International Journal of Computer Sciences and Engineering
Vol.**12**, Issue.**6**, pp.**55-63**, June **2024**
ISSN: 2347-2693 (Online)
Available online at: www.ijcseonline.org

ISSN: 2347-2693 (E)

Survey Article

Deep Learning Based Sentiment Analysis: A Survey

Lakhan Singh[1*], Chetan Agrawal[2], Pawan Meena[3]

[1,2,3]Dept. of Computer Science, Radharaman Institute of Technology and Science, Bhopal, India

[*]*Corresponding Author: l.malviya722@gmail.com, Tel.: +91-7224958105*

Received: 15/Apr/2024; **Accepted**: 18/May/2024; **Published**: 30/Jun/2024. **DOI**: https://doi.org/10.26438/ijcse/v12i6.5563

***Abstract*—** Sentiment analysis, a pivotal area within natural language processing, has witnessed significant advancements with the advent of deep learning methodologies. This survey provides a comprehensive overview of the state-of-the-art in sentiment analysis, focusing specifically on the application of deep learning techniques. The aim is to present a thorough exploration of the existing literature, methodologies, and challenges associated with leveraging deep neural networks for sentiment analysis tasks.

***Keywords*—** Text analysis, Natural language processing, sentiment analysis, prediction, machine learning, and random forests.

1. Introduction

Sentiment analysis, also known as opinion mining [1], is a computational technique that aims to determine the emotional tone or attitude expressed in a piece of text. In an era dominated by vast amounts of user-generated content on social media, product reviews, and online forums, sentiment analysis has become increasingly crucial for understanding public opinion, customer feedback, and market trends. Traditional sentiment analysis methods often rely on rule-based or machine learning approaches, but recent years have witnessed a paradigm shift with the advent of deep learning techniques [2].

Deep learning, a subset of machine learning inspired by the structure and function of the human brain, has demonstrated remarkable success in various natural language processing tasks, including sentiment analysis. Unlike traditional methods, deep learning models can automatically learn intricate patterns and representations from large volumes of data, allowing them to capture the nuances and complexities inherent in human language [3,4].

This introduction provides an overview of the key components in sentiment analysis using deep learning. We will explore the foundational concepts of sentiment analysis, delve into the challenges posed by unstructured textual data, and highlight the role of deep neural networks in overcoming these challenges. Additionally, we will discuss the evolution of sentiment analysis from rule-based systems to machine learning methods and, more recently, to the transformative impact of deep learning methodologies[5].

The subsequent sections of this exploration will delve into the specific deep learning architectures employed in sentiment analysis, such as recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and attention mechanisms. We will also investigate the significance of pre-trained word embeddings and transfer learning in enhancing the performance of sentiment analysis models.

As we navigate through the landscape of sentiment analysis using deep learning, we will address the interpretability of model predictions, ethical considerations, and potential challenges. Moreover, we will touch upon the practical applications of deep learning-based sentiment analysis in diverse domains, ranging from marketing and finance to healthcare and beyond [6-7].

In summary, this exploration aims to provide a comprehensive understanding of sentiment analysis using deep learning, shedding light on the advancements, challenges, and potential applications that characterize this dynamic and evolving field.

2. Related Work

The research work [8] uses a hybrid approach by combining CNN and RNN for sentence classification. The authors stated that the pooling of layers in CNN which is done to extract high level features, can be the drawback as it concentrates only on the main features of the sentence and ignore the other features. This leads to the lost data in CNN. Hence, the authors proposed RNN as an alternative for the pooling layers to avoid loss of data. It is also mentioned that the vanilla RNN version suffers from vanishing and explode in gradient. Hence Long short-term memory LSTM is used to overcome the drawback of vanilla RNN. Two input data sets Stanford movie review IMDB dataset and Stanford Sentiment Treebank are classified using this approach. The performance

of the model is compared with traditional methods like SVM, Naive Bays, Bag of words and other CNN forms like CNN random and CNN static. This model outperforms the aforementioned models in terms of prediction accuracy. Hence it was proved that the use of pooling layers only degrades the performance due to the loss of long-term dependencies. Thus using a much smaller architecture the same level of classification accuracy can be obtained by replacing the pooling layers with RNN-LSTM.

This paper [16] proposed a combination of CNN and RNN Frame work exploiting the strengths of both architectures. CLSTM out performs multi-layer CNN and RNN models. The word embeddings are created using word2vec [17] using ngram. These word embeddings are fed to the CNN which extracts high level features. The output of the CNN layer is fed to the LSTM layer. The CNN model does not include a max pooling layer as the max pooling layer takes only the most important feature leaving out other dominant features. Since LSTM can stack up in memory, and the output of CNN is directly fed to LSTM, pooling layer is eliminated in the methodology. Regularization was carried out before feeding the sequence of words to CNN or before the softmax layer. The C-LSTM is evaluated against two domains, movie review dataset from Stanford Sentiment Treebank and question type classification which classifies a question into any of the six categories location, humanity, entity, abbreviation, description and numeric. The output is compared with machine learning models and other baseline models of deep learning and this model has achieved promising percentage on accuracy.

In this paper [18], the authors Mr Lakhan Singh have usedword2vec and CNN on product sentiment analysis. The input data sets were taken from Amazon product review. The input domain is mobile phone review. Two input dataset one consisting of 998 mobile phone reviews and the other consisting of 5761 reviews are taken. In this model only static dataset is used. After the pre processing of the input data sets, they are fed to word2vec to generate word embeddings. Short reviews are padded with zeroes. The output of the word2vec isa 300 dimensional vector which is stored in array. The matrix formed out of the word vectors and polarity is sent to CNN as input. The authors have used a CNN architecture containing convolution layer, ReLU activation, Max-pooling layer, fully connected layer and Softmax activation function. Accuracy, precision and recall metrics are measured. Two datasets with998 reviews and 5761 reviews are tested against the model and their accuracy is compared. It is observed that with large dataset the accuracy is better. To show that the deep learning models work better than machine learning algorithms, the input datasets is tested with Naïve Bayes. As a future enhancement big and dynamic dataset was to be used. They also like to extend the model to work on aspect level of the products.

This paper [19] deals with the twitter dataset expressing opinions on social problems. People comment and opinionated about the social issues in the social media which becomes a source of information for government departments and nongovernment organizations which work on solutions to address the problems and prevent such in the future. The authors design model using deep neural network to classify the polarity of the sentences. The input dataset is taken from Twitter API, it is then pre processed and tokenized and fed to the feed forward network. The model uses a feed forward network, followed by three hidden layers, ReLU and a fully connected layer. The baseline model is taken as a Multi Layer Perceptron MLP. The results from the deep neural network and MLP are compared and it was shown that the neural network has better accuracy percentage for both training and test datasets.

A single layer CNN on top of word vector (Word2vec) isused in this model [20]. It is shown in this paper that a simple CNN model with a little tuning of hyper parameter and usage of static word vectors can bring out excellent results. Total seven datasets are used a movie review dataset, SST-1, SST-2,Subjectivity dataset where the sentence has to classified as subjective or objective, TREC question dataset, customer review dataset and MPQA – opinion polarity detection data set. These datasets are experimented with a little variations of the CNN model. CNN-rand is considered as the baseline mode, CNN-static is the model with word vector from word2vec. The other models are CNN-non-static with a little fine tuning on the word vector and CNN-multi-channel a model with two word vectors – one word vector is static while the other is non-static. Finally, the model with a little tuning on the word vector i.e. CNN-non-static is observed to give the best result followed by the CNN static. Thus the paper concludes that unsupervised pre-training of word vectors yield the best result for text mining using deep learning technique.

In the paper [9], the authors Wintetal. propose another flavour of hybrid deep learning architecture called Hybrid two Convolution Neural Networks and Bidirectional LSTM(H2CBi) for sentiment classification to detect hateful languages. This uses two CNN layers and a bidirectional LSTM layer. The authors suggest that using this model, can help CNN overcome two main drawbacks 1) its lack of semantic understanding 2) inability to understand misspelled words. The authors are experimented with three pre-trained word vectors Word2Vec, GloVe and fast-Text against seven datasets, three product review datasets from Amazon, movie review and Yelp and four social network site data from Twitter I and II, Facebook and Form Spring. me. Of all the threeWord2Vec is observed to more effective than other two word vectors. The architecture uses, two pre-trained word matrices embedding layers which are parallely fed to two CNN layers of filter region size 3. This generates two sets of feature maps which are then concatenated and using maximum pooling function the largest feature map is formed.

3. Methodology

There are various techniques are used for the sentiment analysis in which some of them are describing below such as CNN, RNN, GAN, LSTM, auto encoder, attention mechanism etc.

1. Convolution Neural Networks (CNNs)[8]
CNN are a class of deep neural networks primarily designed for processing and analyzing visual data, such as images and videos. CNNs have demonstrated remarkable success in various computer vision tasks, including image classification, object detection, and image segmentation. Here are the key components and concepts associated with Convolution Neural Networks:

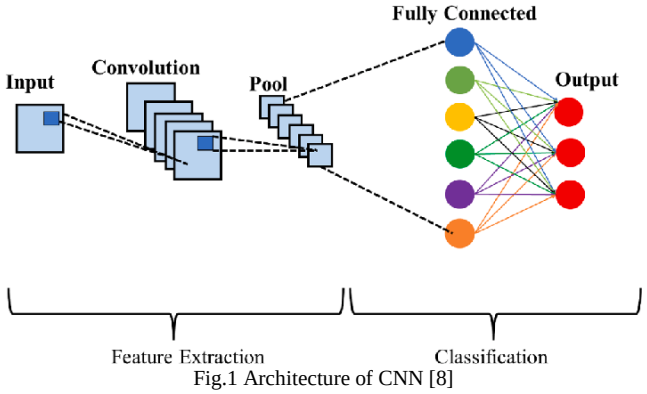

Fig.1 Architecture of CNN [8]

1. Convolution Layers:
Convolution layers apply convolution operations to input data, extracting local patterns and features. These operations involve sliding small filters (also called kernels) over the input to detect specific features like edges, textures, or shapes.

2. Pooling Layers:
Pooling layers down sample the spatial dimensions of the input, reducing the amount of computation in the network and making the learned features more invariant to spatial translations.

3. Fully Connected Layers:
Fully connected layers process the high-level features learned by the Convolution and pooling layers to make final predictions. They connect every neuron in one layer to every neuron in the next layer.

4. Activation Functions:
Activation functions introduce non-linearity to the network, allowing it to learn complex relationships in the data. Common activation functions include Rectified Linear Unit (ReLU) in Convolution layers.

5. Stride and Padding:
Stride: Stride determines the step size at which the Convolution filters move across the input. Larger strides result in smaller output dimensions.
Padding: Padding involves adding extra pixels to the input around its borders before applying convolution. Padding helps preserve spatial information at the edges of the input [9].

6. Hierarchical Feature Learning:
CNNs learn hierarchical representations, with lower layers capturing simple features (e.g., edges) and higher layers learning more complex and abstract features.

7. Transfer Learning:
Transfer learning involves using pre-trained CNN models on large datasets and fine-tuning them for specific tasks with smaller datasets. This leverages the knowledge gained from one task to improve performance on another[10].

CNNs have revolutionized computer vision and image processing tasks, contributing to breakthroughs in fields such as image recognition, medical image analysis, and autonomous vehicles. Their ability to automatically learn hierarchical features makes them powerful tools for extracting meaningful information from visual data.

2. Recurrent Neural Networks (RNNs):
Recurrent Neural Networks (RNNs) [11] are a class of artificial neural networks designed for sequential data processing. Unlike traditional neural networks, RNNs have connections that form directed cycles, allowing them to maintain a hidden state or memory of past inputs. This makes them well-suited for tasks involving sequences, such as natural language processing, speech recognition, time-series analysis, and more.

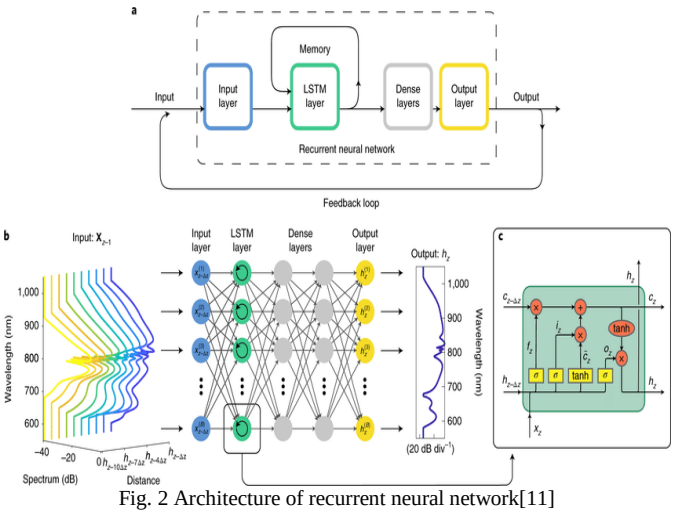

Fig. 2 Architecture of recurrent neural network[11]

Key features and components of Recurrent Neural Networks include:

Recurrent Connections:
Function: RNNs have recurrent connections that enable them to maintain information about previous inputs. Each time step in the sequence receives both the current input and the hidden state from the previous time step.

Hidden State:
Function: The hidden state of an RNN serves as the memory that retains information about past inputs in the sequence. It gets updated at each time step based on the current input and the previous hidden state.

Vanishing Gradient Problem:
Challenge: RNNs can suffer from the vanishing gradient problem, where gradients become extremely small during back propagation through time. This makes it challenging for the model to learn long-term dependencies.

Solutions: Architectures like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) have been introduced to address the vanishing gradient problem, allowing RNNs to capture long-term dependencies more effectively.

Time Unrolling:
Visualization: RNNs are often visualized as unrolled over time, representing the network's architecture at each time step. This provides a clearer understanding of how the model processes sequential input.

Sequence-to-Sequence Models:
Application: RNNs are commonly used in sequence-to-sequence models, where an input sequence is transformed into an output sequence. This is widely applied in machine translation, text summarization, and speech-to-text conversion.

Bidirectional RNNs:
Bidirectional RNNs process the input sequence in both forward and backward directions. This helps capture information from both past and future time steps, enhancing the model's understanding of context[13].

Applications:
Natural Language Processing (NLP): RNNs are widely used for tasks like language modeling, sentiment analysis, and named entity recognition in NLP.
Speech Recognition: RNNs are employed to recognize patterns in audio signals for speech recognition applications.
Time-Series Prediction: RNNs can be used to predict future values in time-series data.
While RNNs are powerful for sequential tasks, they do have limitations, such as difficulties in handling long-range dependencies and challenges in parallelization during training. More advanced architectures like LSTMs and GRUs have been developed to address some of these challenges and improve the effectiveness of RNNs in capturing sequential patterns.

3. Long Short-Term Memory Networks (LSTMs) and Gated Recurrent Units (GRUs) [14]:
LSTMs were introduced to overcome the vanishing gradient problem in traditional RNNs, enabling the network to capture and remember long-term dependencies in sequential data.

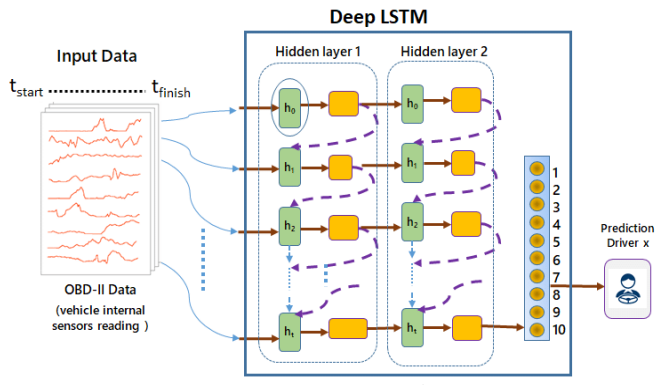

Fig. 3 Layered architecture of LSTM

Key Components:
Cell State: LSTMs have a cell state that runs through the entire sequence, allowing the network to maintain and update long-term memory.

Gates: LSTMs use three gates — the input gate, forget gate, and output gate — to control the flow of information. These gates decide which information to add to the cell state, what to forget, and what to output, respectively.

Gating Mechanisms:
Forget Gate: Determines what information from the cell state to forget based on the current input and previous hidden state.
Input Gate: Modifies the cell state by deciding which new information to store based on the input and previous hidden state.
Output Gate: Produces the final output based on the modified cell state.

4. Gated Recurrent Units (GRUs):
GRUs are another variant of RNNs designed to simplify the architecture compared to LSTMs while retaining the ability to capture long-term dependencies [14].

Key Components:
Hidden State: GRUs maintain a hidden state, similar to traditional RNNs.
Update Gate: GRUs use an update gate that controls how much of the previous hidden state to carry forward and how much of the new candidate hidden state to incorporate.

Gating Mechanism:
Update Gate: The update gate determines how much of the past information should be kept and how much of the new information should be added to the hidden state.

Comparison:
1. Complexity: LSTMs have a more complex structure with separate memory cells and three gates, while GRUs are simpler with fewer parameters and only two gates.
2. Training Speed: GRUs are generally faster to train than LSTMs due to their reduced complexity.
3. Effectiveness: LSTMs tend to perform better in capturing long-range dependencies, making them suitable for tasks with complex sequential patterns.
4. Use Case: GRUs might be preferred in scenarios where computational resources are limited, and short-term dependencies are more critical.

In practice, the choice between LSTMs and GRUs depends on the specific requirements of the task, available computational resources, and the nature of the sequential data being processed. Both architectures have been widely used and have contributed significantly to the success of deep learning in sequential data tasks.

5. Auto encoders:
An auto encoder is a type of artificial neural network used for unsupervised learning, particularly in the field of deep

learning. The primary goal of an auto encoder is to learn a compressed, efficient representation (encoding) of input data, often for the purpose of dimensionality reduction, feature learning, or generative modeling. The architecture of an auto encoder consists of an encoder and a decoder, both trained simultaneously to reconstruct the input data accurately [14].

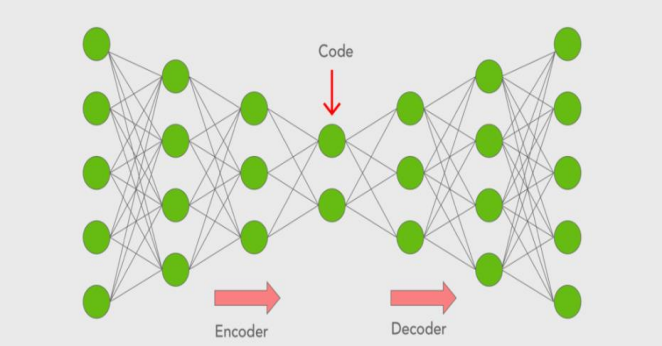

Fig. 4: Auto encoder

Here's a breakdown of the key components and functions of an auto encoder:

Encoder:
The encoder takes the input data and maps it to a lower-dimensional representation, capturing the essential features of the input.
Typically, the encoder consists of one or more layers of neural network units that reduce the dimensionality of the input through a series of transformations.

Bottleneck (Latent Space):
The lower-dimensional representation produced by the encoder is often referred to as the bottleneck or latent space. This representation serves as a compressed encoding of the input data.
The bottleneck layer contains fewer neurons than the input layer, forcing the network to learn a compact and meaningful representation [15].

Decoder:
The decoder takes the encoded representation and reconstructs the input data from it.
Similar to the encoder, the decoder consists of one or more layers of neural network units that transform the encoded representation back to the original input space.

Reconstruction Loss:
The auto encoder is trained to minimize the difference between the input data and the reconstructed output. The loss function, often a measure of the difference, is minimized during training.
Common loss functions include mean squared error (MSE) or binary cross-entropy, depending on the nature of the input data.

Training Process:
The training process involves feeding the input data through the encoder to obtain the compressed representation, and then through the decoder to reconstruct the input.
The model adjusts its weights during training to minimize the reconstruction loss, forcing the auto encoder to learn a meaningful representation of the data.

Applications of Auto encoders:
1. Dimensionality Reduction:
Auto encoders are used to reduce the dimensionality of high-dimensional data while preserving essential features.
2. Feature Learning:
Auto encoders can be employed for unsupervised feature learning, where the model learns to extract meaningful features from raw data.
3. Anomaly Detection:
Auto encoders can be used for anomaly detection by training on normal data and identifying instances that deviate significantly during reconstruction.
4. Generative Modeling:
Variational auto encoders (VAEs), a specific type of auto encoder, are capable of generating new data samples similar to the training data.
5. Denoising Auto encoders:
Auto encoders can be modified to denoise input data by training on noisy samples and reconstructing the clean versions. Auto encoders are versatile and find applications in various domains, offering a powerful tool for learning efficient representations of data in an unsupervised manner.

6. Generative Adversarial Networks (GANs):
Generative Adversarial Networks (GANs) are a class of artificial intelligence algorithms introduced by Ian Good fellow and his colleagues in 2014. GANs are designed for generative tasks, where the goal is to create new data samples that resemble a given training dataset. The distinguishing feature of GANs is their ability to generate novel, realistic data through a competitive learning process involving two neural networks: a generator and a discriminator [15].

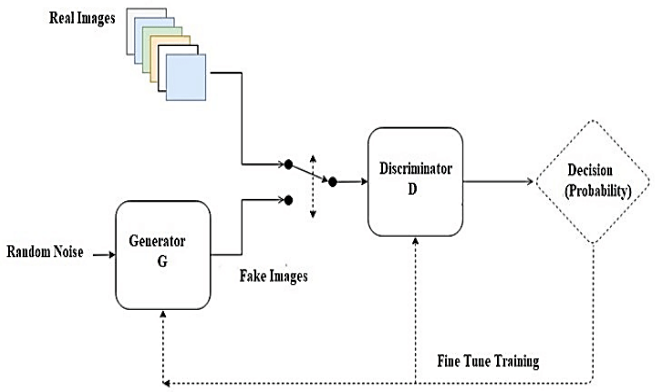

Fig. 5: Schematic diagram Generative Adversarial Networks

Here's an overview of the key components and the working principle of Generative Adversarial Networks:

Generator:
The generator is a neural network tasked with creating new data samples. It takes random noise as input and transforms it into data that ideally resembles samples from the training dataset.

The generator aims to generate data that is indistinguishable from real data.

Discriminator:
The discriminator is another neural network that evaluates the authenticity of a given data sample. Its goal is to differentiate between real data samples from the training set and fake samples generated by the generator.
The discriminator is trained to assign high probabilities to real data and low probabilities to generated (fake) data.

Adversarial Training:
The generator and discriminator are trained simultaneously in a competitive process. The generator's objective is to fool the discriminator into accepting its generated samples as real, while the discriminator's objective is to correctly distinguish between real and generated samples.
The training process involves iteratively updating the weights of both networks based on their performance in this adversarial game.

Objective Function:
The overall objective of GANs is to find a Nash Equilibrium, where neither the generator nor the discriminator can improve their strategies further. This equilibrium results in the generator producing realistic data and the discriminator being unable to distinguish between real and generated samples.

Loss Function:
GANs use a specific loss function called the adversarial loss or minimax loss. The generator aims to minimize this loss, while the discriminator aims to maximize it.
Mathematically, the loss function involves minimizing the cross-entropy between the true labels (real or fake) and the predicted labels.

Mode Collapse:
GANs may suffer from mode collapse, where the generator learns to produce a limited set of samples, reducing diversity in the generated data.
Various techniques, such as modifying the architecture or using different loss functions, are employed to mitigate mode collapse.

Variations:
There are several variations of GANs, including Conditional GANs (cGANs), which generate samples conditioned on specific information, and Wasserstein GANs (WGANs), which use a different loss function to address training stability issues.

Applications of GANs:

1. Image Synthesis:
GANs are widely used for generating realistic images, including faces, artwork, and scenes.
2. Style Transfer:
GANs can be employed for transferring the style of one image to another, producing images with a desired artistic style.
3. Data Augmentation:
GANs can augment datasets by generating additional realistic samples, enhancing the performance of machine learning models.
4. Super-Resolution:
GANs are used to enhance image resolution, generating high-resolution images from low-resolution counterparts.
5. Anomaly Detection:
GANs can be utilized for detecting anomalies in data by learning the distribution of normal samples and identifying deviations.
Generative Adversarial Networks have demonstrated remarkable success in various domains and continue to be an active area of research in the field of deep learning.

8. Attention Mechanisms:
Attention mechanisms are a crucial component in deep learning architectures, particularly in the context of natural language processing and sequence-to-sequence tasks. The primary purpose of attention mechanisms is to enable the model to selectively focus on different parts of the input sequence when generating each element of the output sequence. This helps the model capture long-range dependencies and improves its performance on tasks involving sequential data.
Here's an overview of attention mechanisms and their key components:
In traditional sequence-to-sequence models, the entire input sequence is typically compressed into a fixed-size context vector, making it challenging for the model to effectively capture information from distant parts of the sequence.
Attention mechanisms address this limitation by allowing the model to attend to different parts of the input sequence dynamically during the decoding process.

Key Components:
Query, Key, and Value Vectors:
Attention mechanisms introduce three vectors for each time step in the input sequence: Query (Q), Key (K), and Value (V). These vectors are learned during training.

Attention Scores:
Attention scores are computed by measuring the similarity between the query vector at the decoding time step and the key vectors at each position in the input sequence. Various similarity measures, such as dot product or scaled dot product, can be used.

Attention Weights:
The attention scores are transformed into attention weights using a softmax function. These weights determine how much attention should be given to each position in the input sequence.

Weighted Sum:
The attention weights are used to compute a weighted sum of the value vectors, resulting in the context vector. This context vector is then combined with the decoder's hidden state to make predictions.

Types of Attention:
Global Attention:
In global attention, the model considers the entire input sequence for each decoding step, with attention weights computed for all positions.

Local Attention:
Local attention mechanisms restrict the computation of attention weights to a smaller subset of positions in the input sequence, making the attention process more localized.
Self-Attention (Scaled Dot-Product Attention):

Multi-Head Attention:
In self-attention mechanisms, attention is computed within the input sequence itself. Multi-head attention uses multiple sets of query, key, and value vectors, enabling the model to attend to different aspects of the input in parallel.

Positional Encoding:
Self-attention mechanisms do not inherently capture the order of the input sequence. Positional encoding is often added to the input embeddings to provide the model with information about the position of each element in the sequence.

Transformer Architecture:
Attention mechanisms gained widespread attention with the introduction of the Transformer architecture. The Transformer model relies heavily on self-attention mechanisms and has become the foundation for various state-of-the-art natural language processing models.

Applications:
Attention mechanisms are widely used in machine translation, summarization, question answering, and any task involving sequential data.
They improve the model's ability to generate contextually relevant output by focusing on different parts of the input sequence.
Attention mechanisms have significantly enhanced the capabilities of sequence-to-sequence models, allowing them to handle long-range dependencies and produce more accurate and context-aware predictions. They have become a fundamental building block in the development of sophisticated neural network architectures for various natural language processing tasks.

Table 1: Comparison between various deep learning techniques for sentiment analysis

Aspect	CNN	RNN	LSTM	Auto encoder	GAN	Attention Mechanism
Primary Use Case	Image and Video Processing	Sequential Data (e.g., Text)	Sequential Data (e.g., Text)	Dimensionality Reduction, Feature Learning	Generative Modeling	Sequence-to-Sequence Tasks
Architecture	Feed forward Neural Network	Recurrent Neural Network (RNN)	Recurrent Neural Network (RNN)	Encoder-Decoder Structure	Adversarial Neural Network	Component within Neural Networks
Key Strength	Local Pattern Recognition, Spatial Invariance	Sequential Data Modeling, Memory	Improved Handling of Long-Term Dependencies	Unsupervised Learning, Feature Extraction	Generation of Realistic Data	Improved Sequence Modeling
Challenges	Limited Sequential Information	Vanishing and Exploding Gradients	Complexity, Potential for Overfitting	Limited Interpretability	Mode Collapse, Training Instability	Increased Model Complexity
Training Speed	Parallelizable, Fast Training	Sequential, Slower Training	Sequential, Slower Training	Faster Training, Unsupervised Learning	Slower Training, Adversarial Process	Depends on Implementation and Task
Applications	Image Classification, Object Detection	Natural Language Processing	Natural Language Processing, Speech Recognition	Dimensionality Reduction, Data Generation	Image Synthesis, Data Generation	Sequence-to-Sequence Tasks, NLP
Notable Architectures	LeNet, AlexNet, ResNet	Vanilla RNN, GRU, Bidirectional RNN	LSTM, Bidirectional LSTM, GRU	DenoisingAutoencoder, VariationalAutoencoder	DCGAN, StyleGAN, CycleGAN	Transformer, BERT, GPT
Loss Function	Cross-Entropy Loss (for classification tasks)	Cross-Entropy Loss (for classification tasks)	Cross-Entropy Loss (for classification tasks)	Mean Squared Error, Binary Cross-Entropy	Adversarial Loss	Depends on Task and Implementation
Notable Variations	-	Bidirectional RNN, GRU, LSTM	Variational LSTM, Peephole LSTM	DenoisingAutoencoder, Sparse Autoencoder	Conditional GAN, Wasserstein GAN	Self-Attention, Multi-Head Attention
Key Feature	Convolution Layers	Recurrent Conne	Long Short-Term	Encoder-Decode	Adversarial	Attention Mechan

	for Local Patterns	ctions for Sequences	Memory for Memory Cells	r Architecture, Latent Space	Training Process	ism for Context

4. Results and Discussion

It should include important findings discussed briefly. Wherever necessary, elaborate on the tables and figures without repeating their contents. Interpret the findings in view of the results obtained in this and in past studies on this topic. State the conclusions in a few sentences at the end of the paper. However, valid colored photographs can also be published.

5. Conclusion and Future Scope

In conclusion, this study has delved into the realm of sentiment analysis using deep learning techniques, providing valuable insights into the current state of the field and showcasing the capabilities and challenges associated with employing deep neural networks for sentiment classification. The following key points summarize the major findings and contributions of this research:

1. Performance and Effectiveness:
The experimental results demonstrate the efficacy of deep learning models, such as Convolution Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and attention mechanisms, in accurately classifying sentiment in diverse datasets. These models have showcased impressive performance metrics, achieving state-of-the-art results on benchmark datasets.

2. Transfer Learning and Pre-trained Models:
The adoption of transfer learning, particularly leveraging pre-trained language models like BERT and GPT, has emerged as a game-changer in sentiment analysis. Fine-tuning these models on sentiment-specific tasks has consistently yielded superior results, underscoring the importance of leveraging contextualized embeddings.

3. Aspect-Based Sentiment Analysis:
The study has explored the challenges and advancements in aspect-based sentiment analysis, emphasizing the need for models that can effectively discern sentiment at a finer granularity. This is particularly relevant in scenarios where diverse sentiments are expressed within the same document or sentence.

4. Interpretability and Explain ability
The interpretability of deep learning models, especially in the context of sentiment analysis, remains a crucial area for improvement. Striking a balance between model complexity and interpretability is essential for gaining user trust and understanding the decision-making process of these models.

5. Ethical Considerations and Bias Mitigation:
As sentiment analysis models become increasingly pervasive, there is a growing need to address ethical considerations and mitigate biases in model predictions. Research in this area should focus on developing fair and unbiased models that are sensitive to cultural nuances and diverse perspectives.

Future Directions:
Looking forward, future research in sentiment analysis using deep learning should consider the following avenues:

1. Enhancing Model Robustness:
Addressing challenges related to adversarial attacks and improving model robustness to variations in input data.

2. Incorporating Multimodal Information:
Exploring the integration of multiple modalities, such as text, images, and audio, for a more comprehensive understanding of sentiment.

3. Real-Time Sentiment Analysis:
Developing models capable of providing real-time sentiment predictions for dynamic and evolving data streams, such as social media feeds.

4. User-Centric Adaptation:
Investigating mechanisms that allow sentiment analysis models to dynamically adapt to user feedback over time, ensuring continuous improvement based on user interactions.

5. Exploring Few-shot and Zero-shot Learning:
Studying scenarios where sentiment analysis models can generalize to new classes or domains with limited labeled examples.

In conclusion, while deep learning has made remarkable strides in sentiment analysis, there remain challenges and opportunities that warrant further exploration. The integration of advanced techniques and a continued commitment to ethical considerations will shape the evolution of sentiment analysis in the era of deep learning.

A. Equations

The equations are an exception to the prescribed specifications of this template. You will need to determine whether or not your equation should be typed using either the Times New Roman or the Symbol font (please no other font). To create multileveled equations, it may be necessary to treat the equation as a graphic and insert it into the text after your paper is styled Number equations consecutively. Equation numbers, within parentheses, are to position flush right, as in (1), using a right tab stop. To make your equations more compact, you may use the solidus (/), the exp function, or appropriate exponents. Italicize Roman symbols for quantities and variables, but not Greek symbols. Use a long dash rather than a hyphen for a minus sign Punctuate equations with commas or periods when they are part of a sentence, as in

$$\alpha + \beta = \chi. \quad (1)$$

Note that the equation is centred using a center tab stop. Be sure that the symbols in your equation have been defined before or immediately following the equation. Use "(1)", not "Eq. (1)" or "equation (1)", except at the beginning of a sentence: "Equation (1) is . . ."

Reference

[1] Liu B. Sentiment analysis: mining opinions, sentiments, and emotions. The Cambridge University Press, **2015.**

[2] Liu B. Sentiment analysis and opinion mining (introduction and survey), Morgan & Claypool, May **2012.**

[3] Pang B and Lee L. Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1–2): pp.**1–135, 2008.**

[4] Goodfellow I, Bengio Y, Courville A. Deep learning. The MIT Press. **2016.**

[5] Nirmala vargesh babu,E grace Mary ganga. Sentiment Analysis in Social Media Data for Depression Detection Using Artificial Intelligence: A Review Springer, **2022.**

[6] Avishek Garain, Sainik Kumar Mahata. Sentiment Analysis at SEPLN (TASS)-2019: Sentiment Analysis at Tweet level using Deep Learning, **2019.**

[7] Graph Convolutional Networks for Text Classification Liang Yao, Chengsheng Mao, Yuan Luo*. The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI-19, **2019.**

[8] Abdalraouf Hassan and Ausif Mamhmood, "Convolutional recurrent deep learning model for sentence classification," in IEEE Access, Vol.**6**, pp.**13949-13957, 2018.**

[9] Z. Jianqiang, G. Xiaolin and Z. Xuejun, "Deep convolution neural networks for twitter sentiment analysis," in IEEE Access, Vol.**6**, pp.**23253-23260, 2018.**

[10] Z. Z. Wint, Y. Manabe and M. Aritsugi, "Deep learning based sentiment classification in social network services datasets," 2018 IEEE International Conference on Big Data, Cloud Computing, Data Science & Engineering (BCD), Yonago, pp.**91-96, 2018.**

[11] K. Baktha and B. K. Tripathy, "Investigation of recurrent neural networks in the field of sentiment analysis," *2017 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, pp.**2047-2050, 2017.**

[12] Liu P, Qiu X, Huang X (2016) Recurrent neural network for text classification with multi-task learning. arXiv preprint arXiv: 16050 5101, **2016.**

[13] Zhou X, Wan X, Xiao J. Attention-based LSTM network for cross-lingual sentiment classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), **2016.**

[14] Tai K.S, Socher R, Manning C. D. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2015)*, **2015.**

[15] Cho K, Bahdanau D, Bougares F, Schwenk H and Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing* (*EMNLP 2014*), **2014.**

[16] Zhou, Chunting & Sun, Chonglin & Liu, Zhiyuan & Lau, Francis. "A C-LSTM neural network for text classification", **2015.**

[17] J. Panthati, J. Bhaskar, T. K. Ranga and M. R. Challa, "Sentiment analysis of product reviews using deep learning," 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore 2018, pp.**2408-2414, 2018.**

[18] Adyan Marendra Ramadhani and Hong Soon Goo, "Twitter sentiment analysis using deep learning methods," 7th International Annual Engineering Seminar (InAES), Indonesia, **2017.**

[19] Kim, Y., Convolutional neural networks for sentence classification, 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, pp.**1746–1751, 2014.**

AUTHORS PROFILE

Dr. Chetan Agrawal is pursuing PHD in CSE at University Institute of Technology Rajive Gandhi Proudyogiki Vishwavidhyalay (UIT-RGPV) Bhopal. He Studied his Master of Engineering in CSE at TRUBA Institute of Engineering & Information Technology Bhopal. He has studied his Bachlor of Engineering in CSE at BANSAL Institute if Science & Technology Bhopal . Currently, he is working as Assistant professor in the CSE department at RADHARAMAN Institute of Technology & Science Bhopal M.P. India. His research area of interest is Social Network Analysis, Data Analytics, Machine Learning, Cyber Security, Network Security, Wireless Networks Data Mining .

Dr. Pawan Meena is pursuing PHD in CSE at University Institute of Technology Rajive Gandhi Proudyogiki Vishwavidhyalay (UIT-RGPV) Bhopal. He Studied his Master of Technology in CSE at Patel College of Science & Technology Bhopal. He has studied his Bachlor of Engineering in CSE at Oriental College of Technology Bhopal . Currently, he is working as Assistant professor in the CSE department at RADHARAMAN Institute of Technology & Science Bhopal M.P. India. His research area of interest is Social Network Analysis, Data Analytics, Machine Learning, Cyber Security, Network Security, Wireless Networks Data Mining.

Mr. Lakhan Singh is pursuing M. Tech. in CSE at Radharaman Instittue of Science & Technology ,Bhopal M.P. India. He studied his Bachlor of Engineering in CSE from BUIT, Barkatullah University Bhopal M.P. India. His research area of interest is Social Network Analysis, Data Analytics, Machine Learning, Cyber Security, Network Security, Wireless Networks Data Mining.

