataset	6	67259
IndicNLP news classification dataset	3	10492
Tamil short text classification dataset	6	13955
Tamil movie review dataset	2	601

which represent the Number of target classes and number of each samples in dataset. For instance, the

TABLE IV: DATASET DESCRIPTION OF TSC

Category	Number of samples
TamilNadu	4807
India	3337
Cinema	1937
Sports	1657
Politics	1172
World	1045

Tamilmurasu news article classification dataset comprises six target classes with a substantial 67,259 samples. Similarly, the Indic NLP news classification dataset presents three target classes with 10,492 samples. The details of other datasets, including the Tamil short text classification dataset and the Tamil movie review dataset, are also elaborated in Table I, providing a comprehensive overview of the dataset characteristics.

Table I presents the details about data sets, which represent the Number of target classes and number of samples in each dataset. For instance, the Tamilmurasu news article classification dataset comprises six target classes with a substantial 67,259 samples. Similarly, the Indic NLP news classification dataset presents three target classes with 10,492 samples. The details of other datasets, including the Tamil short text classification dataset and the Tamil movie review dataset, are also elaborated in Table I, providing a comprehensive overview of the dataset characteristics.

1) Tamilmurasu news classification data set (TNC)

The data set is a subset of the publicly available Tamilmurasu dataset which contains 127,000 text documents along with their respective categories. To reduce the class unbalance, a subset of the data set containing 67,259 news articles was created. The selected articles fall into six categories, namely, India, Sports, World, Crime, District and Cinema and the corresponding number of samples in each classes are described in Table II: Dataset description of TNC.

TABLE II: DATASET DESCRIPTION OF TNC

Category	Number of samples
India	16935
Sports	8230
World	7477
Crime	16290
District	9079
Cinema	9248

2) IndicNLP news classification data set (INC)

The data set contains 10491 news articles, which are categorized into Entertainment, Politics and Sports. The data set is balanced across classes. The corresponding number of samples for each categoryis detailed in Table III.

TABLE III: DATASET DESCRIPTION OF INC

Category	Number of samples
Entertainment	3497
Politics	3497
Sport	3497

3) Tamil short text classification data set (TSC)

The data set contains 13955 news headlines with labels which were collected from Puthiyathalaimurai online Tamil news website. The headlines are categorized into six classes, namely, TamilNadu, India, Cinema, Sports, Politics and World. Table IV: DATASET DESCRIPTION OF TSC provides a detailed account of the number of samples within each category in the dataset, offering a comprehensive overview of the distribution across various content categories.

4) Tamil movie review data set-TMRC

This data set contains 601 Tamil movie reviews which are collected from various online Tamil movie review websites. Movie reviews have a score rate between 1 to 5. This data set is used for sentiment classification task and falls into Positive and Negative categories. Table V: DATASET DESCRIPTION OF TMC provides a detailed account of the number of samples in each category. Specifically, the dataset includes 222 samples classified as Positive and 379 samples classified as Negative. This breakdown offers insights into the distribution of sentiments within the TMRC dataset.

TABLE V: DATASET DESCRIPTION OF TMC

Category	Number of samples
Positive	222
Negative	379

B. Evaluation methods

In this study, the precision, Recall, F1 score, and Accuracy are selected as the evaluation metrics. To compare the performance of the baseline model and different pre-trained word vectors, the accuracy score is used since it can be considered as the most intuitive and comprehensive evaluation indicator. The evaluation metrics are defined below:

$$\text{Precision} = \frac{TP}{(TP + FP)}$$

$$\text{Recall} = \frac{TP}{(TP + FP)}$$

$$\text{F1} - \text{score} = \frac{(2 \times Precision \times Recall)}{(Precision + Recall)}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

where, TP is true positive, TN is true negative, FP is false positive, and FN is false negative

C. Analysis on Pre-trained word vectors for Tamil text representation

The study conducted multiple experiments to analyse effectiveness of word embedding method against the dimension of representation for Tamil text. Fifteen pre-trained word vectors were generated using CBOW, Skip-gram, and FastText models with dimensions of 50, 100, 200,

300, and 400. The performance of these models were evaluated using a simple neural network model consisting of one fully connected layer, which was applied to classify all the selected datasets.

Stanford's GloVe and Facebook's FastText are some popular pre-trained word vectors which can be readily used to represent words [20-23]. Facebook released pre-trained word vectors for 157 languages including Tamil using FastText which is the only one publicly available pre-trained word vector for Tamil language [94]. The performance of the pre-trained word vector released by Facebook for the Tamil language is compared with the FastText models generated in this study.

D. Parameter Settings:

The study aimed to identify the optimal hyperparameters for Tamil classification tasks, as they depend on the characteristics of each dataset. Several trials were conducted to choose the parameters, as Tamil NLP has a lack of research in deep learning to find the optimal hyperparameters. To train the embedding models, the text needed to be zero-padded such that all the text has a specific maximum length.

To build the models, the study used a simple one-dimensional convolution layer with a filter size of 128 and applied kernel sizes in the range of 3 to 5 for different datasets. The output dimension of the Bi-GRU was fixed as 100. The CNN and Bi-GRU layers used *Relu* and *Tanh* activation functions, respectively. Before the output softmax layer, a fully connected layer with a *Relu* activation function was used. TABLE VI outlines the specific parameter configurations employed in all deep learning-based models analysed in this study. These parameter settings serve as crucial inputs for the training and optimization of the deep learning models, contributing to the overall performance and effectiveness of the classification process. The learning rate is set to 0.001, the epoch size is 8, the batch size is 32, the loss function is Binary Crossentropy, and the optimizer used is Adam.

TABLE VI: PARAMETER SETTINGS FOR DEEP LEARNING-BASED MODELS

Parameters	value
Learning rate	0.001
Epoch size	8
Batch size	32
Loss function	Binary_crossentropy
Optimizer	Adam

V. RESULTS AND DISCUSSIONS

A. Baseline performance

1) Performance of Classical models:

This study considers the baseline performance for NLP tasks as performance of models using classical NLP features namely BOW and TF-IDF along with classifiers LR, NB and SVM. The implementation of these models was carried out using the scikit-learn library with default parameters. Table VII illustrates the testing accuracy scores for different models on distinct datasets, employing both BOW and TF-IDF representations. The models used LR, NB, and SVM as classifier. The accuracy scores are detailed as follows:

TABLE VII: ACCURACY SCORE OF CLASSICAL APPROACHES

Dataset	BOW			TF-IDF		
	LR	NB	SVM	LR	NB	SVM
TMNC	0.8216	0.8011	0.8236	0.8450	0.7145	0.8583
INC	0.9512	0.9496	0.9344	0.9565	0.9493	0.9599
TSC	0.6674	0.6643	0.5984	0.5984	0.5984	0.5984
TMRC	0.5847	0.5818	0.5517	0.5867	0.5308	0.6402

For the Tamilmurasu news article classification dataset, SVM obtained better accuracy than LR and NB classifiers and showed nearly equal performance for both BOW and TF-IDF features. All the models obtained high accuracy for IndicNLP news classification.

All three classifiers obtained insufficient accuracy for Tamil movie review dataset and Tamil short text classification. For Tamil movie review dataset, the accuracy was even less for TF-IDF features compared with BOW. The models obtained accuracy score in the range of 0.55 to 0.65 for Tamil short text classification. Here SVM showed comparatively high performance with TF-IDF feature.

2) Performance for Multilingual BERT model:

The study used the BERT multilingual base model (uncased) [95] pre-trained with the largest Wikipedia in 102 languages to obtain the baseline performance for deep-learning approaches. For this analysis, the parameters, namely, the number of training epochs, maximum sequence length, batch size and learning rate are set as 4, 128, 32, and 3e-5, respectively. TABLE VIII provides a comprehensive overview of the performance metrics, including accuracy and F1-score, for the Multilingual BERT model across diverse datasets. For TMNC Dataset, The model achieved an accuracy and F1-score of 82.79%, showcasing its robust performance on this dataset. The Multilingual BERT model exhibited a high accuracy and F1-score of 92.24% and 92.25%, respectively, underscoring its effectiveness in classifying instances within the INC dataset. In the TSC dataset, the model's performance was notably lower with an accuracy and F1-score of 44.34% and 44.36%, suggesting challenges in accurately classifying texts in this particular context. The model performed reasonably well on the TMRC dataset, achieving an accuracy and F1-score of 62.66%. While not as high as some other datasets, it still demonstrates moderate effectiveness in this context. These comparisons highlight the Multilingual BERT model's adaptability but also underscore the dataset-specific nature of its performance. It excels in certain contexts but may face challenges in others, reinforcing the importance of considering dataset characteristics in model selection.

TABLE VIII: RESULTS OF MULTILINGUAL BERT MODEL

Data set	Accuracy	F1-score
TMNC	0.8279	0.8279
INC	0.9224	0.9225
TSC	0.4434	0.4436
TMRC	0.6266	0.6266

B. Performance of Pre-trained word vectors for Tamil text representation.

This section presents the results obtained for various pre-trained word vectors. Specifically, the accuracy scores for CBOW and Skip-gram models are presented in TABLE IX. Upon analysing the results, it is evident that the accuracy of the word embedding models improved with an increase in the dimension of the vector, peaking at 300. However, beyond 300, the accuracy either decreased or showed minimal improvement. Across all four selected datasets, Skip-gram outperformed CBOW and FastText in terms of accuracy. Notably, the Tamil movie review dataset showed almost the same low accuracy across all dimensions in CBOW, Skip-gram, and FastText. This could be attributed to the relatively small sample size of this dataset. The Skip-gram model can represent rare words compared with the CBOW model, but it is slower than the CBOW model. The FastText model can solve the out of vocabulary issue since it can construct the embedding vector for an unseen word using the character n-grams of other word.

TABLE IX: ACCURACY SCORE FOR TEXT CLASSIFICATION USING A SIMPLE DEEP-LEARNING MODEL WITH PRE-TRAINED WORD2VEC VECTORS

Dataset	Word2Vec-cbow					Word2Vec-skip gram				
	50	100	200	300	400	50	100	200	300	400
TMNC	0.7138	0.76	0.7476	0.8023	0.8129	0.7811	0.7861	0.7836	0.83	0.8285
INC	0.8502	0.8917	0.9199	0.9269	0.9272	0.9083	0.9085	0.9134	0.9145	0.9160
TSC	0.5159	0.5381	0.5682	0.5711	0.5558	0.5806	0.5914	0.6114	0.6126	0.6234
TMRC	0.5138	0.5856	0.6077	0.6077	0.6077	0.6077	0.6077	0.6077	0.6077	0.6077

TABLE X: ACCURACY SCORE FOR FASTTEXT PRE-TRAINED WORD VECTORS WITH DIFFERENT DIMENSIONS

Dataset	FastText					Facebook's FastText
	50	100	200	300	400	300
TMNC	0.7702	0.7868	0.7851	0.8368	0.8317	0.7075
INC	0.8856	0.913	0.933	0.939	0.9392	0.8904
TSC	0.5701	0.6112	0.6277	0.6523	0.6408	0.5455
TMRC	0.6077	0.6077	0.6077	0.6077	0.6077	0.6077

Additionally, this study conducted a performance comparison between a FastText model trained with a local corpus and the existing FastText pre-trained word vector released by Facebook for the Tamil language. TABLE X displays the obtained classification accuracy. It is evident from the results that the accuracy achieved using Facebook's pre-trained word vectors is inferior to the pre-trained word vectors generated in this study. As mentioned earlier, the corpus used to generate the pre-trained word vectors contained around 3 million unique words, whereas Facebook's model utilized 2 million unique words.

These results indicate that increasing the number of training samples can result in a more effective word embedding representation of text. Moreover, it is worth noting that the study also observed a performance peak at the dimension of 300, which further highlights the importance of selecting an appropriate embedding dimension for effective text classification.

C. Results for Hybrid CNN and Bi-GRU models

In order to ensure uniformity in the length of training samples, the study set a maximum text length of 1000 for the datasets except Tamil short text dataset, for which it is set as 30. For texts that were shorter than this length, zero padding was used. This length was determined based on the average number of words in the text. The study used a 300-dimensional FastText model as the embedding model for all four proposed models, with the selection of the embedding model and dimension based on the analysis conducted in the previous section. The analysis involved the examination of jointly learning deep learning-based models, namely CNN and Bi-GRU, alongside four additional proposed deep learning-based models. The implementation, training, and application of these models were conducted to perform classification tasks on the selected datasets. The goal of this analysis was to assess the performance and effectiveness of these models in handling the given datasets and to draw insights into their comparative capabilities for the specified classification tasks.

Tables XI presents the performance metrics for different models in the task of TamilMurasu News Article Classification. The CNN exhibits high precision, recall, and accuracy, indicating its ability to correctly classify news articles, with a well-balanced F1-score suggesting overall effectiveness. The Bi-GRU model demonstrates similar high performance as CNN, showcasing its efficiency in capturing sequential dependencies and nuances in news articles. The model combining Parallel CNN and RNN surpasses the individual models, achieving higher precision, recall, and accuracy, indicating enhanced discriminatory power. The combination of CNN and Bi-GRU further improves performance emphasizing the synergy between different neural network architectures in news article classification. In this alternative combination, Bi-GRU followed by CNN maintains high precision and recall, demonstrating the robustness of the architecture order in the ensemble. The ensemble model, aggregating predictions from various models, achieves the highest precision, recall, accuracy, and

F1-score, showcasing its effectiveness in comprehensive news article classification. In summary, all models perform exceptionally well, with the ensemble model standing out as the top-performer in accurately classifying TamilMurasu news articles.

The performance results for Tamil Short Text Classification, as summarized in Table XIV, illustrate the effectiveness of diverse models in handling the nuances of short texts. While CNN demonstrates moderate performance, the Bi-GRU exhibits improved precision, recall, accuracy, and F1-score, emphasizing its capability in short text classification. The combination of Parallel CNN and RNN further enhances the performance results for IndicNLP News Classification, as outlined in Table XII, reveal a nuanced evaluation of various models.

Moreover, the Bi-GRU models exhibit moderate yet commendable precision, recall, accuracy, and F1-scores, showcasing their efficacy in classifying IndicNLP news articles. The combination of Parallel CNN and RNN proves to be synergistic, surpassing individual models in terms of precision, recall, and accuracy.

TABLE XI: RESULTS FOR TAMILMURASU NEWS ARTICLE CLASSIFICATION

	Precision	Recall	Accuracy	F1-score
CNN	0.9026	0.9032	0.9032	0.9027
Bi-GRU	0.9050	0.9048	0.9042	0.9042
Parallel CNN & RNN	0.9124	0.9101	0.9086	0.9124
CNN+Bi-GRU	0.9133	0.9132	0.9124	0.9124
Bi-GRU+CNN	0.9062	0.9065	0.9060	0.9060
Ensemble model	0.9160	0.9162	0.9156	0.9156

TABLE XII: RESULTS FOR INDICNLP NEWS CLASSIFICATION

	Precision	Recall	Accuracy	F1-score
CNN	0.7007	0.7081	0.7081	0.6999
Bi-GRU	0.7091	0.7165	0.7165	0.7101
Parallel CNN &RNN	0.7348	0.7370	0.7370	0.7298
CNN+Bi-GRU	0.7465	0.7480	0.7480	0.7410
Bi-GRU+CNN	0.7553	0.7526	0.7526	0.7531
Ensemble model	0.7570	0.7538	0.7538	0.7454

The combination of Parallel CNN and RNN proves to be synergistic, surpassing individual models in terms of precision, recall, and accuracy. Furthermore, ensemble approaches, both in the form of CNN+Bi-GRU and Bi-GRU+CNN, demonstrate improved performance, emphasizing the significance of model combination. The Ensemble Model, aggregating predictions from various architectures, emerges as the top performer, achieving the highest precision, recall, accuracy, and F1-score.

This comprehensive analysis suggests that the ensemble approach significantly enhances the overall performance, providing a robust solution for IndicNLP news article classification.

The performance results for Tamil Movie Review Classification, as summarized in Table XIII, demonstrate outstanding proficiency across various models. Both the CNN and Bi-GRU exhibit exceptional precision, recall, accuracy, and F1-score, showcasing their effectiveness in capturing nuanced features of Tamil movie reviews. The Parallel CNN and RNN combination maintains high performance, while the ensemble approaches, CNN+Bi-GRU and Bi-GRU+CNN, further enhance classification accuracy.

Notably, the Ensemble Model, aggregating predictions from diverse architectures, stands out as the top performer, achieving the highest precision, recall, accuracy, and F1-score. Additionally, it is worth considering that the results were influenced by the datasets used for evaluation. This comprehensive analysis underscores the robustness of the ensemble approach in achieving superior performance for Tamil Movie Review Classification, providing a well-rounded solution for accurate and reliable classification of sentiments in movie reviews.

TABLE XIII: RESULTS FOR TAMIL MOVIE REVIEW CLASSIFICATION

	Precision	Recall	Accuracy	F1-score
CNN	0.9689	0.9684	0.9684	0.9685
Bi-GRU	0.9720	0.9718	0.9718	0.9718
Parallel CNN & RNN	0.9664	0.9661	0.9661	0.9661
CNN+Bi-GRU	0.9722	0.9722	0.9722	0.9722
Bi-GRU+CNN	0.9714	0.9714	0.9714	0.9714
Ensemble model	0.9725	0.9726	0.9726	0.9725

TABLE XIV RESULTS FOR TAMIL SHORT TEXT CLASSIFICATION

	Precision	Recall	Accuracy	F1-score
CNN	0.6910	0.6961	0.6961	0.6811
Bi-GRU	0.7253	0.7293	0.7293	0.7219
Parallel CNN & RNN	0.7550	0.7459	0.7459	0.7290
CNN+Bi-GRU	0.7379	0.7403	0.7403	0.7320
Bi-GRU+CNN	0.7628	0.7403	0.7403	0.7257
Ensemble model	0.7529	0.7535	0.7535	0.7532

Notably, the Ensemble Model emerges as a strong contender, achieving high precision, recall, accuracy, and F1-score, showcasing the robustness of the ensemble approach in effectively classifying short texts. This comprehensive analysis underlines the suitability of various models for short text classification, with the ensemble strategy providing a reliable solution for improved accuracy across multiple evaluation metrics.The performance across diverse text classification tasks is summarized as follows: For the TamilMurasu news article classification dataset, the deep

learning models using the FastText word vectors obtained an increment of accuracy by around 10% compared to the classical models. In deep learning-based models, the hybrid CNN and Bi-GRU models showed improved performance compared to individual CNN and Bi-LSTM models. For the IndicNLP news classification dataset, the deep learning-based models increased the performance by around 2% compared to the classical models, even though the classical methods gave satisfactory results of around 94% accuracy. Bi-GRU model achieved better performance than CNN and the CNN+Bi-GRU. The ensemble models obtained a slightly improved accuracy compared to individual models.

For the Tamil movie review dataset, the deep learning models with pre-trained embedding showed significant improvement, since the performance of classical models was extremely low. For the Tamil short text classification dataset, the deep learning-based model outperformed the classical models by about 15% of accuracy. In addition, the hybrid CNN and Bi-GRU models slightly improved the performance and efficiency of the models compared to individual CNN and Bi-GRU. The high-performance model in the context of the provided tables appears to be the Ensemble Model, as indicated by the consistently superior metrics across precision, recall, accuracy, and F1-score in multiple text classification tasks. This Ensemble Model is a fusion of different architectures, possibly combining CNN, Bi-GRU, and other components. The model leverages the strengths of each constituent part, resulting in a comprehensive and robust system for text classification.

Specifically, in Tables XI, XII, XIII, and XIV, the Ensemble Model consistently achieves top-notch performance across various datasets, showcasing its adaptability and effectiveness in handling diverse text types, including news articles, movie reviews, and short texts. The high precision, recall, accuracy, and F1-score values collectively indicate the model's ability to provide accurate and well-balanced predictions.

The success of the Ensemble Model underscores the significance of integrating diverse approaches to text classification, capitalizing on the unique strengths of each individual model. This approach ensures a more comprehensive understanding of the underlying patterns in

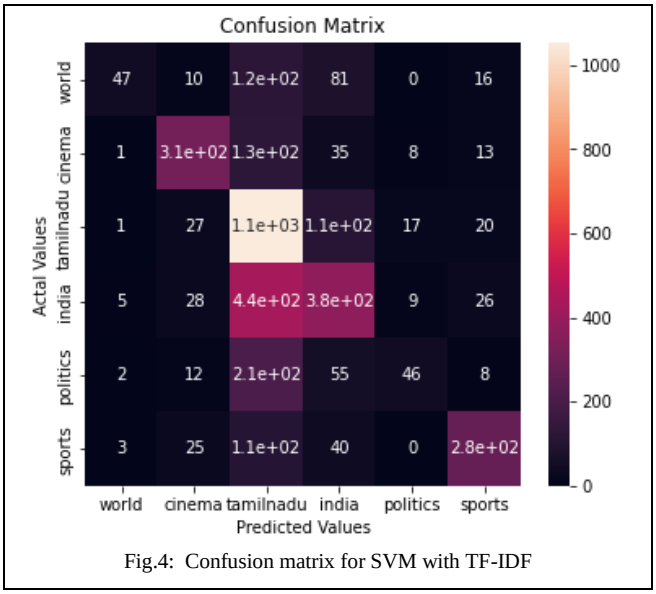

Fig.4: Confusion matrix for SVM with TF-IDF

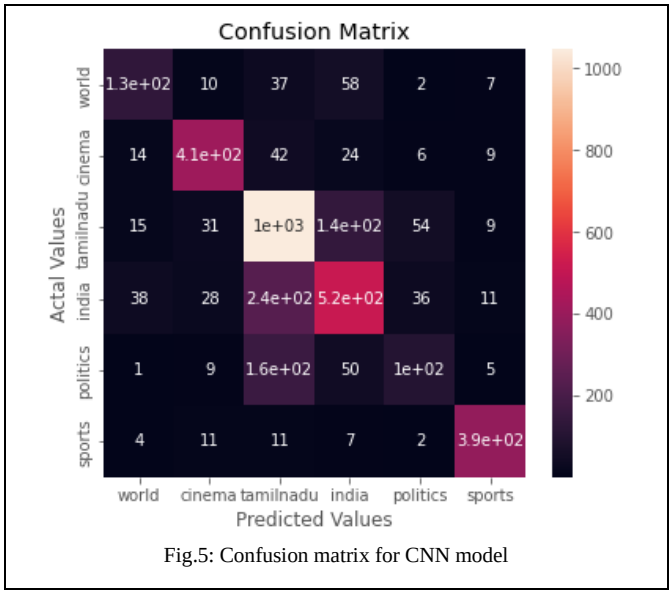

Fig.5: Confusion matrix for CNN model

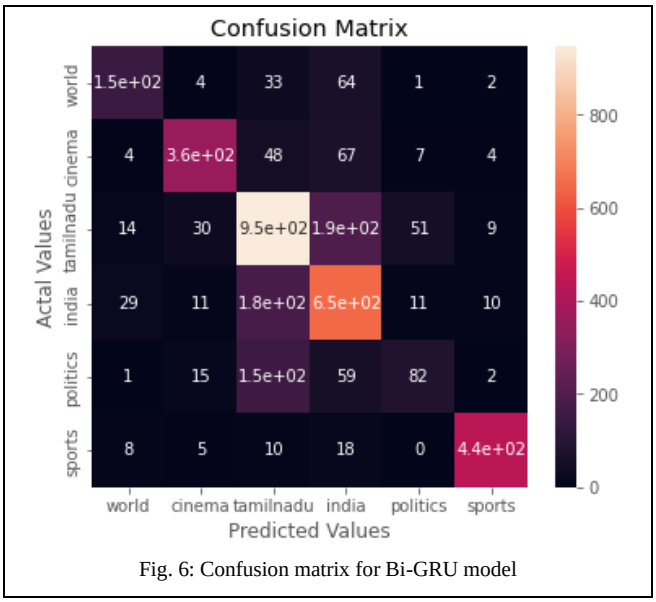

Fig. 6: Confusion matrix for Bi-GRU model

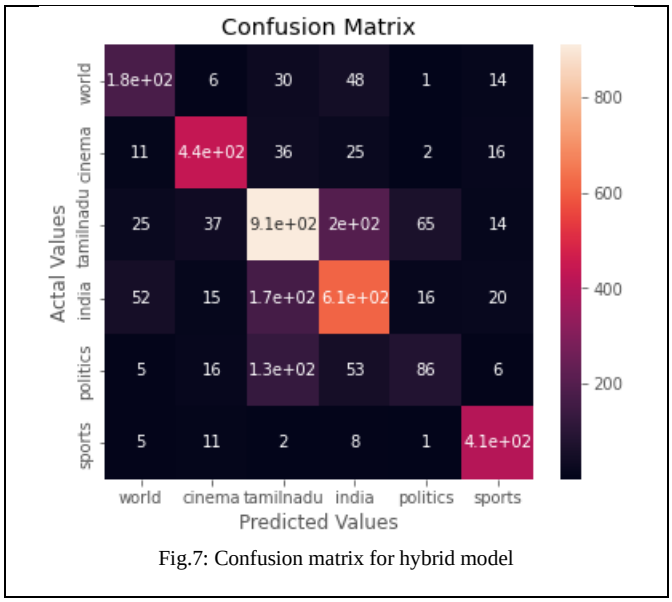

Fig.7: Confusion matrix for hybrid model

the data, leading to improved performance compared to relying on a single architecture.

1) Analysis of models on Movie review dataset

The study specifically chose to analyse the Tamil short text classification task, which is a multi-class classification problem, and revealed notable variations in performance across different models. The confusion matrices for each of the analysed models are presented in Figures 4, 5, 6, and 7. The classical model, which uses SVM with TF-IDF, exhibited poor performance in correctly predicting samples belonging to the "Tamilnadu," "India," "Politics," and "Sports" classes. This is because most of the texts in these categories fall under severely overlapping domains.

However, the deep learning models, specifically the CNN and Bi-GRU, demonstrated improved accuracy in predicting samples in the "Sports" and "Politics" classes. The Bi-GRU model was able to resolve ambiguity in the "Politics" and "Tamilnadu" classes, possibly due to its ability to retain sequential information. The CNN model increased the overall correct predictions in other classes by utilizing word-level n-gram features. As a result, the hybrid model that combined both CNN and Bi-GRU exhibited better overall predictions than the individual models.

VI. Conclusion

In recent years, natural language processing (NLP) has achieved remarkable advancements that were once thought to be possible only in movies or science fiction. These advancements have been made possible due to the development of new techniques such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), and the availability of powerful hardware. However, many of these advancements were not achieved in low-resource languages. This study aimed to analyse the performance of deep learning-based approaches for NLP tasks in the Tamil language, particularly in text classification. The outcome of this study are listed below:

1. Reviewed existing research on text classification in Tamil NLP and conducted a performance analysis of classical models.
2. A corpus containing 254633 Tamil articles collected from Wikipedia and other web sources.
3. Word embedding models for Word2Vec (cbow and skipgram) and FastText with varying dimensions (100 to 500) generated using the above corpus.
4. Four hybrid deep-learning models were built based on the literature study.
5. Performance analysis summary of the word embedding vectors and the proposed hybrid models.

The analysis conducted on the capability of various word embedding approaches revealed that training the embedding vectors with a task-related corpus could lead to better performance. In addition, the analysis shows that an increase in corpus size would result in an improved performance.

Although increasing the dimensions of text representation improves effectiveness, the study found that efficiency declines or shows a minor increase in higher dimensions. FastText with 300 dimensions was identified as a better-performing embedding model for Tamil text classification.

Based on the analysis performed on CNN, Bi-GRU, and four hybrid CNN and Bi-GRU models using four different Tamil datasets, it can be concluded that deep learning-based models can significantly improve the performance for NLP tasks, especially Tamil text classification. The jointly learning models are capable of outperforming models that generate embedding and train the classifier separately.

Moreover, the hybrid models with CNN and Bi-GRU gave slightly better performance than other individual CNN and Bi-GRU deep learning models. Furthermore, the proposed ensemble model has produced comparatively better results for the selected four datasets. Based on the study on the selected datasets, it can be indicated that when the performance is the important metric, hybrid CNN and Bi-GRU models can be utilised. Further, with the error analysis, the study indicates that with an adequately large dataset, the performance of the hybrid deep-learning models can be well highlighted. The study indicates that producing resources such as corpuses and task-related datasets for low-resourced languages is necessary to increase the performance of deep-learning based approaches. This would support low resourced languages to be on par with the improvements achieved in English and other highly resourced languages.

References

[1] Y. Bengio, R. Ducharme, and P. Vincent, “A neural probabilistic language model,” in Advances in Neural InformationProcessing Systems, 2001, pp. 932–938.
[2] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, “A neural probabilistic language model,” Journal of machine learningresearch, vol. 3, no.Feb, pp. 1137–1155, 2003.
[3] T. Mikolov, K. Chen, G. Corrado, and J. Dean., “Efficient estimation of word representations in vector space”, 2013.
[4] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed Representations of Words and Phrases andtheir Compositionality”, Advances in Neural Information Processing Systems, vol. 26, 10, 2013.
[5] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching Word Vectors with Subword Information”,Transactionsof the Association for Computational Linguistics, vol. 5, 07, 2016.
[6] J. L. Elman, “Finding structure in time”, Cognitive science, vol. 14(2), p.179–211,1990.
[7] S. Hochreiter and J. Schmidhuber, “Long short-term memory”,Neural computation, vol. 9, pp. 1735–80, 12, 1997.
[8] M. Schuster and K. Paliwal, “Bidirectional recurrent neural networks”, Signal Processing, IEEE Transactions on, vol. 45,pp. 2673 – 2681, 12, 1997.
[9] K. Cho, B. Merrienboer, D. Bahdanau, and Y. Bengio, “On the Properties of Neural Machine Translation: Encoder-Decoder Approaches” 09, 2014.
[10] X. Zhang, J. Zhao, and Y. LeCun, “Character-level convolutional networks for text classification,” Advances in neuralinformation processing systems, vol. 28, pp. 649–657, 2015.
[11] N. Kalchbrenner, E. Grefenstette, and P. Blunsom, “A convolutional neural network for modelling sentences”,arXivpreprint arXiv:1404.2188, 2014.
[12] Y. Kim, “Convolutional Neural Networks for Sentence Classification”, in Proceedings of the 2014 Conference onEmpirical Methods in Natural Language Processing (EMNLP), Association for Computational Linguistics, 10,pp. 1746–1751, 2014.
[13] J. Weston, S. Bengio, and N. Usunier, “Wsabie: Scaling up to large vocabulary image annotation”,IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011.
[14] P. Turney and P. Pantel, “From frequency to meaning: Vector space models of semantics”, Journal of Artificial Intelligence Research, vol. 37, 03,2010.

[15] R. Socher, C. Lin, A. Ng, and C. Manning, “Parsing natural scenes and natural language with recursive neural networks”, 01, pp. 129–136, 2011.
[16] E. Cambria, S. Poria, A. Gelbukh, and M. Thelwall, “Sentiment analysis is a big suitcase,” IEEE Intelligent Systems,vol. 32, pp. 74–80, 11 2017.
[17] R. Collobert and J. Weston, “A unified architecture for natural language processing: Deep neural networks with multitask learning,” vol. 8, 01, pp. 160–167, 2008.
[18] Pennington, Jeffrey and Socher, Richard and Manning, Christopher, “Glove: Global Vectors for Word Representation”, vol. 14, 01, pp. 1532–1543, 2014.
[19] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, “Bag of tricks for efficient text classification”, 07, 2016.
[20] Y. Kim, “Convolutional Neural Networks for Sentence Classification”, in Proceedings of the 2014 Conference onEmpirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics,pp. 1746–1751, 2014.
[21] J. Camacho-Collados, M. T. Pilevar, and R. Navigli, “Nasari: Integrating explicit knowledge and corpus statistics for amultilingual representation of concepts and entities”, Artificial Intelligence, vol. 240, 08 ,2016.
[22] Y. Zhang and B. Wallace, “A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification” 10,2015.
[23] Y. Liu, B. Liu, L. Shan, and X. Wang, “Modelling context with neural networks for recommending idioms in essay writing”, Neurocomputing vol. 275, pp. 2287–2293, 2018.
[24] J. Howard and S. Ruder, “Universal Language Model Fine-tuning for Text Classification”, in ACL (1), 01, pp.328–339, 2018.
[25] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, “Deep Contextualized Word Representations”, in NAACL-HLT, 01, pp. 2227–2237, 2018.
[26] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving Language Understanding by Generative PreTraining”, 2018.
[27] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in NAACL-HLT, pp. 4171–4186), 2019.
[28] G. Lample and A. Conneau, “Cross-lingual Language Model Pretraining, journalNeurIPS”, 01, 2019.
[29] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to Sequence Learning with Neural Networks”, Advances in NeuralInformation Processing Systems, vol. 4, 2014.
[30] A. Graves, A. rahman Mohamed, and G. E. Hinton, “Speech recognition with deep recurrent neural networks”, IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 6645–6649, 2013.
[31] L. Guo, D. Zhang, L. Wang, H. Wang, and B. Cui, “CRAN: A Hybrid CNN-RNN Attention-Based Model for Text Classification”, 37th International Conference, ER 2018, Xi’an, China, Proceedings, 09, pp. 571–585, 2018.
[32] O. Alharbi, “A deep learning approach combining cnn and bi-lstm with svm classifier for arabic sentiment analysis”, International Journal of Advanced Computer Science and Applications, vol. 12, 2021.
[33] W. Zhang, M. Wang, Y. Zhu, J. Wang, and N. Ghei, “A hybrid neural network approach for fine-grained emotion classification and computing”, Journal of Intelligent Fuzzy Systems, vol. 37, pp. 1–11, 05, 2019.
[34] A. Ombabi, W. Ouarda, and A. Alimi, “Deep learning cnn–lstm framework for arabic sentiment analysis using textualinformation shared in social networks”, Social Network Analysis and Mining, vol. 10, 12, 2020.
[35] Liu, Jin & Yang, Yihe&Lv, Shiqi& Wang, Jin & Chen, Hui. , “Attention-based BiGRU-CNN for Chinese question classification”, Journal of Ambient Intelligence and Humanized Computing, 2019.
[36] Y. Fan, L. Gongshen, M. Kui and S. Zhaoying, "Neural Feedback Text Clustering With BiLSTM-CNN-Kmeans", in IEEE Access, vol. 6, pp. 57460-57469, 2018.
[37] SenarathYasas, JihanNadheesh&RanathungaSurangika, “A Hybrid Approach for Aspect Extraction from Customer Reviews”, International Journal on Advances in ICT for Emerging Regions (ICTer), 2019.
[38] Goonathilake, Pathum&Pathirage, Nandana, “Stance-Based Fake News Identification on Social Media with Hybrid CNN and RNN-LSTM Models”, International Journal on Advances in ICT for Emerging Regions (ICTer), 2023.
[39] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, SharanNarang, Michael Matena, Yanqi Zhou, Wei Li, Peter J. Liu, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer", Journal of Machine Learning Research (JMLR), 2020.
[40] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, RuslanSalakhutdinov, "XLNet: Generalized Autoregressive Pretraining for Language Understanding", NeurIPS 2019.
[41] Xian, Tobias Lorenz, BerntSchiele, ZeynepAkata, "A Survey on zero-Shot Learning: From Theory to Implementation", Journal: arXiv preprint arXiv:2002.05002,2020.
[42] M. Swamy, P. Hanumanthappa, and J. Mohan, “Indian language text representation and categorization using supervised learning algorithm”,pp. 406–410, 11, 2014.
[43] U. Reshma, B. G. Hb, M. Kumar, and S. Kp, “Supervised methods for domain classification of tamil documents”, ARPNJournal of Engineering and Applied Sciences, vol. 10, pp. 3702–3707, 01, 2015.
[44] Z. Xiao, T. Mcenery, P. Baker, and A. Hardie, “Developing asian language corpora: standards and practice”,01, 2004.
[45] S. Arunselvan, M. Kumar, and S. Kp, “Sentiment analysis of tamil movie reviews via feature frequency count”, vol. 10,pp. 17 934–17 939, 01, 2015.
[46] S. Seshadri, M. Kumar, and S. Kp, “Analyzing sentiment in Indian languages micro text using recurrent neural network”,vol. 7, pp. 313–318, 01, 2016.
[47] S. Phani, S. Lahiri, and A. Biswas, “Sentiment analysis of tweets in three indian languages”, in WSSANLP@COLING,2016.
[48] E. Nivedhitha, S. Sanjay, M. Kumar, and S. Kp, “Unsupervised word embedding based polarity detection for tamiltweets,” vol. 9, pp. 4631–4638, 01, 2016.
[49] S. Seshadri, M. Kumar, and S. Kp, “Analyzing sentiment in Indian languages micro text using recurrent neural network”, vol. 7, pp. 313–318, 01, 2016.
[50] S. Ramraj, R. Arthi, S. Murugan, and M. Julie, “Topic categorization of Tamil News Articles using PreTrained Word2VecEmbeddings with Convolutional Neural Network”, in 2020 International Conference on Computational Intelligence forSmart Power System and Sustainable Energy (CISPSSE). IEEE, pp. 1–4.,2020.
[51] N. Rajkumar, T. Subashini, K. Rajan, and V. Ramalingam, “AN ENSEMBLE OF FEATURE SELECTION WITH DEEPLEARNING BASED AUTOMATED TAMIL DOCUMENT CLASSIFICATION MODELS”, International Journal of Electrical Engineering and technology (IJEET), vol. 11, pp. 70–91, 11, 2020.
[52] A. Kumar, S. Saumya, and J. Singh, “Nitp-ai-nlp@hasoc-dravidian-codemix-fire2020: A machine learning approach toidentify offensive languages from dravidian code-mixed text”, 12, 2020.
[53] S. K, P. B, and S. Kp, “Amrita_cen_nlp@dravidianlangtech-eacl2021: Deep learning-based offensive language identification in malayalam, tamil and kannada”, in DRAVIDIANLANGTECH, 2021.
[54] S. Sai and Y. Sharma, “Siva@ HASOC-Dravidian-CodeMix-FIRE-2020: Multilingual Offensive Speech Detection in Code-mixed and Romanized Text”, in FIRE (Working Notes), pp. 336–343, 2020.
[55] C. Vasantharajan and U. Thayasivam, “Towards offensive language identification for tamil code-mixed youtube comments and posts”, 08, 2021.
[56]Bhargava, Neeraj&Johari, Anantika., “Enhancing Deep Learning Approach for Tamil English Mixed Text Classification”, 2023.
[57] KogilavaniShanmugavadivel, SaiHarithaSampath, PramodNandhakumar, PrasathMahalingam, Malliga Subramanian, Prasanna Kumar Kumaresan, RubaPriyadharshini,“An analysis of machine learning models for sentiment analysis of Tamil code-mixed data”,Computer Speech & Language, Volume 76, 2022.
[58]Suba Sri Ramesh Babu., “Sentiment Analysis In Tamil Language Using Hybrid Deep Learning Approach”, Master’s thesis, Dublin, National College of Ireland, 2022.
[59] Abhinav Kumar, Sunil Saumya, and Ashish Singh, “Detecting Dravidian Offensive Posts in MIoT: A Hybrid Deep Learning Framework”, ACM Trans. Asian Low-Resour. Lang. Inf. Process., 2023.
[60]C. Kumaresan, P. Thangaraju, “ELSA: Ensemble learning based sentiment analysis for diversified text”, Measurement: Sensors, Volume 25, 2023.
[61]Shibly, F.H.A., Sharma, U., Naleer, H.M.M. , “An Efficient Method for Detecting Hate Speech in Tamil Tweets Using an Ensemble Approach”, International Conference on Innovative Computing and Communications. ICICC, 2023.
[62]JothiPrakash V and Arul Antran Vijay S., “Cross-lingual Sentiment Analysis of Tamil Language Using a Multi-stage Deep Learning Architecture”, ACM Trans. Asian Low-Resour. Lang. Inf. Process. 22, 12, Article 254, 2023.
[63]Eduri Raja, BadalSoni, Candy Lalrempuii, Samir Kumar Borgohain, “An adaptive cyclical learning rate based hybrid model for Dravidian fake news detection”, Expert Systems with Applications, Volume 241, 2024.

[64] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. v. d. Oord, A. Graves, and K. Kavukcuoglu, "Neural machine translation in linear time", arXiv preprint arXiv:1610.10099, 2016.
[65] J. Bradbury, S. Merity, C. Xiong, and R. Socher, "Quasi-recurrent neural networks", arXiv preprint arXiv:1611.01576,2016.
[66] J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang, "Dimensional sentiment analysis using a regional CNN-LSTM model", inProceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pp. 225–230, 2016.
[67] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequencemodeling", 12, 2014.
[68] L. Zhou and X. Bian, "Improved text sentiment classification method based on bigru-attention", Journal of Physics:Conference Series, vol. 1345, p. 032097, 11, 2019.
[69] B. Anbalagan, J. Thomas, and P. Kesavan, "Embedded bi-directional gru and lstmlearning models to predict disasterson twitter data",Procedia Computer Science, vol. 165, pp. 511–516, 01, 2019.
[70] T. Zhang and R. Xu, "Performance comparisons of bi-lstm and bi-gru networks in chinese word segmentation", inICDLT 2021: 5th International Conference on Deep Learning Technologies, Qingdao, China, July 23 - 25, 2021.
[71] A. Mishra, R. K. Mundotiya, and S. Pal, "IIT (BHU)@ IECSIL-FIRE2018: Language Independent Automatic Frameworkfor Entity Extraction in Indian Languages", in FIRE (Working Notes), pp. 145–152, 2018.
[72] K. Kaur, "Khushleen@ IECSIL-FIRE-2018: Indic Language Named Entity Recognition using Bidirectional LSTMs withSubword Information", in FIRE (Working Notes), pp. 153–157, 2018.
[73] S. N. Bhattu, N. S. Krishna, and D. V. Somayajulu, "idrbt-team-a@ IECSIL-FIRE-2018 Named Entity Recognition of Indian Languages using bi-LSTM", in FIRE (Working Notes), pp. 158–165, 2018.
[74] V. Hariharan, M. Kumar, and S. Kp, Named Entity Recognition in Tamil Language Using Recurrent Based Sequence Model,01, pp. 91–99, 2019.
[75] S. Ramraj, R. Arthi, S. Murugan, and M. Julie, "Topic categorization of Tamil News Articles using PreTrained Word2VecEmbeddings with Convolutional Neural Network", in 2020 International Conference on Computational Intelligence forSmart Power System and Sustainable Energy (CISPSSE). IEEE, pp. 1–4, 2020.
[76] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg, "Feature hashing for large scale multitask learning.", p. 140, 2009.
[77] K. Jones, "A statistical interpretation of term specificity in retrieval", Journal of Documentation, vol. 60, pp. 493–502, 2004.
[78] C. Cortes and V. Vapnik, "Support-vector networks", Chem. Biol. Drug Des., vol. 297, pp. 273–297, 2009.
[79] Joachims, T, "Text Categorization with Support Vector Machines: Learning with Many Relevant Features.", Springer, 1998.
[80] T. Joachims, "Learning to Classify Text Using Support Vector Machines: Methods, Theory, and Algorithms", vol. 29, 2002.
[81] V. N. Vapnik, "Statistical Learning Theory. Wiley-Interscience," 1998.
[82] S. Hoi, R. Jin, and M. Lyu, "Large-scale text categorization by batch mode active learning", pp. 633–642, 2006.
[83] J. Zhang, R. Jin, Y. Yang, and A. Hauptmann, "Modified logistic regression: An approximation to svm and its applications in large-scale text categorization", vol. 2, pp. 888–895, 2003.
[84] P. Komarek and A. Moore, "Making logistic regression a core data mining tool a practical investigation of accuracy,speed, and simplicity",2005.
[85] M. Tahrawi and R. Zitar, "Polynomial networks versus other techniques in text categorization." IJPRAI, vol. 22, pp.295–322, 2008.
[86] Z. Zheng, X. Wu, and R. Srihari, "Term selection for text categorization on imbalanced data", SIGKDD Explorations,vol. 6, pp. 80–89, 2004.
[87] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-oneloss", Machine Learning - ML, vol. 29, pp. 103–130, 1997.
[88] D. Lewis, C. Info, L. Studies, and M. Ringuette, "A comparison of two learning algorithms for text categorization", Third Annual Symposium on Document Analysis and Information Retrieval, 1996.
[89] W. Yin, K. Kann, M. Yu, and H. Schütze, "Comparative study of cnn and rnn for natural language processing", ArXiv, vol. abs/1702.01923, 2017.
[90] Yao, Hongdou& Zhang, Xuejie& Zhou, Xiaobing& Liu, Shengyan, "Parallel Structure Deep Neural Network Using CNN and RNN with an Attention Mechanism for Breast Cancer Histology Image Classification", 2019.
[91] J. Krishnaraj., "Tamil news classification.", 2020 [Online]. Available: https://www.kaggle.com/code/krishnarajj/tamilnewsclassification/data?select=tamilmurasu_dataset.csv
[92] S. Rajkumar. Tamil nlp.[Online]. Available: https://www.kaggle.com/datasets/sudalairajkumar/tamil-nlp?select=tamil_movie_reviews_test.csv
[93] A. Kunchukuttan, D. Kakwani, S. Golla, G. N.C., A. Bhattacharyya, M. M. Khapra, and P. Kumar, "Ai4bharat-indicnlpcorpus: Monolingual corpora and wordembeddings for indic languages", arXiv preprint arXiv:2005.00085, 2020.
[94] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov, "Learning word vectors for 157 languages", in Proceedingsof the International Conference on Language Resources and Evaluation (LREC 2018), 2018.
[95] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding",CoRR, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805