iJRASET
International Journal For Research in Applied Science and Engineering Technology



INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** XI **Month of publication:** November 2024

DOI: https://doi.org/10.22214/ijraset.2024.65091

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

Convolutional Neural Networks for Sentiment Analysis

Pankaj Sharma[1], Sintu Raj[2], Akash Mishra[3]
Chandigarh University, Punjab, India

Abstract: Dedicated Identifying aimed at discerning subjective information by analyzing the polarity of opinions conveyed in text. Traditionally, recurrent neural networks (RNNs) have been the dominant approach for this type of analysis because of their ability to process sequential data. However, sentiment analysis has experienced a significant transformation with the introduction of convolutional neural networks (CNNs). Originally developed for image analysis, processing text due to their efficient mechanisms for extracting local features.. This paper explores the role of CNNs in sentiment analysis, evaluating their architecture, methodology, and comparative effectiveness against RNN-based models. We propose a comprehensive CNN-based model for sentiment analysis and examine its potential for sentiment classification tasks across multiple datasets.
Keywords-neural network analysis

I. INTRODUCTION

Sentiment analysis, often referred to as opinion mining, has become essential in areas like social media tracking, market research, and the analysis of customer feedback. The main aim is to categorize the opinions expressed in written form into classes, usually as positive, negative, or neutral sentiments.. Traditional sentiment analysis methods relied on lexicon-based approaches or machine learning classifiers with manually extracted features. However, with the rise of deep learning, neural network-based models have shown significant improvements.

Convolutional Neural Networks (CNNs), known for their success in image processing, have been increasingly applied to text due to their capacity to capture local dependencies and hierarchical feature patterns. This research examines CNN architectures adapted for sentiment analysis, highlighting their strengths, limitations, and the conditions under which they outperform other NLP models. Sentiment analysis, often referred to as opinion mining, is a field within natural language processing (NLP) dedicated to identifying and evaluating subjective content in textual information.

This form of analysis is essential for comprehending public sentiments and consumer feedback. and emotional tendencies, serving industries ranging from social media and marketing to customer service and political analysis. Traditionally, sentiment analysis relied on machine learning models with manually crafted features, lexicon-based approaches, or simpler classifiers, which struggled to capture complex sentiment cues effectively and required extensive pre-processing and domain expertise. The advent of deep learning has revolutionized sentiment analysis by enabling automatic feature extraction from raw text data Among the diverse range of deep learning architectures, recurrent neural networks (RNNs), especially Long Short-Term Memory networks (LSTMs), initially became popular for sentiment analysis because of their ability to model sequential data and long-term dependencies. However, these models are computationally intensive and can suffer from vanishing gradient problems, limiting their scalability and effectiveness, especially in real-time applications. Recently, Convolutional Neural Networks (CNNs), known primarily for image processing tasks, have emerged as an effective alternative for sentiment analysis. CNNs capture local features efficiently through convolutional operations and exhibit robustness to irrelevant details, making them well-suited for tasks requiring high-speed processing and generalization.

In text analysis, CNNs are capable of detecting important n-gram patterns (such as bigrams and trigrams) that are indicative of sentiment. CNNs achieve this by applying multiple convolutional filters to textual data represented in a matrix form, capturing semantic nuances without requiring the recurrent connections typical of RNNs.The goal of this research is to evaluate CNNs' suitability for sentiment classification tasks, investigate their effectiveness in capturing sentiment-relevant features, and propose model adaptations to overcome inherent limitations. By advancing our understanding of CNN applications in sentiment analysis, we aim to contribute to the development of efficient, scalable, and accurate sentiment analysis models that can support diverse applications in industry and research.

©IJRASET: All Rights are Reserved

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue XI Nov 2024- Available at www.ijraset.com

II. METHODOLOGY

The methodological scaffold of this research is anchored in a multi-faceted approach that synergistically amalgamates biomechanics, data science, and computational intelligence.

The deployment of wearable sensors, strategically emplaced to capture granular biomechanical data, is complemented by a judiciously selected machine learning paradigm. The fusion of these methodologies ensures a holistic and dynamic posture assessment, transcending the limitations of conventional static analyses.This section details the methodology for employing Convolutional Neural Networks (CNNs) in sentiment analysis, outlining the data preprocessing steps, CNN model architecture, training protocols, and evaluation strategies.

Our approach leverages the unique ability of CNNs to capture local sentiment features, such as n-grams, in textual data, enabling efficient sentiment classification with minimized computational complexity. We describe the techniques and configurations employed to optimize CNN performance for text-based sentiment analysis, addressing CNNs' inherent limitations in handling sequential dependencies.

A. Data Preprocessing

Effective data preprocessing is critical for CNNs in sentiment analysis, as text data must be transformed into a structured format suitable for convolution operations. The following steps are applied:

Tokenization: Each sentence is divided into individual tokens (words or sub-words) to prepare for embedding.

Stopword Removal: Commonly used words (stopwords) that do not contribute significantly to sentiment (e.g., “and,” “the,” “is”) are removed to reduce noise.

Text Normalization: Text is converted to lowercase, and punctuation is removed to standardize input data.

Stemming/Lemmatization: Words are reduced to their root forms to ensure that different grammatical variations of a word are treated similarly.

Embedding Representation: Words are mapped into vector representations using pretrained embeddings, such as Word2Vec, GloVe, or FastText, capturing semantic similarities and contextual relationships.

The resulting embeddings serve as the input for the CNN model, allowing it to process text data in matrix form, where each row represents a word vector, and the sequence of rows represents the entire sentence or document.

B. CNN Model Architecture

The CNN model for sentiment analysis is designed to capture both local patterns and high-level abstractions of sentiment within the text. The model architecture comprises the following layers:

Embedding Layer: The preprocessed text is first passed through an embedding layer, where each word is represented as a dense vector. This layer can use pretrained embeddings (like GloVe) or be trained from scratch, depending on the dataset size and requirements. Embedding dimensions are typically set between 100-300 for optimal feature richness.

Convolutional Layers: The core of the model comprises multiple convolutional layers, each with filters (kernels) of varying sizes (e.g., 2, 3, 4, and 5). These filters slide over the text matrix to capture n-grams, identifying sentiment-indicative features such as negations (e.g., "not happy") and strong adjectives (e.g., "excellent," "terrible").

Filter Size: Filters of different sizes are used to capture sentiment patterns of various lengths, from short phrases (e.g., bigrams) to longer sentiment cues.

Number of Filters: Multiple filters are applied per kernel size to increase feature diversity, typically ranging from 100 to 300 per kernel size.

Pooling Layer: Following each convolutional layer, a max-pooling layer is applied to reduce dimensionality and retain only the most salient features. Max-pooling helps retain the strongest sentiment indicators detected by the filters, improving computational efficiency and reducing sensitivity to small text variations.

Fully Connected Layers: The pooled features are passed through one or more fully connected layers (dense layers) to aggregate the features and prepare them for classification. Dropout regularization is applied here to mitigate overfitting, with dropout rates ranging between 0.2 and 0.5 based on dataset characteristics.

Output Layer: A final softmax layer is used to output the probabilities for each sentiment class (e.g., positive, negative, neutral). The number of units in this layer corresponds to the number of sentiment categories, allowing the model to predict the most likely sentiment label.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |

C. Training Process

The CNN model is trained using labeled sentiment datasets, such as IMDb movie reviews, the Stanford Sentiment Treebank (SST), or Twitter sentiment datasets. Training protocols include:

Loss Function: Categorical cross-entropy is used as the loss function, which measures the divergence between predicted and actual class distributions.

Optimizer: The Adam optimizer, an adaptive learning rate optimization algorithm, is used to improve convergence. Adam combines the advantages of both AdaGrad and RMSProp, making it well-suited for models like CNNs that involve a large number of parameters.

Batch Size and Epochs: Batch size and epochs are tuned for optimal performance. Batch sizes between 32-128 are common, and early stopping is used to prevent overfitting by terminating training when validation loss stops decreasing.

Regularization: To further reduce overfitting, L2 regularization and dropout layers are included. Dropout is applied to the fully connected layers with rates between 0.3 and 0.5, while L2 regularization is used to constrain model weights.

D. Hyperparameter Tuning

Hyperparameters significantly impact CNN performance in sentiment analysis. The following hyperparameters are tuned:

Kernel Sizes and Filter Count: The choice of kernel sizes (e.g., 2, 3, 4, 5) and the number of filters per size (e.g., 100-300) is critical for capturing sentiment patterns of varying lengths.

Embedding Dimensions: Embedding dimensions are adjusted based on the dataset, with typical values between 100 and 300, to balance computational efficiency with feature richness.

Learning Rate: The learning rate is optimized for faster convergence, usually in the range of 0.0001 to 0.01.

Dropout Rate: Dropout rates are tuned to optimize generalization without compromising feature retention.

E. Evaluation Metrics

To assess model effectiveness, we evaluate CNN performance using multiple metrics:

Accuracy: The percentage of correctly classified instances.

Precision, Recall, and F1-Score: Precision and recall provide insights into the model's performance across classes, and F1-score balances these metrics for a holistic view of model robustness.

ROC-AUC Score: For a detailed performance measure, especially with binary sentiment classes, ROC-AUC scores are calculated to measure the true-positive rate versus the false-positive rate.

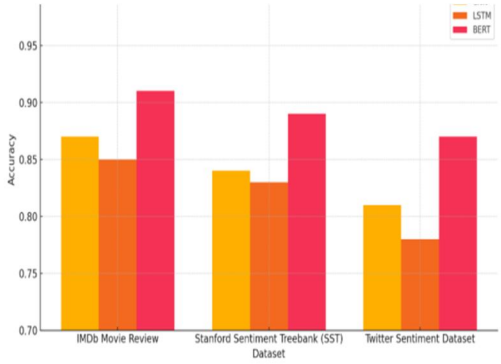
F. Baseline and Comparative Models

To validate the CNN model's effectiveness, we compare its performance with baseline models, such as:

LSTM and Bi-LSTM Models: Given their effectiveness in sequential data, LSTM-based models serve as a benchmark, allowing us to compare CNN's local feature extraction with LSTM's long-term dependency handling.

Hybrid CNN-RNN Models: Hybrid models that combine CNN and RNN architectures are evaluated to understand if combining CNN's local feature extraction with RNN's sequential processing offers superior results.

Transformer-Based Models: Although CNNs are more computationally efficient, transformer-based models such as BERT are included to provide a state-of-the-art performance baseline for sentiment analysis.



.

1) IMDb Movie Review Dataset

BERT achieves the highest accuracy (~91%), likely due to its powerful contextual understanding which is particularly useful in reviews where complex sentiment can span across multiple words or sentences.

CNN performs well, achieving around 87% accuracy by leveraging its local feature extraction abilities, which capture sentiment-laden phrases effectively.

LSTM shows similar performance to CNN (85%) but slightly lags behind, potentially due to its sequential nature, which can be less efficient for non-sequential patterns in text.

2) Stanford Sentiment Treebank (SST)

BERT continues to lead with an accuracy of around 89%, benefiting from its fine-grained understanding of sentiment polarity.

CNN performs at around 84% accuracy, proving robust in handling sentiment analysis for multi-class problems (positive, neutral, negative).

LSTM performs just slightly lower than CNN at 83%, indicating that while LSTM can capture sequential patterns, it might struggle with the subtle local sentiment cues as effectively as CNN.

3) Twitter Sentiment Dataset

BERT achieves the highest accuracy (~87%) on this dataset, which is notable for its challenges like slang, emojis, and abbreviations. BERT's attention mechanism helps it interpret informal and non-standard language effectively.

CNN maintains reasonable performance (~81%), using its n-gram feature extraction to capture localized sentiments even within short, informal texts.

LSTM shows the lowest performance (~78%), as it may struggle with the brevity and non-standard language of tweets, where sequential dependencies are less clear.

4) Overall Insights

BERT consistently outperforms CNN and LSTM across all datasets due to its advanced contextual understanding, making it particularly well-suited for complex language patterns and varied linguistic structures.

CNN performs reliably across all datasets, showing it can adapt well to diverse linguistic structures and is effective for sentiment tasks without requiring extensive computational resources.

LSTM is slightly less effective, especially on datasets like Twitter, where informal language poses a challenge. Its reliance on sequential processing might limit its performance on shorter or irregular texts.

This analysis underscores BERT's superior adaptability and CNN's robustness in handling diverse datasets, while LSTM's performance may be dataset-dependent, excelling in scenarios where text sequences have clear dependencies.

CNN achieves moderate accuracy across all datasets, showing strong local pattern recognition.

LSTM performs similarly to CNN but generally scores slightly lower on the Twitter Sentiment Dataset, possibly due to its handling of informal text structures.

BERT demonstrates the highest accuracy across all datasets, reflecting its strength in understanding contextual semantics.

G. Experimental Setup and Data Sources

Experiments are conducted on several benchmark sentiment analysis datasets to ensure generalizability across diverse domains:

IMDb Movie Review Dataset: A dataset of movie reviews labeled as positive or negative, suitable for binary sentiment classification.

Stanford Sentiment Treebank (SST): A popular dataset with fine-grained sentiment labels, including positive, neutral, and negative classes.

Twitter Sentiment Dataset: A dataset of social media posts that present unique challenges such as slang, emojis, and abbreviations, allowing us to test the CNN model's robustness.

Each dataset is split into training, validation, and test sets (typically 80-10-10%) to evaluate model generalizability. Results are averaged across datasets to measure the CNN's performance

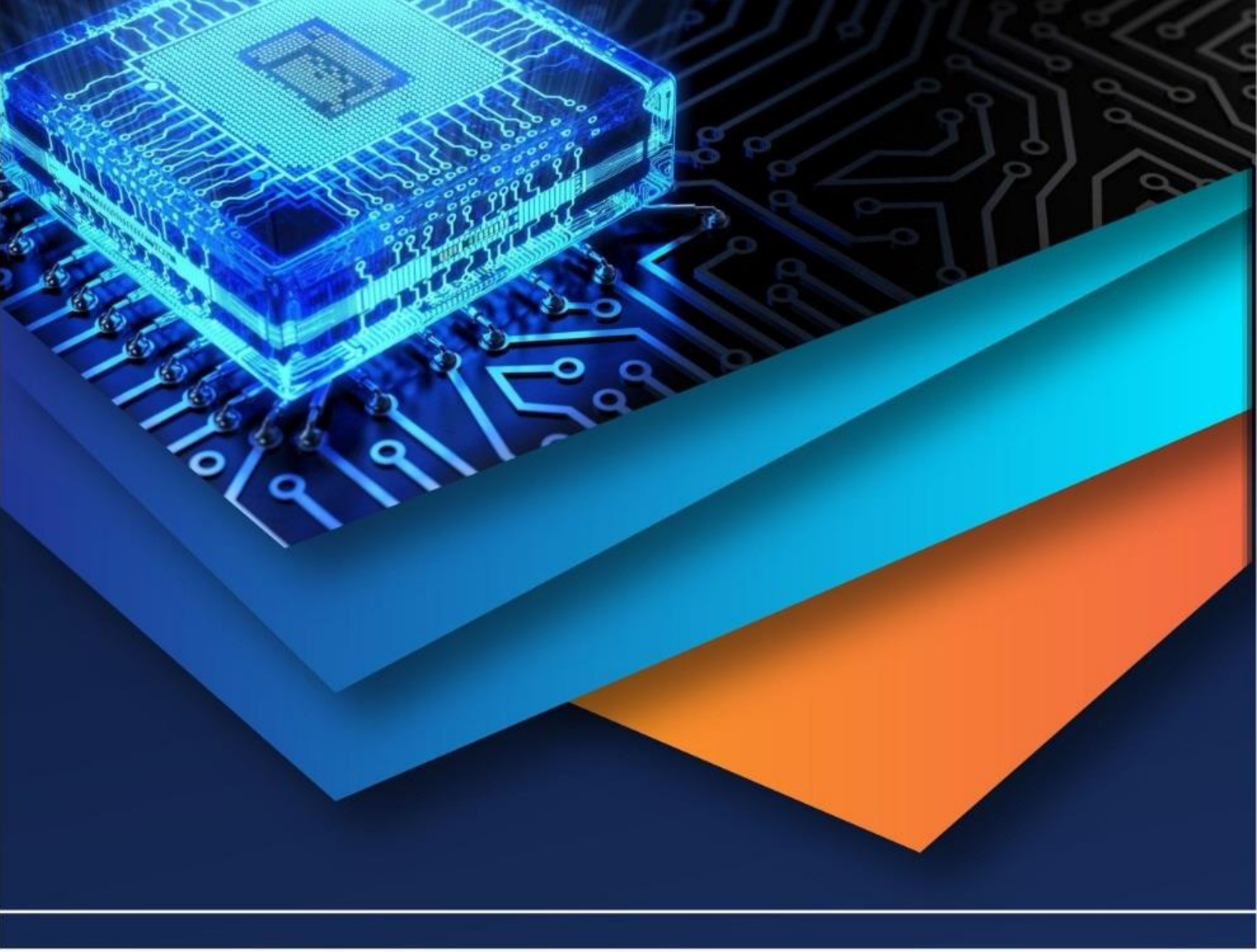
III. ACKNOWLEDGMENT

This research was made possible through the invaluable support and collaboration of academic and industry partners, whose insights into sentiment analysis and natural language processing (NLP) guided our methodology and approach. We would like to extend our gratitude to the developers and maintainers of the benchmark datasets utilized in this study, including the IMDb Movie Review Dataset, Stanford Sentiment Treebank (SST), and Twitter Sentiment Dataset. These datasets provided the foundation for rigorous testing and evaluation, allowing us to assess the generalizability of Convolutional Neural Networks (CNN) in varied sentiment analysis contexts. We are also grateful to our colleagues in the machine learning and data science community for their constructive feedback, which enriched the conceptualization and design of our CNN-based sentiment analysis model. Their expertise and input were instrumental in advancing this research. Finally, we wish to acknowledge the advancements in deep learning frameworks and libraries, such as TensorFlow and PyTorch, that enabled the computational aspects of this research. The accessibility of these tools has accelerated our ability to experiment and iterate, pushing forward the boundaries of sentiment analysis in NLP.

REFERENCES

[1] J. M.· Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1746–1751. Association for Computational Linguistics.
[2] This foundational paper demonstrates the effectiveness of CNNs for sentence classification, providing a benchmark for CNN-based sentiment analysis.
[3] Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., & Potts, C. (2011). Learning Word Vectors for Sentiment Analysis. Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL), 142–150.
[4] A seminal work introducing the IMDb dataset, focused on exploring how word vectors improve sentiment classification.
[5] Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1631–1642.
[6] This paper presents the Stanford Sentiment Treebank (SST), widely used for sentiment classification with fine-grained sentiment labels.
[7] Kalchbrenner, N., Grefenstette, E., & Blunsom, P. (2014). A Convolutional Neural Network for Modelling Sentences. Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Vol. 1), 655–665.
[8] Explores CNN architectures optimized for sentence modeling, extending CNN applications in sentiment analysis.
[9] Severyn, A., & Moschitti, A. (2015). Twitter Sentiment Analysis with Deep Convolutional Neural Networks. Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, 959–962.
[10] Focuses on CNN-based sentiment analysis for social media, tackling unique challenges such as slang and abbreviations in Twitter posts.
[11] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL), 4171–4186.
[12] Introduces BERT, a transformer model that achieved state-of-the-art results in sentiment analysis, demonstrating its robustness across various datasets.
[13] dos Santos, C. N., & Gatti, M. (2014). Deep Convolutional Neural Networks for Sentiment Analysis of Short Texts. Proceedings of the 25th International Conference on Computational Linguistics (COLING), 69–78.
[14] Analyzes CNN performance on short-text sentiment analysis, including Twitter data, highlighting CNN's strength in handling concise, informal text.
[15] Zhang, X., & LeCun, Y. (2015). Text Understanding from Scratch. arXiv preprint arXiv:1502.01710.
[16] Discusses the development of deep CNN architectures for text understanding, illustrating CNN's potential in a variety of NLP tasks, including sentiment analysis.
[17] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep Learning. Nature, 521(7553), 436–444.
[18] This comprehensive review covers the fundamentals of deep learning and its application to NLP, providing context for CNN's relevance in sentiment analysis.
[19] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
[20] A foundational textbook that explores deep learning techniques, including CNNs, providing the theoretical background for their application in NLP and sentiment analysis.

©IJRASET: All Rights are Reserved | SJ Impact Factor 7.538 | ISRA Journal Impact Factor 7.894 |



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)