[1]Hiteshkumar Barot*,
Viral Parekh2,
Mehul Barot3,
Yaashu Dave4

Enhancing Sentiment Analysis Using CNN: Evaluative Study of Traditional Models

Journal of Electrical Systems

Abstract: This paper forms a research work in which a novel method for performing sentiment analysis is introduced in the form of convolutional neural networks and tested against other established algorithms including logistic regression, Naive Bayes, and decision tree classifiers. Our proposed method is to apply deep learning approaches to classify appropriate sentiments associated with the textual data, which are crucial for estimating users' satisfaction, their mood changes depending on the environment, or how they respond to specific events. CNNs have shown their capability in image analysis through the feature of being able to extract and analyze local features with robustness. This capability is extended to text data as well while at the same time preserving the context within which the sentiment of the data is being expressed. For multiple datasets, it has been observed that accuracy and F1 score were higher for the proposed custom CNN model as compared to traditional methods. As confirmed by these results, CNNs can contribute to raising the sentiment analysis and sentiment prediction accuracy in the different application contexts.

Keywords: *Sentiment Analysis, Convolutional Neural Net-works (CNN), Logistic Regression, Naive Bayes, Decision,* Trees, Natural language processing (NLP).

INTRODUCTION

On social media sites, you can find a wide variety of information. A lot of new data is being created. Twitter is very popular, with over 330 million monthly users in 2023. There are so many people on it, and their tweets are in real time. People post millions of tweets every day, giving us a ton of data that we can look at to understand how people are feeling. By looking at Twitter, we can learn what users are thinking, how happy they are, how they are emotionally reacting to events, products, places, etc.

Sentiment analysis is important. The goal is to classify pieces of text into positive, negative, or just plain neutral emotion. This type of factor is very useful for finding out what people are thinking and processing language intelligently. Old fashioned techniques like Support Vector Machines and Naive Bayes are often used for this type of tasks. They learn each word one at a time using one, but they may miss the bigger picture because key words are selected without considering the words around them.

The world in particular experiencing an amazing field of image display and speech recognition are the most recent flashy deep learning models. Convolutional neural networks, which are most commonly used for image content recognition, also enhance the performance on language tasks. Now, they treat words as if they were pixels in a picture and fit groups of them together to see how different parts are related. In this way, they are able to extract salient information from text much better than conventional methods.

The foundation for Twitter sentiment analysis was laid by Go et al. (1), who introduced distant supervision as a method for classifying sentiment in tweets. In this research, we endorse a CNN-primarily based on technique for sentiment evaluation on Twitter data. Our CNN version leverages the convolutional operation to extract functions from sequences of words, thinking about their interrelatedness to enhance sentiment category accuracy. We

[1]Research scholar, Department of Technology and Engineering-CE, C U Shah University, Wadhawan, Gujarat, India
*ORCID : https://orcid.org/0009-0001-9873-4402
[2]Dean-Faculty of Engineering, Monark University, Gujarat, India, 3Department of Computer Engineering, LDRP-Institute of Technology and Research, Gujarat, India, 4Department of Computer Engineering, LDRP- Institute of Technology and Research, Gandhinagar-382016, Gujarat, India.
*Corresponding Author's Email:hitesh.ldrp@gmail.com

compare the overall performance of our CNN version in opposition to conventional devices gaining knowledge of models, along with Logistic Regression, Naive Bayes, and Decision Trees, by using the benchmark datasets. The effects display that our CNN version outperforms those conventional techniques in phrases of accuracy and F1-score, highlighting its capability for powerful sentiment evaluation.

Through the harnessing of CNNs, our method seeks to obtain more accurate sentiment predictions, and thereby improve the general comprehension of people's sentiments and emotions considered on the social media data. The findings obtained from this evaluation may be valuable for enterprises, authorities, and researchers who want to understand the public perception of a tray of things and events.

The paper is structured as follows: illustrates the different methods for related works then explains our proposed methodology. Finally discuss about our experiments and results then perform evaluative study on different dataset which we have used, along with a comparison with other methods. Finally, we conclude our research with its performance.

RELATED WORK

Recent years have seen developing interest in the application of Sentiment analysis among researchers as well as organizations; mainly sentiment analysis on social media such as twitter, face book, etc. This growing interest is due to the massive amount of data created by users that facilitates the observation of trends in opinions and attitudes. Four papers take a different approach than the conceptual articles cited above, with works seeking to refine and advance the technique of sentiment analysis through the use of other methodologies to ascertain revised and more reliable sentiment analysis results.

SVM and Naive Bayes algorithms are some of the oldest algorithms that have been applied for SA. Such approaches are mostly the process of extracting specific features from the data materials and classification. For example, Go, Bhayani and Huang (1) used Naive Bayes and SVM classifiers for the sentiment classification of the tweets leading to moderate gain. Still, these traditional methods are used not very effectively because they fail to identify the context of the textual data, which in turn affects the sentiment analysis.

The new methods of deep learning have added more complications to the processes of sentiment analysis. CNNs have proved efficient in many NLP tasks as they can model local and abstract information present within the language data. CNNs were also proven by Kim (2) to be effective in sentence classification where the authors reported better results than the conventional techniques. Likewise, Severyn and Moschitti (3) used a CNN model relating to twitter sentiment analysis based on its capacity to out-compete typical strategies.

Word embedding have played a significant role in natural language processing tasks and have significantly improved most text classification models; the Word2Vec model by Mikolov et al. (4) and the GloVe model by Pennington, Socher, and Manning (5).

The process of sentiment analysis has advanced over time especially with the enhancement of deep learning models that have enhanced the classification aspect Sentiment analysis in its initial stages was achieved using distant supervision for tweets classification as seen with Go, Bhayani, and Huang in their study (1).

Apart from the developments in model architecture, several studies were performed in aspects of pre-processing for sentiment analysis improvement. Word vectors like Word2Vec and GloVe are commonly used to map words into a dense numerical vector space by preserving semantics of the words for efficiency. Mikolov et al. (4) and Pennington, Socher, and Manning (6) have proved the utilization of these embedding in different NLP activity, for example sentimental analysis. In developing our approach for sentiment analysis using convolutional neural networks (CNNs), we drew a little inspiration from several key studies in the field. Notably, Kim (2) demonstrated the effectiveness of CNNs for sentence classification, showing how convolutional layers can be used to extract meaningful features from text (2).

Thus, the present contribution expands from these developments by using a CNN-based framework for sentiment analysis on data extracted from Twitter. We choose the CNN model because it shows the best results when it used to capture the local relationships and hierarchies in text. As for the case of text sequences, the similar consideration to image data enlightens our model for analyzing relationships among words.

All in all, it can be stated that although the traditional ma- chine learning approaches provided the foundation for the sentiment analysis, the deep learning methods, more specifically the CNNs, add tangible improvements in the textual nuances' representation along with better classification accuracy. Thus, in what follows, the proposed paper aims at building a novel CNN approach for Sentiment Analysis specifically designed for textual data, and comparing it with existing traditional techniques in order to showcase its efficiency.

METHODOLOGY OF BASELINE MODELS

In our research, we developed a Convolutional Neural Net- work (CNN) model for sentiment analysis and compared its performance against several well-known pre-trained models: There are many of them, but five of the most used are; Logistic Regression, Naive Bayes, Decision Tree, Support Vector Machine, and k- Nearest Neighbors. These traditional models are used for the sentiment analysis tasks mainly because of their simplicity and efficiency. Regarding these models, we tried to set up the idea of the baseline by which we would be able to compare the performance of the custom CNN model built in the further parts of the study. Hence we decided to compare our custom CNN model with Logistic Regression, Naive Bayes and Decision Tree.

Logistic Regression

Logistic Regression can be defined as a basic classifying model frequently applied to problems of binary classification. It reproduces the likelihood of a certain binary value depending on one or more independent variables. In our study, we employed the following approach for Logistic Regression:

- **Data preprocessing:** Text Cleaning: Basically, the text data was cleaned where the URLs, mentions, hashtags and special characters were deleted. Lowercase conversion of the text was also done as well as the removal of single characters and excessive spaces.
- **Feature Extraction:** Using of TF-IDF (Term Frequency-Inverse Document Frequency) vectorization was chosen to transform text into numeric characteristics. The case was reduced to the maximum of 3000 features and the presence of English stop words were excluded.
- **Model Training and Evaluation:** Training: The model which is Logistic Regression is in the sklearn from the python library. Linear model was performed with the options to run a maximum of 1000 iterations. Evaluation: The dataset is already given separately for train and test for both IMDB and Amazon dataset. Cross entropy and accuracy were measured by accuracy score, recall, precision, and F1-score by classification report.

Naive Bayes

Naive Bayes is a probabilistic classifier that has actually been evolved on the basis of the Bayes theorem, but have strong assumption that is made between the features of different classes or categories. For sentiment analysis, we implemented the following steps:

- **Data Preprocessing:** Text Cleaning: As in the case of Logistic Regression models, similar data processing was done as a part of the data preparation.
- **Feature Extraction:** TF-IDF Vectorization: Naive Bayes model took the maximum of 5000 features for predicting the sentiment of the used reviews.
- **Model Training and Evaluation: Training**: Multinomial Naive Bayes classifier is from the sklearn package of python. Naive bayes was used. Evaluation: Finally, the model accuracy together with other classification measurements was established using prediction analysis on the test set.

Decision Tree

Decision Tree is simple to implement classification method where the structure of a tree is built to make prediction based on value of a feature. The approach used for Decision Tree in our study included:

- **Data Preprocessing:** Text Cleaning: Preprocessing was similar to the one applied in the case of Logistic Regression and Naive Bayes.
- **Feature Extraction:** TF-IDF Vectorization: For Decision Tree model, the maximum features was taken as 5000.

- **Model Training and Evaluation:** Training: It is the Decision Tree Classifier from sklearn. Tree was employed with the ‘entropy’ criterion.
- **Evaluation:** Thus, accuracy and classification metrics were calculated basing on the predictions made on the test set.

In these models, the classification is defined as binary classification with two distinct labels: Positive and Negative

- **Positive**: The Positive label is given to those texts where the opinion is positive in terms of approval/satisfaction or happiness.
- **Negative**: The Negative label is assigned to text that contains a negative attitude and suggests disapproval, or disappointment or even upset.

In sentiment analysis these binary sentiment labels are important since they are used for measuring the performance of the models basing the performance metrics on the accuracy, precision, recall as well as the F score. When it comes to the machine learning models applied to the text classification process, the process includes the identification of the proper patterns and other triggers that determine if the reviewed samples are positively or negatively oriented.

PROPOSED METHODOLOGY BY USING CNN

As we move onto the construction of the CNN based models for sentiment analysis part, we took some inspiration from a few seminal research works in this line. Interestingly, Kim (1).established the possibility of using CNNs in sentence classification asserting that convolutional layers are able to identify and capture relevant features from the text (2). Our methodology builds upon the work of Severyn and Moschitti (3), who successfully employed deep CNNs for Twitter sentiment analysis.

Data Collection and Preprocessing

Dataset Description: here we both have used IMDB as well as Amazon polarity dataset; it primarily help us to analyze the performance of our CNN model more exhaustively for different dataset size and difficulty level. The IMDB dataset is used as a baseline whereby information on the performance of the model working on a small, well-proportioned, dataset is received. However, the Amazon Polarity, which contains a greater amount of data and encompasses a greater spectrum of topics, tests the model’s ability to perform well in a larger and more diversified example set. Such a setup of utilization of two different datasets would allow for the overall assessment while pointing out the benefits and possible weaknesses of the proposed model under different circumstances.

Data Preprocessing:

- The text is processed by tokenizing them using the ’basic English’ tokenize of the torch text library.
- All the text/reviews were truncated or padded to the maximum of 256 tokens so that all the CNN model input data had the same size.
- Vocabulary was formed based on the training set where the tokens’ frequency less than 5 were not considered and two additional special tokens unk and pad were applied.
- Reviews were tokenized and the reviews sequences were transformed into sequences of integer indices referring to the vocabulary created during the processing phase.
- Full review on the data collection and processing shown in Figure 1 as architecture of data processing pipeline.

Data Splitting:

- IMDB Dataset: First, to assess the effectiveness of the given proposed CNN model we initially start evaluation on IMBD dataset which is quite popular and often used to evaluate Affect Analysis algorithms. IMDB data set is the corpus of 50000 movie reviews with positive and negative labels. It consists of 25000 Training samples and 25000 Testing samples. To optimize our model we split the training set into training set and validation set in 75-25 split ratio. This led training sample to 18,750 and the validation sample to 6,250.

- Amazon Polarity Dataset: Besides, the used IMDB dataset we also used the Amazon Polarity dataset to evaluate the presented model on a larger and more diverse data set. There are the datasets that include loading the Amazon Polarity which focuses on millions of the Amazon customer reviews categorized as the positive polarity as well as the negative polarity. In our case for data filtering we dropped a large number of data from the Amazon Polarity dataset and used only 400,000 review for experiments. Training samples include 360,000 and test samples include 40,000 samples respectively. As the same with the IMDB dataset, for this dataset, we also used training and validation split of 75-25. This in return provided 270000 samples for training and 90000 samples for undergoing validation.

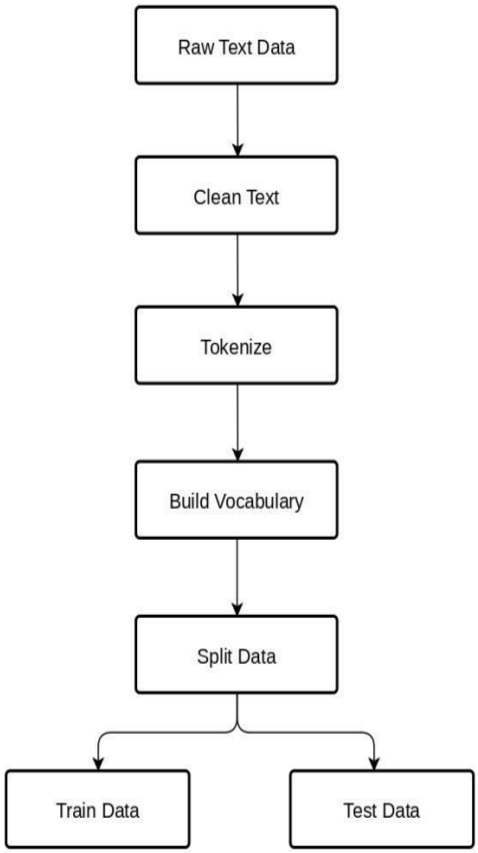

Figure 1: Architecture of Data Processing Pipeline

Model Architecture:

Embedding Layer: An embedding layer of size 300 is used to convert the input tokens in the text into dense vectors. We utilize word embeddings as proposed by Mikolov et al. (4), which have been instrumental in improving the performance of NLP models, and further enhance them using the GloVe model by Pennington et al. (5) .To initiate the embedding layer, trained GloVe word embeddings were used because they improve the model's performance in understanding semantical mean- ings of words.

Convolutional Layers: For the extraction of the fea- tures, three sets of parallel 1D convolution layers with the filter size of 3, 5 and 7 is used. Every convolutional layer applied 100 filters in order to detect various features in the text data.Performing non-linearity, ReLU activation functions were used in the model.

Pooling Layer:

1. To eliminate the spatial dimensions and keep the most relevant features, some max-pooling operations were performed on the convoluted layer outputs. Fully Con-nected Layer.

2. The levels of the pooling layers were joined and fed into a dense layer with a dropout layer (drop out rate of 0.25) to mitigate the over fit. The last layer gave output logits that were associated with the two sentiment classes; positive and negative.

Training Process:

Loss Function and Optimizer:Cross entropy loss was used for training the model as this is appropriate for a multi-class classification problem.The Adam optimizer was used as the optimization procedure following gra- dient followers for efficient calculation.

Training Loop: The training was performed during 10 epochs, adjusting the parameters of the model in every iteration's end. Each epoch, the accuracy of the model on the validation set was tested in order to detect over-fitting and modify the training process. The learning rate was regulated with the help of the ReduceLROnPlateau scheduler, which decreased the learning rate when the validation loss stagnated.

Model Evaluation:Since the best validation accuracy occurred in the model with the least validation loss, that model was employed in evaluating the test accuracies on the test set.

From the performance of the model,the following errors; accuracy, precision, recall, F1- score and the confusion matrix were calculated.

Figure 2 represent the basic-overview kinds of flow related to CNN model. Figure 5 represents the architecture of the training process.

Figure 3 and Figure 4 represents the training and validation plots over number of epochs for IMDB dataset and Amazon dataset. There is a small chance it is a little overfitted but will address that in the future when future work will be done.

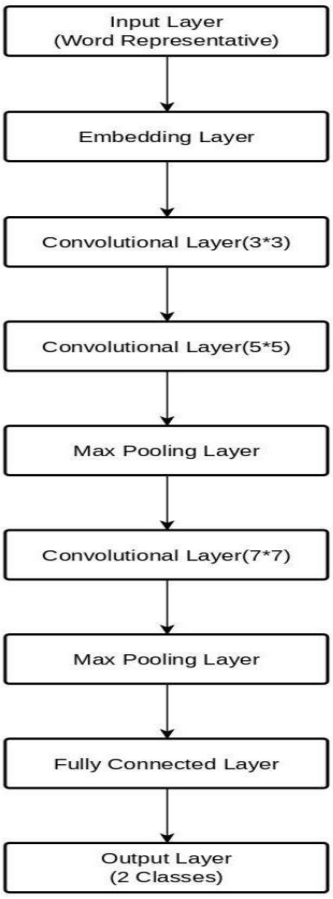

Figure 2: Architecture of CNN Model.

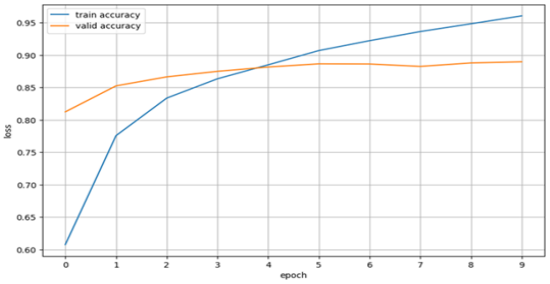

Figure 3: Training and Validation plot for IMDB

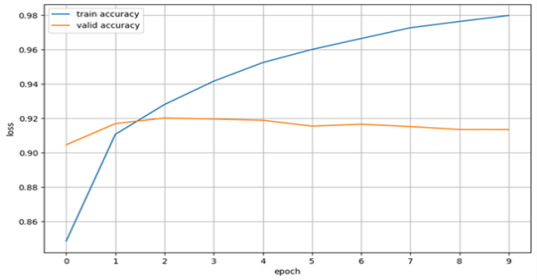

Figure 4: Training and Validation plot for Amazon

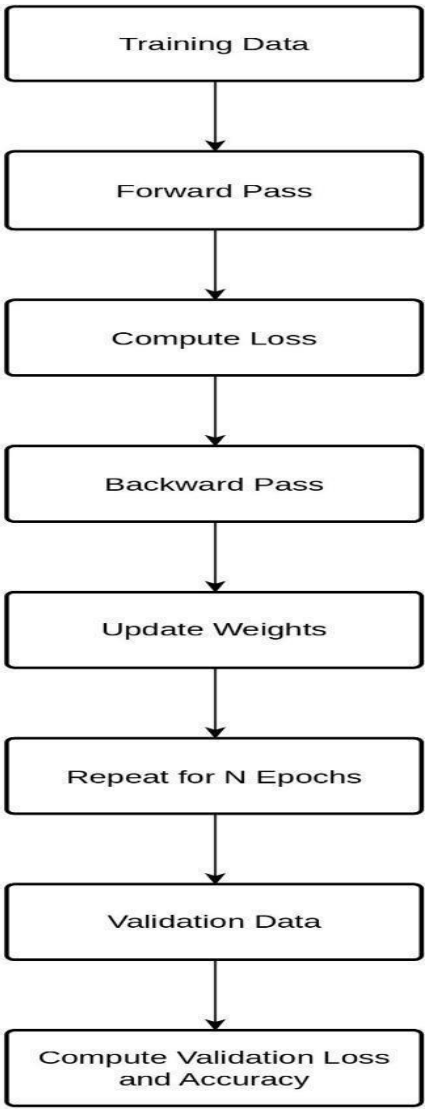

Figure 5: Architecture of training process

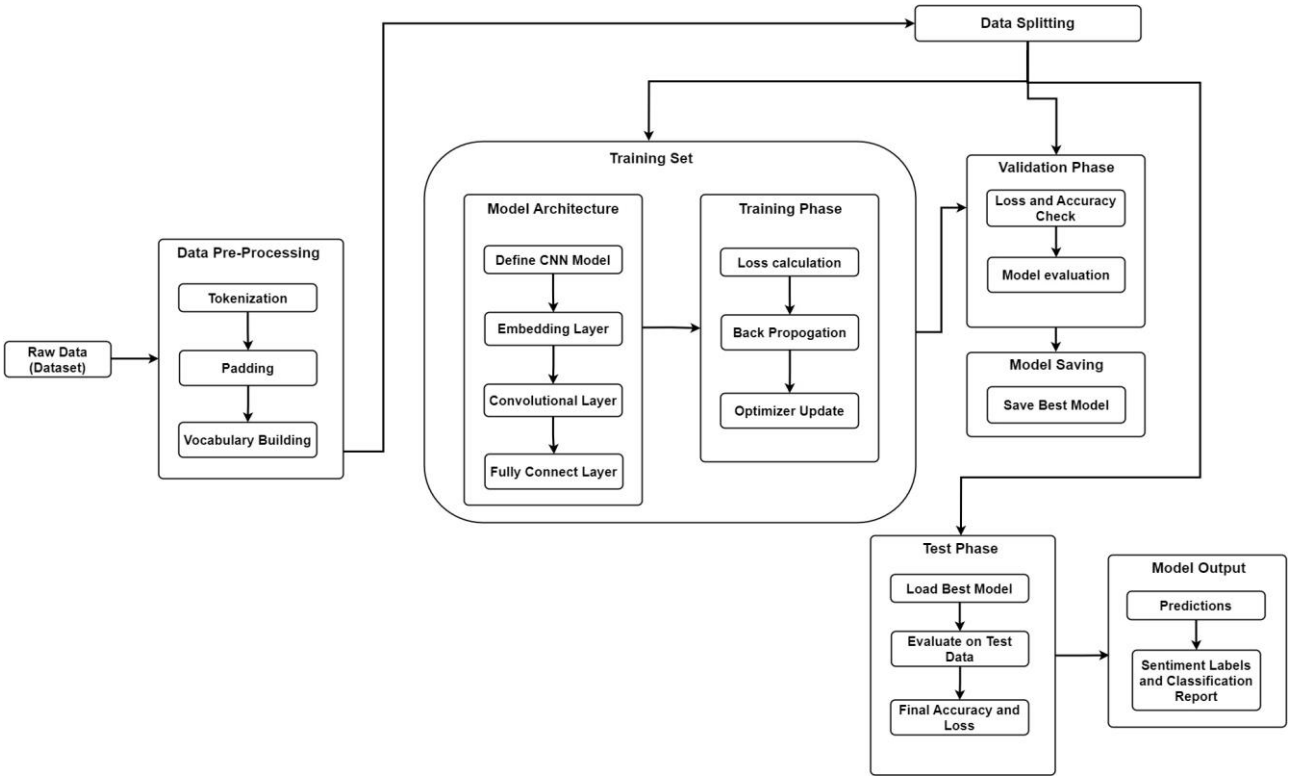

Figure 6: The Detailed Flowchart representing all the processes

RESULTS, COMPARISON AND ANALYSIS FOR

IMDB Dataset

Results:

Logistic Regression Model: The Figure 7 shows the confusion matrix for the Logistic Regression model. The confusion matrix illustrates the number of reviews that have been classified.From Figure 7, we can see that the Logistic Regression model is correctly able to detect the reviews, and the results are as follows:

- True Negative:10424
- True Positive:10814

However, the Logistic Regression model also misclassified reviews, and the results are as follows:

- False Negative:1686
- False Positive:2076

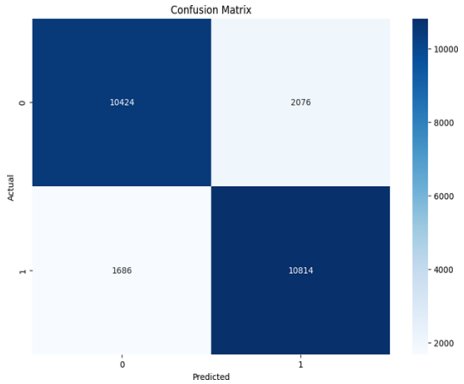

Figure 7: Confusion Matrix for Logistic Regression model.

Naive Bayes Model: The Figure 8 shows the confusion matrix for the Naive Bayes model. The confusion matrix

shows the number of reviews which have been classified. The labels on which the reviews have been classified are Positive and Negative.

From Figure 8, we can see that the Naive Bayes model is correctly able to detect the reviews, and the results are as follows:

- True Negative:10391
- True Positive:10476

Naive Bayes also has misclassified reviews and the results are as follows:

- False Negative:2024
- False Positive:2109

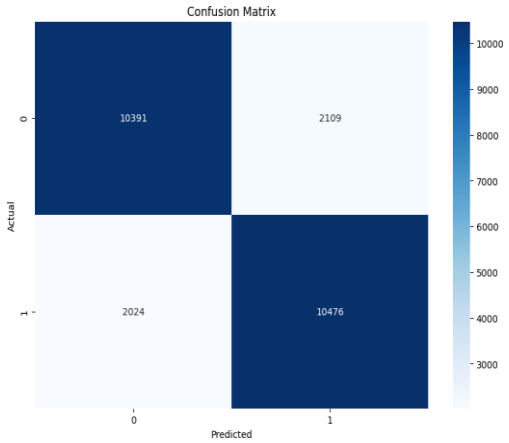

Figure 8: Confusion Matrix for Naive Bayes model.

Decision Tree Model: The Figure 9 shows the confusion matrix for the Decision Tree model. The confusion matrix shows the number of reviews which have been classified. The labels on which the reviews have been classified are the same as described above.

From Figure 9, we can see that the Decision Tree model is correctly able to detect the reviews, and the results are as follows:

- True Negative:8863
- True Positive:8834

The Decision Tree model also has misclassified reviews, and the results are as follows:

- False Negative:3666
- False Positive:3637

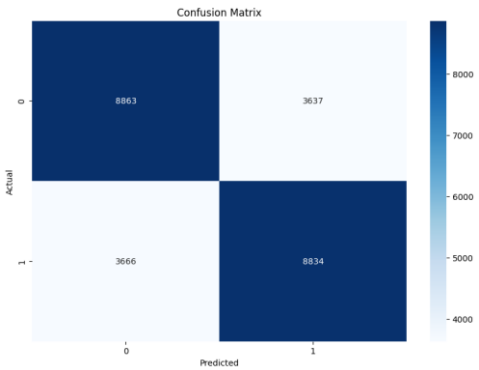

Figure 9: Confusion Matrix for Decision Tree model.

Custom CNN Model: The Figure 10 shows the confusion matrix for our CNN model. From the Figure 10, we can see that our CNN based model is correctly able to detect the reviews, and the results are as follows:

- True Negative:11189
- True Positive:10691

Our CNN model also has misclassified reviews, and the results are as follows:

- False Negative:1809
- False Positive:1311

Thus, the confusion matrix of CNN based model as given below:

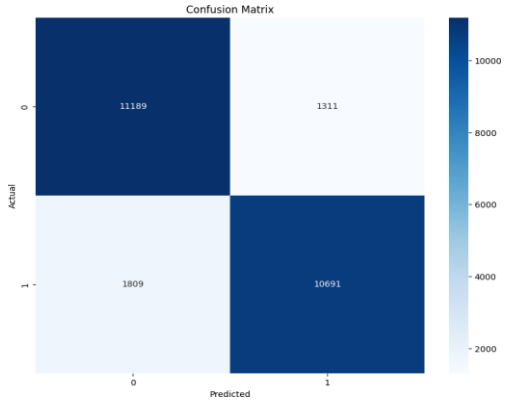

Figure 10: Confusion Matrix for Custom CNN model.

Results Comparison:

The chart for showing the f1-score comparison of our CNN model with base reference models like Logistic Regression, Naive Bayes, and Decision Tree is presented in Figure 11 for the IMDB Dataset

By analyzing Figure 11, it can be observed that the f1-score of the Logistic Regression model is about 0.85, the Naive Bayes model has an f1-score of about 0.83, an f1-score of about 0.70 for the Decision Tree model and an f1-score of 0.87 for our proposed CNN model which is even higher than the three aforementioned methods.

The evaluation of the models as to the accuracy, f1-score, Precision and Recall is presented in the Table I. From this table, it was observed that our proposed CNN model has higher accuracy and higher f1-score than Logistic Regression, Naive Bayes, and Decision Tree models.

The bar chart compares the F1-scores of four different models: Logistic Regression, Naive Bayes, Decision Tree, and a Custom CNN. The Custom CNN achieves the highest F1-score at 0.8727, followed by Logistic Regression with 0.8518, and Naive Bayes with 0.8352. The Decision Tree model has the lowest F1-score at 0.7075. This indicates that the Custom CNN model performs the best among the evaluated models in terms of F1-score.

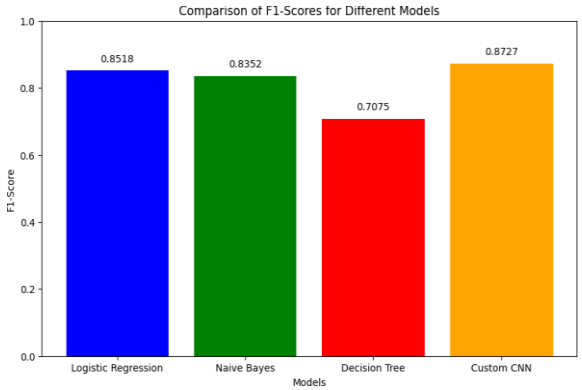

Figure 11: f1-score comparison with reference models.

Table 1: Detailed Comparison of Models

	Logistic Regression	Naive Bayes	Decision Tree	CNN Model
Accuracy	0.84952	0.83468	0.70788	Train: 0.961 Valid: 0.890
F1-score	0 - 0.847135 1 - 0.851831	0 - 0.834116 1 - 0.835240	0 - 0.708218 1 - 0.707541	0 - 0.87763746 1 - 0.87266346
Precision	0 - 0.860776 1 - 0.838945	0 - 0.836971 1 - 0.832420	0 - 0.707399 1 - 0.708363	0 - 0.86082474 1 - 0.89076821
Recall	0 - 0.833920 1 - 0.865120	0 - 0.831280 1 - 0.838080	0 - 0.709040 1 - 0.706720	0 - 0.89512 1 - 0.85528

Table 1, Shows the comparison of metrics like Precision, Recall, F1-Score, and Accuracy for all four models for the IMDB Dataset

Analysis:

The four proposed methods of evaluation show that our CNN model is better than Logistic Regression, Naive Bayes, and Decision Tree method because the feature extraction techniques used in the CNN model are advanced and the CNN architecture is deeper and thus has a better generalization performance. With the help of convolution possible in the architecture of CNN, the model is capable of extracting even higher-order patterns and relationships from the data, thus achieving the better accuracy, precision, recall, and F1 score. Also, what we think is that bigger the dataset better will be our results.

Hence, it is observed from Table-1, Figure 11: our proposed model not only gains higher f1-score for the identification of labels but also having less misclassification for recognizing the labels as compared to the other methods.

RESULTS, COMPARISON AND ANALYSIS FOR AMAZON DATASET

Results:

Logistic Regression Model: The Figure 12 shows the confusion matrix for the Logistic Regression model. The confusion matrix illustrates the number of reviews that have been classified.

From Figure 12, we can see that the Logistic Regression model is correctly able to detect the reviews, and the results are as follows:

- True Negative:16894
- True Positive:17296

However, the Logistic Regression model also misclassified reviews, and the results are as follows:

- False Negative:2863
- False Positive:2947

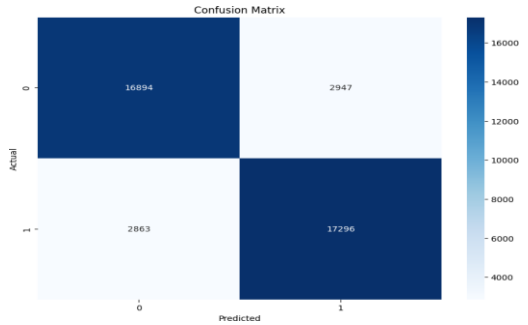

Figure 12: Confusion Matrix for Logistic Regression model.

Naive Bayes Model: Precisely, there of Figure 13provides the confusion matrix of the Naive Bayes model. The confusion matrix indicates the overall reviews classified in terms of their corresponding category.

From Figure 13, we can see that the Naive Bayes model is correctly able to detect the reviews, and the results are as follows:

- True Negative:15774
- True Positive:16296

Naive Bayes also has misclassified reviews and the results are as follows:

- False Negative:3863
- False Positive:4067

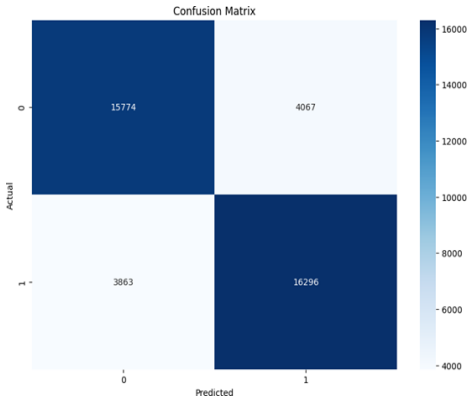

Figure 13: Confusion Matrix for Naive Bayes model.

Decision Tree Model: The Figure 14 shows the confusion matrix for the Decision Tree model. The confusion matrix shows the number of reviews which have been classified. The labels on which the reviews have been classified are the same as described above.

From Figure 14, we can see that the Decision Tree model is correctly able to detect the reviews, and the results are as follows:

- True Negative:14540
- True Positive:14745

The Decision Tree model also has misclassified reviews, and the results are as follows:

- False Negative:5414
- False Positive:5301

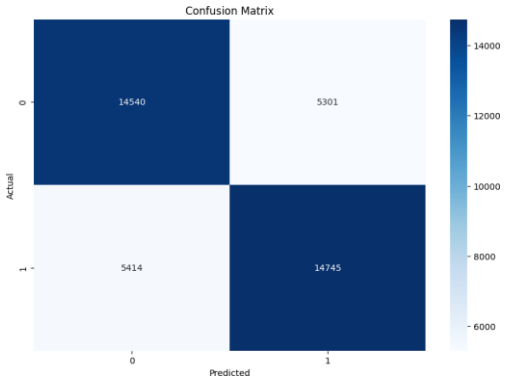

Figure 14: Confusion Matrix for Decision Tree model.

Custom CNN Model: The Figure 15, shows the confusion matrix for our CNN model.

From Figure 15, we can see that our CNN model is correctly able to detect the reviews, and the results are as follows:

- True Negative:18241
- True Positive:18467

Our CNN model also has misclassified reviews, and the results are as follows:

- False Negative:1692
- False Positive:1600

The confusion matrix of our model is given below:

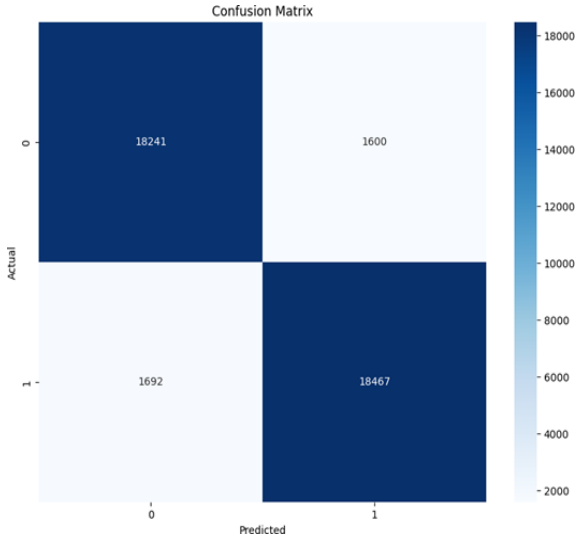

Figure 15: Confusion Matrix for CNN model.

Results Comparison:

Here, for the amazon dataset we obtained significantly higher results for our proposed model as model was able to train better because Amazon dataset is a large dataset.

The chart for showing the f1-score comparison of our CNN model with base reference models like Logistic Regression, Naive Bayes, and Decision Tree is presented in Figure 16.

By analyzing bar chart of figure 16, it can be observed that the f1-score of the Logistic Regression model is about 0.85, the Naive Bayes model has an f1-score of about 0.80, an f1-score of about 0.73 for the Decision Tree model and an f1-score of 0.92 for our proposed CNN model which is significantly higher than the other three mentioned methods.

The evaluation of the models as to the accuracy, f1-score, Precision and Recall is presented in the Table II. From this table, it was observed that our proposed CNN model has immensely high accuracy and high f1-score than Logistic Regression, Naive Bayes, and Decision Tree models.

The bar chart shows the F1-scores of four different model like: Logistic Regression, Naive Bayes, Decision Tree, and Custom CNN. The Custom CNN model achieves the highest F1-score at 0.9182, followed by Logistic Regression with 0.8562, and Naive Bayes with 0.8043. The Decision Tree model has the lowest F1-score at 0.7352. This indicates that the Custom CNN model outperforms the other models in terms of F1-score, suggesting it is the most effective among the evaluated models in Figure 16.

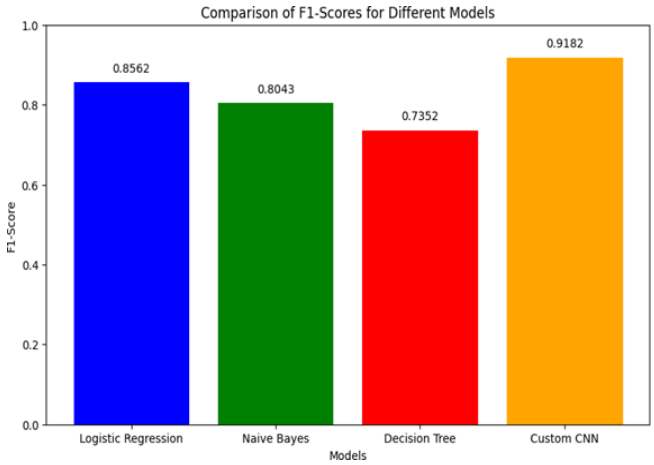

Figure 16: f1-score comparison with reference models.

Table 2, Shows the comparison of metrics like Precision, Recall, F1-Score, and Accuracy for all four models for the Amazon dataset

Table 2: Detailed Comparison of metrics Models

	Logistic Regression	Naive Bayes	Decision Tree	CNN Model
Accuracy	0.85475	0.80175	0.734311	train: 0.980 valid: 0.913
F1-score	0 - 0.853275 1 - 0.856195	0 - 0.799129 1 - 0.804304	0 - 0.733452 1 - 0.735164	0 - 0.91732326 1 - 0.91816238
Precision	0 - 0.855089 1 - 0.854419	0 - 0.803280 1 - 0.800275	0 - 0.730164 1 - 0.738469	0 - 0.91511564 1 - 0.92026711
Recall	0 - 0.851469 1 - 0.857979	0 - 0.795020 1 - 0.808373	0 - 0.736770 1 - 0.731890	0 - 0.91935890 1 - 0.91606727

Analysis:

The four proposed methods of evaluation show that our CNN model is immensely better than Logistic Regression, Naive Bayes, and Decision Tree method because the feature extraction techniques used in the CNN model are advanced and the CNN architecture is deeper and thus has a better generalization performance. With the help of convolution possible in the architecture of CNN, the model is capable of extracting even higher-order patterns and relationships from the data, thus achieving the better accuracy, precision, recall, and F1 score.

As it is observed from Table II, Figure 16 our proposed model not only gains significantly higher f1- score for the identification of labels but also having very less misclassification for recognizing the labels as compared to the other methods.

CONCLUSION

Here, for the amazon dataset we obtained significantly higher results for our proposed model as the model was able to train better because the Amazon dataset is a large dataset.

The chart for showing the f1-score comparison of our CNN model with base reference models like Logistic Regression, Naive Bayes, and Decision Tree is presented in Figure 16

Based on the analysis of different types of models used for sentiment analysis assessment, one can conclude that, in compare with the standard machine-learning algorithms, such as Logistic Regression, Naive Bayes, Decision

Trees, the most appropriate is using the convolutional neural network (CNN). Remember that CNNs have the capacity of decomposing the local relations and hierarchies with regard to the text data and they can therefore be viewed as valuable tools when it comes to detection of sentiment within the textual content of social media posts and reviews. Observing the IMDB and Amazon Polarity datasets clearly indicate that the proposed CNN model is helpful to get rid of the problem of handling all the data and the model is quite responsive when it comes to handling large datasets. Our approach, leveraging CNNs for sentiment analysis, demonstrates the effectiveness of deep learning techniques initially highlighted by Kim (2) and further developed by Severyn and Moschitti (3). The incorporation of word embeddings (4, 5), robust pre-training methods (6, 10), and advanced language models (7, 8, 9), underline the robustness and accuracy of our proposed model. The combining of advanced word representations and CNN architecture has been notably effective for Sentiment Analysis tasks, thus, we also confirms the approach proposed by Go, Bhayani, and Huang (1), Mikolovet al. (4), and Pennington et al. (5). The kind of work presented above exemplifies how CNNs can be harnessed in the advancement of sentiment analysis tasks that potentially be beneficial in numerous industries and studies. As discussed in future work, other forms of CNN models might be enhanced, and other deep learning structures, such as Recurrent Neural Networks or Transformers, might be integrated.

Acknowledgement
we would like to thank C U Shah University, Monark University, and LDRP - Institute of Technology and Research for their support in this research.

Funding
this research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors. (Not applicable)

Conflict of Interest
the authors declare no conflict of interest.

Author Contributions

Hiteshkumar Barot conceptualized the study, with data preprocessing and experimental validation, also apply methodology and doing data analysis, and wrote the manuscript. Viral Parekh and Mehul Barot provided technical guidance and supervised the research, Yaashu Dave helped with data preprocessing and experimental validation. As we did this work as a team.

Ethics Approval
Not applicable for this study as no human or animal subjects were involved.

Data Availability
The datasets used in this study are publicly available: IMDB dataset and Amazon Polarity dataset.

Abbreviations
CNN - Convolutional Neural Network
SA - Sentiment Analysis
SVM - Support Vector Machine
TF-IDF - Term Frequency-Inverse Document Frequency

NLP- Natural Language Processing

Figures and Tables

All figures and tables are created by the authors based on achieved results.Figures and tables are appropriately mentioned in the manuscript and include proper legends.

REFERENCES

[1] Go A, Bhayani R, Huang L. Twitter sentiment classification using distant supervision. Processing. 2009;150(12):1-6.

[2] Kim Y. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882. 2014.

[3] Severyn A, Moschitti A. Twitter sentiment analysis with deep convolutional neural networks. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2015. p. 959-62.

[4] Mikolov T, Chen K, Corrado G, Dean J. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781. 2013.

[5] Pennington J, Socher R, Manning CD. GloVe: Global vectors for word representation. In: Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. 2014. p. 1532-43.

[6] Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, et al. RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692. 2019.

[7] Brown T, Mann B, Ryder N, Subbiah M, Kaplan J, Dhariwal P, et al. Language models are few-shot learners. In: Advances in Neural Information Processing Systems. 2020.

[8] Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov R, Le QV. XLNet: Generalized autoregressive pretraining for language understanding. In: Advances in Neural Information Processing Systems. 2019. p. 5753-63.

[9] Peters M, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, et al. Deep contextualized word representations. arXiv preprint arXiv:1802.05365. 2018.

[10] Devlin J, Chang M, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2019;1:4171-86.