# Leveraging CNN-RNN Hybrid Architectures for Sentiment Analysis Task

**Sachin Sharma**
University of Birmingham
and IIT Madras

## Abstract

This study explores deep learning models for sentiment analysis on the Sentiment140 dataset, containing 1.6 million labeled tweets. We examine CNN-only, RNN-only, CNN-RNN hybrid, and attention-based hybrid models, leveraging pre-trained GloVe 6B and GloVe 27B embeddings. Our CNN-RNN hybrid model performs comparably to DistilBERT, offering similar accuracy with faster training. Surprisingly, the simpler RNN model outperforms the hybrid architectures, highlighting the importance of sequential dependencies in noisy, short Twitter texts. This research emphasizes optimizing deep learning models for social media sentiment analysis.

## 1 Introduction

Social media platforms, particularly Twitter, have become vital spaces for expressing public sentiment on diverse issues. Understanding these sentiments aids in fields such as marketing, public health, and policymaking. Sentiment analysis, a key NLP task, classifies textual data as positive, negative, or neutral. However, the informal and often noisy nature of Twitter data presents unique challenges for traditional sentiment analysis models.

Recent advancements in NLP include pre-trained word embeddings, like GloVe, which improve models' ability to capture semantic relationships. GloVe 27B, trained on large-scale Twitter data, outperforms smaller embeddings, such as GloVe 6B, in capturing nuanced sentiment (Pennington et al., 2014; Tang et al., 2014). Hybrid models combining Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have also gained attention for their complementary strengths: CNNs capture local patterns, while RNNs handle sequential dependencies. The introduction of attention mechanisms further improves model interpretability and performance (Bahdanau et al., 2015; Vaswani et al., 2017).

Transformer models like DistilBERT have set new performance benchmarks in NLP but come with higher computational costs, limiting their applicability in resource-constrained environments (Sun et al., 2019). This study explores the effectiveness of CNN-RNN hybrid models using GloVe embeddings, comparing them to DistilBERT. Additionally, an error analysis of misclassified tweets provides insights into model limitations and challenges in sentiment analysis.

### 1.1 Contributions

This research offers:

- Evaluation of CNN-RNN hybrid architectures with GloVe embeddings, including attention mechanisms.
- Insights into the effects of embedding size, model complexity, and attention on performance.
- A comparative analysis with DistilBERT to assess hybrid models versus transformer-based models.
- An error analysis to identify misclassification causes and suggest future improvements.

These contributions advance sentiment analysis methodologies and provide insights for real-world applications.

## 2 Literature Review

### 2.1 Evolution of Sentiment Analysis

Sentiment analysis, a key NLP task, involves extracting opinions and emotions from text. Early methods used traditional machine learning models like Support Vector Machines (SVMs) and Naive Bayes classifiers (Pang et al., 2002), which struggled with the informal language and semantic complexity of social media. Recent advancements

have leveraged deep learning models like Convolutional Neural Networks (CNNs) for local word patterns (Kim, 2014), and Recurrent Neural Networks (RNNs), including LSTMs, for sequential dependencies (Hochreiter and Schmidhuber, 1997). Hybrid CNN-RNN architectures and attention mechanisms have further improved performance by capturing both local and long-range dependencies.

### 2.2 Pre-trained Word Embeddings

Pre-trained embeddings such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) have revolutionized text representation. These embeddings capture semantic relationships using co-occurrence statistics from large corpora, improving generalization. GloVe embeddings trained on Twitter data, like GloVe 27B, are particularly effective for social media sentiment analysis due to their ability to model informal language and context (Tang et al., 2014).

### 2.3 Hybrid CNN-RNN Architectures

Hybrid CNN-RNN models combine the strengths of CNNs for local pattern recognition and RNNs for sequential dependencies (Zhou et al., 2016). Attention mechanisms further enhance these models by focusing on sentiment-relevant parts of the text (Bahdanau et al., 2015). These hybrid models have been highly effective in handling noisy social media data, offering both high accuracy and better interpretability.

### 2.4 Transformer Models and Comparisons

Transformer-based models like BERT (Devlin et al., 2019) and DistilBERT (Sanh et al., 2019) have set new performance benchmarks in NLP, using self-attention to capture long-range dependencies. Fine-tuned transformer models achieve excellent results, with accuracy up to 89.7% on the Sentiment140 dataset (Tan et al., 2022). However, these models are computationally expensive (Sun et al., 2019), making them less feasible for real-time or low-resource applications. Hybrid CNN-RNN models, while less accurate, offer a more efficient alternative.

### 2.5 Challenges and Research Gaps

Key challenges in sentiment analysis include handling slang, abbreviations, and emojis, which often lead to misclassifications. While transformer models provide high accuracy, their resource demands make them impractical for real-time or low-resource scenarios. This study addresses these challenges by evaluating hybrid CNN-RNN architectures and comparing them with DistilBERT, offering insights into the trade-offs between accuracy, computational efficiency, and interpretability.

## 3 Methodology

### 3.1 Dataset

The Sentiment140 dataset, widely used in sentiment analysis tasks, contains over 1.6 million labeled tweets with binary sentiment values: 0 for negative and 1 for positive. In this study, the positive label was remapped from 4 to 1 for consistency. The dataset is balanced, with roughly equal numbers of positive and negative sentiments, offering a fair evaluation of the models.

#### 3.1.1 Structure of the Dataset

The dataset consists of six columns:

- **sentiment:** The sentiment label (0 for negative, 1 for positive).
- **id:** A unique tweet identifier.
- **date:** The tweet's posting time.
- **user:** The Twitter user ID.
- **flag:** A placeholder field labeled "NO QUERY".
- **text:** The tweet content, which may include hashtags, mentions, URLs, emojis, and abbreviations.

#### 3.1.2 Relevance of the Dataset

The Sentiment140 dataset is highly relevant for sentiment analysis due to its real-world challenges, including informal language, brevity, and diverse expressions. These characteristics make it ideal for evaluating advanced techniques like hybrid CNN-RNN architectures and attention mechanisms in sentiment classification tasks.

### 3.2 Dataset Overview and Exploratory Data Analysis (EDA)

#### 3.2.1 Dataset Composition and Format

The dataset is stored in CSV format with key columns:

- **Text:** The tweet content, which may include various social media-specific elements.

- **Sentiment Label:** A binary sentiment classification (0 for negative, 1 for positive).

Neutral tweets (labeled as 2) were excluded or relabeled as positive to simplify the binary classification task.

#### 3.2.2 Dataset Characteristics

The Sentiment140 dataset reflects the informal and noisy nature of Twitter content, which presents several challenges:

- **Tweet Composition:** Tweets vary widely, containing slang, emoticons, hashtags, and mentions. Preprocessing is required to standardize the data, such as converting emojis to text and handling hashtags or mentions.
- **Diversity of Topics:** The dataset covers diverse topics, from politics to personal expressions, introducing challenges in sentiment expression across different domains.
- **Preprocessing Needs:** Due to noisy content, preprocessing steps like tokenization, removal of irrelevant symbols, and normalization of abbreviations are essential for effective model training.

#### 3.2.3 Initial Insights from Exploratory Data Analysis (EDA)

An in-depth Exploratory Data Analysis (EDA) was conducted to gain insights into the dataset, informing subsequent modeling decisions. The following key findings were made:

#### 3.2.4 Word Count Distribution

The histogram of word counts in the tweets (Figure 1) shows a peak around 10-15 words, with some longer tweets extending beyond 15 words. This aligns with Twitter's 280-character limit, though longer tweets sometimes offer more explicit sentiment-bearing words, which could enhance sentiment classification.

#### 3.2.5 Sentiment Label Distribution

The Sentiment140 dataset has a balanced distribution of positive and negative sentiments (Figure 2), which is beneficial for model training by avoiding class imbalance issues and ensuring more accurate generalization.

These EDA findings guided the feature engineering and model selection, ensuring models aligned with the dataset's structure. The dataset's noisy nature emphasized the need for comprehensive preprocessing.

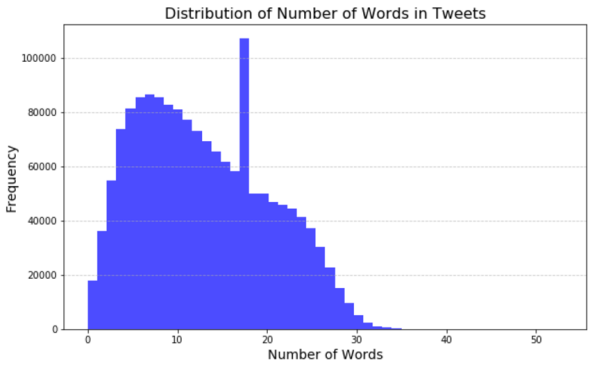


Figure 1: Word Count Distribution in Tweets

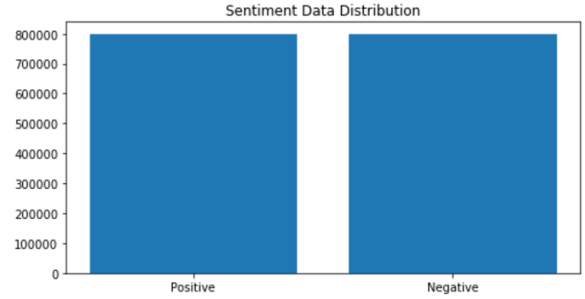


Figure 2: Sentiment Label Distribution in the Sentiment140 Dataset

In conclusion, the EDA reinforced that while the Sentiment140 dataset is appropriate for sentiment analysis, effective preprocessing is crucial to address its informal and noisy characteristics. The insights gained shaped the feature extraction and model development processes.

The following flowchart (Figure 3) outlines the key steps in the research methodology, from dataset selection to evaluation.

### 3.3 Data Preprocessing and Feature Engineering

Data preprocessing is an essential step in preparing the Sentiment140 dataset for sentiment analysis. The goal is to clean, normalize, and structure the textual data while retaining sentiment-relevant information. The preprocessing pipeline includes cleaning, expanding, tokenizing the text, and handling emoticons, slang, and stop words.

#### 3.3.1 Data Cleaning

Text cleaning was performed using regular expressions (regex) to remove unnecessary elements, which included:

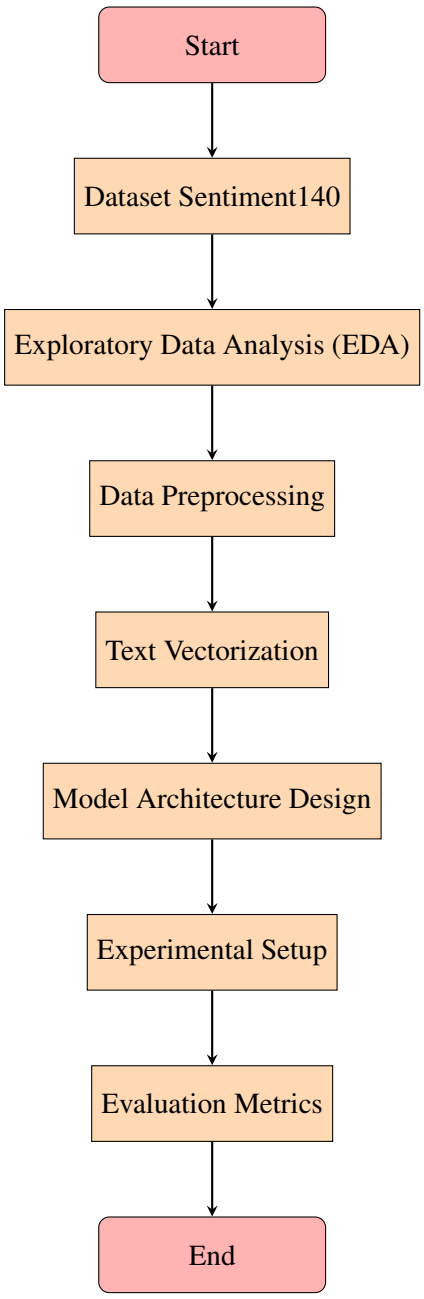


Figure 3: Flowchart illustrating the research methodology.

- **Mentions:** User mentions were removed (e.g., `@username`).
- **Hyperlinks:** URLs were removed (e.g., `http://example.com`).
- **Special Characters:** Non-alphanumeric symbols were eliminated to focus on meaningful content.

Numerical values, which did not contribute meaningfully to sentiment classification, were also removed.

#### 3.3.2 Emoticon Replacement

Emoticons, often used to convey sentiment, were replaced with their corresponding textual representations:

- `:)` was replaced with "smile."
- `:(` was replaced with "sad."
- `:'(` was replaced with "cry."

This step ensured that the sentiment conveyed by emoticons was preserved.

#### 3.3.3 Slang and Abbreviation Expansion

A predefined slang dictionary was used to expand abbreviations and slang terms into their full forms. For example:

- `"thx"` was expanded to "thanks."
- `"luv"` was expanded to "love."
- `"b4"` was expanded to "before."

This preprocessing step prevented informal language from affecting the model's understanding.

#### 3.3.4 Stop Word Handling

Stop words were generally removed to focus on meaningful tokens. However, certain stop words important for sentiment analysis, such as "not," "never," and "but," were retained. This selective removal ensured that the model could capture sentiment context effectively.

#### 3.3.5 Optional Stemming

Stemming was optionally applied using the Snowball Stemmer, which reduced words to their root forms, e.g., "running" to "run." This step helped balance reducing data sparsity while maintaining contextual meaning.

| | sentiment | text |
|---|---|---|
| 0 | Negative | @switchfoot http://twitpic.com/2y1zl - Awww, t... |
| 1 | Negative | is upset that he can't update his Facebook by ... |
| 2 | Negative | @Kenichan I dived many times for the ball. Man... |
| 3 | Negative | my whole body feels itchy and like its on fire |
| 4 | Negative | @nationwideclass no, it's not behaving at all.... |

Figure 4: Before Data Preprocessing

| | sentiment | text |
|---|---|---|
| 0 | Negative | awww that s a bummer you shoulda got david car... |
| 1 | Negative | is upset that he can t update his facebook by ... |
| 2 | Negative | i dived many times for the ball managed to sav... |
| 3 | Negative | my whole body feels itchy and like its on fire |
| 4 | Negative | no it s not behaving at all i m mad why am i h... |

Figure 5: After Data Preprocessing

### 3.4 Tokenization

After preprocessing, the text was tokenized into individual words to prepare for further analysis and conversion into embeddings.

#### 3.4.1 Sequence Length Standardization

Given the variation in tweet lengths, the sequences of word embeddings were standardized to a uniform length of 30 tokens to meet the input requirements of deep learning models.

#### 3.4.2 Padding

Tweets with fewer than 30 tokens were padded with zero vectors at the end of the sequence, ensuring a uniform input length while preserving the natural order of words.

#### 3.4.3 Truncation

Tweets exceeding 30 tokens were truncated, retaining only the first 30 tokens. This choice was supported by the EDA, which indicated that most tweets were short, and essential sentiment information is typically located near the beginning.

#### 3.4.4 Rationale for Length Selection

The sequence length of 30 tokens was selected based on the distribution of tweet lengths in the dataset, which showed that most tweets fell within this range. This minimized the impact of padding or truncation on the dataset's integrity.

### 3.5 Text Vectorization

After preprocessing, the cleaned text was converted into numerical vectors using pre-trained GloVe embeddings (Global Vectors for Word Representation). These embeddings capture the semantic meaning of words by mapping them to high-dimensional spaces, enabling the model to understand relationships between words.

#### 3.5.1 Embedding Sources

The study utilized the following pre-trained GloVe embeddings:

- **GloVe 6B:** Trained on Wikipedia 2014 and Gigaword 5, using 6 billion tokens and a 400,000-word vocabulary, this version provides general-purpose embeddings. GloVe 6B offers embeddings in four dimensions: 50, 100, 200, and 300. Due to its diverse training corpus, it captures a broad range of semantic relationships, making it suitable for various tasks.

- **GloVe 27B:** Trained on 2 billion tweets with 27 billion tokens and a vocabulary of 1.2 million words, this version is tailored to social media text. It provides embeddings in 25, 50, 100, and 200 dimensions and excels in capturing the nuances of slang, abbreviations, and informal language common in Twitter data.

The inclusion of both GloVe 6B and GloVe 27B allows for comparison between general-purpose and Twitter-specific embeddings to evaluate which one yields better results for sentiment analysis on social media text.

#### 3.5.2 Token-to-Vector Mapping

- Each token was mapped to its corresponding embedding vector from the GloVe matrix.

- If a token was absent from the GloVe vocabulary, it was assigned either a random vector or a generic unknown vector.

### 3.6 Contextual Representation

- GloVe embeddings ensure that semantically similar words have close representations in the vector space (e.g., *happy* and *joyful*), while antonyms like *happy* and *sad* are distant.

- These embeddings also capture critical nuances such as negation (*not good* versus *good*) and emphasis (*very bad* versus *bad*), which are essential for sentiment analysis.

#### 3.6.1 Sequence Length Standardization

- Sequences of word embeddings were padded or truncated to a uniform length to meet the input requirements of deep learning models.
- Zero vectors were used for padding to avoid introducing noise into the model.

### 3.7 Model Architecture

This section outlines the architecture of the model employed in this study, detailing its components and the theoretical foundations behind the design. The model consists of an embedding layer, a convolutional neural network (CNN), a bidirectional long short-term memory (BiLSTM) layer, an attention mechanism, and dense layers for sentiment classification. Each component plays a crucial role in enabling the model to capture relevant patterns and dependencies in the text data for accurate sentiment analysis.

#### 3.7.1 Embedding Layer

The input sequence, consisting of tokenized text, is first passed through an embedding layer, which converts each token into a dense vector representation. This layer uses pre-trained GloVe embeddings, which capture semantic relationships between words learned from large corpora. The embedding layer provides a fixed-size representation for each word, ensuring that words with similar meanings are represented by similar vectors. The embedding layer is represented as:

$$\mathbf{E} = \text{Embedding}(x)$$

Where $\mathbf{E}$ represents the embedding matrix and $x$ represents the input sequence of tokenized words.

#### 3.7.2 Convolutional Neural Network (CNN)

To capture local patterns such as n-grams in the text, a convolutional layer is employed. A 1D convolution is applied to the sequence of word embeddings, where each filter in the convolutional layer is convolved with the input sequence to detect local features, such as word combinations or phrases that contribute to the sentiment of the text. The convolution operation is expressed as:

$$y_i = W * X_i + b$$

Where:

- $W$ is the convolutional filter,
- $X_i$ represents a sliding window over the input sequence,
- $b$ is the bias term,
- $*$ denotes the convolution operation.

The output is passed through a ReLU activation function to introduce non-linearity:

$$\text{ReLU}(x) = \max(0, x)$$

This helps the model detect important features while capturing the complexity of language in the text.

#### 3.7.3 Bidirectional Long Short-Term Memory (BiLSTM)

After capturing local features using the CNN layer, the model processes the sequence using a Bidirectional LSTM (BiLSTM). The BiLSTM learns long-range dependencies by processing the input sequence in both forward and backward directions, enabling the model to understand context from both past and future words. The hidden state at time $t$ is computed using the recurrence relation:

$$h_t = \sigma(W_h \cdot h_{t-1} + W_x \cdot x_t + b)$$

Where:

- $h_t$ is the hidden state at time step $t$,
- $x_t$ is the input at time step $t$,
- $W_h$ and $W_x$ are the weight matrices,
- $\sigma$ is the activation function (e.g., tanh or sigmoid).

The BiLSTM processes the sequence in both directions, producing a comprehensive representation of the input sequence, which helps capture both forward and backward contextual information.

#### 3.7.4 Attention Mechanism

To enhance the model's ability to focus on the most relevant parts of the input, an attention mechanism is employed. The attention mechanism assigns a weight to each hidden state $h_t$ based on its importance in predicting the sentiment of the sequence. The attention score $\alpha_t$ for each time step is calculated as:

$$\alpha_t = \frac{\exp(e_t)}{\sum_{t=1}^{T} \exp(e_t)}$$

Where $e_t$ is the attention score for time step $t$, and $T$ is the total number of time steps in the sequence. The final context vector is computed as a weighted sum of the hidden states:

$$h_{\text{att}} = \sum_{t=1}^{T} \alpha_t \cdot h_t$$

This context vector $h_{\text{att}}$ is then used to make the final sentiment prediction. The attention mechanism helps the model focus on the most important words, improving sentiment classification performance.

#### 3.7.5 Dense Layers and Output Layer

After the attention mechanism, the output is passed through dense layers to refine the learned representation. A fully connected layer with 512 units and ReLU activation is followed by a dropout layer to prevent overfitting. Finally, a dense output layer with a sigmoid activation function classifies the sentiment as positive or negative:

$$\text{Output} = \text{sigmoid}(W_{\text{out}} \cdot h_{\text{att}} + b)$$

Where $W_{\text{out}}$ is the weight matrix, $h_{\text{att}}$ is the context vector, and $b$ is the bias term.

The model is compiled using the Adam optimizer with a learning rate, binary cross-entropy loss, and accuracy as the evaluation metric:

$$\text{Loss} = \text{binary\_crossentropy}(y_{\text{true}}, y_{\text{pred}})$$

$$\text{Accuracy} = \frac{\text{correct predictions}}{\text{total predictions}}$$

### 3.8 Experimental Setup

We used deep learning models for sentiment analysis of tweets with GloVe 6B and GloVe 27B embeddings, trained on a Kaggle GPU.

#### 3.8.1 Data Preprocessing

Text data was tokenized and cleaned using regular expressions to remove noise (e.g., URLs, special characters).

#### 3.8.2 Stemming

Stemming was omitted, as GloVe embeddings capture semantic relationships, making stemming unnecessary.

#### 3.8.3 Stopword Removal

Stopwords like "not," "no," and "very" were retained, as they influence sentiment analysis.

#### 3.8.4 Model Architecture

We combined CNNs for local patterns and RNNs for sequential dependencies, with attention mechanisms for better focus on critical text parts.

#### 3.8.5 Hyperparameter Tuning

Grid search was used to optimize batch size, learning rate, LSTM units, and dropout rate. Models were trained for 75 epochs with the Adam optimizer and Binary Cross-Entropy loss.

#### 3.8.6 Dataset Split

The dataset was split 80% for training and 20% for testing.

### 3.9 Evaluation Method

We evaluated the models using binary accuracy and F1 score.

#### 3.9.1 Binary Accuracy

Measures the proportion of correctly classified examples:

$$\text{Accuracy} = \frac{\text{Correctly Labeled Examples}}{\text{Total Examples}}$$

#### 3.9.2 F1 Score

The harmonic mean of Precision and Recall, useful for imbalanced classes:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

## 4 Results

This section presents the performance of sentiment analysis models using GloVe embeddings (6B and 27B). Models evaluated include Hybrid, Hybrid with Attention, RNN, and CNN, with the F1 score as the primary metric due to its balance between precision and recall, crucial for imbalanced tasks like sentiment analysis.

| Model | Embedding | F1 Score | Precision | Recall | Accuracy |
|---|---|---|---|---|---|
| Hybrid Model | GloVe 6B | 0.8277 | 0.8207 | 0.8392 | 0.8278 |
| Hybrid Model (2 CNN Layers) | GloVe 27B | 0.8320 | 0.8386 | 0.8228 | 0.8320 |
| Hybrid Model | GloVe 27B | 0.8365 | 0.8384 | 0.8334 | 0.8362 |
| Hybrid Model with Attention | GloVe 27B | 0.8399 | 0.8471 | 0.8301 | 0.8399 |
| RNN Model | GloVe 27B | 0.8412 | 0.8573 | 0.8192 | 0.8412 |
| CNN Model | GloVe 27B | 0.8257 | 0.8344 | 0.8131 | 0.8257 |

Table 1: Performance metrics for various models.

### 4.1 Key Findings

1. **RNN (GloVe 27B):** The RNN achieved the highest F1 score of 0.8412, excelling in capturing long-term dependencies, ideal for tweet data. It uses a BiLSTM architecture for bidirectional dependencies.

2. **Hybrid Models:** The Hybrid Model with GloVe 27B (F1: 0.8365) outperformed GloVe 6B (F1: 0.8277). Adding two CNN layers (F1: 0.8320) showed diminishing returns in feature extraction.

3. **Hybrid with Attention (GloVe 27B):** Attention improved performance (F1: 0.8399, Precision: 0.8471), enabling the model to focus on key tokens.

4. **CNN (GloVe 27B):** The CNN model had the lowest F1 score (0.8257), limited by its inability to model long-range dependencies.

### 4.2 Confusion Matrix

### 4.3 Confusion Matrix Analysis

The confusion matrix for the hybrid model with attention (GloVe 27B), shown in Figure 6, highlights the model's performance in classifying positive and negative sentiments. The model achieved a true negative rate of 85.10%, indicating effective identification of negative sentiments, and a true positive rate of 82.93%, reflecting strong but slightly lower performance in recognizing positive sentiments. However, approximately 14.90% of negative samples were misclassified as positive, likely due to ambiguous or context-dependent language, and 17.07% of positive samples were classified as negative, suggesting the need for better handling of subtle cues in positive sentiments. Overall, the results indicate balanced performance, with slightly better detection of negative sentiments. Future improvements could focus on reducing misclassifications by integrating advanced attention mechanisms or incorporating richer contextual embeddings.

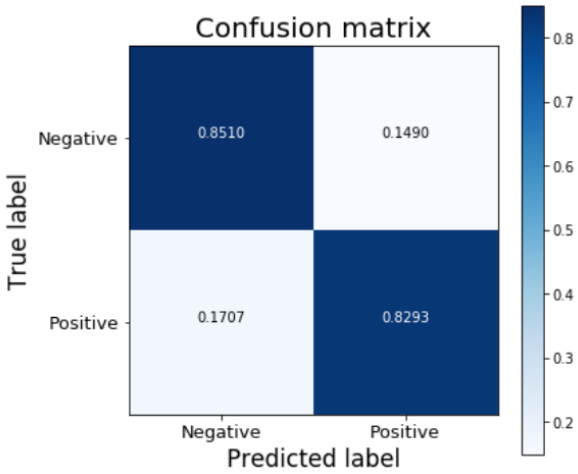


Figure 6: Confusion Matrix for Hybrid Model with attention (GloVe 27B).

### 4.4 Learning Curves

## 5 Accuracy and Loss Analysis

The learning curves for loss and accuracy during training for the hybrid model with attention (GloVe 27B), shown in Figure 7, provide insights into the model's training performance over 75 epochs. The loss curve demonstrates a steady decrease in error, while the accuracy curve reflects consistent improvements in classification performance. The training accuracy reached approximately 84% by the final epoch, with validation accuracy closely matching it, indicating strong generalization to unseen data and no signs of overfitting. Similarly, training and validation loss curves show minimal divergence, with the validation loss stabilizing at around 0.36 by the final epoch, further confirming the model's robustness. These results demonstrate a well-balanced hybrid CNN-RNN architecture with attention, capable of handling both training and unseen data effectively. Due to space constraints, only the learning curves and confusion matrix for the hybrid model with attention (GloVe 27B) are presented, while results for other models are omitted.

## 6 Discussion

The results of this study offer important insights into the performance of various model architectures and the role of pre-trained embeddings in sentiment analysis tasks. Below are the key observations drawn from the findings:

1. **Significance of Pre-trained Embeddings:** Models utilizing **GloVe 27B embeddings** con-

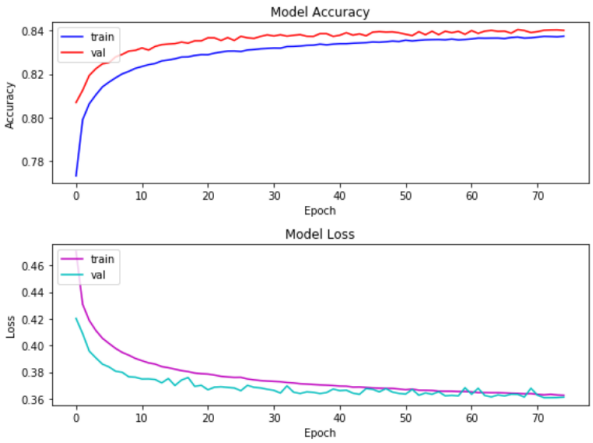


Figure 7: Learning Curve for Hybrid Model with attention (GloVe 27B).

sistently outperformed those that used **GloVe 6B embeddings**. This can be attributed to the richer vocabulary and contextual information embedded in GloVe 27B, which was trained on a much larger corpus of Twitter data. The larger corpus allows the model to capture more complex word relationships and nuanced meanings, which ultimately leads to better performance in sentiment analysis. The **Hybrid Model GloVe 27B** and the **Hybrid Model with Attention GloVe 27B** achieved F1 scores of 0.8362 and 0.8399, respectively, demonstrating the advantage of using high-quality embeddings for sentiment analysis tasks.

2. **Performance of Hybrid Architectures:** The **hybrid models**, which combine **CNN** and **RNN** layers, performed competitively across both embeddings. The **Hybrid Model GloVe 27B**, with an F1 score of 0.8362, demonstrated the benefit of combining local feature extraction (via CNN) and sequential dependency modeling (via RNN). When an **attention mechanism** was added, the performance improved slightly, with the **Hybrid Model with Attention GloVe 27B** achieving the highest F1 score of 0.8399. This suggests that the attention mechanism helps the model focus on critical parts of the text, enhancing its ability to capture complex sentiment relationships.

3. **Impact of Attention Mechanisms:** The addition of the **attention mechanism** resulted in a small but significant improvement in performance. The **Hybrid Model with Attention GloVe 27B** achieved an F1 score of 0.8399, which was slightly higher than the 0.8362 achieved by the non-attention hybrid model. This indicates that attention mechanisms are effective in allowing the model to emphasize more relevant parts of the input text, especially in sequences where context is crucial for determining sentiment. In sentiment analysis, where certain words or phrases can dramatically change the meaning of the sentence, attention mechanisms play a crucial role.

4. **Performance of Standalone Models:** Among the standalone models, the **RNN Model GloVe 27B** outperformed both the **CNN Model GloVe 27B** and the hybrid models, achieving the highest F1 score of 0.8412. This reinforces the idea that **RNNs** are particularly effective in capturing the sequential nature of text, where understanding how sentiment evolves over time is essential. In contrast, the **CNN Model GloVe 27B** achieved the lowest F1 score of 0.8257, highlighting that while CNNs are effective at capturing local patterns (such as n-grams), they struggle with modeling long-range dependencies, which are necessary for accurately interpreting sentiment in text.

5. **Effect of Model Complexity:** Introducing additional complexity to the hybrid model by adding an extra CNN layer resulted in only a marginal improvement in performance. The **Hybrid Model with 2 CNN Layers GloVe 6B** achieved an F1 score of 0.8301, slightly higher than the **Hybrid Model GloVe 6B**'s F1 score of 0.8277. This suggests that simply increasing the depth of the model does not always lead to significant gains. In fact, increasing complexity may introduce diminishing returns, making it important to balance model depth with efficiency to avoid overfitting, especially when working with large datasets.

### 6.1 Comparison with DistilBERT

In addition to the hybrid CNN-RNN models, the performance of DistilBERT, a more efficient variant of BERT, was also evaluated for sentiment analysis. DistilBERT achieved an accuracy of 0.8417, precision of 0.8442, and recall of 0.8392, demonstrating strong performance in classifying senti-

ment. Despite the computational efficiency of DistilBERT compared to the original BERT model, the hybrid CNN-RNN model with GloVe 27B embeddings delivered similar results, particularly in terms of F1 score, which was 0.8399 for the hybrid model with attention.

This comparison highlights a key observation: while DistilBERT offers competitive performance, especially in terms of accuracy and precision, the hybrid CNN-RNN model provides a viable alternative, achieving nearly equivalent results with significantly reduced computational overhead. The ability to achieve similar performance with less training time and fewer resources makes the hybrid CNN-RNN approach a strong contender, particularly in environments where computational efficiency is crucial.

### 6.2 Analysis of Misclassified Texts

To evaluate the model's performance, an analysis of misclassified examples was conducted. The dataset includes three columns: `text_x` (input text), `sentiment_x` (true sentiment), and `Predicted Sentiment` (model's prediction). Misclassified examples reveal the model's limitations, particularly in challenging scenarios.

### 6.3 Observations from Misclassified Examples

1. **Example 1**: *"where the f are my pinking shears? rararararrrarrrrraarrrrgh!!!"*
   **True Sentiment**: Negative
   **Predicted Sentiment**: Positive
   **Analysis**: The informal and noisy text with exaggerated spellings misled the model.

2. **Example 2**: *"Song Of My Life Now...Your Love Is A Lie-Simple Plan"*
   **True Sentiment**: Negative
   **Predicted Sentiment**: Positive
   **Analysis**: The presence of "love" and "song" caused the model to overlook the negative context.

3. **Example 3**: *"@hostile_bioform I dropped your books off in the library :)"*
   **True Sentiment**: Positive
   **Predicted Sentiment**: Negative
   **Analysis**: Despite the positive emoji, the word "dropped" suggested negativity.

4. **Example 4**: *"Can't feel my bottom and is now going to bed."*
   **True Sentiment**: Positive
   **Predicted Sentiment**: Negative
   **Analysis**: The phrase "can't feel my bottom" was misinterpreted negatively despite the humorous tone.

#### 6.3.1 Identified Challenges

- **Ambiguity and Context Dependence**: Many texts were ambiguous, complicating sentiment identification.
- **Informal and Noisy Text**: Informal expressions and exaggerated spellings often led to errors.
- **Emoji Interpretation**: Emojis, crucial for sentiment, were not always accurately processed.
- **Domain-Specific Vocabulary**: Uncommon phrases, such as "UFC Undisputed," were not well-understood due to limited training data.

#### 6.3.2 Recommendations for Improvement

1. **Data Augmentation**: Include informal language, emojis, and domain-specific terms in the training set.
2. **Use of Contextual Models**: Leverage pretrained models like DistilBERT or RoBERTa for better handling of ambiguous inputs.
3. **Emoji Sentiment Mapping**: Preprocess emojis to map them to corresponding sentiments.
4. **Balanced Dataset**: Ensure the dataset includes examples with ambiguity, sarcasm, and domain-specific content.

### 6.4 Future Work

While this study emphasizes the efficacy of hybrid CNN-RNN models for sentiment analysis, several promising avenues remain unexplored, offering opportunities to enhance the current work:

#### 6.4.1 Exploration of Transformer-Based Models

Expanding the scope to include transformer-based architectures, such as BERT, RoBERTa, or GPT, could provide new insights into sentiment analysis performance. These models, with their ability to capture intricate language patterns, can be fine-tuned on domain-specific datasets, such as product reviews, movie reviews (IMDB), or healthcare-related sentiment datasets, to achieve higher accuracy and broader applicability.

#### 6.4.2 Evaluation Across Diverse Datasets

Future research could assess the generalizability of the proposed models by testing them on datasets beyond Sentiment140. For instance, datasets like IMDB for longer text, Amazon Reviews for product feedback, or Multilingual Sentiment Analysis datasets for non-English languages would offer a more comprehensive understanding of the model's strengths and limitations across different contexts and domains.

#### 6.4.3 Domain Adaptation and Multilingual Applications

Adapting the hybrid model to domain-specific or multilingual sentiment analysis tasks presents an exciting direction for further research. Incorporating multilingual embeddings such as mBERT or XLM-R would enable the model to perform in diverse linguistic contexts, particularly for low-resource languages where labeled data is scarce. This could also involve domain adaptation techniques to customize the model for specialized fields, such as financial sentiment analysis or healthcare reviews.

#### 6.4.4 Mitigating Model Errors

Handling nuanced sentiment cases, such as sarcasm, mixed sentiment, or ambiguous phrasing, remains a challenge. Incorporating advanced techniques like context-aware embeddings, external knowledge graphs, and sarcasm detection mechanisms could significantly enhance model performance. Exploring methods to better understand and address errors from misclassified samples would also provide actionable insights for model improvement.

#### 6.4.5 Real-Time Sentiment Analysis and Deployment

Optimizing the hybrid CNN-RNN model for real-time sentiment analysis would expand its applicability to scenarios requiring instantaneous feedback, such as social media monitoring, customer service, or event tracking. Future work could focus on reducing inference time and computational overhead, making the model suitable for deployment in resource-constrained environments like mobile devices or embedded systems.

## 7 Conclusion

This study examined the effectiveness of various deep learning architectures for sentiment analysis, using GloVe embeddings and comparing them with DistilBERT. Models with **GloVe 27B** consistently outperformed those with **GloVe 6B**, demonstrating the advantage of richer, larger corpus-based embeddings in capturing nuanced sentiment.

**Hybrid CNN-RNN models**, particularly those with **attention mechanisms**, achieved competitive results, with an F1 score of 0.8399. However, standalone **RNN models** achieved the highest F1 score of 0.8412, underscoring the importance of sequential modeling in sentiment tasks.

DistilBERT performed well with an accuracy of 0.8417 and precision of 0.8442, though hybrid models delivered near-equivalent results with significantly lower computational overhead. The study also found that increasing model complexity by adding layers yielded diminishing returns, and CNN models struggled with long-range dependencies compared to RNNs.

Analysis of misclassified examples revealed challenges, such as handling noisy text, informal language, sarcasm, and domain-specific phrases. These insights suggest that future research should focus on improving model handling of ambiguous and context-dependent sentiment.

Overall, the research highlights the value of high-quality embeddings, hybrid CNN-RNN models, and balancing model complexity. Future work could integrate fine-tuned embeddings, explore advanced transformers, and address challenges like sarcasm and informal language to improve model robustness.

## Limitations

This study has the following key limitations:

1. **Dependency on Pre-trained Embeddings:** The models relied on pre-trained embeddings like GloVe, which may not capture domain-specific terms or nuanced sentiments. Fine-tuning embeddings on domain-specific datasets could improve performance.

2. **Handling Noisy and Informal Text:** The models struggled with noisy or informal text, such as misspelled words or excessive punctuation. Tailored preprocessing techniques could enhance performance on such data.

3. **Inability to Capture Sarcasm and Ambiguity:** Despite advanced architectures, models

struggled with sarcasm and ambiguity. Further exploration of contextual embeddings is needed for better detection.

4. **Dataset Limitations:** While the research emphasizes the relevance of the Sentiment140 dataset, it does not address potential biases in the dataset or how they might affect generalizability. Future work should investigate dataset biases and their impact on model performance.

These limitations highlight areas for future work, such as fine-tuning embeddings, improving handling of informal text, enhancing sarcasm detection, and optimizing models for efficiency.

## References


Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

J. Devlin, M. W. Chang, K. Lee, and K. Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: Sentiment classification using machine learning techniques. In *Proceedings of the ACL Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79–86.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Chi Sun, Luyao Huang, and Xipeng Qiu. 2019. Utilizing bert for aspect-based sentiment analysis via constructing auxiliary sentence. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 380–385.

K.L. Tan, C.P. Lee, K.S.M. Anbananthen, and K.M. Lim. 2022. Roberta-lstm: A hybrid model for sentiment analysis with transformer and recurrent neural network. *IEEE Access*, 10:21517–21525.

D. Tang, F. Wei, B. Qin, T. Liu, and M. Zhou. 2014. Learning sentiment-specific word embedding for twitter sentiment classification. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008.

Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. 2016. A c-lstm neural network for text classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2870–2876.