Doi: https://doi.org/XXXX/XXXX/JFNC202501004

SentCNN - A Lightweight and Efficient Deep Learning Model for Accurate Text Classification

[1]Iram Javed and [2]Arulmurugan Ramu
[1]School of Information and Software Engineering, University of Electronics Science and Technology of China, Hefei, Anhui, 230026, P.R.China.
[2]Mattu University, Metu, Ethiopia.
[1]iram.javed1@hotmail.com, [2]arul.murugan@meu.edu.et

Correspondence
Arulmurugan Ramu
arul.murugan@meu.edu.et

Article Info
Journal of Future Networks and Communications
(https://sepub.tw/journals/jfnc/jfnc.html)

Received 25 October 2024
Revised from 10 December 2024
Accepted 18 December 2024
Available online 05 January 2025

Abstract – Text categorization plays a crucial role in various NLP applications, including sentiment analysis, subject labeling, and question answering. Machine-learning-based text classification is a key research subject with numerous applications, including spam detection, hate speech identification, reviews, rating summarization, sentiment analysis, and topic modeling. Machine learning algorithms for text classification have proven to be quite reliable and are widely employed. We introduce SentCNN, based on convolutional neural network (CNN) for text classification that addresses NLP challenges. SentCNN's simplified architecture ensures outstanding performance while lowering processing costs. In this work, Stanford Sentiment Treebank dataset is used for text classification. The proposed SentCNN achieved remarkable results with an accuracy of 95.2% with 95.6% recall, and an F1 score of 95.2% and compared with sophisticated models like DPCNN, RCNN, and TextCNN. The comparative results shows that the proposed SentCNN outperformed existing models and effective for text classification.

Keywords – Text Classification, Natural Language Processing, Convolutional Neural Networks and Stanford Sentiment Treebank.

1. INTRODUCTION

Over the course of the last ten years, there has been a substantial rise in the number of digital documents and complex text data. There are several applications of natural language processing (NLP), such as sentiment analysis, subject labeling, and question answering, in which text categorization plays an important purpose. It is essential for organizations and sales to do sentiment research since it offers essential insights and enables informed decision-making. Within the context of the information age that we live in, the manual processing and classification of enormous amounts of textual data can be both time-consuming and complicated. It is possible for human variables, such as fatigue and a lack of topic knowledge, to have an impact on the accuracy of manual text classification. The reliability of categorization processes can be improved by the use of machine learning approaches combined with automation. A significant portion of this is performed through the use of rule-based approaches that classify texts according to specific subject knowledge, machine learning (ML) techniques, and pre-established rules [1].

This is accomplished mostly by automatic labeling or annotation. The categorization of text is a key component of natural language processing (NLP) due to the fact that it serves a multitude of purposes, including text classification, web search, ranking, data retrieval, and spam filtering techniques. Researchers have recently shown an interest in a variety of models that are based on neural networks, and the methodologies that they have developed have resulted in successful outcomes in applications that are used in the real world. Nevertheless, they are only useful for really large datasets because they appear to be extremely slow throughout both the training and testing phases. Because of developments in machine learning techniques, more complex models can now be trained on significantly larger datasets. Most of the time, the performance of these models is superior to that of simpler models [2].

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

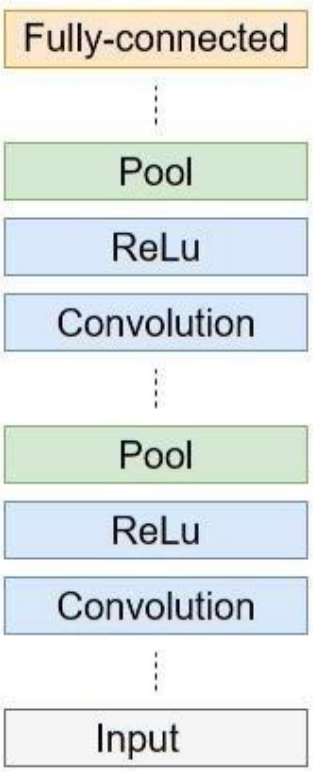

Figure 1. General CNN Architecture

Figure 1 shows the general architecture of CNN. In recent years, deep learning has made considerable advancements in natural language processing (NLP), which has attracted a lot of interest for its application in text processing [3]. During the training process, language models that are based on neural networks are taught to generate word vectors, which are extra outputs of the models. One of the most important ideas is to make predictions about the terms that are most likely to appear in a specific corpus context. Essentially, this technique involves learning the statistical properties of the words that appear together and then mapping those words into a space with fewer dimensions. Although text classification can be done manually, doing so is a process that is not only labor-intensive but also expensive, time-consuming, and time-consuming. Methods that are carried out manually are prone to errors, which are typically the consequence of human error and a lack of expertise regarding the subject matter.

Text classification underwent a significant transformation as a consequence of the implementation of machine learning models such as Support Vector Machines (SVM), Naive Bayes (NB), and Random Forest (RF) [4,5]. These models not only increased the accuracy of categorization but also reduced the amount of time and money required. Significant research has been conducted ever since the method of text categorization was first developed with the goal of enhancing, optimizing, and refining it in order to achieve even greater levels of efficiency and precision [6].

The article is organized as follows. **Section II** overviews the existing text classification methods based on deep learning. **Section III** describes the proposed text classification method termed as SentCNN and the results are explained in **Section IV**. The detailed comparative analysis is done in **Section IV** with concluding statements in **Section V.**

2. RELATED WORKS

Over the past few years, there has been a significant advancement in the classification of texts [7]. By way of illustration, Prabhakar et al. [8] suggested two hybrid deep learning models that make advantage of attention mechanisms in order to achieve excellent classification accuracy across a wide range of datasets. A transformer encoder-decoder-based method was presented by Duan et al. for the purpose of multilabel text categorization [9]. This method successfully extracts dependencies between texts and labels. Li et al. [10] used a BERT-based automatic classification model in the energy sector policy area, and it performed better than other models in terms of accuracy, recall, and F1 scores. This was the case in comparison to other models. A dual-channel XLNet–CNN–GRU technique was developed by Wu et al. [11] for the purpose of aspect-level sentiment classification of comments. This technique surpassed five benchmark neural network models in natural language processing (NLP) respectively in terms of accuracy and F1 scores.

Word2Vec is a model that was presented by Mikolov et al. [12], which provides a representation of words as dense vectors with low dimensions. By utilizing the distances that exist between them, these vectors are able to quantify the similarity of words, so successfully capturing the semantic linkages that exist between words. Using this as a foundation, Kim et al. [13] introduced the TextCNN model for text classification, which involved the use of pre-trained Word2Vec

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

embeddings. Despite the fact that TextCNN has shown impressive performance in text classification tasks, it has difficulty successfully capturing local textual nuances and contextual linkages.

2.1 Deep Learning Based Text Classification

This restriction was overcome by Liu et al. [14], who developed a Recurrent Neural Network (RNN) for the purpose of text classification. When it comes to extracting deeper contextual features from text, RNNs are particularly effective at capturing sequential and historical information inside the sentence. On the other hand, they have difficulties remembering information, particularly when it comes to digesting lengthy word sequences. An improvement in text classification was achieved by Lee et al. [15] by combining RNN and CNN models. This was accomplished by improving text representations. A Very Deep Convolutional Neural Network (VDCNN) was developed by Conneau et al. [16] with the intention of achieving superior performance. This CNN handles input data at the character level, as opposed to the conventional word-level preprocessing. The RCNN, which is a mix of the RNN and the CNN, was proposed by Lai et al. [17] in order to further increase the accuracy of text classification.

Zhang et al. [18] presented two character-level CNN models for text categorization and compared them to classic natural language processing (NLP) techniques like bag-of-words and n-grams, as well as deep learning techniques such as word vector-based CNNs and recurrent neural networks (RNNs). According to their findings, traditional approaches perform better on smaller datasets, whereas CNNs do exceptionally well with user-generated data, with semantics playing a limited role in the process. In addition to developing a Recursive Neural Tensor Network, Socher et al. [19] built the Sentiment Treebank dataset, which had 11,855 phrases that were labeled for the purpose of sentiment analysis. Their approach beat earlier methods and captured negation in a way that was not possible before, which improved sentiment identification. These models, on the other hand, have difficulty capturing long-range dependencies and preserving information in long sequences. In addition, models like as recurrent neural networks (RNN), support vector machines (SVM), and hybrid models have a tough time completely comprehending local textual elements and intricate contextual relationships when it comes to jobs that require fine-grained analysis.

CNN architectures that make use of domain-specific word embeddings have reportedly demonstrated good results in multi-label classification tasks that involve brief texts, as stated by Parwez [20]. Similarly, Tang et al. [21] highlight the need of using sentiment-based word embedding models in order to successfully differentiate between words with opposite polarities that are used in contexts that are comparable. The textual information as well as the word contexts are both captured by these models. In this method, a CNN is used to extract features in order to build high-quality phrase representations. This is accomplished by utilizing enhanced word embedding techniques that are trained on the co-occurrence of words within a corpus or dataset. The practice of expressing the words in a document in an R-dimensional vector space is known as word embedding. This technique is essential to natural language processing since it enables the capture of semantics, word similarity, and syntactic information. The primary contribution of this study is the development of a lightweight convolutional neural network (CNN) model for text classification that is referred to as SentCNN. This model is utilized for character-level embeddings. When it comes to accuracy and the word error rate (WER), the performance of the model that has been proposed has the potential to be efficient.

3. PROPOSED FRAMEWORK FOR CLASSIFICATION

SentCNN is an architecture that processes text data in various layers, each of which has special characteristics to boost its learning power, as shown in **Figure 1**. When the sentence is represented by the input layer, it is represented as a matrix of dimensions N×d. Here, N represents the number of words in the phrase, and dd denotes the embedding dimension (for example, 300 for GloVe embeddings). For the purpose of capturing n-gram information, the convolutional layer employs numerous filters, also known as kernels, with widths k that range from two to four respectively. In order to learn a wide variety of features, each filter generates a feature map, and f filters (for example, 100–256 filters) are employed for each kernel size.

The amount of distance that the filter travels over the input is determined by the stride, which is commonly set to 1. Not only does a ReLU activation function introduce non-linearity after convolution, but it also makes it possible for the network to learn complicated patterns. In order to minimize the feature maps, the pooling layer, which frequently makes use of max-pooling, selects the maximum value in each zone. This helps to condense the information while also reducing the dimensionality. Global max-pooling is a technique that is frequently utilized, which ultimately leads to a vector of size f that is of a fixed length for the fully connected layers. The collected features are incorporated into a high-level representation by these layers, and the final output layer, which utilizes a softmax activation function, computes probabilities for each class, which enables text categorization. The hierarchical feature extraction and dimensionality reduction of this architecture are efficiently combined for the purpose of conducting robust text analysis.

As can be seen in **Figure 2,** the suggested model for text classification follows a straightforward flow. A sentence matrix with dimensions N×d is provided as the input. Here, NNN represents the total number of words, and d represents the embedding dimension. Additionally, in order to enhance the stability of training and incorporate non-linearity, two convolutional layers with 3×3 filters (sizes f1 and f2) are utilized. Each of these layers is then followed by batch normalization and ReLU activation process. Each convolution is followed by a down sampling of the feature maps using the max pooling layers (2×2, stride 2). After that, the network moves on to fully linked levels, which include 128 and 64

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

neurons respectively, and both of these layers use ReLU activation. A softmax-activated output layer is the final layer that is responsible for predicting class probabilities for the target classification task. **Table 1** provides a summary of the architecture that has been offered.

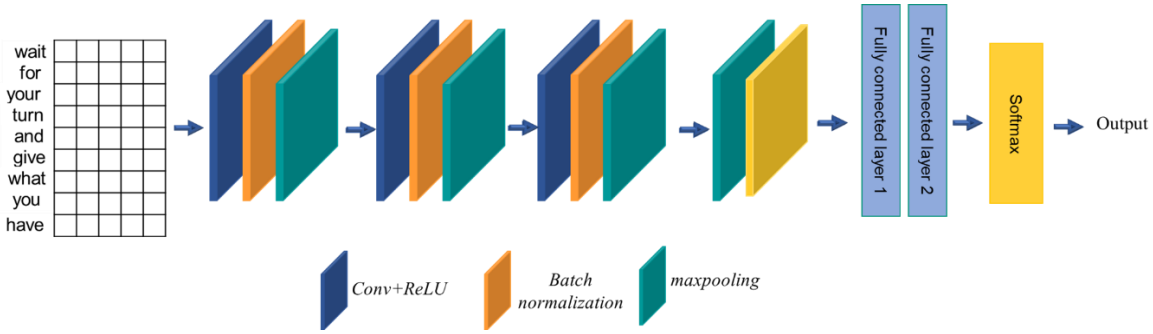

Figure 2. SentCNN Architecture

SentCNN, or Sentiment Convolutional Neural Network, is a lightweight and efficient model designed for text classification tasks within the field of Natural Language Processing (NLP). The architecture of SentCNN is intentionally simplified to ensure high performance while minimizing computational overhead. At its core, the SentCNN model begins with an embedding layer. This layer converts words into dense vectors of fixed size, capturing semantic meanings and relationships between words. By using pre-trained word embeddings (such as Word2Vec or GloVe), the model leverages prior knowledge to represent the input text in a numerical format that the neural network can process effectively.

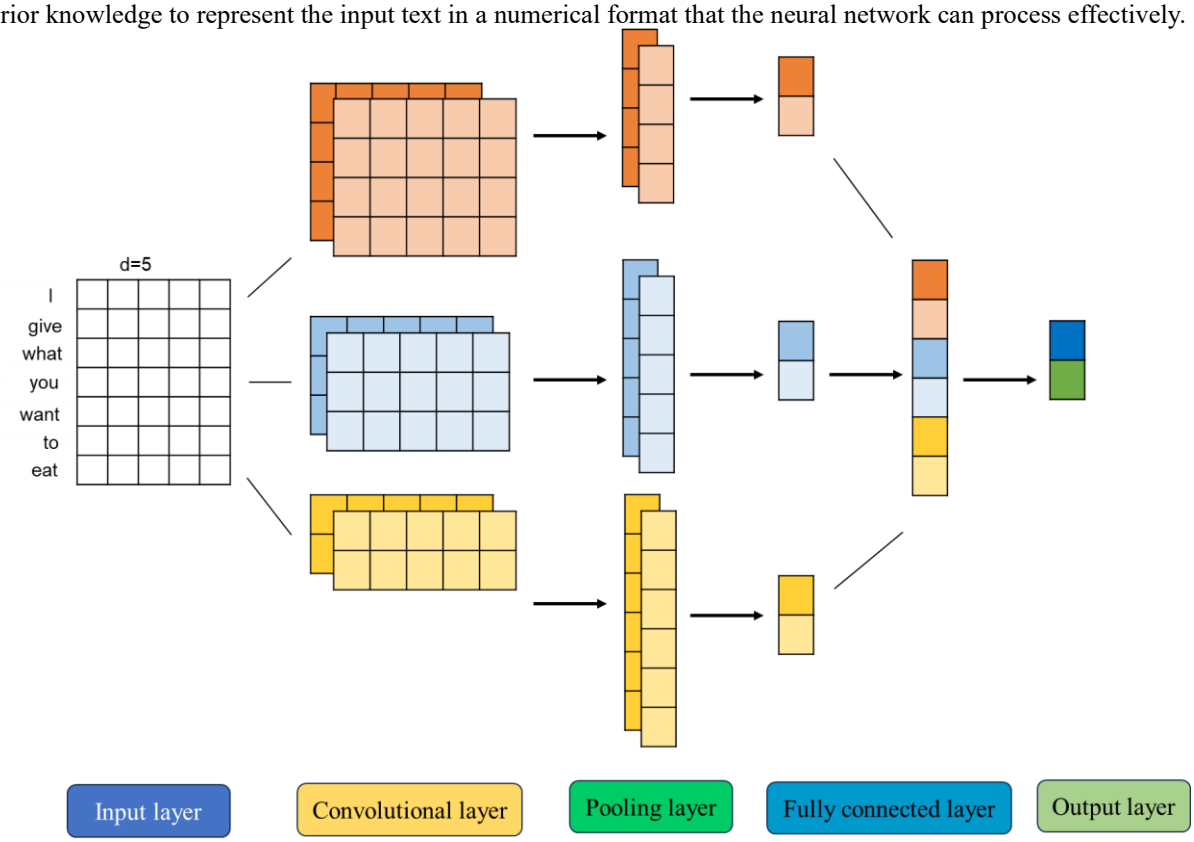

Figure 3. Flow Diagram of Proposed SentCNN Architecture for Text Classification

Following the embedding layer, SentCNN employs **Figure 3** one or more convolutional layers. These layers apply convolutional filters across the text data to detect local patterns and n-grams (sequences of words). The filters slide over the text, capturing relevant features such as phrases or sentiment-bearing terms. This process helps the model identify important structures within the text that contribute to classification decisions. To reduce the dimensionality of the feature maps generated by the convolutional layers, pooling layers are introduced. Specifically, global max-pooling is used to extract the most significant features from each filter. This operation significantly reduces the computational burden by downsampling the data, while still retaining the most crucial information for classification.

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

The pooled features are then fed into one or more fully connected (dense) layers. These layers act as high-level feature extractors, further refining the representations learned by the convolutional layers. By introducing dropout regularization, the model combats overfitting by randomly disabling a fraction of the neurons during training. This enhances the model's generalization capability, making it more robust to unseen data. The output of the fully connected layers is passed through a softmax activation function (or sigmoid for binary classification), which produces the final classification probabilities. This output layer ensures that the model can assign a confidence score to each class, allowing for accurate and interpretable predictions.

Table 1. Summary of Proposed Architecture

Layer	Size	Other Parameters
Input	Sentence Matrix (N x d)	N = number of words, d = embedding dimension
Convolution 1	3 x 3 x f1	Activation: ReLU
Batch Normalization	-	Applied after Convolution 1
Max Pooling 1	2 x 2	Stride: 2
Convolution 2	3 x 3 x f2	Activation: ReLU
Batch Normalization	-	-
Max Pooling 2	2 x 2	Stride: 2
Fully Connected 1	128 neurons	Activation: ReLU
Fully Connected 2	64 neurons	Activation: ReLU
Output	Number of classes	Activation: Softmax

The **Table 1** provides a comprehensive overview of the architecture for a text classification model, specifically detailing the layers involved in the SentCNN model, along with their sizes and other relevant parameters. By analyzing this architecture, several key inferences can be drawn about the design and functionality of the model. The SentCNN model is designed to be both lightweight and efficient. The use of essential layers such as convolutional, pooling, and fully connected layers suggests an architecture that prioritizes performance without excessive complexity. This simplification helps in reducing computational costs while maintaining high accuracy.

The inclusion of multiple convolutional layers (Convolution 1 and Convolution 2) with ReLU activation functions indicates a robust mechanism for feature extraction. These layers are capable of capturing intricate patterns and structures within the text, which are crucial for accurate classification. The application of batch normalization layers after each convolutional layer highlights the model's focus on stability and performance optimization. Batch normalization helps in normalizing the output of convolutional layers, which accelerates the training process and enhances model convergence.
Max pooling layers are used to reduce the spatial dimensions of the feature maps generated by the convolutional layers. This dimensionality reduction retains the most important features while decreasing the computational burden, enabling the model to process data more efficiently. The fully connected layers (Fully Connected 1 and Fully Connected 2) play a crucial role in forming high-level feature representations. With 128 and 64 neurons respectively, these layers further refine the features extracted by the convolutional and pooling layers, enhancing the model's classification capabilities.

The output layer, equipped with a softmax activation function, ensures that the model can provide probabilistic predictions for multiple classes. This is critical for text classification tasks, where the goal is to assign each input text to a specific category with a high degree of confidence. The architecture's emphasis on simplicity and efficiency makes SentCNN a practical choice for real-world applications in text classification. The model's design ensures that it can be deployed on various platforms without requiring extensive computational resources.

4. ANALYSIS AND DISCUSSION

4.1 Evaluation Environment

Google Colab and TensorFlow with its high-level API, Keras, as the deep learning framework, were used in this experiment. There are two datasets are used.

Table 2. Performance Metrics of Proposed SentCNN for Text Classification

Dataset	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)
Stanford Sentiment Treebank (SSTb)	95.2	94.8	95.6	95.2
Customized dataset	94.7	94.2	95.1	94.6

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

The Stanford Sentiment Treebank (SST) is a renowned dataset extensively used for natural language processing (NLP) tasks, particularly sentiment analysis. Introduced by Socher et al., SST is unique for its fine-grained sentiment annotations and syntactic tree structure. The dataset comprises 11,855 sentences from movie reviews, where each sentence is parsed into a binary tree structure. Every node in this tree represents a phrase, enabling sentiment analysis at both the sentence and phrase levels. Each phrase is annotated with one of five sentiment categories: very negative, negative, neutral, positive, or very positive, allowing for detailed sentiment classification. The customized dataset, which includes 36 English words, is used to generate training and testing data. The dataset is split into 9:1 ratio for testing and training, respectively. The performance of proposed model is shown in **Table 2**.

The table illustrates the high performance of the SentCNN model on two datasets: the Stanford Sentiment Treebank (SSTb) and a customized dataset. On the SSTb dataset, the model achieves an accuracy of 95.2%, precision of 94.8%, recall of 95.6%, and an F1 score of 95.2%. Similarly, on the customized dataset, the model performs admirably with an accuracy of 94.7%, precision of 94.2%, recall of 95.1%, and an F1 score of 94.6%. These metrics highlight the model's robustness and efficacy in text classification tasks, demonstrating its ability to generalize well across different datasets and maintain consistently high performance.

4.2 Performance Indicators

Accuracy measures the proportion of correctly classified sentences out of the total sentences in the dataset. It is the most intuitive metric but may not reflect the model's performance well when dealing with imbalanced datasets.

$$Accuracy = \frac{True\ Positives\ (TP) + True\ Negatives\ (TN)}{Total\ Predictions\ (TP + TN + FP + FN)} \tag{1}$$

Precision evaluates the proportion of true positive predictions out of all positive predictions made by the model. It is crucial when minimizing false positives is important.

$$Precision = \frac{True\ Positives\ (TP)}{TTrue\ Positives\ (TP) + False\ Positives\ (FP)} \tag{2}$$

Recall measures the proportion of true positives correctly identified by the model out of all actual positive instances. It is essential when minimizing false negatives is a priority.

$$Recall = \frac{True\ Positives\ (TP)}{TTrue\ Positives\ (TP) + False\ Negatives\ (FN)} \tag{3}$$

The F1 score is the harmonic mean of precision and recall given by,

$$F1 - score = 2 \times \frac{Precision * Recall}{Precision + Recall} \tag{4}$$

Table 3 compares the performance of the proposed SentCNN model with that of the other deep learning frameworks, namely DPCNN, RCNN, TextCNN, and FastText. The comparison assesses parameters like accuracy, recall, and F1 score and demonstrates the effectiveness of proposed method in text categorization tasks. DPCNN and RCNN measured 90.9% accuracy with almost similar results in terms of recall and F1 score. Although they achieved better results, these models limit their classification ability as they cannot capture local features accurately. TextCNN achieved an accuracy of 91.28%, placing it slightly ahead of these models to show how well it captures semantic patterns by convolutional processes. With 93.08% F1-score and 92.57% accuracy, FastText demonstrated a clear improvement due to its layered softmax and ease of data classification, considering the fact that the data's deep semantic and syntactic information was underutilized.

SentCNN has performed much better than all other models with an accuracy of 95.2%, recall of 95.6%, and F1 score of 95.2%. The main reason for this increase in performance was the ability of SentCNN to represent local and contextual features quite efficiently through convolutional architecture optimized to extract quality signals. Further, its straightforward architecture promotes speed in training while remaining strong across different datasets. Performance results illustrate the ability of SentCNN to process complicated language patterns, making it relevant in text classification.

Table 3. Comparative analysis of proposed SentCNN with existing text classification models

Model	Accuracy	Recall	F1-score
DPCNN [24]	90.9	90.9	90.9
RCNN [17]	90.9	90.7	90.83
TextCNN [13]	91.28	91.27	91.26
FastText [23]	92.57	92.88	93.08
Proposed SentCNN	**95.2**	**95.6**	**95.2**

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

The **Table 3** showcases the performance of various text classification models, including DPCNN, RCNN, TextCNN, and FastText, alongside the proposed SentCNN model. It is evident that the proposed SentCNN model significantly outperforms the other models across all three metrics: accuracy, recall, and F1-score. While DPCNN and RCNN both achieve 90.9% accuracy, they fall short in recall and F1-score. TextCNN slightly improves with an accuracy of 91.28%, and FastText shows a considerable improvement with an accuracy of 92.57%, recall of 92.88%, and an F1-score of 93.08%. However, SentCNN surpasses all with a remarkable accuracy of 95.2%, recall of 95.6%, and an F1-score of 95.2%, highlighting its superior performance and robustness in text classification tasks. This performance demonstrates SentCNN's effectiveness and efficiency in capturing and classifying text data accurately, making it a highly suitable choice for various NLP applications.

5. CONCLUSION

This work proposed a CNN based text classification model called as SentCNN which achieved effective and straightforward in solving the extraction of both localized and general context information from the textual asset. The proposed SentCNN surpassed widely recognized models such as DPCNN, RCNN, or TextCNN by achieving 95.2% in accuracy, 95.6% in recall, and 95.2% for F-measure using Stanford Sentiment Treebank dataset. One of the most significant features of SentCNN is its adaptability to a range of NLP tasks, which includes sentiment analysis, spam filtering, or topic classification. It affords strength in its reliability versatility that will be illustrated through the consistent performance of the model across a variety of datasets. The model is also very ideal for application on low-resource conditions such as mobile devices or edge computing platforms since it incurs very little processing effort. This opens new promising future directions for research, such as expanding the application of SentCNN to specific and multilingual domains, combining attention mechanisms to support better processing of context, or even exploring hybrid architectures that merge CNNs and transformer models. Indeed, SentCNN presents a promising framework for developing effective and efficient NLP solutions capable of confronting difficult linguistic scenarios in realistic environments.

CRediT Author Statement

The authors confirm contribution to the paper as follows:
Conceptualization: IJ, AR; **Methodology:** IJ, AR; **Software:** AR; **Data Curation:** IJ; **Writing- Original Draft Preparation:** IJ, AR; **Visualization:** IJ; **Supervision:** AR; **Validation:** IJ, AR; **Writing- Reviewing and Editing:** IJ, AR; **Writing- Original Draft:** IJ, AR; All authors reviewed the results and approved the final version of the manuscript.

Data Availability

The datasets generated during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interests

The authors declare that they have no conflicts of interest regarding the publication of this paper.

Funding

No funding was received for conducting this research.

Competing Interests

The authors declare no competing interests.

References

[1]. Palanivinayagam, C. Z. El-Bayeh, and R. Damaševi˘cius, “Twenty Years of Machine-Learning-Based Text Classification: A Systematic Review,” Algorithms, vol. 16, p. 236, 2023, doi: 10.3390/a16050236.
[2]. F. Sebastiani, “Machine Learning in Automated Text Categorization,” ACM Comput. Surv., vol. 34, pp. 1–47, 2002.
[3]. H. Kim and Y.-S. Jeong, “Sentiment Classification Using Convolutional Neural Networks,” Appl. Sci., vol. 9, p. 2347, 2019, doi: 10.3390/app9112347.
[4]. K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, “Text Classification Algorithms: A Survey,” Information, vol. 10, p. 150, 2019.
[5]. D. W. Otter, J. R. Medina, and J. K. Kalita, “A Survey of the Usages of Deep Learning for Natural Language Processing,” IEEE Trans. Neural Netw. Learn. Syst., vol. 32, no. 2, pp. 604–624, Feb. 2021, doi: 10.1109/TNNLS.2020.2979670.
[6]. E. O. Arkhangelskaya and S. I. Nikolenko, “Deep Learning for Natural Language Processing: A Survey,” J. Math. Sci., vol. 273, pp. 533–582, 2023, doi: 10.1007/s10958-023-06519-6.
[7]. L. He, S. Tan, F. Xiang, J. Wu, and L. Tan, “Research and development of deep learning-based text classification,” Comput. Eng., vol. 47, pp. 1–11, 2021.
[8]. S. K. Prabhakar, H. Rajaguru, and D. O. Won, “Performance Analysis of Hybrid Deep Learning Models with Attention Mechanism Positioning and Focal Loss for Text Classification,” Sci. Program., vol. 2021, p. 2420254, 2021.
[9]. L. Duan, Q. You, X. Wu, and J. Sun, “Multilabel Text Classification Algorithm Based on Fusion of Two-Stream Transformer,” Electronics, vol. 11, p. 2138, 2022.
[10]. Q. Li, Z. Xiao, and Y. Zhao, “Research on the Classification of New Energy Industry Policy Texts Based on BERT Model,” Sustainability, vol. 15, p. 111, 2023.
[11]. D. Wu, Z. Wang, and W. Zhao, “XLNet-CNN-GRU dual-channel aspect-level review text sentiment classification method,” Multimed. Tools Appl., vol. 83, pp. 5871–5892, 2024.

Doi: https://doi.org/XXXX/XXXX/JFNC202501004

[12]. T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in Adv. Neural Inf. Process. Syst., vol. 26, 2013.
[13]. Y. Kim, "Convolutional Neural Networks for Sentence Classification," in Proc. 2014 Conf. Empirical Methods Natural Lang. Process., Doha, Qatar, Oct. 2014, pp. 1746–1751.
[14]. P. Liu, X. Qiu, and X. Huang, "Recurrent Neural Network for Text Classification with Multi-Task Learning," arXiv, 2016, arXiv:1605.05101.
[15]. J. Y. Lee and F. Dernoncourt, "Sequential short-text classification with recurrent and convolutional neural networks," arXiv, 2016, arXiv:1603.03827.
[16]. A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," arXiv, 2016, arXiv:1606.01781.
[17]. S. Lai, L. Xu, K. Liu, and J. Zhao, "Recurrent convolutional neural networks for text classification," in Proc. Twenty-Ninth AAAI Conf. Artif. Intell., Austin, TX, USA, Jan. 2015, pp. 2267–2273.
[18]. X. Zhang, J. Zhao, and Y. LeCun, "Character-level convolutional networks for text classification," Adv. Neural Inf. Process. Syst., vol. 2015, 2015.
[19]. R. Socher, A. Perelygin, et al., "Recursive deep models for semantic compositionality over a sentiment treebank," in Proc. Conf. Empirical Methods Natural Lang. Process., 2013, pp. 1631–1642.
[20]. M. A. Parwez and M. J. I. A. Abulaish, "Multi-label classification of microblogging texts using convolution neural network," IEEE Access, vol. 7, pp. 68678–68691, 2019.
[21]. D. Tang, F. Wei, B. Qin, N. Yang, T. Liu, M. Zhou, and D. Engineering, "Sentiment embeddings with applications to sentiment analysis," IEEE Trans. Knowl. Data Eng., vol. 28, pp. 496–509, 2015.
[22]. "Stanford Sentiment Treebank Dataset," [Online]. Available: https://nlp.stanford.edu/sentiment/code.html/. [Accessed: Nov. 11, 2024].
[23]. W. Hu, J. Xiong, N. Wang, F. Liu, Y. Kong, and C. Yang, "Integrated Mod el Text Classification Based on Multineural Networks," Electronics, vol. 13, p. 453, 2024, doi: 10.3390/electronics13020453.
[24]. R. Johnson and T. Zhang, "Deep Pyramid Convolutional Neural Networks for Text Categorization," in Proc. 55th Annu. Meeting Assoc. Comput. Linguist., 2017, pp. 562–570.