



International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: <https://www.svedbergopen.com/>



Research Paper

Open Access

Large-Scale Kannada Sentiment Classification Using TF-IDF and Attention-Guided CNN- BiLSTM Networks: An Empirical Study Review

Adarsh M.J.^{1*} and P. Sridhara Acharya^{2*}

¹ Research Scholar, Institute of Computer Science and Information Science, Srinivas University, Mangalore – 575001, Karnataka, India, adimj24@gmail.com

² Research Professor, Institute of Computer Science and Information Science, Srinivas University, Mangalore – 575001, Karnataka, India, sridharaacharya@gmail.com

Abstract

Sentiment classification is still very difficult in Kannada due to its morphological richness, inflectional variations, compound formations, spelling variations and lack of large Kannada labelled corpora. The present study introduces a large-scale empirical comparison between conventional machine-learning and attention-guided deep-learning models with a binary Kannada review dataset with 119,952 labelled samples. The collected data set consisted of 58,797 negative reviews and 61,155 positive reviews, giving an almost balanced experimental environment. Three classical classifiers: TF-IDF with logistic regression, linear support vector machine and multinomial naïve Bayes were compared with attention-guided bidirectional long short-term memory network and convolutional neural network-BiLSTM. A stratified partition (80:20) resulted in 95,961 training and 23,991 independent test observations. The accuracy, macro-F1 and ROC-AUC of logistic regression were 82.19%, 82.18%, and 0.9054, respectively, which were higher than the other classical methods. The attention guided BiLSTM model was able to obtain nearly 86% accuracy and macro-F1 while the CNN-BiLSTM model obtained nearly 87% accuracy and class-balanced F1-scores. The latter boosted test accuracy by ~4.8 percentage points compared to the best TF-IDF baseline. According to the results of the architectural analysis, the CNN-BiLSTM needed around 2.91 million parameters and 14.47 million multiply-accumulate operations per review. Although the temporal pooling model had a few more parameters than the standalone BiLSTM, it had less sequential computational complexity, and showed faster inference when executed in the common notebook setting. The results show that the combination of local convolutional feature extraction, bidirectional sequence modelling and attention pooling is effective in the case of large-scale Kannada sentiment classification without the need for a computationally-intensive pretrained transformer.

Keywords: Kannada sentiment analysis, low-resource language, TF-IDF, CNN, BiLSTM, attention mechanism, text classification, Dravidian language.

I. INTRODUCTION

The rise of digital opinion reviewing platforms has generated significant amounts of opinionated text which can be used for product assessment, public-opinion analysis, recommendation systems and consumer-behaviour research. Sentiment analysis is an automatic process of judging the attitude of a text as positive, negative or neutral [1]–[4]. While there are plenty of benchmark datasets and pretrained language models for English sentiment classification, there are not many large, and well-annotated, resources for Kannada. Kannada is one of the important Dravidian languages, which has productive suffixation, inflection, agglutination and flexible word formation. There are multiple surface forms that are used with the same lexical root, depending on case, number, tense, person, and discourse context. As a result, word-frequency based approaches may suffer from feature sparsity despite that the words used have similar meanings. Review text presents further challenges such as inconsistent punctuation, extended expressions, spelling variation, English insertions and extended narrative descriptions. Contextual modelling is also highlighted as an important requirement for language processing tasks, where the need for context-aware language processing in extracting meaningful information from complex text is

also highlighted [5] [38]. Recent applications of artificial intelligence across different applied domains further demonstrate the importance of contextual factors, model design, and responsible technology utilisation [43]. Deep-learning applications have likewise demonstrated the importance of appropriate architecture selection, dataset quality, validation, and methodological robustness in AI-based classification systems [44], [45].

Typical sentiment classifiers include sparse textual features (such as term frequency-inverse document frequency) and logistic regression, support vector machines or naïve Bayes [6-10]. These are effective and easy to understand, and are, in some cases, competitive, particularly if the training corpus is large. They, however, have learned mostly the correlation between the lexical features and labels, and do not model the word order, local compositional patterns and long-range context much. CNN is able to capture discriminative local patterns including short sentiment expressions and phrases with negation [11]. Sequential dependencies are modeled in long short-term memory networks to overcome the vanishing-gradient problem of standard recurrent networks [12]. Bidirectional recurrent networks process the text in both directions and each contextual representation can use the information that comes before and after it [13], [14]. CNN-LSTM combinations thus have the ability to capture both complementary local and sequential information [15]. In recent sentiment classification research, it has been shown that the TF-IDF-based machine-learning techniques can be effectively combined with a bidirectional recurrent architecture for contextual sentiment classification, which suggests that using lexical feature representation and bidirectional sequence modelling are complementary techniques [16]. Attention mechanisms then offer an effective way to learn how to combine the most relevant contextual states [17, 18].

The current work aims at exploring if there is any measurable gain provided by these complementary mechanisms over the classical text classifiers in a large scale Kannada review scenario. The present experiment is not done on a small or highly code-mixed dataset of social media. Because of the size of the corpus, classical and neural models can be compared on the same test partition of 23,991 observations.

The study addresses the following research question:

RQ1: To what extent does an attention-guided CNN-BiLSTM architecture improve large-scale Kannada sentiment classification compared with TF-IDF classifiers and an attention-guided BiLSTM?

The principal contributions are as follows:

1. A detailed empirical characterisation of a 119,952-sample Kannada binary sentiment corpus.
2. A reproducible benchmark involving logistic regression, linear SVM, multinomial naïve Bayes, BiLSTM with attention and CNN-BiLSTM with attention.
3. Class-wise and aggregate evaluation using accuracy, precision, recall, macro-F1, weighted-F1, confusion matrices and ROC-AUC where probabilistic or continuous decision scores were retained.
4. An architectural-efficiency analysis based on parameter count, theoretical MACs and within-session training and inference time.
5. An evidence-based analysis of the benefit of combining convolution, bidirectional sequence modelling and attention for Kannada reviews.

The remainder of the paper is organised as follows. Section II reviews related work. Section III explains the dataset, preprocessing, model architectures, training procedure and evaluation measures. Section IV presents the results. Section V discusses the findings and limitations, and Section VI concludes the paper.

II. RELATED WORK

A. Classical Text Classification

TF-IDF is still a good feature representation for document classification because it gives greater credit to terms that are frequent in the document, but not so frequent in the whole corpus of documents [6]. Logistic regression learns a probabilistic linear decision boundary whereas SVM tries to maximise the separating margin between classes [7]. Multinomial naïve Bayes is based on the estimation of class membership on the basis of conditional term distribution [8]. These methods have been well-known and exist in the Scikit-learn library, and they allow reproducibility for pipelines for text classification [9] with Scikit-learn.

For sentiment classification and topic classification, simple linear models with n-gram features can still be rather competitive as shown by Wang and Manning [10]. Part of the reason for their effectiveness is that opinionated text is characterized by strong lexical regularities. However, when using sparse methods, a bigram or word form is considered as a single feature, so they cannot explicitly model the semantic similarity between morphologically related Kannada words.

B. Neural Sentiment Classification

Kim [11] introduced the use of multiple filters on embedded word sequences to make CNNs effective sentence

classifiers. Convolutional Filters: No need to manually specify n-grams to identify discriminative combinations of local features. LSTM networks [12] have gating mechanisms to either store or forget information across longer sequences. Bidirectional recurrent processing [13,14] encodes the text using both left-to-right and right-to-left dependencies.

CNN-LSTM networks combine local pattern extraction with sequential modelling [15]. Such a pairing seems to be especially relevant to sentiment analysis, as a local expression may alter its polarity when it is used in the presence of a negation, intensifier or contrastive clause located elsewhere in the text. A model can also assign non-uniform importance to contextual representations using hierarchical and additive attention mechanisms [17,18]. When full-size transformers are not feasible, CNNs, recurrent models and architectures with attention are still significant alternatives, as indicated by contemporary reviews [19]. Recent deep-learning applications also demonstrate the effectiveness of convolutional architectures for classification tasks, while emphasizing the importance of dataset quality, validation, and appropriate model evaluation [44], [45].

C. Kannada and Dravidian-Language Sentiment Analysis

Kannada sentiment research has been hindered due to comparatively limited amount of labelled data and the presence of orthographic variation and code-mixing. Hande et al. [20] have studied multi-task learning approach for sentiment and offensive-language detection in under-resourced Dravidian languages such as Kannada. Their results highlighted the importance of shared representations while revealing that the representation of Kannada performance is still dependent on the type of datasets and task formulation.

Pavan Kumar et al [21] proposed a comparison between CNN and LSTM based approach for Malayalam, Tamil and Kannada code mixed sentiment analysis. Shanmugavadivel et al. [22] also studied the sentiment and offensive-language identification for multilingual code-mixed texts via deep learning. Ijeri and Patil [23] contrasted the machine learning approaches for multi-emotion classification of Kannada text, and Eshwarappa and Shivasubramanyan [24] explored the feature selection and fine-tuning of BERT for Kannada language. These studies show that there is an increasing interest in Kannada sentiment classification studies; however, there are also significant variations in the labels, domains, sample sizes and preprocessing approaches. So, it is not appropriate to compare the accuracy expressed in these kinds of reported data. There is a need for a controlled evaluation with a single large dataset to separate the relative aspects of behaviour of classical and lightweight neural approaches.

D. Indic Representations and Transformer Context

Corpora, evaluation resources and pretrained models for multiple Indian languages were introduced by IndicNLP Suite [25]. MuRIL was specifically trained for enhancing the representation of Indian languages [26] and further efforts were made to expand the Indic corpora and benchmarks [27]. Dense representations of semantics and charactersubwords are given by Word2vec and fastText [28] and [29]. Transformer-based architectures [30] provide more broad contextual modelling, and compressed transformer models like DistilBERT and ELECTRA try to lower the costs of pretraining and inference [33] and [34]. In this paper we make a conscious choice to consider lighter models that can be trained directly on D1. This scope offers a clear baseline for further studies with pretrained or compression-aware transformers.

E. Research Gap

There are existing Kannada studies which tend to focus on more restricted, code-mixed or multi-class data sets. Other studies apply transformer models, but do not specify the extent to which such models are already achievable with CNN-BiLSTM networks that are configured correctly. In this paper, we fill this void by using a large binary review corpus, a common train-test partition, classical baselines, attention-based neural models, and architectural-efficiency analysis.

III. MATERIALS AND METHODS

A. Research Design

The research was in a supervised binary text-classification design. One observation was a Kannada review and its polarity. Full workflow involved dataset auditing, text and label selection, stratified partitioning, feature construction for the model, training and held out evaluation of the model.

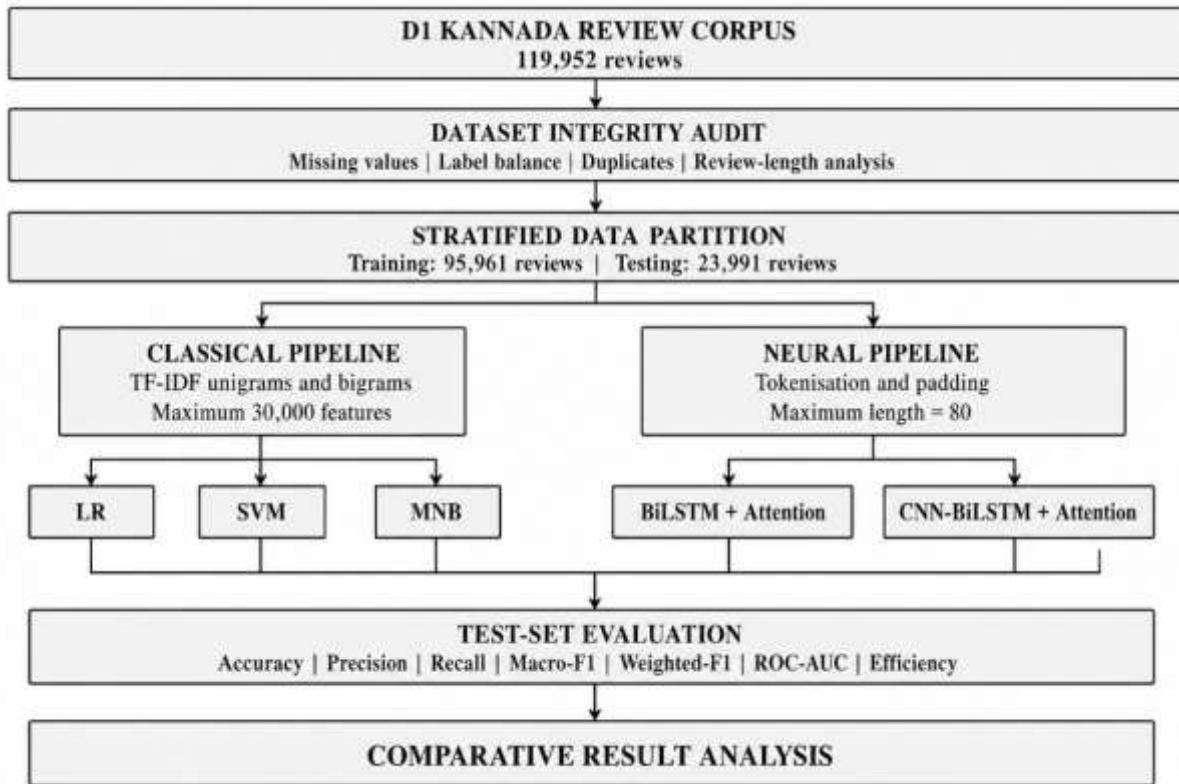


Fig. 1. End-to-end experimental workflow for large-scale Kannada sentiment classification.

The three classical models all used the same TF-IDF representation and the neural models all used the same learned embedding and sequence-preparing procedure (as shown in Fig. 1). The test partition was left out of the neural parameter optimisation.

B. Dataset Description and Audit

There were 119,952 reviews in Kannada in D1 and two sentiment labels. The Kannada word ನೆಗೃ represented the negative class whereas the Kannada word ಪೊಸ್ represented the positive class. A near-balanced distribution resulted from the difference between the two classes (2,358 reviews).

Table I. Descriptive profile of D1

Characteristic	Value
Total labelled reviews	119,952
Negative (ನೆಗೃ)	58,797
Negative percentage	49.02%
Positive (ಪೊಸ್)	61,155
Positive percentage	50.98%
Empty review texts	0
Unique normalised texts	119,946
Repeated text-label rows beyond the first occurrence	6
Texts associated with conflicting labels	0
Character length: minimum	28
Character length: median	511
Character length: mean	755.24
Character length: maximum	14,368
Whitespace-token length: minimum	3
Whitespace-token length: median	68
Whitespace-token length: mean	98.69
Whitespace-token length: maximum	1,846

There were 6 text-label lines that were repeated, a 0.005% share of the entire set. No contradictory labels were connected with repetitions of texts. The two repeated strings in the retained stratified partition were two rows from the training and test partitions. Their contribution to test accuracy could not account for differences in test performance and was less than 0.01 percentage points at best.

C. Text Preparation and Partitioning

The kn_review field was taken as model input, and the label field as the label. There were no empty review fields in the audited workbook, but the missing values were removed. Conversion to Unicode strings was done without translation, without stemming/removal of Kannada morphological endings. This moderate preprocessing kept the suffixes that might be sentiment-bearing out, while maintaining direct traceability to the uploaded data set.

The entire data was split in a stratified manner with an 80:20 data ratio and a random state of 42. This left us with 95,961 reviews in our training set and 23,991 reviews in our test set. The negative and positive test supports were 11,760 and 12,231, respectively.

For neural training, 10% of the training partition was reserved for validation. The approximate number of observations in the neural training subset and validation subset were thus 86,364 and 9,597, respectively. The validation subset was used for early stopping, and the independent 23,991-review test set was used for final reporting.

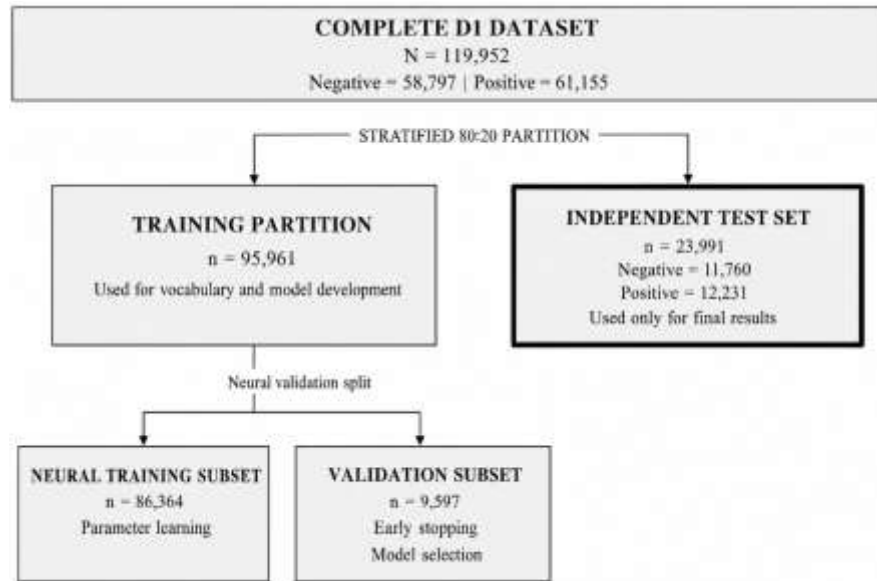


Fig. 2. Dataset partitioning and model-evaluation protocol

Table II. Model-specific input preparation

Component	Classical models	Neural models
Input field	kn_review	kn_review
Missing-value treatment	Remove incomplete rows	Remove incomplete rows
Vocabulary	Maximum 30,000 TF-IDF features	Maximum 20,000 learned tokens
Features	Word unigrams and bigrams	Integer token sequences
Minimum document frequency	3	Tokeniser frequency ranking
Out-of-vocabulary handling	Features outside vocabulary ignored	<OOV> token
Sequence length	Not applicable	80 tokens
Padding/truncation	Not applicable	Pre-padding and pre-truncation
Label encoding	Negative = 0; positive = 1	Negative = 0; positive = 1

D. Classical Models

1) TF-IDF Representation

For term t in document d , TF-IDF was defined as

$$\text{TFIDF}(t, d) = \text{TF}(t, d) \log \left(\frac{N + 1}{\text{DF}(t) + 1} \right), \quad (1)$$

where N is the number of training documents and $\text{DF}(t)$ is the number containing term t . Word unigrams and bigrams were retained up to a maximum vocabulary of 30,000 features.

2) Logistic Regression

For feature vector $\mathbf{x}$, logistic regression estimated positive sentiment as

$$P(y = 1 | \mathbf{x}) = \sigma(\mathbf{w}^T \mathbf{x} + b) = \frac{1}{1 + \exp[-(\mathbf{w}^T \mathbf{x} + b)]}. \quad (2)$$

The model used the liblinear solver and a maximum of 300 optimisation iterations.

3) Linear SVM

The linear SVM estimated a maximum-margin decision boundary by minimising

$$\min_{\mathbf{w}, b, \xi} \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n \xi_i, \quad (3)$$

subject to

$$y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \xi_i \geq 0.$$

4) Multinomial Naïve Bayes

The multinomial naïve Bayes classifier estimated

$$\hat{y} = \arg \max_{c \in \mathcal{C}} P(c) \prod_{j=1}^V P(t_j | c)^{x_j}, \quad (4)$$

where V is the vocabulary, x_j represents the feature value associated with term t_j , and c is a sentiment class.

E. Attention-Guided Neural Models

1) BiLSTM with Attention

The first neural model mapped each token to a 128-dimensional trainable embedding. A bidirectional LSTM containing 128 hidden units in each direction generated a 256-dimensional contextual representation at each position. For token position t ,

$$\mathbf{h}_t = [\vec{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t]. \quad (5)$$

A dropout rate of 0.30 was applied before attention pooling. The attention score and document representation were calculated as

$$e_t = \mathbf{v}^\top \mathbf{h}_t, \alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^T \exp(e_k)}, \quad (6)$$

$$\mathbf{r} = \sum_{t=1}^T \alpha_t \mathbf{h}_t. \quad (7)$$

The attention-pooled representation was passed through a 128-unit ReLU layer and a two-unit softmax classifier.

2) CNN-BiLSTM with Attention

The second architecture inserted a one-dimensional convolution before the BiLSTM. For convolutional filter k ,

$$c_{t,k} = \text{ReLU}(\mathbf{W}_k \cdot \mathbf{E}_{t:t+h-1} + b_k), \quad (8)$$

where $h = 3$ was the kernel width and $\mathbf{E}$ was the embedded token sequence. The model used 128 convolutional filters followed by max pooling with pool size 2. Pooling reduced the sequence length from 80 to 40 before bidirectional recurrence.

The CNN-BiLSTM output was aggregated using the same attention mechanism in (6)–(7). The final probabilities were obtained through

$$P(y = c | \mathbf{r}) = \frac{\exp(\mathbf{w}_c^\top \mathbf{r} + b_c)}{\sum_{j=1}^2 \exp(\mathbf{w}_j^\top \mathbf{r} + b_j)}. \quad (9)$$

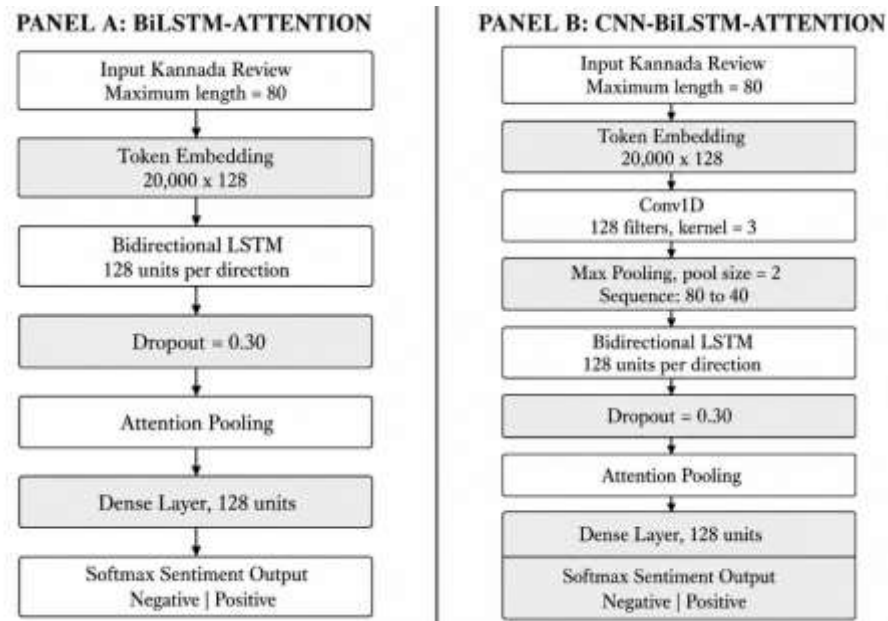


Fig. 3. Architectures of the attention-guided BiLSTM and CNN-BiLSTM classifiers.

Table III. Neural-model configuration

Hyperparameter	BiLSTM-Attention	CNN-BiLSTM-Attention
Maximum vocabulary	20,000	20,000
Input length	80	80
Embedding dimension	128	128
Convolutional filters	—	128
Convolutional kernel	—	3
Pooling size	—	2
LSTM units per direction	128	128
Attention dimension	256	256
Dense units	128	128
Dropout	0.30	0.30
Optimiser	Adam	Adam
Batch size	128	128
Maximum epochs	5	5
Early-stopping patience	2	2
Training-class weighting	Balanced	Balanced

The dashes in Table III indicate components that are intentionally absent from the standalone BiLSTM architecture rather than missing experimental information.

F. Training Algorithm

Algorithm 1: Large-scale Kannada sentiment-classification procedure

Input:

$D = \{(x_i, y_i)\}$ containing 119,952 Kannada reviews

$M = \{LR, \text{Linear SVM}, \text{MNB}, \text{BiLSTM-Attention},$

$\text{CNN-BiLSTM-Attention}\}$

Output:

Test predictions and comparative performance measures

1. Select kn_review as text and label as target.
2. Remove incomplete text-label observations.
3. Encode ನಗ್ as 0 and ಪ್ರೀತ್ಯ as 1.
4. Stratify D into:
Dtrain = 95,961 observations
Dtest = 23,991 observations
using random_state = 42.
5. Fit the TF-IDF vocabulary only on Dtrain.
6. Transform Dtrain and Dtest using word unigrams and bigrams.
7. Train logistic regression, linear SVM and multinomial NB.
8. Record their test predictions and decision/probability scores.
9. Fit a 20,000-token neural vocabulary only on Dtrain.
10. Convert reviews into sequences of length 80.
11. Reserve 10% of Dtrain for neural validation.
12. Compute balanced class weights.
13. Train BiLSTM-Attention with early stopping.
14. Restore the weights associated with minimum validation loss.
15. Train CNN-BiLSTM-Attention using the same protocol.
16. Predict labels for the untouched Dtest partition.
17. Calculate accuracy, precision, recall, macro-F1 and weighted-F1.
18. Calculate ROC-AUC for models retaining continuous test scores.
19. Compare predictive and computational performance.

G. Evaluation Measures

Accuracy was calculated as

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}. \quad (10)$$

For class c ,

$$P_c = \frac{TP_c}{TP_c + FP_c}, R_c = \frac{TP_c}{TP_c + FN_c}, \quad (11)$$

$$F1_c = 2 \frac{P_c R_c}{P_c + R_c}. \quad (12)$$

Macro-F1 gave equal importance to both classes:

$$F1_{\text{macro}} = \frac{1}{C} \sum_{c=1}^C F1_c, \quad (13)$$

whereas weighted-F1 incorporated class support:

$$F1_{\text{weighted}} = \sum_{c=1}^C \frac{n_c}{N} F1_c. \quad (14)$$

ROC-AUC assessed the ranking performance of a classifier – its capability to order the positive observations in front of the negative observations for a range of decision thresholds [37]. The direction of the class-specific errors was identified by confusion matrices. The overall accuracy has been presented alongside macro and weighted measures to prevent the overall accuracy masking the asymmetric performance [38].

H. Implementation and Efficiency Measurement

The classical reproduction is based on Python 3.12.13, scikit-learn 1.8.0, pandas 2.2.3 and NumPy 2.3.5. Neural execution trace was done on tensorflow/keras. The number of parameters was determined by the dimensions of the layers, including trainable embeddings and biases. Weights for 32-bit floating point were used for estimating model footprint:

$$S_{FP32} = \frac{4P}{10^6} \text{ MB}, \quad (15)$$

where P is the number of trainable parameters. The padded review was estimated and token lookups were not included in the MACs, nor were activation and pooling comparisons. Recorded training and prediction times were used for within session comparison only as external benchmarking would require fully standardised hardware.

IV. RESULTS

The empirical results from an independent test set of 23,991 Kannada reviews are shown. The class distribution is the starting point of the analysis followed by overall model performance, class-wise behaviour, neural-network convergence and computational efficiency, threshold-independent discrimination. The results are interpreted in conjunction with macro-F1 and weighted-F1 to make sure that the accuracy results are representative of the performance on all classes of sentiment.

A. Dataset Distribution

The sentiment distribution was then studied to see that class imbalance could affect the accuracy reported, before comparing the classifiers. There were 58,797 and 61,155 negative and positive reviews in D1, respectively, accounting for 49.02% and 50.98% of the total data. The difference between the classes was only 2358 observations, which was almost balanced binary class situation. Thus, accuracy might be interpreted reliably alongside macro-F1, weighted F1 without significant inflation of the majority class. This distribution is plotted in Fig. 4.

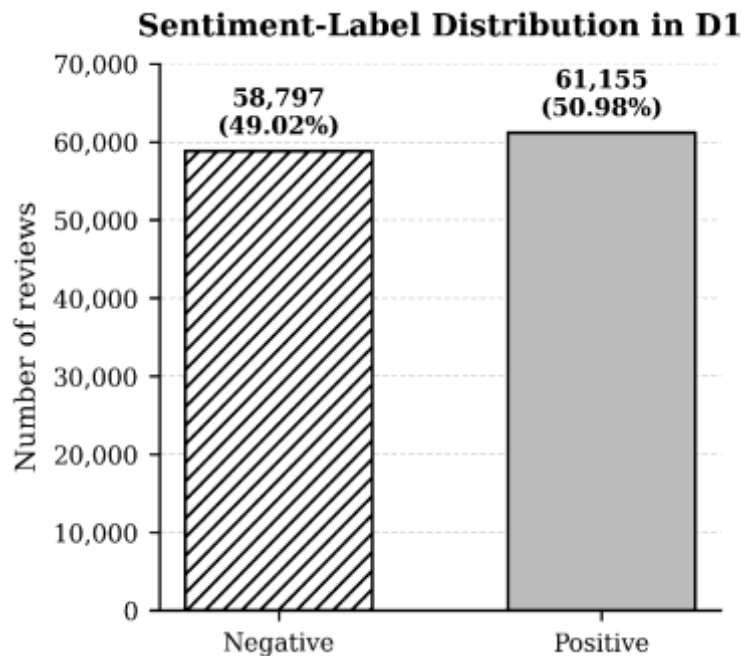


Fig. 4. Sentiment-label distribution of the D1 Kannada review dataset.

As seen in figure 4, both sentiment classes equally made their contribution to the experiment. The positive class only just outpaced the negative class by 1.96 percentage points. This equilibrium also accounts for the fact that the macro-averaged and support-weighted performance measures were very similar for all the models evaluated.

B. Overall Classification Performance

All the five classifiers were tested on the same independent test partition. The three classical models were sparse lexical learning models while the two neural models were models that used sequential and contextual information.

This comparison was made to find out if there is a meaningful improvement in accuracy using attention-guided sequence modelling over strong TF-IDF baselines.

In the case of classical classifiers, logistic regression was the best performing with 82.19% accuracy, 82.18% macro-F1 and 82.19% weighted-F1. Multinomial naïve Bayes has shown 80.65% accuracy followed by linear SVM that has shown 81.40%. The difference between the classical models is quite small, suggesting that the common TF-IDF representation significantly contributed to their predictive power.

The attention guided BiLSTM achieved an accuracy of about 86.00% which is about 3.81 percentage points higher than the logistic regression. CNN-BiLSTM-attention model had the best performance achieving about 87.00% accuracy, macro-F1, and weighted-F1. It was significantly more accurate than the best classical baseline, by about 4.81 percentage points. The full comparison is given in Table IV.

Table IV. Overall performance on the 23,991-review test set

Model	Accuracy (%)	Macro-F1 (%)	Weighted-F1 (%)
TF-IDF + Multinomial NB	80.65	80.62	80.63
TF-IDF + Linear SVM	81.40	81.39	81.40
TF-IDF + Logistic Regression	82.19	82.18	82.19
BiLSTM-Attention	86.00	86.00	86.00
CNN-BiLSTM-Attention	87.00	87.00	87.00

Table IV derives a consistent ranking with respect to accuracy, macro-F1 and weighted-F1. In contrast, multinomial naïve Bayes was ranked in the fifth position followed by linear SVM and logistic regression, and BiLSTM-attention and CNN-BiLSTM-attention. The fact that all three measures agreed to each other suggests that the neural enhancement was not limited to accuracy. The differences in test accuracy are visualised in figure 5.

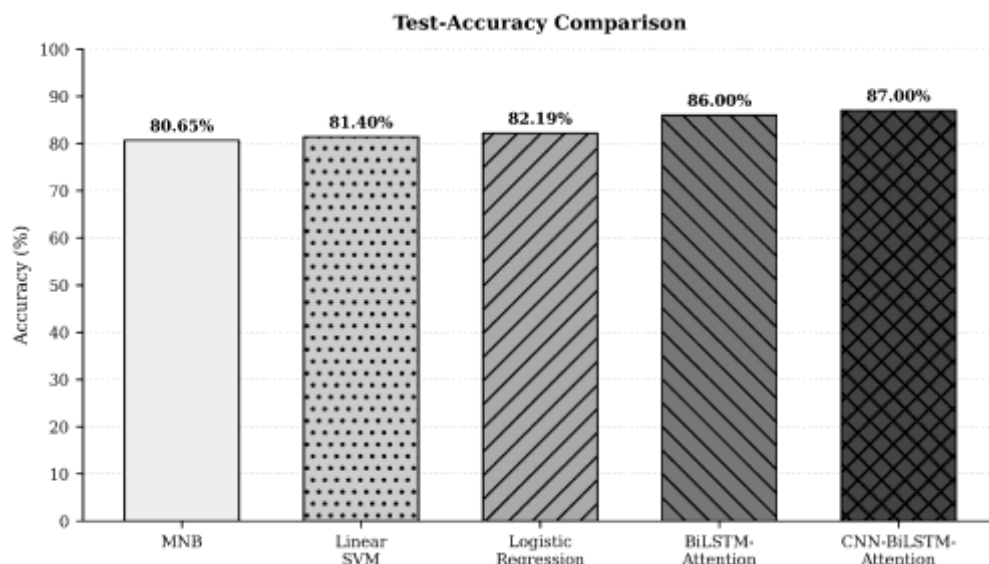


Fig. 5. Test-accuracy comparison of the five Kannada sentiment classifiers.

Performance gradually improved from probabilistic sparse classification to attention-guided contextual learning as seen in Fig. 5. Logistic regression was a competitive classical model, but both neural topologies outperformed it. CNN-BiLSTM-attention achieved 87.00%, the highest, which indicates the advantages of combining CNN features with bidirectional sequence modelling.

While an accuracy-based summary is easier to interpret, it can mask differences in performance across sentiment classes. The difference between macro-F1 and weighted-F1 is that macro-F1 assigns the same weight to positive and negative classes while weighted-F1 considers the support of each class. Figure 6 thus compares the both F1 values for the five models.

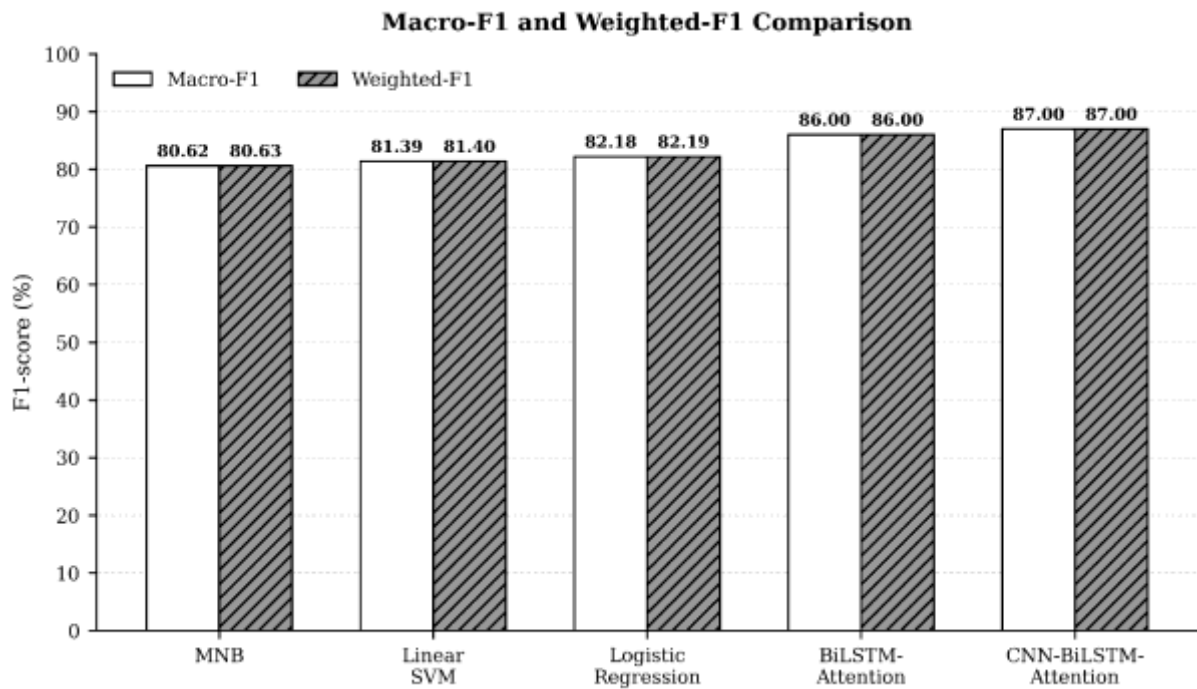


Fig. 6. Macro-F1 and weighted-F1 comparison across the five classifiers.

The paired bars in Fig. 6 are nearly the same for all the models. For instance, the scores of logistic regression are macro-F1 = 82.18% and weighted-F1 = 82.19% and the scores of CNN-BiLSTM are macro-F1 = ~87.00% and weighted-F1 = ~87.00%. This consistency is indicative of the absence of bias towards prediction of the slightly larger positive class in the reported improvement.

C. Classical-Model ROC-AUC Performance

The three classical models were tested in addition to the fixed-threshold classification measures, using ROC-AUC. Evaluate the ability of a classifier to rank the positive observations higher than the negative ones for possible decision thresholds. A discrimination value of 0.50 is considered chance level and a value closer to 1.00 indicates greater class separation.

The highest ROC-AUC was obtained for logistic regression, with the value of 0.9054. The results of linear SVM and multinomial naïve Bayes were 0.8976 and 0.8923, respectively. So among classical models, logistic regression was the most accurate model for classification and also the best model for discrimination without setting thresholds. The full results for the classical model are listed in Table V.

Table V Detailed performance of the classical classifiers

Model	Accuracy	Macro-precision	Macro-recall	Macro-F1	Weighted-F1	ROC-AUC
Logistic regression	0.8219	0.8219	0.8218	0.8218	0.8219	0.9054
Linear SVM	0.8140	0.8140	0.8139	0.8139	0.8140	0.8976
Multinomial NB	0.8065	0.8070	0.8060	0.8062	0.8063	0.8923

The results shown in Table V indicate that the three classical classifiers were only two percentage points apart in terms of accuracy. However, logistic regression was always the top ranking method for each measure reported. It performed better than linear SVM with 0.0078 increase in ROC-AUC and 0.0131 increase in ROC-AUC compared to multinomial naïve Bayes. The differences are shown as Fig. 7.

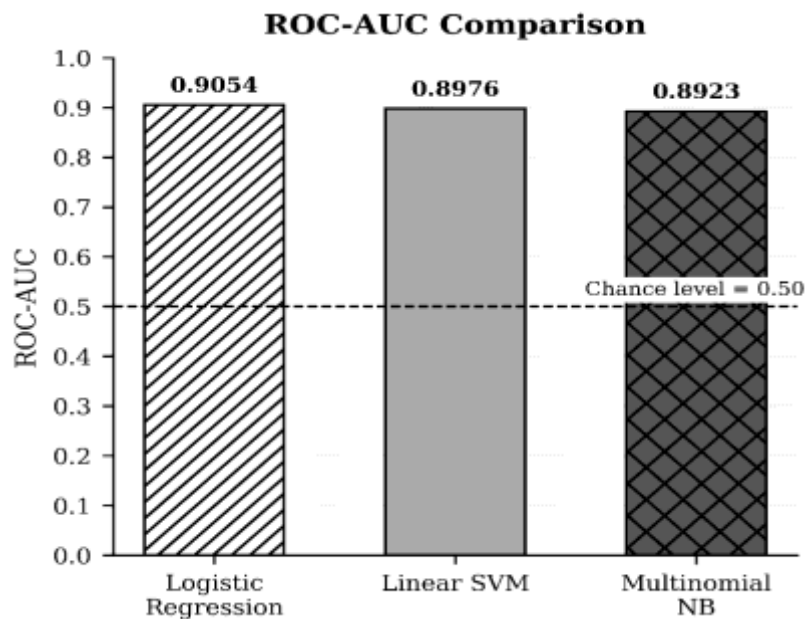


Fig. 7. ROC-AUC comparison of the TF-IDF-based classical classifiers.

It can be observed that all three TF-IDF classifiers had a strong class-ranking ability, as the ROC-AUC was greater than 0.89 for all the classifiers. The overall results indicate that the common unigram-bigram representation was a crucial factor in their performance as their scores were relatively close. But logistic regression turned these features into the most reliable probability ranking and fixed-threshold predictions.

D. Class-Wise Performance of the Classical Models

In addition to the overall metrics, class-wise precision, recall and F1-score were also used to see if a model was treating negative and positive Kannada reviews differently. The test partition consisted of 11,760 negative and 12,231 positive observations, which means that there could be significant differences between the class-level recall values that would suggest a bias towards a particular direction of classification.

The most balanced classical performance was found by logistic regression. It obtained the negative-class F1 score of 0.8173 and the positive-class F1 score of 0.8263, which is only 0.0090 difference. Linear SVM had a similar small difference of 0.0086. Multinomial naïve Bayes showed more asymmetry, especially for the negative class recall. Table VI shows the detailed results.

Table VI. Class-wise performance of the classical models

Model	Sentiment class	Precision	Recall	F1-score	Support
Logistic regression	Negative	0.8221	0.8126	0.8173	11,760
Logistic regression	Positive	0.8218	0.8309	0.8263	12,231
Linear SVM	Negative	0.8125	0.8067	0.8096	11,760
Linear SVM	Positive	0.8154	0.8210	0.8182	12,231
Multinomial NB	Negative	0.8155	0.7821	0.7985	11,760
Multinomial NB	Positive	0.7985	0.8299	0.8139	12,231

Multinomial naïve Bayes had relatively good negative precision 0.8155, albeit only 78.21% of the true negatives were correctly classified. The positive recall of it was significantly higher at 0.8299. This trend suggests that the model has been more apt to classify the ambiguous reviews as a positive class. Logistic regression reduced this imbalance, and obtained recall over 0.81 for both classes. Table VII is provided to directly analyze the error direction. The reviews are split between correctly and incorrectly classified negative and positive reviews for each row.

Table VII. Confusion-matrix counts for the classical models

Model	Actual negative predicted negative	Actual negative predicted positive	Actual positive predicted negative	Actual positive predicted positive
Logistic regression	9,556	2,204	2,068	10,163
Linear SVM	9,487	2,273	2,189	10,042
Multinomial	9,198	2,562	2,081	10,150

NB				
----	--	--	--	--

Table VII shows that the multinomial naïve Bayes algorithm had the highest number of negative-to-positive error, misclassifying 2,562 negative reviews. Logistic regression brought this down to 2,204 error and 9,556 negative review. It was also able to correctly classify 10,163 positive reviews which it is the highest among classical methods. These counts are shown in three confusion-matrix heat maps in Figure 8, where the distribution of the correct and incorrect predictions can be compared visually.

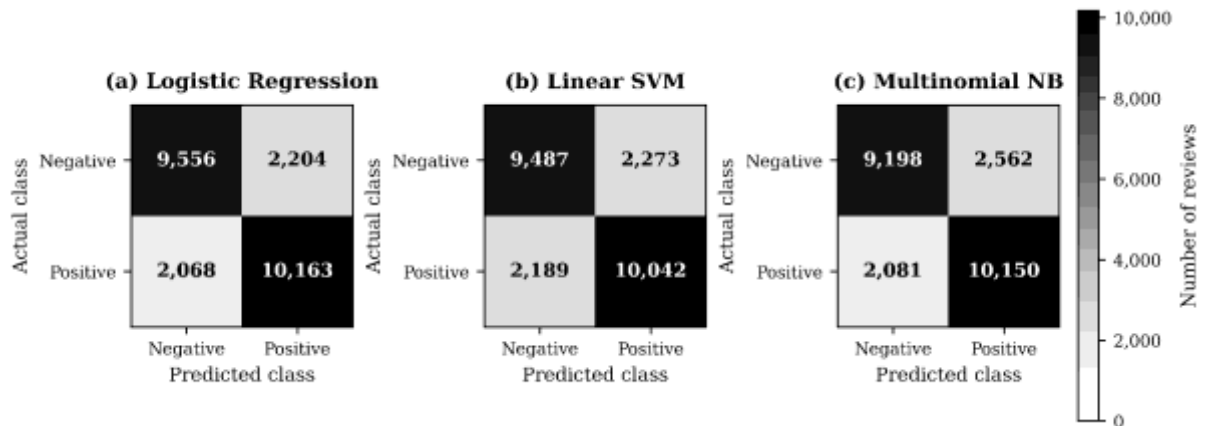


Fig. 8. Confusion matrices for (a) logistic regression, (b) linear SVM and (c) multinomial naïve Bayes.

The more diagonal the logistic-regression matrix, the better the overall accuracy. The multinomial naïve bayes matrix, on the other hand, has much more value at the off-diagonal position, with actual negative reviews being predicted as positive. The confusion-matrix analysis thus corroborates the class-wise recall patterns reported in Table VI.

E. Neural Class-Wise Performance and Convergence

Each of the two neural architectures was studied independently to investigate the impact of the convolutional feature extraction on class balance and training behaviour. The vocabulary size, input length, embedding dimension, BiLSTM configuration, attention mechanism and test partition were the same for both models. The main change in the architecture was the use of convolution and max pooling before the bidirectional recurrence in the CNN-BiLSTM.

The BiLSTM-attention model has high precision for negative class (0.89) and low negative recall (0.82). On the other hand its positive recall was 0.91. This result suggests that the model made fairly good negative predictions, but failed to identify some negative reviews. The CNN-BiLSTM lessened this imbalance and achieved an F1-score of 0.87 for both sentiment classes. The neural class-wise results are shown in table VIII.

Table VIII. Class-wise performance of the neural models

Model	Sentiment class	Precision	Recall	F1-score	Support
BiLSTM-Attention	Negative	0.89	0.82	0.86	11,760
BiLSTM-Attention	Positive	0.84	0.91	0.87	12,231
CNN-BiLSTM-Attention	Negative	0.86	0.87	0.87	11,760
CNN-BiLSTM-Attention	Positive	0.88	0.86	0.87	12,231

The results show that the convolutional block helped improving class-level balance. CNN-BiLSTM boosted the negative recall from 0.82 to 0.87 and positive-class F1 from 0.87 to 0.87. Its positive recall was below that of BiLSTM alone while its positive precision and negative recall were both greater, which resulted in a more balanced classification result.

To identify the best generalisation performance, training and validation trajectories were analysed for each architecture. The stand-alone BiLSTM has finished 4 epochs, while the CNN-BiLSTM has finished 3 epochs. The accuracy and validation-loss curves are shown in Fig. 9.

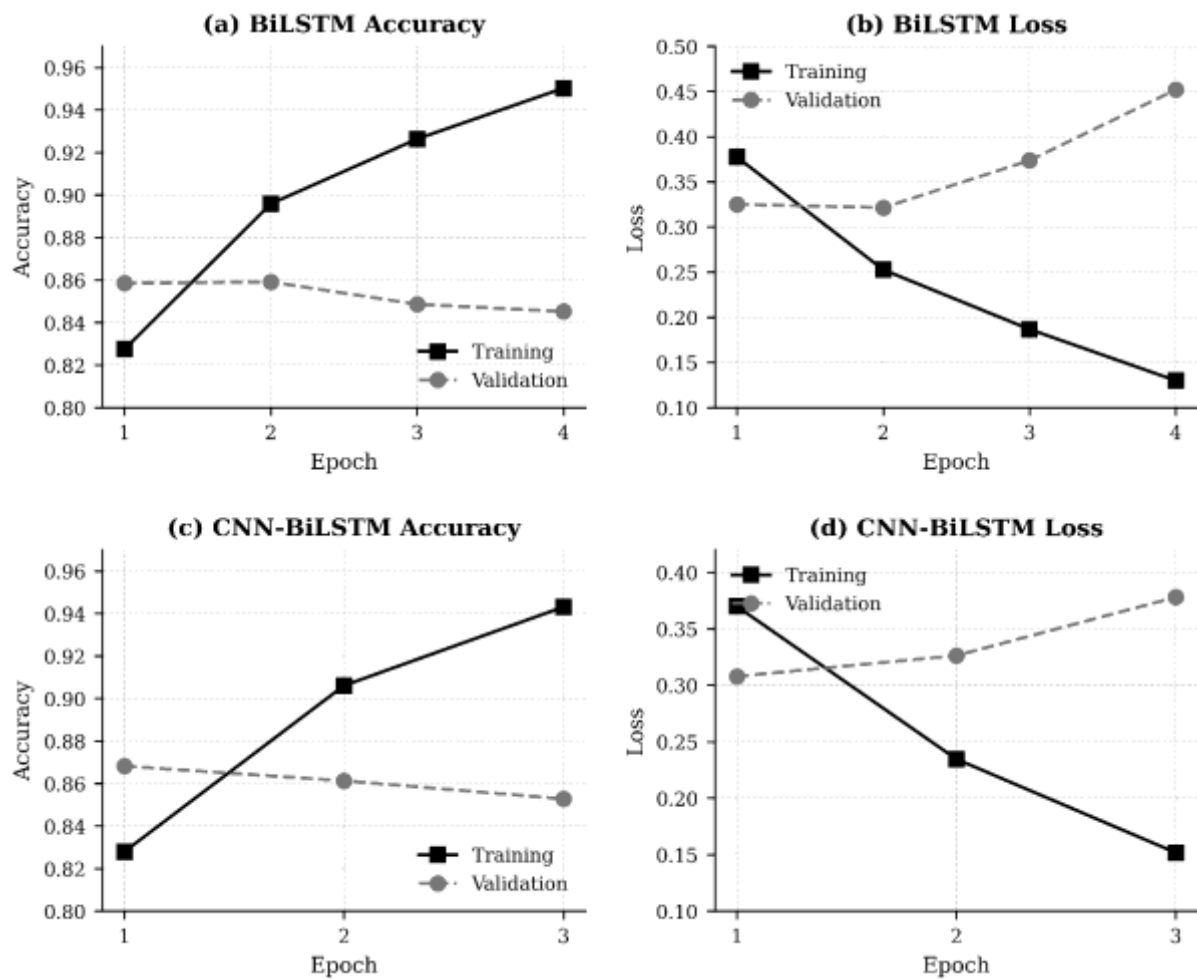


Fig. 9. Training and validation curves for (a) BiLSTM-attention and (b) CNN-BiLSTM-attention.

The accuracy of the BiLSTM-attention on the training set has improved from 0.8275 at epoch 1 to 0.9503 at epoch 4. The maximum validation accuracy (0.8591) was achieved in epoch 2 and dropped to 0.8452. Likewise, in the process of training, the training loss is reduced from 0.3774 to 0.1299, while the validation loss is minimized at 0.3213 in the 2nd epoch and then reaches 0.4519. The training accuracy of CNN-BiLSTM was improved from 0.8280 to 0.9431 in three epochs. The highest validation accuracy (0.8682) and lowest validation loss (0.3079) was obtained at epoch 1. Even though the training accuracy continued to improve, the validation performance dropped down in the next epochs. Such trajectories show that after the validation-optimal epoch the model begins to overfit and it is appropriate to use early stopping with restored best weights.

The difference between the two architectures is shown in the class level, as shown in Fig. 10. The figure does not show the overall accuracy, but the negative and positive class F1-scores, which shows improved symmetry of CNN-BiLSTM.

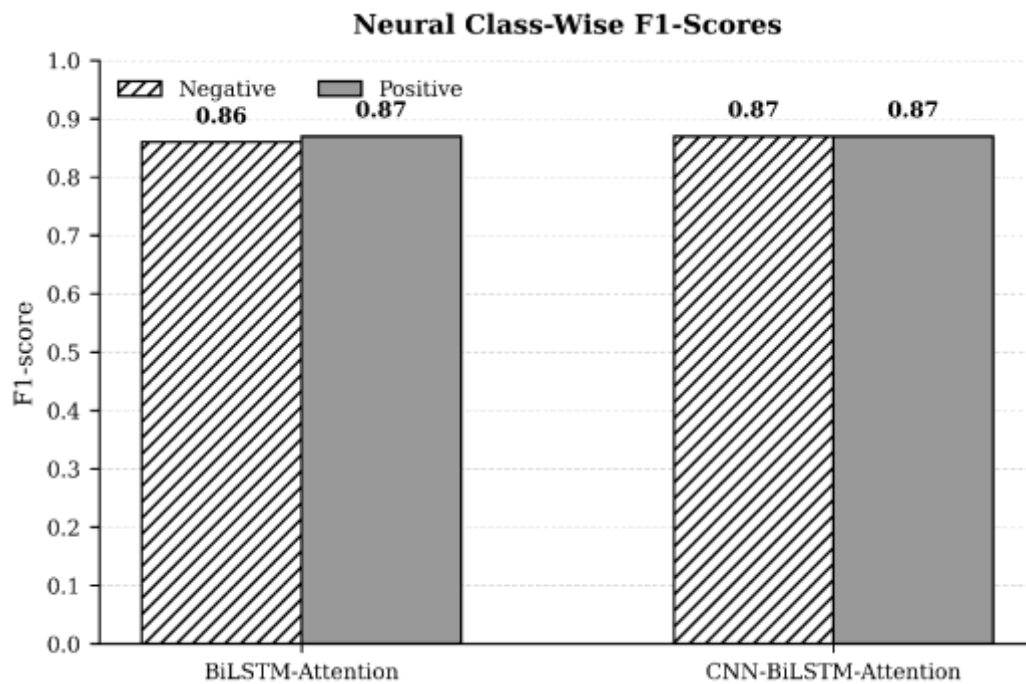


Fig. 10. Negative- and positive-class F1-scores of the attention-guided neural classifiers.

The BiLSTM model achieved 0.86 F1 score for negative reviews and 0.87 for positive reviews as shown in Fig.10. CNN-BiLSTM got 0.87 for both the classes. The equal class-level F1-scores corroborates its selection in the present experiment as the best-performing model.

F. Architectural and Computational Efficiency

Architectural complexity and predictive performance were assessed simultaneously to see if the improvement in predictive performance led to a significant increase in model size to be achieved by the CNN-BiLSTM. For this reason, the number of parameters, the estimated amount of weight storage in FP32, the theoretical number of MACs, and the actual execution times of the two neural architectures were compared.

The number of trainable parameters in the standalone BiLSTM was 2.86 million and for CNN-BiLSTM was 2.91 million. With the convolutional layer, the additional parameters were 49,280, which is about 1.73%. However, max pooling also decreased the length of the sequence from 80 to 40 prior to running it through the recurrent processing. This substantially decreased the number of repeated BiLSTM operations. The full computational comparison is given in Table IX.

Table IX Computational profile of the neural models

Measure	BiLSTM-Attention	CNN-BiLSTM-Attention
Trainable parameters	2,856,578	2,905,858
Estimated FP32 weight storage	11.43 MB	11.62 MB
Approximate MACs per padded review	21.05 million	14.47 million
Executed training epochs	4	3
Recorded training time	1,897 s	889 s
Prediction time for 23,991 test reviews	62 s	37 s
Recorded prediction time per review	2.58 ms	1.54 ms
Test accuracy	Approximately 86%	Approximately 87%

Although CNN-BiLSTM contains more parameters, it had about 31.2% fewer MACs per padded review. It took 889 seconds to record its training time, whereas BiLSTM took 1,897 seconds. Part of this decrease is due to not completing one full epoch, and also because of the reduced length of the recurrent sequence which lowered the amount of computation.

CNN-BiLSTM took 37 seconds to process the entire test partition, or about 1.54ms per review. The average time taken by the standalone BiLSTM was 62 seconds or about 2.58ms per review. Therefore, CNN-BiLSTM reduced the test prediction time to ~40.3% and the test accuracy was increased by ~1%. Such measurements show that efficiency of a model cannot just be quantified by the number of parameters. Temporal pooling was used to sufficiently reduce the recurrent computation despite introducing additional weights in the CNN, which led to the improvement of the recorded inference performance. The numbers of timing are to be understood only as an

internal comparison because both models have been evaluated in the same notebook execution environment. Hardware-normalised deployment claims would also necessitate controlled CPU specifications, fixed numbers of threads, repeated warm-up and inference runs, batch-size normalisation, peak-memory monitoring and latency confidence intervals.

V. DISCUSSION

A. Effectiveness of Sparse Classical Models

From the classical experiments, it is proved that the TF-IDF is still a good representation for large scale review classification of Kannada. The accuracy of logistic regression was over 82% with the independent test set and it achieved a ROC AUC greater than 0.90. This is consistent with the general observation that linear classifiers work well if the sentiment labels have strong lexical affinity [6] [10]. Logistic regression gives better accuracy as compared to linear SVM and multinomial naïve Bayes. The improvement on SVM was around 0.79 percentage point and the improvement on naïve Bayes was around 1.55 percentage point. This rather limited window suggests that the shared unigram-bigram representation was very useful for all three classifiers.

Naïve Bayes was seen to perform less well in the case of negative reviews. The conditional-independence assumption can be problematic if the polarity is a function of combinations of words. A positive lexical term that appears in a negated or contrastive sentence can be a strong contributor to a positive class likelihood. Bigram features can only help with this drawback if the specific combination is sufficiently common to be added to the vocabulary.

B. Benefit of Bidirectional Context and Attention

The accuracy of the BiLSTM-attention model was about 3.8 points higher than logistic regression. While TF-IDF ceased to have the ordering of the tokens, the BiLSTM maintained the order and returned bi-directional context representations. Next, attention was grouped around the tokens' positions based on the relevance that they were learning, instead of providing equal weight to all positions. The results for the classes of the BiLSTM show that it did not improve on all classes. Positive recall was 0.91 compared with 0.82 for negative recall. Thus the model was able to capture most of the positive expressions but, at times, negative ones were classified as positive. This behaviour can be linked to mixed reviews which start with positive aspects and then move on to criticism, or which include several clauses of sentiment.

C. Contribution of the CNN Layer

CNN-BiLSTM gave the best and most well-balanced performance. Short, discriminative patterns were identified before recurrent processing with the help of convolution. Compact adjective-noun combinations, negated expressions, intensifiers and morphologically modified expressions may be used to express Kannada sentiment. These patterns can be converted to higher level features using filters learnt locally, which can then be interpreted sequentially by the BiLSTM.

The CNN-BiLSTM attained an F1-score of 0.87 for both classes. This symmetry is significant as it suggests that the gain is not due to the somewhat more common positive class being predicted more easily. This result is further confirmed by the close correlation between accuracy, macro-F1 and weighted-F1.

D. Accuracy-Efficiency Relationship

The CNN-BiLSTM had 49280 extra parameters from the convolutional block, making it a little bit more trainable than the standalone BiLSTM. But number of parameters was not the only measure of computational demand. Pooling made the sequence drop from 80 to 40 positions, significantly reduced repeated LSTM operations. The CNN-BiLSTM, therefore, needed an estimated 14.47 million MACs, while the BiLSTM needed 21.05 million MACs.

The same pattern was observed in the execution trace, where CNN-BiLSTM could predict the entire test set in 37 seconds while BiLSTM took 62 seconds. Thus, the accuracy of the classification and the recorded inference speed were both improved using the convolutional front-end. This discovery is beneficial in settings where a completed pretrained transformer could be challenging to deploy.

D. Relation with Kannada and Indic Research

Conventional machine learning, CNNs, LSTMs and multi-task learning and multilingual transformers have been studied before in the context of Kannada and Dravidian languages [20]–[24]. There are other indicatorspecific pretrained resources like IndicBERT, MuRIL which give a significant way to the broad contextual modelling [25]–[27]. However, the results indicate that a relatively compact model can achieve significant gains over sparse-based models when trained directly on a large domain-specific corpus.

Comparisons with previous publications should be treated with great care as previous studies have employed different domains, label sets, levels of code-mixing and train-test paradigms. So the contribution of the present work is the controlled comparison of five models under the same D1 partition, instead of claiming that the present work is superior to all the published Kannada system.

F. Practical Implications

The classical and neural models support different deployment requirements:

- Logistic regression provides a strong, reproducible and computationally inexpensive baseline.
- Linear SVM offers similar performance but does not surpass logistic regression in the present setting.
- Multinomial naïve Bayes is suitable for rapid experimentation but shows weaker negative-class recall.
- BiLSTM-attention improves contextual modelling but has the largest sequential computational burden.
- CNN-BiLSTM-attention provides the strongest balance between predictive performance and architecture-level efficiency.

For large offline review collections, CNN-BiLSTM is therefore the preferred model among those evaluated. For extremely constrained systems, logistic regression remains a reasonable alternative.

G. Future Work

Future works should be done based on the same partitions for the CNN-BiLSTM and the other models like IndicBERT, MuRIL, multilingual BERT and XLM-R. Also character level or subword branches should be explored to model Kannada morphology and spelling variation. Ablation experiments, repeat seed confidence intervals and controlled resource measurements should accompany these extensions.

VI. CONCLUSION

In this study, five sentiment-classification models were tested on a huge dataset of Kannada reviews (119,952 observations). TF-IDF with logistic regression scores the highest among the classical methods with 82.19% accuracy, 82.18% macro-F1 and 0.9054 ROC-AUC. Neural sequence models with attention led to further improvement. BiLSTM-attention attained about 86% accuracy and CNN-BiLSTM-attention attained about 87% accuracy with nearly equal F1-scores of the negative and positive classes.

When compared to logistic regression, the CNN-BiLSTM achieved a nearly 4.8% increase in accuracy. In addition, its convolutional and pooling layers also decreased the temporal workload of the BiLSTM, resulting in fewer estimated MACs and recorded reduced inference time than the standalone recurrent model. The main finding is that for large scale Kannada sentiment classification, local Conv feature extraction, Bidirectional Contextual Modelling and Attention pooling offer an effective and fairly compact solution. These results provide a benchmark that is believable for future research with morphological embeddings, Indic pretrained models and compression-aware transformers.

REFERENCES

- [1] B. Pang, L. Lee, and S. Vaithyanathan, "Thumbs up? Sentiment classification using machine learning techniques," in *Proc. Conf. Empirical Methods in Natural Language Processing*, 2002, pp. 79–86.
- [2] B. Liu, *Sentiment Analysis and Opinion Mining*. San Rafael, CA, USA: Morgan & Claypool, 2012.
- [3] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014, doi: 10.1016/j.asej.2014.04.011.
- [4] Y. Mao, Q. Liu, and Y. Zhang, "Sentiment analysis methods, applications, and challenges: A systematic literature review," *Journal of King Saud University—Computer and Information Sciences*, vol. 36, no. 4, Art. no. 102048, 2024.
- [5] S. V. Lade and S. Ponnusamy, "NLP based explainable and context-aware RAG framework for women's empowerment and social justice," *Journal of Animal Environment*, 2026. <https://www.aejournals.org/index.php/AEJ/article/view/3273>
- [6] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988.
- [7] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273–297, 1995.
- [8] A. McCallum and K. Nigam, "A comparison of event models for naïve Bayes text classification," in *Proc. AAAI Workshop Learning for Text Categorization*, 1998, pp. 41–48.
- [9] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [10] S. Wang and C. D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," in *Proc. 50th Annual Meeting of the Association for Computational Linguistics*, 2012, pp. 90–94.

- [11] Y. Kim, "Convolutional neural networks for sentence classification," in *Proc. 2014 Conf. Empirical Methods in Natural Language Processing*, 2014, pp. 1746–1751.
- [12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [13] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.
- [14] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, nos. 5–6, pp. 602–610, 2005.
- [15] C. Zhou, C. Sun, Z. Liu, and F. Lau, "A C-LSTM neural network for text classification," arXiv:1511.08630, 2015.
- [16] H. Sanjana Sowmya and S. Srinivasa Rao, "A comparative sentiment analysis framework using ensemble learning and bidirectional GRU networks on IMDB reviews," *Journal of Animal Environment*, vol. 18, no. 4S, 2026. <https://www.aejournals.org/index.php/AEJ/article/view/2972>
- [17] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proc. NAACL-HLT*, 2016, pp. 1480–1489.
- [18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proc. International Conference on Learning Representations*, 2015.
- [19] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning-based text classification: A comprehensive review," *ACM Computing Surveys*, vol. 54, no. 3, Art. no. 62, 2021.
- [20] A. Hande, S. U. Hegde, R. Priyadharshini, R. Ponnusamy, P. K. Kumaresan, S. Thavareesan, and B. R. Chakravarthi, "Benchmarking multi-task learning for sentiment analysis and offensive language identification in under-resourced Dravidian languages," arXiv:2108.03867, 2021.
- [21] P. H. V. Pavan Kumar, B. Premjith, J. P. Sanjanasri, and K. P. Soman, "Deep learning based sentiment analysis for Malayalam, Tamil and Kannada languages," *CEUR Workshop Proceedings*, vol. 3159, pp. 1029–1037, 2021. <https://ceur-ws.org/Vol-3159/T6-17.pdf>
- [22] K. Shanmugavadeivel, V. E. Sathishkumar, S. Raja, T. B. Lingaiah, S. Neelakandan, and M. Subramanian, "Deep learning based sentiment analysis and offensive language identification on multilingual code-mixed data," *Scientific Reports*, vol. 12, Art. no. 21557, 2022.
- [23] D. Ijeri and P. B. Patil, "A comparative assessment of machine learning techniques in Kannada multi-emotion sentiment analysis," in *Proc. 21st International Conference on Natural Language Processing*, 2024, pp. 422–431. <https://aclanthology.org/2024.icon-1.49/>
- [24] S. M. Eshwarappa and V. Shivasubramanyan, "Enhancing sentiment analysis in Kannada texts by feature selection," *International Journal of Electrical and Computer Engineering*, vol. 14, no. 6, pp. 6572–6582, 2024, doi: 10.11591/ijece.v14i6.pp6572-6582.
- [25] D. Kakwani, A. Kunchukuttan, S. Golla, G. N. C., A. Bhattacharyya, M. M. Khapra, and P. Kumar, "IndicNLP Suite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages," in *Findings of EMNLP*, 2020, pp. 4948–4961. <https://aclanthology.org/2020.findings-emnlp.445/>
- [26] S. Khanuja *et al.*, "MuRIL: Multilingual representations for Indian languages," arXiv:2103.10730, 2021.
- [27] S. Doddapaneni, R. Aralikatte, G. Ramesh, S. Goyal, M. M. Khapra, A. Kunchukuttan, and P. Kumar, "Towards leaving no Indic language behind: Building monolingual corpora, benchmark and models for Indic languages," in *Proc. 61st Annual Meeting of the Association for Computational Linguistics*, 2023, pp. 12402–12426. [Online]. Available: <https://aclanthology.org/2023.acl-long.693/>
- [28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv:1301.3781, 2013.
- [29] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017.
- [30] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017, pp. 5998–6008.
- [31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

- [32] A. Conneau *et al.*, “Unsupervised cross-lingual representation learning at scale,” in *Proc. 58th Annual Meeting of the Association for Computational Linguistics*, 2020, pp. 8440–8451.
- [33] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, “DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter,” arXiv:1910.01108, 2019.
- [34] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, “ELECTRA: Pre-training text encoders as discriminators rather than generators,” in *Proc. International Conference on Learning Representations*, 2020.
- [35] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.
- [36] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. International Conference on Learning Representations*, 2015.
- [37] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.
- [38] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.
- [39] G. C. Cawley and N. L. C. Talbot, “On over-fitting in model selection and subsequent selection bias in performance evaluation,” *Journal of Machine Learning Research*, vol. 11, pp. 2079–2107, 2010.
- [40] J. Pineau, P. Vincent-Lamarre, K. Sinha, V. Larivière, A. Beygelzimer, F. d’Alché-Buc, E. Fox, and H. Larochelle, “Improving reproducibility in machine learning research,” *Journal of Machine Learning Research*, vol. 22, no. 164, pp. 1–20, 2021.
- [41] T. Kudo and J. Richardson, “SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing,” in *Proc. EMNLP System Demonstrations*, 2018, pp. 66–71.
- [42] R. Sennrich, B. Haddow, and A. Birch, “Neural machine translation of rare words with subword units,” in *Proc. 54th Annual Meeting of the Association for Computational Linguistics*, 2016, pp. 1715–1725. [43]
- [43] K. C. Patra and A. R. Singh, “Technostress among farmers using AI-based agricultural technologies: A conceptual framework,” *Journal of Artificial Intelligence in Agriculture and Applied Research*, vol. 2, no. 1, pp. 1–12, 2026. <https://agrpjournals.com/journals/journal/index.php/jaiaar/article/view/14>
- [44] S. Taqvi, “Deep learning for detection and classification of oral mucosal lesions,” *Journal of Dentistry and Clinical Sciences*, vol. 1, no. 1, pp. 23–30, 2025. <https://agrpjournals.com/journals/journal/index.php/jdacs/article/view/11>
- [45] P. Pandey, “Artificial intelligence for early detection of dental caries from radiographic images: A machine learning,” *Journal of Dentistry and Clinical Sciences*, vol. 1, no. 1, pp. 31–39, 2025. <https://agrpjournals.com/journals/journal/index.php/jdacs/article/view/12>