Tribhuvan University
Institute of Science and Technology

Central Department of Computer Science and Information Technology
Kirtipur, Kathmandu
2023

Seminar Report on

"Evaluating Sentiment Analysis Models with Gated Recurrent Units"

In partial fulfillment of the requirement for Master's degree in Computer Science and Information Technology (M.Sc. CSIT), 2nd Semester

Submitted to:

Central Department of Computer Science and Information Technology, Tribhuvan University, Kirtipur, Kathmandu, Nepal

Submitted By:

Sujata Shrestha (7915053/079)

Tribhuvan University

Institute of Science and Technology

Supervisor Recommendation

This is to certify that Miss. Sujata Shrestha has submitted the seminar report on the topic **"Evaluating Sentiment Analysis Models with Gated Recurrent Units"** for the partial fulfilment of Masters of Science in Computer Science and Information Technology, second semester. I hereby, declare that this seminar report has been approved.

_____

Supervisor

Asst. Prof. Mr. Bikash Balami

Central Department of Computer Science and Information Technology

Certificate of Approval

This is to certify that the seminar report prepared by Miss. Sujata Shrestha **"Evaluating Sentiment Analysis Models with Gated Recurrent Units"** in partial fulfilment of the requirements for the degree of Masters of Science in Computer Science and Information Technology has been well studied. In our opinion, it is satisfactory in the scope and quality as a project for the required degree.

Evaluation Committee

…….……..	……..……
Asst. Prof. Sarbin Sayami	**Asst. Prof. Bikash Balami**
(H.O.D)	(Supervisor)
Central Department of Computer Science and Information Technology	Central Department of Computer Science and Information Technology

……

(External)

Acknowledgement

The success and final outcome of this report required a lot of guidance and assistance from many people and I am very fortunate to have got this all along the completion. I am very glad to express my deepest sense of gratitude and sincere thanks to my highly respected and esteemed supervisor **Asst. Prof. Bikash Balami,** Central Department of Computer Science and Information Technology for his valuable supervision, guidance, encouragement, and support for completing this paper.

I am also thankful to **Asst. Prof. Sarbin Sayami**, HOD of Central Department of Computer Science and Information Technology for his constant support throughout the period. Furthermore, with immense pleasure, I submit by deepest gratitude to the Central Department of Computer Science and Information Technology, Tribhuvan University, and all the faculty members of CDCSIT for providing the platform to explore the knowledge of interest. At the end I would like to express my sincere thanks to all my friends and others who helped me directly or indirectly.

Thanking you,

Sujata Shrestha (7915053)

Abstract

Sentiment analysis, a crucial task in natural language processing (NLP), involves determining the emotional tone behind a body of text. This paper presents an evaluation of sentiment analysis using a Gated Recurrent Unit (GRU) neural network, focusing on movie reviews. The dataset is preprocessed, including tokenization, padding, and splitting into training and testing sets. The GRU model, featuring an embedding layer, a GRU layer with dropout for regularization, and a dense output layer with a sigmoid activation function, is compiled with the Adam optimizer and binary cross-entropy loss function. The model is trained and validated, achieving an accuracy of 82.61%, precision of 80.74%, recall of 86.04%, F1 score of 83.31%, and specificity of 79.12%. Performance is further evaluated using a confusion matrix and visualized through heatmaps and accuracy/loss plots. The results demonstrate the GRU model's effectiveness in sentiment classification.

Keyword: Confusion Matrix, Gated Recurrent Unit, Movie reviews, Natural language processing, Sentiment analysis

Table of Content

List of Figures

List of Abbreviations

GRU: Gated Recurrent Unit

NLP: Natural Language Processing

LSTM: Long Short-Term Memory

RNN: Recurrent Neural Network

IDE: Integrated Development Environment

IMDB: Internet Movie Database

Chapter 1: Introduction

1.1 Introduction

Sentiment analysis, also referred to as opinion mining, is part of the general area of Natural Language Processing (NLP) that addresses identifying and extracting subjective information from text data. It aims at assessing the attitude, emotions, and opinions stated in a text, the outcome usually being classified as positive or negative. Such analysis is important for a number of applications, including market research, social media monitoring, customer feedback analysis, and more, because it helps organizations to understand the sentiments of their audience and make the right decisions.

One of the advanced techniques used for sentiment analysis is based on the use of deep learning models, in particular, Gated Recurrent Unit (GRU) networks. GRUs are a type of recurrent neural network designed to capture temporal dependencies and process sequential data effectively. They are a simplified variant of Long Short-Term Memory (LSTM) networks, offering comparable performance with fewer parameters and lower computational complexity. GRUs deal with the vanishing gradient problem inherent in traditional RNNs, thus they can learn long-term dependencies. This is what makes them applicable, particularly to sentiment analysis tasks, where understanding the context and sequence of words is an essential part to correctly classify the sentiment. Using the power of GRUs, we can learn robust models that provide state-of-the-art performance for sentiment analysis, even for difficult and subtle pieces of textual data.

In this study, the application of GRU networks for sentiment analysis was explored. The architecture and functioning of GRUs, the dataset used for training and evaluation, the steps of preprocessing involved, and the results obtained from the experiments were examined. The study aimed to demonstrate the effectiveness of GRUs in capturing sentiment nuances and providing profound insights from textual data.

1.2 Problem Statement

The biggest problem is the need for automated sentiment analysis of movie reviews in the tremendous volume of online user-generated movie-related content. Manual assessment of sentiments is impractical because of the volumes involved. Therefore, there is an increasing need for an automated solution using machine learning to classify movie reviews as positive or negative automatically. Developing an accurate and efficient model that could effectively handle the subtleties of the language in movie reviews presented a significant challenge. This report addresses this problem through the application of GRU architecture for sentiment analysis.

1.3Objective

The main objective includes;

- To develop a GRU-based sentiment analysis model for movie reviews.
- To train the model and evaluate its performance.

Chapter 2: Background Study and Literature Review

2.1 Background Study

2.1.1 Recurrent Neural Network:

A Recurrent Neural Network (RNN) is a type of neural network that has an internal memory, allowing it to process sequential data. Unlike feedforward neural networks, which treat each input independently, RNNs have connections that allow information to be passed from one step to the next. This enables them to capture dependencies and patterns across the sequence.

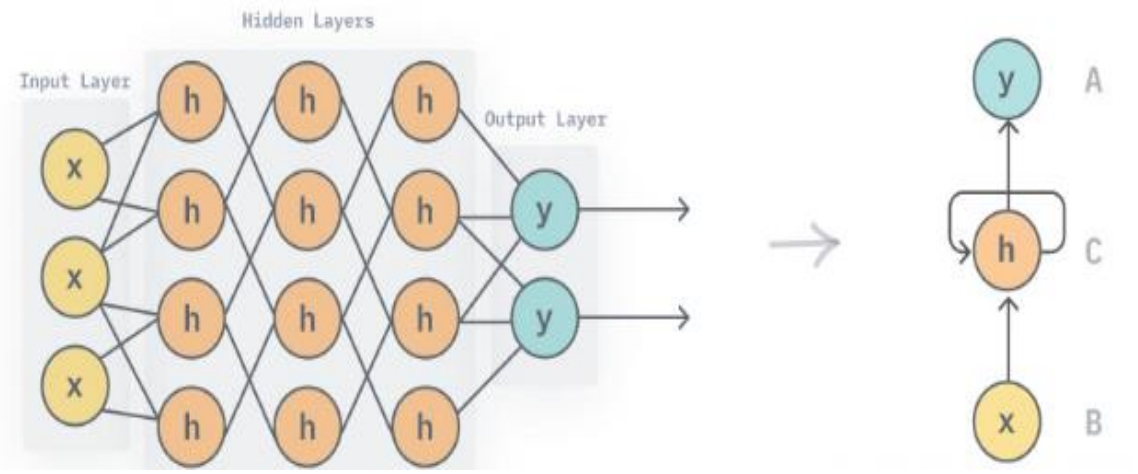

Figure 1:Recurrent Neural Network

RNNs are particularly suited for tasks that involve sequential data, such as handwriting recognition or speech recognition, where the current input's output depends not only on the current input but also on the previous inputs processed by the network. They have connections that fold back on themselves; this allows information from one step to be passed on to the following steps. This makes them well-suited to learning patterns over sequences, which has proved useful for tasks ranging from language translation to speech recognition. In spite of this, RNNs do face serious problems. A major one is the vanishing gradient problem, where the gradients used to update the weights of the network become very small after many time steps; that is, it becomes really hard to learn long-term dependencies in the data.

And thankfully GRU can be used to solve the vanishing gradient problem.

2.1.2Gated Recurrent Unit:

Gated Recurrent Unit (GRU) networks are a type of recurrent neural network (RNN) designed to handle sequential data and capture temporal dependencies. GRUs were introduced to address some of the limitations of traditional RNNs, particularly the vanishing gradient problem, which makes it difficult for RNNs to learn long-term dependencies.

A GRU works in following ways:

1. **Hidden State(h_t):** Like traditional RNNs, a GRU maintains a hidden state vector h_t at each time step t. This hidden state captures information about the sequence of inputs processed so far.
2. **Update Gate(z_t):** At each time step, a GRU computes an update gate z_t that determines how much of the previous hidden state h_{t-1} to retain and how much new information to incorporate from the current input x_t . The update gate is computed using a sigmoid activation function, which squashes the values between 0 and 1.

$$z_t = \sigma\,(W_z \,.\, [h_{t-1}\,, x_t])$$

3. **Reset Gate (r_t):** Additionally, a reset gate r_t is computed at each time step to control how much of the previous hidden state h_{t-1} should be forgotten. Similar to the update gate, the reset gate is also computed using a sigmoid activation function. It controls the contribution of the previous hidden state to the candidate activation.

$$r_t = \sigma\,(W_r \,.[h_{t-1}\,, x_t])$$

4. **Candidate Hidden State ($\tilde{h}_t$):** This represents the new candidate content that will be added to the current state. The reset gate controls how much of the previous state is included in the new candidate.

$$\tilde{h}_t = \tanh\,(W_h \,.\, [r_t * h_{t-1}\,, x_t])$$

5. **Update Hidden State (h_t):** The final hidden state is a combination of the previous hidden state and the new candidate activation, controlled by the update gate. This allows the GRU to decide how much of the new information to incorporate into the hidden state while retaining information from previous time steps.

$$h_t = (1- Z_t)* h_{t-1} + Z_t * \tilde{h}_t$$

Where:

- *σ is the sigmoid function.*
- W_z , W_r , W_h are weight matrices for the update and reset gates, respectively.
- h_{t-1} is the previous hidden state.
- x_t is the current input.
- $\tilde{h}_t$ is the candidate activation.
- * represents elements wise multiplication and . denotes concatenation.

2.2 Literature Review:

Sentiment analysis [1] of such data which comprises of people's views is very important in order to gauge public opinion on a particular topic of interest. This paper reviews a number of techniques, both lexicon-based approaches as well as learning-based methods that can be used for sentiment analysis of text. In order to adapt these techniques for sentiment analysis of data procured from one of the social networking websites, Twitter, a number of issues and challenges need to be addressed, which are put forward in this paper.

Sentiment classification is extensively [2] used in many business domains to improve products or services by understanding the opinions of customers regarding these products. Deep learning achieves state-of-the-art results in various challenging domains. Herein, we present a comparative study of different deep-learning-based sentiment classification model structures to derive meaningful implications for building sentiment classification models. Specifically, eight deep-learning models, three based on convolutional neural networks and five based on recurrent neural networks, with two types of input structures, i.e., word level and character level, are compared for 13 review datasets, and the classification performances are discussed under different perspectives. In this paper [3] we try to investigate the impact of hyper parameters like dropout, number of layers, activation functions. We have analyzed the performance of the model with different neural network configurations and reported their performance with respect to each configuration. IMDB bench mark dataset is used for the experimental studies.

Neural networks, such as long short-term memory (LSTM) and the gated recurrent unit (GRU) [4], are good predictors for univariate and multivariate data. The present paper describes a case study where the performances of long short-term memory and gated

recurrent units are compared, based on different hyperparameters. In general, gated recurrent units exhibit better performance, based on a case study on pulp paper presses. The final result demonstrates that, to maximize the equipment availability, gated recurrent units, as demonstrated in the paper, are the best options.

Chapter 3: Methodology

Sentiment Analysis of movie review is done using GRU recurrent neural network. The detail architecture of the system is presented below.

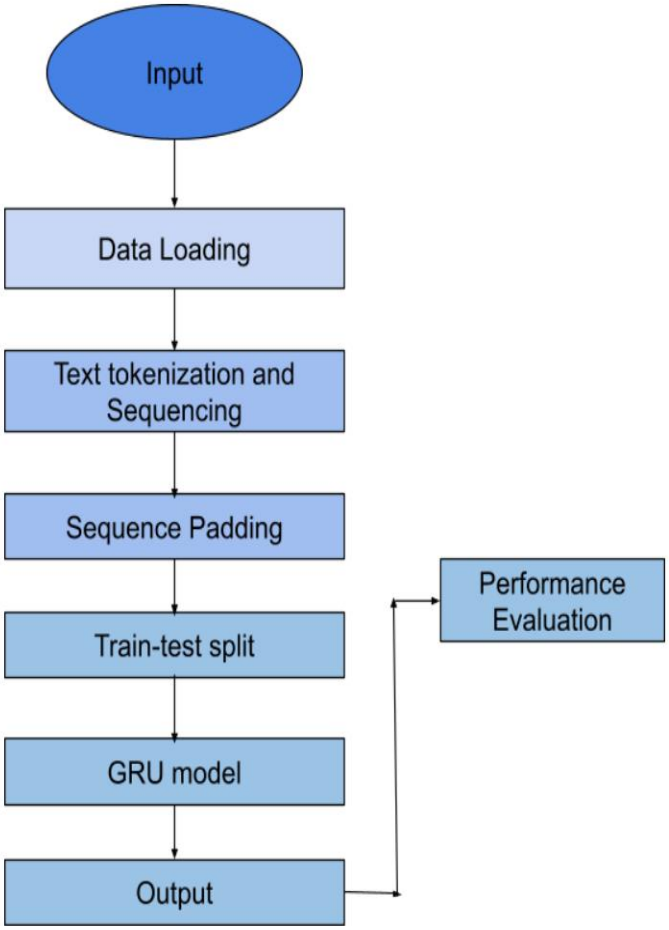

Figure 2: Architecture

Each step in the architecture illustrated in Figure 2 is explained in brief as follows.

3.1 Data Loading:

Input data for this project i.e., Movie review Dataset is taken from Kaggle [5] an online repository known for its diverse collection of datasets. The dataset consists of a total of 8000unique entries, with 4034 entries labeled as positive sentiments (1) and the remaining 3960 entries classified as negative sentiments (0).

text	label
I grew up (b. 1965) watching and loving the Thunderbirds. All my mates at school watched. We played "Thunderbirds" before school,	0
When I put this movie in my DVD player, and sat down with a coke and some chips, I had some expectations. I was hoping that this n	0
Why do people who do not know what a particular time in the past was like feel the need to try to define that time for others? Repla	0
Even though I have great interest in Biblical movies, I was bored to death every minute of the movie. Everything is bad. The movie is	0
Im a die hard Dads Army fan and nothing will ever change that. I got all the tapes, DVD's and audiobooks and every time i watch/liste	1
A terrible movie as everyone has said. What made me laugh was the cameo appearance by Scott McNealy, giving an award to one o	0
Finally watched this shocking movie last night, and what a disturbing mindf**ker it is, and unbelievably bloody and some unforgettab	1
I caught this film on AZN on cable. It sounded like it would be a good film, a Japanese "Green Card". I can't say I've ever disliked an A	0
It may be the remake of 1987 Autumn's Tale after eleven years, as the director Mabel Cheung claimed. Mabel employs rock music as	1
My Super Ex Girlfriend turned out to be a pleasant surprise for me, I was really expecting a horrible movie that would probably be stu	1
I can't believe people are looking for a plot in this film. This is Laural and Hardy. Lighten up already. These two were a riot. Their con	1
If you haven't seen the gong show TV series then you won't like this movie much at all, not that knowing the series makes this a grea	0
I have always been a huge fan of "Homicide: Life On The Street" so when I heard there was a reunion movie coming up, I couldn't wa	1
Greg Davis and Bryan Daly take some crazed statements by a terrorists, add some commentary by a bunch of uber-right reactionarie	0
A half-hearted attempt to bring Elvis Presley into the modern day, but despite a sexy little shower scene and a pseudo-Playboy maga	0
If you want a fun romp with loads of subtle humor, then you will enjoy this flick. I don't understand why anyone wouldn't	1
I really wanted to be able to give this film a 10. I've long thought it was my favorite of the four modern live-action Batman films to d	1
The main problem with "Power" is that it features way too may pointless characters and subplots that add absolutely nothing to the	0
The folks at Disney have a lot to explain. First and foremost, why anyone thought this lesser-sitcom material would ever make even	0
A friend told me of John Fante last summer after we got into a conversation about Charles Bukowski. I did not know that Fante was	0
Ever since I heard of the Ralph Bakshi version of "The Lord of the Rings" I wondered: What the hell is 'rotoscope' animation?!!! Well	1
I sat through this film and i have to say it only just managed to keep my attention. The film would have been a bit more bearable if i	0
I don't care if some people voted this movie to be bad. If you want the Truth this is a Very Good Movie! It has every thing a movie sh	1

Figure 3: Datasets (URL Kaggle)

3.2Text tokenization and Sequencing

In the context of sentiment analysis using GRU, text tokenization and sequencing are two fundamental preprocessing steps. Tokenization means the division of text into individual words or tokens, which are then mapped to some numerical representation. These tokens are then put into fixed-length input vectors, often padded, to make them of equal length across the dataset. Such preprocessing makes the GRU model efficiently process textual data and obtain the time dependence and context. Proper tokenization and sequencing could better extract sentiment nuances from the text and improve the accuracy of sentiment predictions by a GRU model.

3.3 Padding:

Padding is a crucial methodology to ensure that all sequences of movie reviews have a consistent and fixed length before feeding them into a GRU model for training or evaluation. It uses Keras' pad sequence's function. The sequences of tokenized words, representing the movie reviews of various lengths, are filled with zeros up to a predefined maximum sequence length of 100 characters. Padding at the end of sequences is specified as 'post,' ensuring that shorter reviews are extended with zeros while maintaining their original content. This standardized sequence length is essential for efficient processing and modeling of text data using GRU, ensuring that all inputs have uniform dimensions, regardless of their original length.

3.4 Train-test split:

The train-test split is a critical step in machine learning to evaluate the model's performance. It involves dividing the dataset into two subsets: the training set and the testing set. In this code, train-test-split from Scikit-Learn is used to split the data into training and testing sets. The training set (typically a larger portion, 80%of the data) is used to train the GRU model, allowing it to learn from the data. The testing set (remaining 20% of the data) is kept separate and used for evaluating the model's performance. This split helps assess how well the model generalizes to unseen data and provides insights into its ability to make accurate predictions on new, unobserved movie reviews.

3.5 GRU model:

To build the GRU model for sentiment analysis of movie reviews, we start by initializing a Sequential model. We then add an embedding layer that transforms the tokenized sequences into dense vector representations, allowing the model to understand the context and semantics of the words. Next, we incorporate a GRU layer, which is designed to capture temporal dependencies and contextual information from the sequences, effectively learning patterns and trends in the review texts. Here we also include the dropout layer which helps the model generalize better to unseen data and to prevent the overfitting. Finally, we include a Dense layer with a sigmoid activation function to output a probability score, indicating the likelihood of the review being positive. This architecture enables the GRU model to process and analyze the sentiment of movie reviews effectively.

3.6 Performance Evaluation:

The first step in evaluation is to predict the labels for the test dataset using the trained model. This is achieved by using the predict method on the model with the test input data (X_test). The predicted labels are then compared with the true labels in the test dataset to generate a confusion matrix. The confusion matrix visually represents the count of true positive, true negative, false positive, and false negative predictions, enabling an understanding of the model's classification performance.

From the confusion matrix, key evaluation metrics are derived. Accuracy, precision, and recall, F1 score and specificity are calculated based on the counts in the matrix. Accuracy measures the overall correctness of predictions, while precision quantifies the proportion of predicted positives that are actually correct, recall captures the proportion of actual positives that were correctly predicted, f1 score are is the harmonic mean of precision and recall and specificity measures the model's ability to correctly identify negative instances (true negatives) out of all the actual negative instances.

Moreover, loss is computed as the sum of false positives and false negatives, providing insight into misclassifications. These metrics collectively offer a comprehensive view of the model's performance in distinguishing between spam and ham emails. Visualizations, such as plots depicting training and validation loss, accuracy, and other metrics across epochs, aid in tracking the model's learning progression and potential overfitting.

Chapter 4: Implementation

4.1 Implementation

Python is used as the programming language to code the program. Vs Code is used as an IDE. Similarly, different libraries are also used as:

- **NumPy:** It is used for numerical operations.
- **Pandas:** It is used for data manipulation and analysis.
- **Matplotlib:** It is used for plotting and visualizations.
- **Seaborn:** It is used for statistical data visualization.
- **TensorFlow and Keras:** It is used for building and training deep learning models.
- **Scikit-learn:** It is used for preprocessing, model evaluation, and metrics.

4.2 Implementation details

Implementation details includes;

4.2.1Train–Test Split:

Split the dataset into a training set and a test set. The split ratio chosen is 80:20, 80% for training and 20% for testing. This division ensures that the model is trained on a substantial portion of the data while being evaluated on unseen samples, allowing us to measure its ability to generalize the sentiments.

```
n-test split
n, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

✓ 0.1s Python

Figure 4:Code to split data into training and testing

4.2.2 Text Sequencing and Padding:

The text data was transformed into sequences of words using a tokenizer, and each sequence was padded with zeros to have a fixed length of 100 words (SEQUENCE_LENGTH). This step ensured that all sequences had consistent dimensions for efficient processing.

```
# Tokenize the text
tokenizer = Tokenizer(num_words=3000)
tokenizer.fit_on_texts(text)
sequences = tokenizer.texts_to_sequences(text)
word_index = tokenizer.word_index
vocab_size = len(word_index) + 1
```

✓ 19.6s

Figure 5:Code for Sequencing

```
# Pad the sequences
max_length = 100
X = pad_sequences(sequences, maxlen=max_length,padding='post')
y = np.array(encoded_labels)
```

Figure 6: Code for padding

4.2.3 GRU Architecture:

The implemented neural network architecture using GRU begins with the Embedding layer, where input sequences of word indices are translated into dense vectors of the same size 100, defined by `embedding_dim`. That is followed by a Gated Recurrent Unit layer comprising 64 units that processes the sequence data and harbors the temporal dependencies to output a single sequence of fixed length. Further, a Dropout layer is applied, with the rate set at 0.5 to prevent overfitting, where half of the input units are randomly set to zero during training. Finally, a Dense layer comprising a single neuron with the sigmoid activation function yields the output for binary classification.

```
# Define the GRU model
embedding_dim = 100

model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=embedding_dim, input_length=max_length))
model.add(GRU(units=64, return_sequences=False))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
```

Figure 7:Code for GRU Architecture

4.2.4 Model Training and Adam Optimizer:

Before training, the model is compiled. This involves configuring the optimizer, loss function, and evaluation metrics. In the provided code, the Adam optimizer is chosen for training. It adapts learning rates for each parameter, which can lead to faster convergence. Binary cross-entropy is used as the loss function, and accuracy is the evaluation metric.

The model is trained over a fixed number of epochs using mini-batches of data (batch size of 512). A validation dataset is used to monitor performance. During training, the Adam optimizer adjusts the model's weights based on the loss function gradients, helping it learn to distinguish between positive and negative sentiments.

```
# Compile the model
optimizer = Adam(learning_rate=0.0001)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])
```

Figure 8: Code For compiling the model

```
# Train the model
history = model.fit(X_train, y_train, epochs=5, validation_data=(X_test, y_test), batch_size=512)
```

Figure 9: Code for training the model

4.2.5 Model Evaluation:

Following training, the model's performance was evaluated on the test dataset. Metrics such as accuracy, precision, F1 score, specificity and recall were calculated from the confusion matrix to assess its effectiveness in classifying positive and negative sentiments.

```
# Calculate and print additional metrics
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
specificity = tn / (tn + fp)

# Calculate and print additional metrics in percentage
accuracy_percentage = accuracy * 100
precision_percentage = precision * 100
recall_percentage = recall * 100
f1_percentage = f1 * 100
specificity_percentage = specificity * 100

print(f"Accuracy: {accuracy_percentage:.2f}%")
print(f"Precision: {precision_percentage:.2f}%")
print(f"Recall: {recall_percentage:.2f}%")
print(f"F1 Score: {f1_percentage:.2f}%")
print(f"Specificity: {specificity_percentage:.2f}%")
```

0.1s

Figure 10: code for evaluating the model

4.2.6 Sentiment Analysis Function:

A function was defined to classify sentiments as positive or negative based on the trained model. The input text was tokenized, padded, and passed through the model to obtain predictions, with the predicted class label returned.

```
# Function to preprocess new reviews
def preprocess_reviews(reviews, tokenizer, max_length):
    sequences = tokenizer.texts_to_sequences(reviews)
    padded_sequences = pad_sequences(sequences, maxlen=max_length)
    return padded_sequences
```

Figure 11:code for the sentiment analysis function

Chapter 5

Result and Findings

5.1 Overfitting and Generalization

The training and validation loss and accuracy plots displayed a smooth convergence during the training process. There were no significant signs of overfitting or underfitting, suggesting that the model effectively learned from the data without memorizing it.

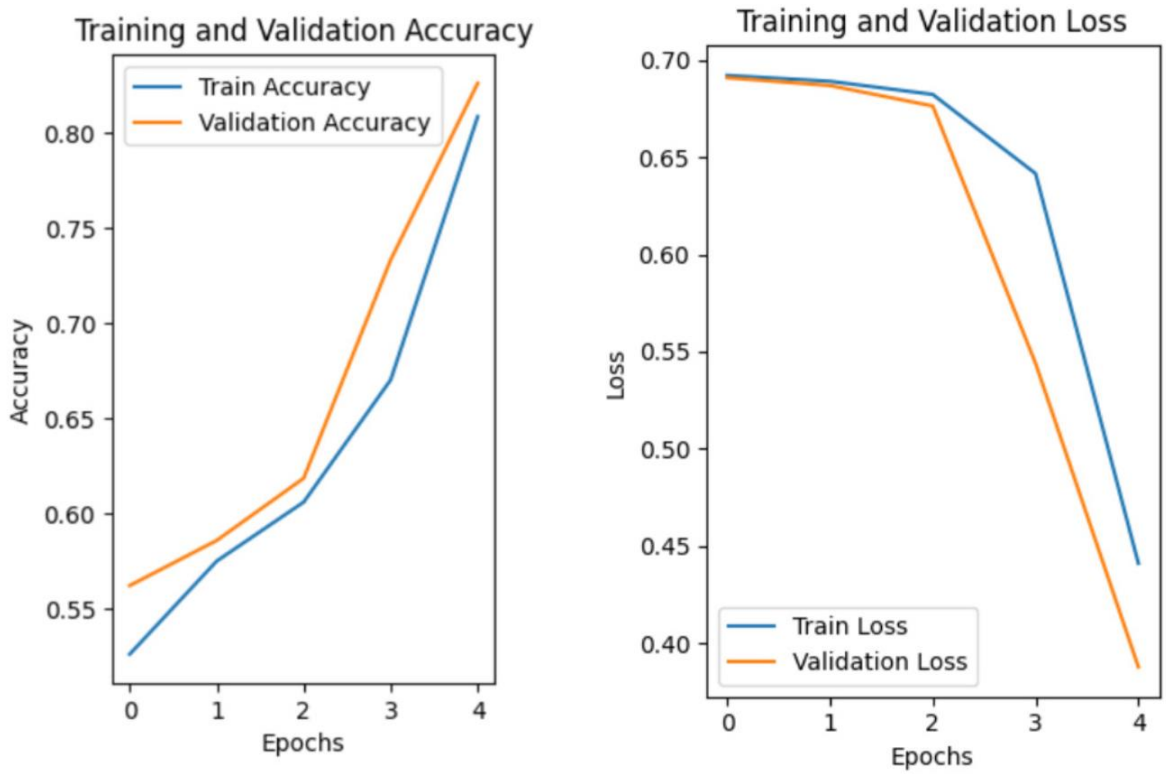

Figure 12: Training and Validation Loss and accuracy plots

5.2 Confusion Matrix

The confusion matrix results provide a detailed view of the classification performance of the model. In this specific scenario, the model correctly identified **3471** instances as true positives, indicating that it accurately recognized as positive sentiment. It also correctly classified **3138** instances as true negatives, signifying its ability to correctly identify as negative sentiment. However, the model had **828** false positives, indicating instances where negative sentiments were mistakenly classified as positive. Additionally, there were **563** false negatives, highlighting cases where positive sentiments were misclassified as negative.

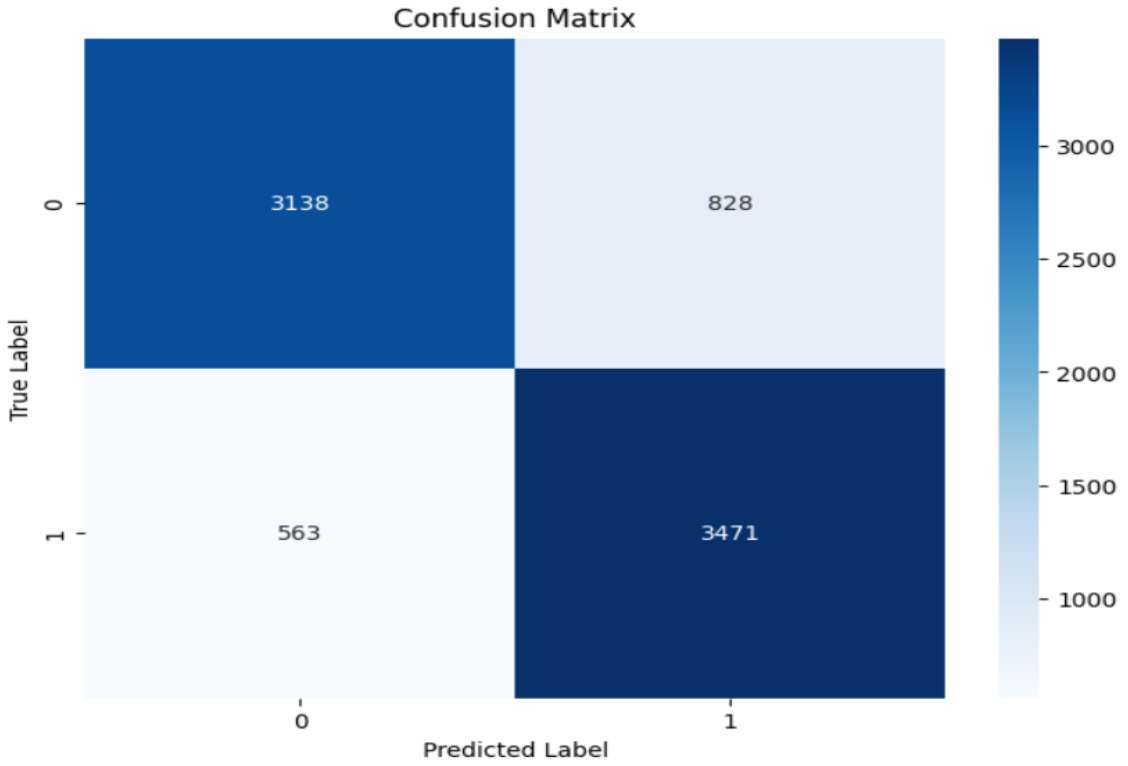

Figure 13: Confusion Matrix

Considering these values, the loss metric, which encompasses both false positives and false negatives, can be calculated. In this case, the loss value would be the sum of false positives **(828)** and false negatives **(563)**, resulting in a total loss of **1391**.

The model is performing well, with a high accuracy, precision, F1 score, specificity and recall. The loss is also relatively low. This suggests that the model is able to learn the patterns of positive and negative sentiments and classify them correctly with a high degree of confidence.

5.3 Model Performance Measure

The Sentiment Classifier using GRU model demonstrated impressive performance in distinguishing between positive and negative sentiments. After training the model for 5 epochs with a batch size of 512 the following evaluation metrics were obtained on the test dataset:

- **Accuracy: 82.61%**
- **Precision: 80.74%**
- **Recall: 86.04%**
- **F1 Score: 83.31%**
- **Specificity: 79.12%**

These metrics indicate that the model achieved a high accuracy in correctly classifying sentiments as positive or negative. The precision score reflects the model's ability to correctly identify positive and negative sentiments, ensuring that the majority of predicted positive sentiments are accurate. The recall score indicates its ability to identify all the actual positive sentiments present in the dataset, capturing the majority of true positive sentiments. The F1 score represents the model's ability to achieve both accurate positive sentiment identification and comprehensive capturing of actual positive sentiments, balancing precision and recall. The specificity score highlights the model's skill in accurately identifying true negatives (negative sentiments) within the dataset, ensuring that most of the negative sentiments are correctly classified.

Chapter 6: Conclusion and Future Recommendation

6.1 Conclusion

In this implementation, the successful handling of sentiment analysis of movie reviews was achieved through the utilization of Gated Recurrent Unit (GRU) architecture. The potency of the GRU-based model in realizing subtleties of the language and in distinguishing between positive and negative sentiments has been depicted well. With an accuracy of 82.61%, precision of 80.74%, recall of 86.04%, F1 score of 83.31%, and specificity of 79.12%, the model showed strong performance. A strong way of performing sentiment analysis is demonstrated with the help of these insightful visualizations, complete metrics, and efficiency of the Adam optimizer in rendering valuable tools for all text classification tasks. This GRU-based model demonstrates its ability in handling sequential data effectively, proving a powerful tool for sentiment analysis and other natural language processing tasks.

6.2 Future Recommendation:

For future improvements, optimize GRU units and learning rates for better performance. Explore data augmentation to diversify training data. Use attention mechanisms for nuanced text analysis. Consider transfer learning with BERT for complex language structures. Combine models with ensemble methods. Test versatility with domain adaptation. Evaluate real-time performance in social media monitoring. Develop explainability for better trust in predictions.

References

[1] V. a. S. V. a. P. A. Sahayak, "Sentiment analysis on twitter data," *International Journal of Innovative Research in Advanced Engineering (IJIRAE),* vol. 2, pp. 178-183, 2015.

[2] S. a. K. C. a. K. H. a. M. K. a. K. P. Seo, "Comparative study of deep learning-based sentiment classification," *IEEE Access,* vol. 8, pp. 6861--6875, 2020.

[3] J. D. a. V. N. a. S. S. N. Bodapati, "Sentiment Analysis from Movie Reviews Using LSTMs," *Ing{\'e}nierie des Syst{\`e}mes d Inf.,* vol. 24, pp. 125--129, 2019.

[4] B. C. a. M. M. a. F. J. T. a. A. R. a. C. A. M. Mateus, "Comparing LSTM and GRU models to predict the condition of a pulp paper press," *Energies,* vol. 14, p. 6958, 2021.

[5] T. B. S. M Yasser H, "IMDB Movie Ratings Sentiment Analysis," vol. 49, 2022.