Text Classification of Tweets using Convolutional Neural Networks and Long-Short-Term Memory

Spandana Kalakonda
Department of Computer Science
Southern Illinois University
Edwardsville, Illinois
skalako@siue.edu

Dr. Mark Mckenney
Department of Computer Science
Southern Illinois University
Edwardsville, Illinois
marmcke@siue.edu

Dr. Eren Gultepe
Department of Computer Science
Southern Illinois University
Edwardsville, Illinois
egultep@siue.edu

ABSTRACT

Social media platforms such as Twitter have become an essential source of user-generated content for researchers, offering valuable insights into user behavior and preferences. One aspect of this is categorizing posts based on location, which can inform us about the origin of a tweet. The ability to accurately classify tweets based on public place has numerous potential applications, such as identifying popular venues or events, tracking the spread of diseases or viral content, and analyzing sentiment and opinions of users in specific locations. In this study, we try to classify the tweets based on textual content of tweet to a public place such as a hospital, restaurant, airport, church, or zoo using machine learning models - Logistic Regression (LR), Long-Short-Term Memory (LSTM), and Convolutional Neural Networks (CNN) and a Parallel CNN – and compare their performance. The aim of our study is to investigate the potential of these models for accurately classifying tweets based on their content. Our findings reveal that the LR model with n-grams achieved an accuracy of 82.8%, the CNN and LSTM models with one hidden layer performed similarly to this baseline, achieving accuracies of 83% and 82%, respectively. On the other hand, we observed that the parallel CNN model, with its advanced architecture and multiple convolutional layers, was the most effective in classifying tweets accurately, achieving an impressive accuracy of 93%. This indicates that the parallel CNN model is better suited for handling the complexities of social media text data and can provide more accurate location-based analysis of tweets. The potential areas for future research can be identifying the crowd of people who are tweeting from a particular public place, which can help in analyzing how public places become popular over time, which can be used to enhance marketing strategies and improve customer experiences in those areas.

1 Introduction

In recent years, there has been a growing interest in the use of machine learning models for text classification in various applications, such as sentiment analysis, spam detection, and content categorization. With the increasing volume of user-generated content on social media platforms, text classification has become a critical task for extracting meaningful insights from social media data. Social media platforms such as Twitter and Facebook have billions of active users generating massive amounts of text data every day, making text classification a challenging task.

One important application of text classification is location-based analysis of social media content. By categorizing social media posts based on their public place of origin, researchers can gain insights into the popularity of different venues, the spread of diseases or viral content, and the sentiment and opinions of users in specific locations. In this context, machine learning models have emerged as powerful tools for accurately classifying social media content based on location.

In this paper, we present an approach to location-based text classification of tweets. Our approach focuses on classifying each tweet that includes a text description containing user posts, mentions, hashtags, and URLs of a physical place as a hospital, restaurant, airport, church, or zoo. For example, a tweet that reads "Spent eight hours at the airport before my flight canceled. We really are back to normal. 😌" would be classified as an airport, while "The lion seems to be hungry today" would be classified as zoo.

This paper is organized into seven main sections. Section 1 introduces the topic and the motivation behind the research. Section 2 discusses the research aim, section 3 and literature related to the topic. Section 4 presents the parallel methods and techniques used in this research. This includes a detailed explanation of the data collection, preprocessing, and machine learning algorithms used for text classification. Section 5 covers the experimental results and analysis, including the evaluation of the model's performance on the validation set and comparison with baseline models. Finally, Section 6 concludes the paper with a summary of the work and its contributions, as well as potential directions for future research.

2 Research Aim

The research involves several key steps as shown in figure 1.

1. Data Collection: Data is collected by downloading tweets related to different accounts such as airports, hospitals, restaurants, churches, and zoos.

2. Data Preprocessing: Once the data is collected, it undergoes through a preprocessing stage where noise and irrelevant information are removed, and the text is cleaned, tokenized, and subjected to stop word removal, lemmatizing and stemming.
3. Feature Extraction: The preprocessed tweets are transformed into numerical representations using techniques such as Term Frequency Inverse Document Frequency (TF-IDF) with N-grams and Word2Vec models to generate embedding matrices.
4. Training, Testing and Model Refinement: The TF-IDF and embedding matrices are used to train, test, and refine machine learning models Logistic Regression (LR), convolutional neural networks (CNN), and long short-term memory (LSTM) for text classification.
5. Model Evaluation: The model's performance is evaluated using various metrics such as accuracy, precision, recall, and F1-score to assess the effectiveness of the trained models.

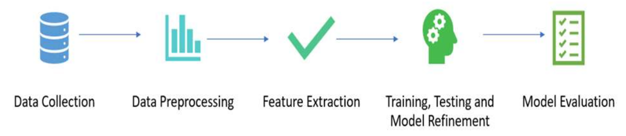


Figure 1: Pipeline of the study

3 Related Work

Research on text classification has been an active area of study in the field of natural language processing (NLP) and machine learning for many years. There have been numerous studies and research papers exploring different approaches and techniques for text classification, as well as their applications in various domains.

One major area of research in text classification has been the development of algorithms and models that can effectively handle large-scale text datasets. This has led to the development of several machine learning algorithms, such as logistic regression, support vector machines (SVM), Naive Bayes, decision trees, and deep learning models like convolutional neural networks (CNNs) and recurrent neural networks (RNNs). Another area of research in text classification has been the exploration of different feature engineering techniques and text representation methods, such as TF-IDF, word embeddings, bag-of-words, and n-grams. These techniques aim to transform raw text data into a numerical representation that can be used as input to a machine-learning model.

In recent years, there has been a growing interest in the use of machine learning models for text classification in various applications, such as sentiment analysis, spam detection, and content categorization. With the increasing volume of user-generated content on social media platforms, text classification has become a critical task for extracting meaningful insights from social media data. Social media platforms such as Twitter and Facebook have billions of active users generating massive amounts of text data every day, making text classification a challenging task.

3.1 Machine Learning Models:

Much research has been done on machine learning algorithms to know the importance of each model on different datasets. Some of the studies are presented in this section. G. Gautam and D. Yadav [1] aim to classify sentence and product reviews based on Twitter data and shows an improved accuracy by incorporating WordNet semantic analysis to 89.9% from 88.2% of naive Bayes compared to maximum entropy's 83.8% and SVM's 85.5% on a unigram model.

M. Wongkar and A. Angdresey [2] aimed to analyze public sentiment towards the Republic of Indonesia's presidential candidates for the 2019-2024 period using tweet data obtained from Twitter. Where Naive Bayes was found to have the highest accuracy of 80.90%, while KNN had an accuracy rate of 75.58%, and SVM had an accuracy rate of 63.99%.

A. Anindditya, M. A. Hasibuan, and E. Sutoyo [3] focus on classifying questions based on the level of Cognitive Level of Bloom's Taxonomy using a dataset of exam questions from the Department of Information System, Telkom University from 2012/2013 to 2018/2019 achieves a high accuracy precision of 85% using Naive Bayes with the TF-IDF N-Gram Level feature.

Pranckevičius, T, and Marcinkevičius, V[4] present a comparison of the Logistic Regression (LR) classification method with and without POS for product-review data. The experimental results show that the LR classification method using POS features has, on average, 3.31% higher classification accuracy compared to the LR classification method without POS. The accuracy of both methods decreases with bigram and trigram models (without POS: min 23.70%, max 47.27%; with POS: min 42.06%, max 56.15%), but increases when a combination of uni/bi/tri-gram models (without POS: min 40.25%, max 78.6%; with POS: min 50.42%, max 78.29%), are used. Overall, the results suggest that the combination of the uni/bi/tri-gram model increases classification accuracy, but the accuracy decreases with larger datasets.

The performance of Naïve Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression for multi-class text classification is compared in [5]. The results show that Logistic Regression achieves the highest classification accuracy (ranging from 32.43% to 58.50%) compared to the other methods. The decision Tree has the lowest average accuracy values (ranging from 24.10% to 34.58%). Naïve Bayes performs slightly better than Random Forest and Support Vector Machines, but the difference is not statistically significant. The use of uni/bi/tri-gram models generally improve classification accuracy.

3.2 Deep Learning Models

Recently neural networks have become a popular way to model text and have led to the development of distributed representations of words. These distributed representations enable more effective text categorization, making neural networks a popular choice for this task. Word2vec [6], GloVe [7], and fastText [8] are popular methods for creating low-dimensional representations of words in neural network models for text categorization. These methods allow the neural network to learn distributed representations of words directly from raw text without the need for manual feature engineering or filtering. This can help to improve the accuracy and generalization of text classification models.

Dharma, E. M., Gaol, F. L., Warnars[9] tested three-word embedding techniques - FastText, GloVe, and Word2Vec on a dataset of 20 newsgroups for text classification using the CNN model. FastText outperformed the other two with an accuracy of 97.2%, while GloVe had 95.8%, and Word2Vec had 92.5%. FastText was also found to be able to handle out-of-vocabulary words, which the other two techniques were not able to do. Although FastText had the best performance, the difference in accuracy between the three techniques was not statistically significant, suggesting they all have competitive performance.

E. Gultepe, M. Kamkarhaghighi, and M. Makrehchi, [10] suggest that LSA (Latent Semantic Analysis) word vectors, when used with CNNs (Convolutional Neural Networks) with small window sizes, can serve as a baseline classifier for document classification. One possible reason for this is that LSA word vectors may be more specific to the domain being analyzed compared to pre-trained w2v (Word2Vec) word vectors. Additionally, using small window sizes can help to leverage the embedding of LSA vectors.

Chen, Y[11] proposed two deep learning methods, Parallel CNN and Deep CNN, for predicting multiple relation candidates of a question. Both models use convolutional layers to capture local semantic features and a max-over-time pooling layer to select global semantic features, followed by a fully connected layer with dropout to summarize the features. Experiments show that both models outperform the traditional Support Vector Classification (SVC) based method by a large margin, with Deep CNN outperforming Parallel CNN. This suggests that the deep structure of Deep CNN provides a stronger semantic learning capacity than the wider but shallower Parallel CNN.

While Convolutional Neural Networks (CNN) can significantly improve the accuracy of short text classification, they may not be effective in processing long sentences or texts. To overcome this limitation, Recurrent Neural Network (RNN) model based on Long Short-Term Memory (LSTM) and Gated Recurrent Unit GRU) are used. This model can memorize long-distance dependencies and retain the main semantic information of the text, allowing for improved processing of longer texts. J. Zhang, Y. Li, J. Tian, and T. Li [12] propose a hybrid LSTM-CNN model for text classification. The model leverages LSTM to preserve historical and context information in long text and overcome the vanishing gradient problem. Additionally, it uses CNN to extract local features of the text. The proposed model combines the strengths of LSTM and CNN and overcomes the shortcomings of CNN. Experimental results show that the proposed model effectively improves the accuracy of text classification by 7% more than basic CNN and 3.5 % more than the basic LSTM model.

In this paper, as the baseline model, we implement the Multinomial Logistic Regression Model [5] with uni, bi, and trigrams and TF-IDF transformation. But n-grams may struggle to capture the meaning of idiomatic expressions, phrases with multiple meanings, or words with complex relationships. The convolutional Neural Networks Model as discussed in [10] with small window sizes has been implemented in this paper to capture more complex relationships and compared with the Long Short-Term Memory model [12] which preserves historical and context information in long text.

4 Material and Methods

In this section, we discuss the research pipeline methods and implementation of our baseline logistic regression, CNN and LSTM models.

4.1 Data Collection

The models are evaluated using an unbalanced class distribution dataset that was collected using a social network scrapping package (snscrape) from python. The dataset consists of a total of 131625 tweets with 129896 unique tweets belonging to 5 different classes. The class distribution is shown in figure 2. The dataset consists of 5 attributes as shown in figure 3, where Time: the post time of the tweet with data, Username: The username of the user who posted the tweet, Tweet: The text of the tweet with a maximum length of 276, Label: the label representing the place category of the tweet, Public_Place: the name or category of a tweet which it is associated with.

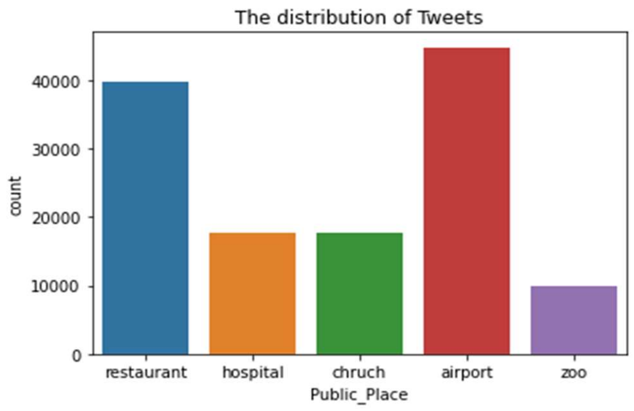
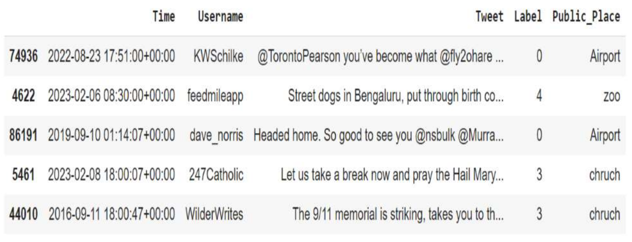


Figure 2: The class distribution of tweets.



	Time	Username	Tweet	Label	Public_Place
74936	2022-08-23 17:51:00+00:00	KWSchilke	@TorontoPearson you've become what @fly2ohare ...	0	Airport
4622	2023-02-06 08:30:00+00:00	feedmileapp	Street dogs in Bengaluru, put through birth co...	4	zoo
86191	2019-09-10 01:14:07+00:00	dave_norris	Headed home. So good to see you @nsbulk @Murra...	0	Airport
5461	2023-02-08 18:00:07+00:00	247Catholic	Let us take a break now and pray the Hail Mary...	3	chruch
44010	2016-09-11 18:00:47+00:00	WilderWrites	The 9/11 memorial is striking, takes you to th...	3	chruch

Figure 3: Sample Data

4.2 Data Preprocessing

The tweets in the dataset were preprocessed before being used for classification. The preprocessing steps involved removing URLs, user mentions, and special characters like punctuations, hashtags, and emoticons from the tweet text. All the letters in the tweets were converted to lowercase, and stop words were removed.

For example, this is a tweet before preprocessing "Spent eight hours at the airport before my flight canceled. We really are back to normal. 😌"

After Preprocessing "spent eight hours airport flight canceled really back normal"

The tweets were then tokenized, Lemmatized, and stemmed. These preprocessing steps were necessary to standardize the format of the tweets and to remove any irrelevant information that could potentially interfere with the classification task.

After Lemmatization and Stemming the tweet is:

"spent eight hour airport flight cancel realli back normal"

4.3 Feature Selection

4.3.1 Bag of Words. Bag of words is a language model technique used in natural language processing that represents text as a collection of individual words. It involves converting a piece of text into a numerical vector that represents the frequency of each word in the text.

4.3.2 Term Frequency Inverse Document Frequency (TF-IDF). The TF-IDF representation of a document which is based on the frequency of words in the document, as well as their frequency in the entire corpus of documents. Specifically, the TF-IDF score of a word in a document is the product of its term frequency (i.e., the number of times the word appears in the document) and its inverse document frequency (i.e., a measure of how much information the word provides across all documents in the corpus). The inverse document frequency is defined as the logarithm of the total number of documents in the corpus divided by the number of documents containing the word.

4.3.3 Word2Vec. Word2vec is a popular natural language processing technique that represents words as vectors in a high-dimensional space. It is based on a neural network algorithm that learns to predict the context of a given word by examining surrounding words. The distance between vectors corresponds to the semantic similarity between words.

4.4 Machine Learning Models

4.4.1 Logistic Regression (LR). Multinomial logistic regression is a variant of logistic regression where the dependent variable can have more than two categories, and the independent variables. The model estimates the probability of each category (1) of the dependent variable based on the values of the independent variables.

$$P(Y = y|X = x) = \frac{e^{\beta 0+\beta 1x1+\beta 2x2+\cdots+\beta pxp}}{1+e^{\beta 0+\beta 1x1+\beta 2x2+\cdots+\beta pxp}} \quad (1)$$

The output of the model is a set of coefficients that indicate the relative influence of each independent variable on each category of the dependent variable. These coefficients are usually represented as a set of conditional logit models by a soft max function, with each model comparing one category of the dependent variable to a reference category. Then classifier uses an argmax function to find the appropriate class based on the coefficients.

4.4.2 Convolutional Neural Networks (CNN). CNNs can also be used for natural languages processing tasks such as text classification and sentiment analysis. In CNN for text classification, the input to the network is a sequence of words, represented as vectors in a high-dimensional space. These word vectors can be generated using techniques such as one-hot encoding or word embeddings like Word2Vec or GloVe. The architecture of a CNN shown in Figure 4 typically consists of multiple layers, such as convolutional, pooling, and fully connected layers. The convolutional layers employ filters that can be adjusted during training to perform convolution operations on the input data, allowing them to detect small patterns and features. The pooling layers then reduce the size of the feature maps created by the convolutional layers, making the data easier to process. Lastly, the fully connected layers take the processed data and perform classification.

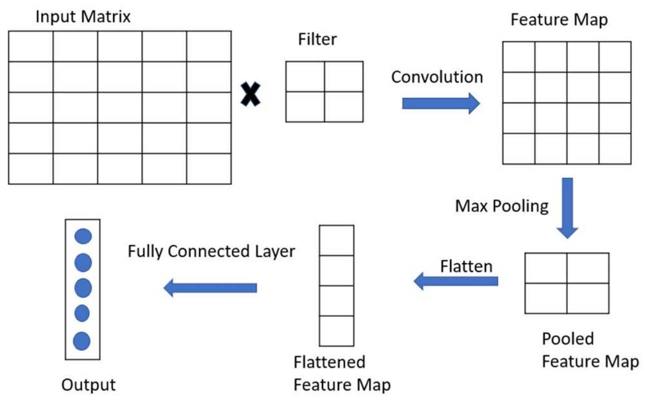


Figure 4: Overview of the CNN Layers

4.4.4 Long Short-Term Memory (LSTM). An LSTM network is composed of memory cells that can store information over long periods of time, input gates that control how much new information is added to the memory cell, forget gates that control how much old information is removed from the memory cell, and output gates that control how much of the memory cell state is used to generate the output. These components make it possible for the LSTM to selectively retain or discard information over time, allowing it to better handle long-term dependencies in sequential data. The LSTM network is typically stacked with multiple LSTM layers, with each layer receiving the output of the previous layer as input. The last LSTM layer may be followed by one or more fully connected layers that perform the final prediction task.

5 Experiments and Results

5.1 Experimental Setup

For the dataset, the vocabulary size is 37620 for all the models. To test the effectiveness of the linear classifier data is split into 80/20 split. We input Word Vector used as an input with TF-IDF transformation and for deep learning models Word2Vec vector is used with word embedding as input. The Adam optimizer is used at a learning rate of 0.0001 for 30 epochs by monitoring the validation loss using Categorical Cross entropy to train the deep-learning models. The macro-averaged F1 score, and accuracy are used to analyze the performance of all the classifiers and the confusion matrix is used to identify the true positives of each class.

5.2 Baseline Logistic Regression

The baseline method involves using the TF-IDF vectorizer to transform the text data into a matrix of features, where each feature corresponds to an n-gram (a sequence of one or more words). The n-grams are specified to be unigrams, bigrams, and trigrams, which means that the features can capture the relationships between adjacent words as well as larger chunks of text. Once the TF-IDF matrix is created, a Multinomial logistic regression model with L2 regularization and the SAGA solver is trained on the data. This model can then be used to predict the labels of new text data based on its n-gram features and corresponding TF-IDF weights. The confusion matrix of this model is shown in figure 5.

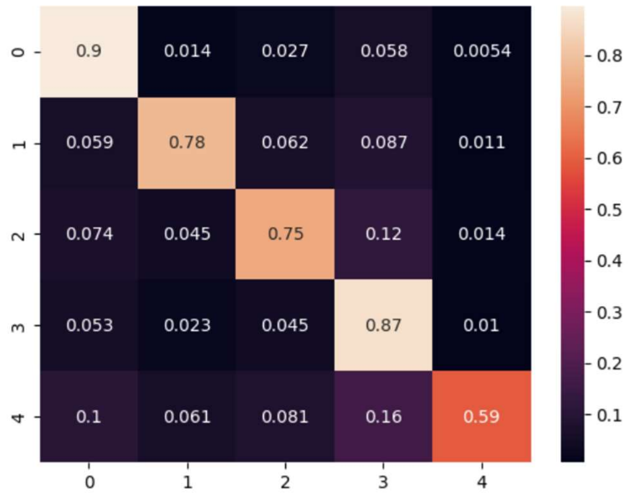


Figure 5: Confusion matrix of LR where the labels 0, 1, 2, 3, and 4 correspond to classes airport, restaurant, hospital, church, and zoo

5.3 Convolutional Neural Network Model

The CNN model uses a combination of an Embedding layer, one Conv1D layer, a MaxPooling1D layer, a dropout layer, and a fully connected dense layer with SoftMax activation. The parameters of each layer are the same as the parallel CNN layer, the only difference is that we have a single conv1D layer with filter size 1 followed by a MaxPooling1D layer. The confusion matrix of this model is shown in figure 6.

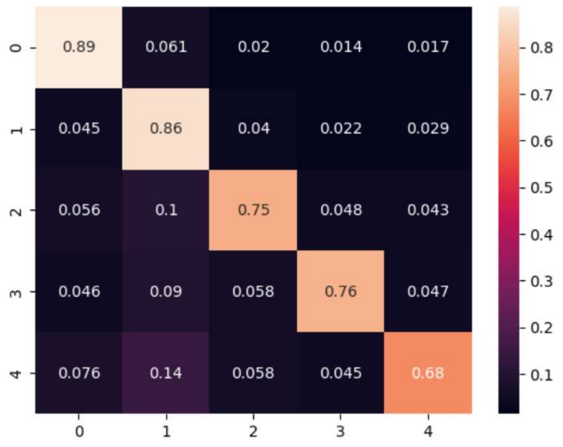


Figure 6. Confusion matrix of CNN with filter size 1 where the labels 0, 1, 2, 3, and 4 correspond to classes airport, restaurant, hospital, church, and zoo

5.4 Long Short-Term Memory Model

The LSTM model uses a combination of an Embedding layer, an LSTM layer, and a fully connected dense layer with SoftMax activation. The first embedding layer is implemented similarly as mentioned in the parallel CNN model. Which is followed by an LSTM layer with a Filter size of 128. Lastly, the output of the LSTM layer is passed through a fully connected dense layer with SoftMax activation to produce the final output. The confusion matrix of this model is shown in figure 7.

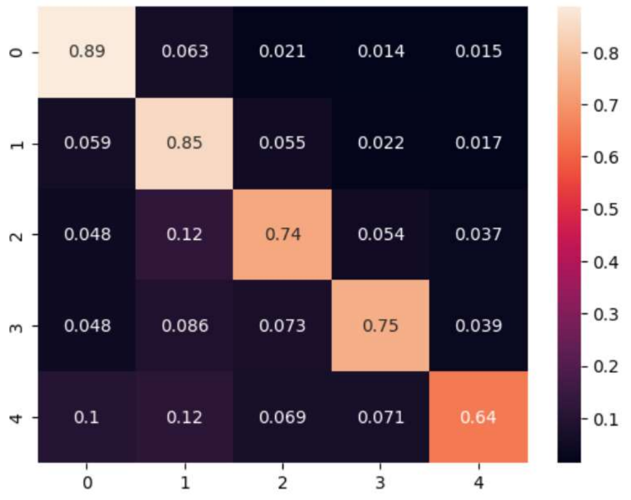


Figure 7: Confusion matrix of LSTM where the labels 0, 1, 2, 3, and 4 correspond to classes airport, restaurant, hospital, church, and zoo

5.5 Parallel CNN Model

The parallel CNN model uses a combination of an Embedding layer, parallel Conv1D layers, MaxPooling1D layers, a dropout layer, and a fully connected dense layer with SoftMax activation. The following are the parameters used for the CNN architecture with a batch size of 128:

Embedding Layer: The embedding matrix is passed with pre-trained word embeddings using word2vec and the embedding dimension is set to 300.

Convolutional Layer: Each conv1D layer has 128 filters with L2 regularization (10e-5), and Rectified Linear (ReLu) activation, and different filter size ranges from 1 to 3, 1 to 4 and 1to 6.

Max Pooling Layer: Each MaxPooling1D layer has a pooling window size of 2 the stride of the pooling operation is 1.

Dropout Layer: dropout of 0.5 is used to regularize the model.

Dense Layer: The final layer with 5 output classes is transformed using a SoftMax function.

The summary of the {1,2,3,4}- Ww2v with CNN model is shown in figure 9 and the confusion matrices of CNN with filter sizes {1,2,3}, {1,2,3,4} and {1,2,3,4,5,6} are shown in figures 8, 10 and 11 respectively.

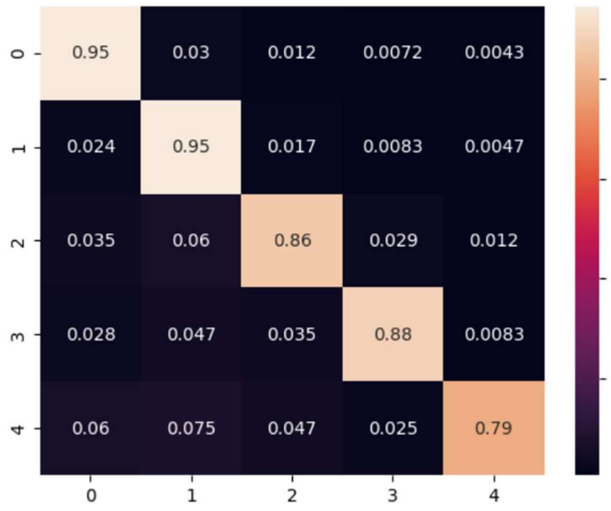
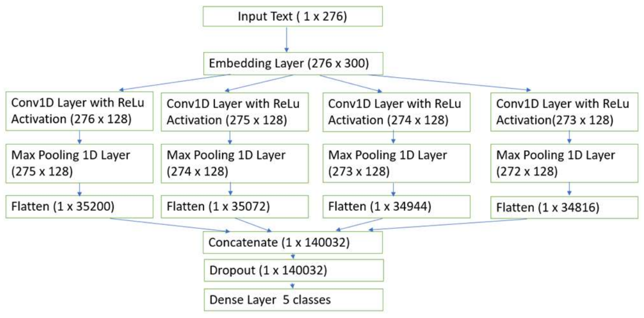


Figure 8: Confusion matrix of Parallel CNN with filter sizes {1,2,3} where the labels 0, 1, 2, 3, and 4 correspond to classes airport, restaurant, hospital, church, and zoo



Input Text (1 x 276)
Embedding Layer (276 x 300)
Conv1D Layer with ReLu Activation (276 x 128)
Conv1D Layer with ReLu Activation (275 x 128)
Conv1D Layer with ReLu Activation (274 x 128)
Conv1D Layer with ReLu Activation(273 x 128)
Max Pooling 1D Layer (275 x 128)
Max Pooling 1D Layer (274 x 128)
Max Pooling 1D Layer (273 x 128)
Max Pooling 1D Layer (272 x 128)
Flatten (1 x 35200)
Flatten (1 x 35072)
Flatten (1 x 34944)
Flatten (1 x 34816)
Concatenate (1 x 140032)
Dropout (1 x 140032)
Dense Layer 5 classes

Figure 8: Summary of Parallel CNN Model with filter sizes {1,2,3,4}

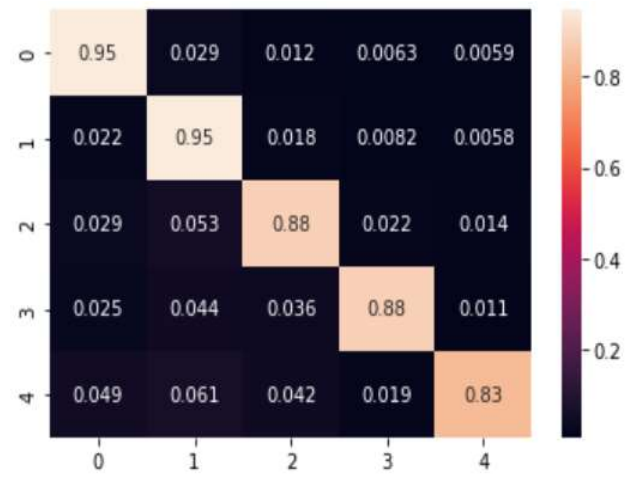


Figure 10: Confusion matrix of Parallel CNN with filter sizes {1,2,3,4} where the labels 0, 1, 2, 3, and 4 correspond to classes airport, restaurant, hospital, church, and zoo

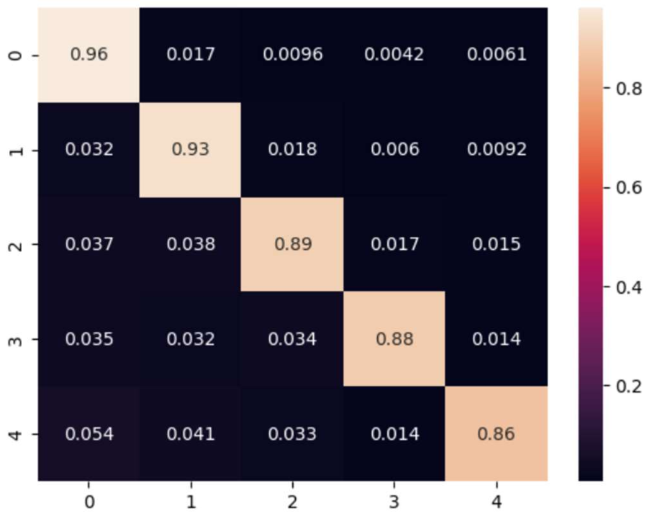


Figure 11: Confusion matrix of Parallel CNN with filter sizes {1,2,3,4,5,6} where the labels 0, 1, 2, 3, and 4 correspond to classes airport, restaurant, hospital, church, and zoo

5.6 Analysis of Results

Table 1 shows the experimental results of the parallel CNN for W_{w2v} models with filter sizes {1,2,3}, {1,2,3,4} and {1,2,3,4,5,6} against the baseline {1,2,3}-$W_{tf\text{-}idf}$ with Logistic Regression, {1}- W_{w2v} CNN and {1}- W_{w2v} LSTM models. The baseline {1,2,3}-$W_{tf\text{-}idf}$ with Logistic Regression, {1}- W_{w2v} CNN and {1}- W_{w2v} LSTM models have almost the same accuracy of 82.81%, 83%, and 82% respectively and the F1-scores are 0.79, 0.79 and 0.78 which are also similar because the logistic regression and Neural network model with one hidden layer almost performed similarly because the logistic regression is a subset of a neural network classifier [13]. The parallel CNN with filter sizes {1,2,3}, {1,2,3,4} and {1,2,3,4,5,6} have an accuracy of 91%, 92% and 93% respectively and F1-Score of 0.90, 0.90 and 0.91 respectively. The best-performing model is {1,2,3,4,5,6}-W_{w2v} with CNN with an accuracy of 93% and F1-Score of 0.91. From this, we can say that the parallel CNN implemented with small filter sizes can perform well compared to all other models as parallel convolutional neural network models can simultaneously extract n -gram patterns with different sizes for exploiting the important features automatically which often enable machines to understand the context of the given text using word embeddings.

Method	Precision	Recall	F1- Score	Accuracy
{1,2,3}-$W_{tf\text{-}idf}$ with LR	0.82	0.78	0.79	82.81%
{1}- W_{w2v} CNN	0.79	0.79	0.79	83 %
{1}- W_{w2v} LSTM	0.79	0.77	0.78	82%
{1,2,3}- W_{w2v} with CNN	0.91	0.89	0.90	91%
{1,2,3,4}- W_{w2v} with CNN	0.91	0.90	0.90	92%
{1,2,3,4,5,6}- W_{w2v} with CNN	0.92	0.91	0.91	93%

Table 1: Table showing the experimental results of all the models

From the confusion matrices shown in figure 5,6,7, 8, 10 and 11 we can say that the parallel CNN models have improved the true positives of all the individual classes. The parallel CNN has shown superiority in performance as compared to models as parallel CNNs can extract more important features by applying filters.

6 Conclusion

This study shows that the parallel CNN model with Word2Vector embedding and small filter sizes can perform better than simple LSTM and Simple CNN Models as the parallel CNNs with different filter sizes can help in extracting the contextual information using n -gram patterns. Future research may focus on identifying the sentiment of population tweeting from a particular public location, which can aid in the analysis of how those locations gain popularity over time and be utilized to better marketing tactics and consumer experiences there.

REFERENCES

[1] G. Gautam and D. Yadav, "Sentiment analysis of twitter data using machine learning approaches and semantic analysis," 2014 Seventh International Conference on Contemporary Computing (IC3), Noida, India, 2014, pp. 437-442, doi: 10.1109/IC3.2014.6897213.

[2] M. Wongkar and A. Angdresey, "Sentiment Analysis Using Naive Bayes Algorithm Of The Data Crawler: Twitter," 2019 Fourth International Conference on Informatics and Computing (ICIC), Semarang, Indonesia, 2019, pp. 1-5, doi: 10.1109/ICIC47613.2019.8985884.

[3] A. Anindityа, M. A. Hasibuan and E. Sutoyo, "Text Mining Approach Using TF-IDF and Naive Bayes for Classification of Exam Questions Based on Cognitive Level of Bloom's Taxonomy," 2019 IEEE International Conference on Internet of Things and Intelligence System (IoTaIS), Bali, Indonesia, 2019, pp. 112-117, doi: 10.1109/IoTaIS47347.2019.8980428.

[4] Pranckevičius, T., & Marcinkevičius, V. (2016, November). Application of logistic regression with part-of-the-speech tagging for multi-class text classification. In 2016 IEEE 4th workshop on advances in information, electronic and electrical engineering (AIEEE) (pp. 1-5). IEEE.

[5] Pranckevičius, T., & Marcinkevičius, V. (2017). Comparison of naive bayes, random forest, decision tree, support vector machines, and logistic regression classifiers for text reviews classification. Baltic Journal of Modern Computing, 5(2), 221.

[6] T. Mikolov, K. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space", Proc. Workshop at ICLR, 2013.

[7] J. Pennington, R. Socher and C. Manning, "Glove: Global vectors for word representation", Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP), pp. 1532-1543, Oct. 2014.

[8] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, "Enriching word vectors with subword information" in arXiv:1607.04606, 2016, [online] Available: http://arxiv.org/abs/1607.04606.

[9] Dharma, E. M., Gaol, F. L., Warnars, H. L. H. S., & Soewito, B. E. N. F. A. N. O. (2022). The accuracy comparison among Word2vec, Glove, and Fasttext towards convolution neural network (CNN) text classification. Journal of Theoretical and Applied Information Technology, 100(2), 31.

[10] E. Gultepe, M. Kamkarhaghighi and M. Makrehchi, "Latent Semantic Analysis Boosted Convolutional Neural Networks for Document Classification," 2018 5th International Conference on Behavioral, Economic, and Socio-Cultural Computing (BESC), Kaohsiung, Taiwan, 2018, pp. 93-98, doi: 10.1109/BESC.2018.8697314.

[11] Chen, Y. (2015). Convolutional neural network for sentence classification (Master's thesis, University of Waterloo).

[12] J. Zhang, Y. Li, J. Tian and T. Li, "LSTM-CNN Hybrid Model for Text Classification," 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Chongqing, China, 2018, pp. 1675-1680, doi: 10.1109/IAEAC.2018.8577620.

[13] https://jamesmccaffrey.wordpress.com/2018/07/07/why-a-neural-network-is-always-better-than-logistic-regression Patricia S. Abril and Robert Plant, 2007.