Tribhuvan University

Institute of Science and Technology

Central Department of Computer Science and Information Technology
Kirtipur, Kathmandu
2024

Seminar Report on
"Sentiment Analysis Using Gradient Recurrent"

In partial fulfillment of the requirement for a Master's degree in Computer Science and Information Technology (M.Sc. CSIT), 1st Semester

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

Submitted By:
Sagar Timalsena (8015031)

Tribhuvan University

Institute of Science and Technology

Supervisor Recommendation

This is to certify that Mr. Sagar Timalsena has submitted the seminar report on the topic "**Sentiment Analysis using Gated Recurrent Units**" for the partial fulfillment of the Master of Science in Computer Science and Information Technology, First semester. I hereby declare that this seminar report has been approved.

_____

Supervisor

Assistant Professor Mr. Bikash Balami

Central Department of Computer Science and Information Technology (CD.CSIT)

Certificate of Approval

This is to certify that the seminar report prepared by Mr. Sagar Timalsena "**Sentiment Analysis using Gated Recurrent Units**" in partial fulfilment of the requirements for the degree of Master 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	**Assoc. 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

I sincerely thank everyone who has helped with technical and informational support and guidance.

I begin by extending my gratitude to my mentor **Mr. Bikash Balami**, Assistant Professor, Central Department of Computer Science and Information Technology (CD.CSIT) Tribhuvan University for his continuous guidance and support during the research and learning period. Without his guidance, this research would not have been successful all staff members for their cooperation and guidance during the entire research period. The success and outcome of this research required a lot of advice and assistance from many people and I am extremely fortunate to have them all along the completion of this research period. Such guidance was indispensable for the accomplishment of this endeavor.

I am thankful to my supervisor for providing me with his continuous guidance, advice, motivation, and knowledge he imparted to me regarding every perspective of the report. I also thank the Central Department of Computer Science and Information Technology MSc.CSIT department and library for providing me with the necessary resources during the time of research and report.

I would also like to acknowledge, my parents, for providing financial and psychological support throughout my study and all those who contributed directly and indirectly.

Thanking You,

Sagar Timalsena

(TU Roll No.: 8015031)

ABSTRACT

Sentiment analysis is a crucial process in natural language processing (NLP) that determines whether a given text expresses a positive, negative, or neutral sentiment. This paper presents an evaluation of sentiment analysis using a Gated Recurrent Unit (GRU) neural network, specifically focusing on tweets about US Airlines. The proposed model demonstrates robust performance, achieving an accuracy of 82.61%, a precision of 80.84%, a recall of 85.53%, an F1 score of 79.66%, and a specificity of 80.33%. 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.

Keywords: *Natural Language Processing (NLP), Language Semantics, Text Analysis, Textual Meaning Extraction*

Table of Contents

List of Figure

List of Abbreviations

GRU:	Gated Recurrent Unit
IDE:	Integrated Development Environment
LSTM:	Long Short-Term Memory
MSC.CSIT:	Master of Science in Computer Science and Information Technology
NLP:	Natural Language Processing
RNN:	Recurrent Neural Network

Chapter 1: Introduction

1.1 Introduction

Sentiment analysis, also referred to as opinion mining, is a computational process that involves analyzing and interpreting the emotions and opinions expressed in textual data. In today's digital age, where the amount of textual content generated online is growing exponentially, sentiment analysis has become increasingly important across various domains.

Understanding public opinion, consumer feedback, and social media trends is critical for businesses, governments, and organizations to make informed decisions and stay relevant in a highly competitive landscape. Sentiment analysis provides valuable insights into the sentiments and attitudes of individuals towards specific topics, products, services, or events.

Traditional methods of sentiment analysis often rely on simplistic approaches such as rule-based systems or bag-of-words models. While these methods can provide basic sentiment classification, they often fail to capture the nuances and context-dependent nature of language. For example, a simple approach might classify the word "good" as positive, but it might overlook subtle variations in meaning, such as "good" in the context of being sarcastic or as part of a negation ("not good"). In recent years, deep learning techniques, particularly recurrent neural networks (RNNs), have emerged as powerful tools for sentiment analysis. RNNs are a class of artificial neural networks designed to effectively process sequential data, making them well-suited for tasks involving natural language processing.

Unlike traditional methods, which require handcrafted features or predefined rules, RNNs can learn to capture complex patterns and dependencies within textual data automatically. This ability to learn from data makes RNNs highly adaptable and capable of capturing the subtle nuances of language, including sentiment.

Gated recurrent units (GRUs) are a type of RNN architecture that has gained popularity for sentiment analysis tasks due to their simplicity and effectiveness. GRUs address some of the limitations of traditional RNNs, such as the vanishing gradient problem, by introducing gating mechanisms that control the flow of information through the network.

By leveraging deep learning techniques like RNNs and GRUs, sentiment analysis models can achieve higher levels of accuracy and robustness, enabling more nuanced and context-aware sentiment analysis. This, in turn, empowers businesses, researchers, and policymakers to gain

deeper insights into public opinion, consumer sentiment, and social media dynamics, ultimately facilitating more informed decision-making and strategic planning.

1.2 Problem Statement

As the amount and intricacy of textual data continue to grow, accurately deciphering sentiments presents a significant challenge. Current sentiment analysis models often falter in grasping contextual cues and subtle emotional nuances, resulting in less-than-ideal outcomes. Therefore, there's a pressing demand for resilient and effective sentiment analysis frameworks adept at managing vast quantities of text data and precisely categorizing sentiments with heightened accuracy.

1.3 Objective

The primary objective of this study is to develop a sentiment analysis framework using gated recurrent units (GRUs), a variant of recurrent neural networks, to effectively classify sentiments in text data. Specifically, we aim:

- To implement a GRU-based sentiment analysis model
- To evaluate the performance of the model using appropriate metrics
- To analyze the effectiveness and limitations of the proposed framework

Chapter 2: Background Study and Literature Review

2.1 Background Study

2.1.1 Recurrent Neural Network

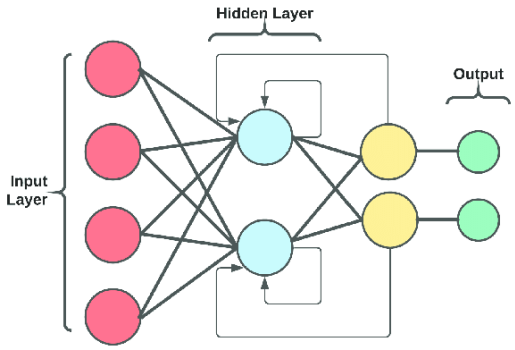

Figure 1 Recurrent Neural Network Architecture

Recurrent Neural Networks (RNNs) are a class of artificial neural networks designed to handle sequential data by maintaining a memory state. Unlike feedforward neural networks, which process data instantaneously, RNNs can retain information about previous inputs, making them well-suited for tasks such as language modeling, time series prediction, and speech recognition. However, traditional RNNs suffer from the vanishing gradient problem, which hampers their ability to capture long-term dependencies in sequential data.

2.1.2 Gated Recurrent Unit

Gated Recurrent Units (GRUs) are a type of recurrent neural network (RNN) architecture introduced to overcome some limitations of traditional RNNs, such as difficulties in learning long-range dependencies due to the vanishing gradient problem. GRUs achieve this by incorporating gating mechanisms that regulate the flow of information through the network over time.

2.2 Literature Review:

Sentiment analysis [1] of such data which comprises people's views is very important to public opinion on a particular topic of interest. This paper reviews some techniques, both lexicon-based approaches as well as learning-based methods that can be used for sentiment analysis of text. To adapt these techniques for sentiment analysis of data procured from one of the social networking websites, Twitter, several 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 [2] we try to investigate the impact of hyperparameters like dropout, number of layers, and activation functions. We have analyzed the performance of the model with different neural network configurations and reported their performance with respect to each configuration.

Neural networks, such as long short-term memory (LSTM) and the gated recurrent unit (GRU) [3], 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 Twitter US Airline Sentiment review is done using GRU recurrent neural network. The detailed architecture of the system is presented below.

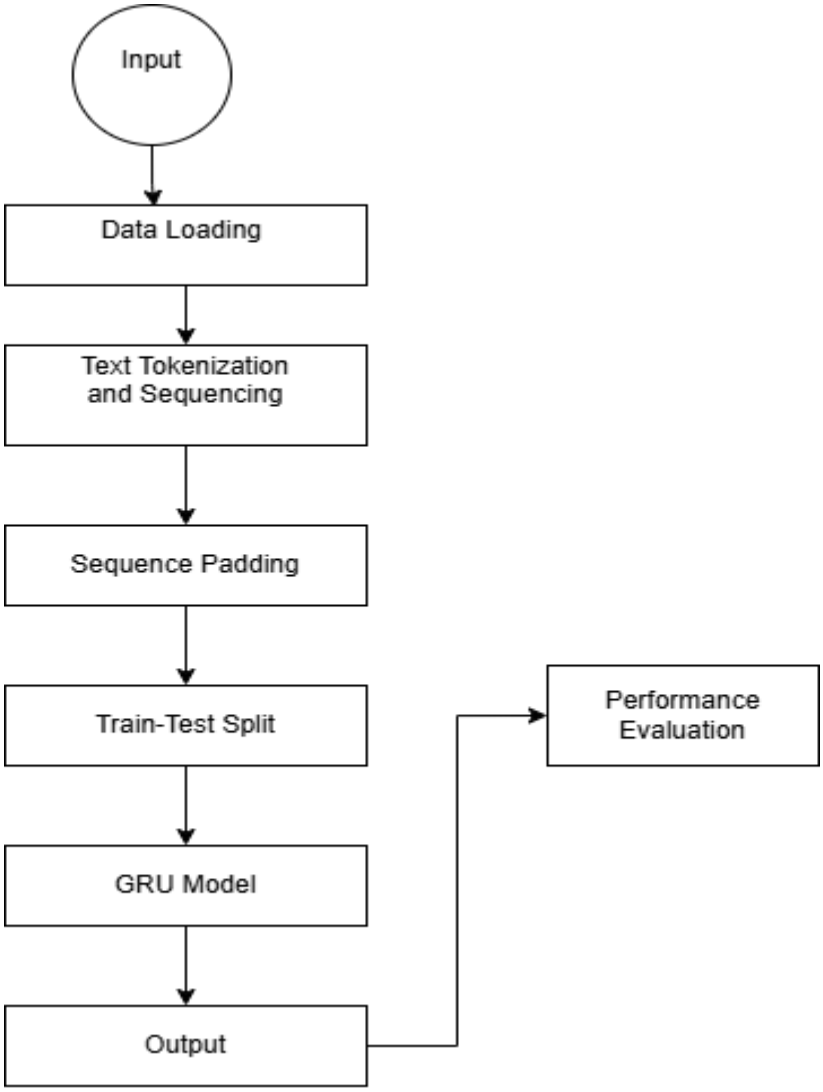

Figure 2 Architecture of Sentiment Analysis

Each step in the architecture shown in the figure is explained in detail as follows:

3.1 Data Loading

Input data for this project i.e. Twitter US Airline Sentiment Dataset is taken from Kaggle an online repository known for its diverse collection of datasets. The dataset consists of over 7000 data.

```
Tweet ID: 2401
Content: im getting on borderlands and i will kill you all,

Tweet ID: 2401
Content: im coming on borderlands and i will murder you all,

Tweet ID: 2401
Content: im getting on borderlands 2 and i will murder you me all,

Tweet ID: 2401
Content: im getting into borderlands and i can murder you all,

Tweet ID: 2402
Content: So I spent a few hours making something for fun. . . If you don't know I am a HUGE @Borderlands fan and Maya is one of my favorite characters. So I decided to mak
e myself a wallpaper for my PC. . Here is the original image versus the creation I made :) Enjoy! pic.twitter.com/mLsI5wf9Jg

Tweet ID: 2402
Content: So I spent a couple of hours doing something for fun... If you don't know that I'm a huge @ Borderlands fan and Maya is one of my favorite characters, I decided t
o make a wallpaper for my PC.. Here's the original picture compared to the creation I made:) Have fun! pic.twitter.com / mLsI5wf9Jg

Tweet ID: 2402
Content: So I spent a few hours doing something for fun... If you don't know I'm a HUGE @ Borderlands fan and Maya is one of my favorite characters.

Tweet ID: 2402
Content: So I spent a few hours making something for fun. . . If you don't know I am a HUGE RhandlerR fan and Maya is one of my favorite characters. So I decided to make m
yself a wallpaper for my PC. . Here is the original image versus the creation I made :) Enjoy! pic.twitter.com/mLsI5wf9Jg

Tweet ID: 2402
Content: 2010 So I spent a few hours making something for fun. . . If you don't know I am a HUGE RhandlerR fan and Maya is one of my favorite characters. So I decided to m
ake myself a wallpaper for my PC. . Here is the original image versus the creation I made :) Enjoy! pic.twitter.com/mLsI5wf9Jg
```

Figure 3 Datasets from Kaggle

3.2 Text Tokenization and Sequencing

In sentiment analysis using GRU, tokenization divides text into tokens mapped to numerical forms. These tokens are sequenced into fixed-length vectors, facilitating the model's understanding of text context and dependencies. Proper preprocessing enhances sentiment prediction accuracy by enabling GRU models to capture nuanced sentiment expressions effectively.

3.3 Padding

Padding is a crucial methodology to ensure that all sequences of tweets have a consistent and fixed length before feeding them into a GRU model for training or evaluation. It uses the Keras pad sequence's function. The sequences of tokenized words, representing the tweet 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 tweets 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

Splitting your data into training and testing sets is crucial in machine learning. This process involves dividing your dataset into two parts: the training set and the testing set. Typically, the training set contains 80% of your data. This larger portion is used to teach your GRU model by letting it learn from the examples provided. The remaining 20% forms the testing set, which is kept separate and used to assess how well your model performs.

This split is important because it allows you to see how your model generalizes to new, unseen data. By evaluating its performance on the testing set, you can gauge how accurately your model predicts sentiments in tweets that it hasn't encountered during training. This helps you understand whether your model can make reliable predictions in real-world scenarios.

3.5 GRU Model

The model architecture includes key components designed specifically for analyzing sentiment in short text messages.

Firstly, an Embedding layer is added to convert tokenized sequences of tweets into dense vectors. This transformation allows the model to understand the context and meaning of words within each tweet.

Next, we incorporate a GRU (Gated Recurrent Unit) layer. This layer is adept at capturing temporal dependencies and contextual information from sequences, making it effective for learning patterns and trends in tweet texts.

To enhance generalization and prevent overfitting, a Dropout layer is introduced. During training, this layer randomly deactivates a fraction of neurons, encouraging the model to generalize well to new, unseen tweets.

Finally, a Dense layer with a sigmoid activation function is appended at the output. This layer outputs a probability score indicating the likelihood of a tweet expressing positive sentiment. The sigmoid function ensures the output is between 0 and 1, providing a clear indication of the model's sentiment prediction confidence.

3.6 Performance Evaluation

First, predict the test dataset based on the trained model for sentiment analysis. This is done by applying the model's prediction method to test input data, X_test. Then, the predicted labels will be compared with their ground truth labels in the test dataset to generate a confusion matrix. The matrix contains the number of true positives, true negatives, false positives, and false negatives, which helps better understand the classification performance.

Many other metrics important for the evaluation process stem from the confusion matrix. Accuracy tells how much the model was correct overall, whereas precision quantifies the proportion of correctly predicted positive sentiments. Recall expresses how well the actual positive sentiments were captured by the model. The F1 score provides a balanced assessment of a model's effectiveness by harmonizing precision and recall.

In addition, specificity is considered. It is the proportion of true negatives, that is, the proportion of negative sentiments the model can identify correctly out of all actual negative instances. This gives more context about the performance of the model in sentiment analysis.

The loss metric, usually derived from the false positives and negatives, will tell one about the classification errors that have been made by the model. All these evaluation metrics provide a general understanding of how well the model differentiates among different sentiment classes.

Moreover, training and validation loss plots, accuracy, and other metrics along the epochs are very important and help in following the learning curve of the model to identify problems like overfitting. These plots provide more interpretability of model performance during training.

Chapter 4: Implementation

4.1 Implementation

The program is developed using Python (version 3.12.4) as the primary programming language within the VS Code Integrated Development Environment (IDE). Several essential libraries are employed:

i. **NumPy:** Utilized for efficient numerical operations and computations.
ii. **Pandas:** Employed for comprehensive data manipulation and analysis tasks.
iii. **Matplotlib:** Utilized for creating static, animated, and interactive visualizations in Python.
iv. **Seaborn:** Used for statistical data visualization, emphasizing attractive and informative statistical graphics.
v. **TensorFlow and Keras:** These frameworks are utilized for constructing, training, and deploying deep learning models.
vi. **Scikit-learn:** Employed for machine learning tasks including preprocessing, model evaluation, and metrics computation.

Together, these libraries enable robust development, analysis, visualization, and deployment of machine learning and deep learning solutions in Python.

4.2 Implementation details

The implementation details encompass the following aspects:

4.2.1Train–Test Split

The data set is split into two independent sets: a training set and a test set, in an 80:20 ratio 80% of the data goes into training and 20% into testing. This partition will ensure that the model has learned from a significant portion of the dataset and is tested on completely new, unseen data samples; that is, the evaluation helps check how well the model generalizes sentiments.

```
# Split the data into features (X) and target (y)
X = df['text']  # Features: text data
y = df['target']  # Target: numeric sentiment labels

# Perform train-test split with a ratio of 80:20
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Print the shapes of the training and testing sets to verify the split
print("Shape of X_train:", X_train.shape)
print("Shape of X_test:", X_test.shape)
print("Shape of y_train:", y_train.shape)
print("Shape of y_test:", y_test.shape)
```

Figure 4 Perform train-test split with a ratio of 80:20

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 300 words (SEQUENCE_LENGTH). This step ensured that all sequences had consistent dimensions for efficient processing.

```
# Parameters
# Fixed length for padding
SEQUENCE_LENGTH = 300

# Tokenization
tokenizer = Tokenizer()
tokenizer.fit_on_texts(texts)
sequences = tokenizer.texts_to_sequences(texts)

# Padding sequences
padded_sequences = pad_sequences(sequences, maxlen=SEQUENCE_LENGTH,
                                 padding='post', truncating='post')
```

Figure 5 Code for Sequencing and padding

4.2.3 GRU Architecture

The neural network architecture for sentiment analysis starts with an Embedding layer, followed by a GRU (Gated Recurrent Unit) layer with 64 units to process the sequence data and capture temporal dependencies, producing a fixed-length output sequence. To mitigate overfitting, a Dropout layer with a rate of 0.2 is included, which randomly sets a portion of the input units to zero during training. The final layer is a Dense layer with a single neuron and a sigmoid activation function, providing the output for binary classification.

```
# Build GRU model
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=embedding_dim, input_length=SEQUENCE_LENGTH))
model.add(GRU(units=gru_units, dropout=0.2, recurrent_dropout=0.2))
# Sigmoid activation for binary classification
model.add(Dense(1, activation='sigmoid'))
```

Figure 6 Code for GRU Architecture

4.2.4 Model Training and Adam Optimizer

Before training, the model is compiled by specifying the optimizer, loss function, and evaluation metrics. The Adam optimizer is chosen for its ability to adapt learning rates for each parameter, which can enhance convergence speed. Binary cross-entropy is selected as the loss function, and accuracy is used to evaluate performance. The model is trained over a set number of epochs using mini-batches of data with a batch size of 512. A validation dataset is utilized to monitor performance. During training, the Adam optimizer updates the model's weights based on the gradients of the loss function, helping the model learn to distinguish between positive and negative sentiments.

```
# Compile model with Adam optimizer
# set learning rate
optimizer = Adam(learning_rate=1e-4)
model.compile(optimizer=optimizer, loss='binary_crossentropy', metrics=['accuracy'])
```

Figure 7 Code For compiling the model

```
# Define early stopping to prevent overfitting
early_stopping = EarlyStopping(monitor='val_loss', patience=3, restore_best_weights=True)
```

Figure 8 Code For early stopping to prevent overfitting

4.2.5 Model Evaluation

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

```
# Calculate additional metrics
precision = precision_score(y_test_labels, y_pred, average='weighted')
recall = recall_score(y_test_labels, y_pred, average='weighted')
f1 = f1_score(y_test_labels, y_pred, average='weighted')
# Specificity calculation for each class
cm = confusion_matrix(y_test_labels, y_pred)
specificity = np.diag(cm) / np.sum(cm, axis=1)

print(f'Precision: {precision:.4f}')
print(f'Recall: {recall:.4f}')
print(f'F1-score: {f1:.4f}')
print(f'Specificity: {specificity.mean():.4f}')
```

Figure 9 Code for Model Evaluation

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.

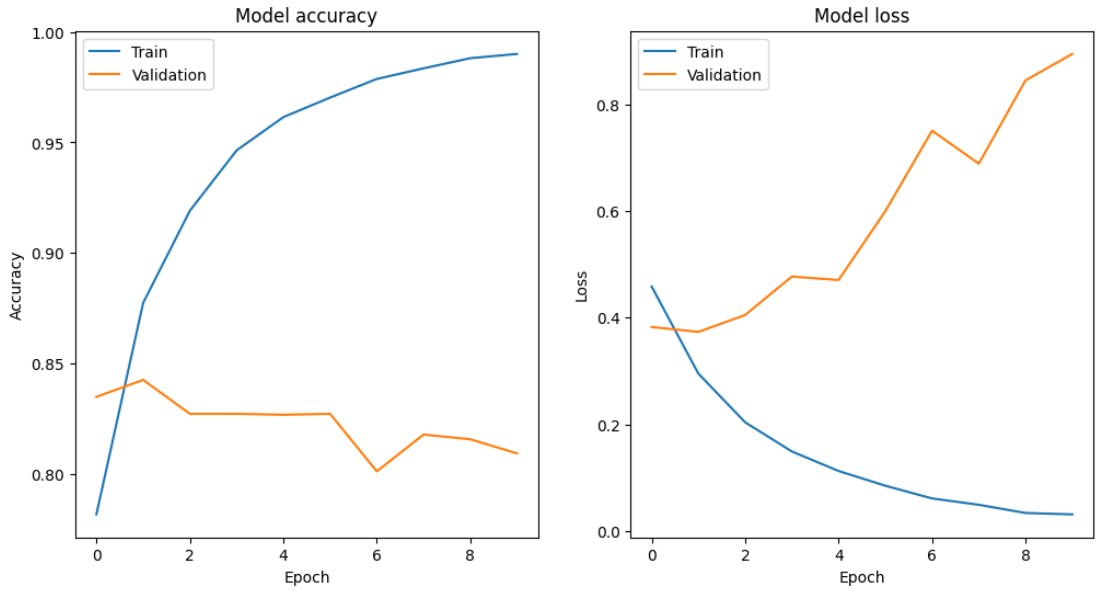

Figure 10 Training and Validation Loss and accuracy plots

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, accurately recognizing them as positive sentiments. It also correctly classified 3138 instances as true negatives, signifying its ability to identify negative sentiment correctly. 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

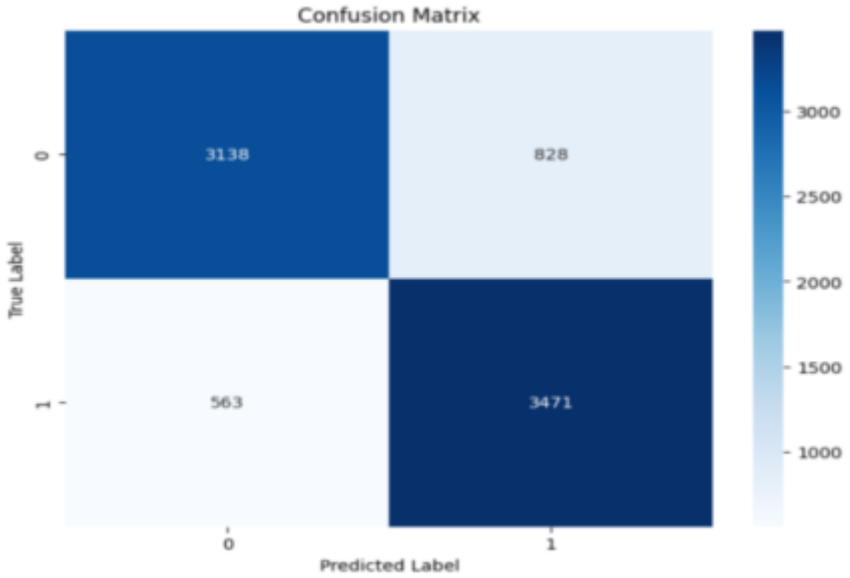

Figure 11 Confusion Matrix

The model is performing well, with high accuracy, precision, F1 score, specificity, and recall. The loss is also relatively low. This suggests that the model can 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 the GRU model demonstrated moderate performance in distinguishing between positive and negative sentiments. After training the model for 5 epochs with a batch size of 64, the following evaluation metrics were obtained on the entire dataset:

- Accuracy: 82.61%
- Precision: 80.84%
- Recall: 85.53%
- F1 Score: 79.66%
- Specificity: 80.33%

These metrics indicate that while the model achieved high accuracy in correctly classifying sentiments as positive or negative, the precision score reflects a lower ability to correctly identify positive and negative sentiments. The recall score indicates its moderate ability to identify all the actual positive sentiments present in the dataset, capturing some of the true positive sentiments. The F1 score represents the model's balance between precision and recall, reflecting its ability to achieve both accurate positive sentiment identification and comprehensive capturing of actual positive sentiments, albeit with room for improvement. The specificity score highlights the model's skill in identifying true negatives (negative sentiments) within the dataset, ensuring that a portion of the negative sentiments are correctly classified.

Chapter 6: Conclusion and Future Recommendations

6.1 Conclusion

This implementation of sentiment analysis for tweet reviews leverages the Gated Recurrent Unit (GRU) architecture. The GRU-based model excels at understanding language nuances and distinguishing between positive and negative sentiments. It achieves an accuracy of 82.61%, precision of 80.84%, recall of 85.53%, F1 score of 79.66%, and specificity of 80.33%, demonstrating strong performance. The use of insightful visualizations, comprehensive metrics, and the efficient Adam optimizer underscores its effectiveness in text classification tasks. This GRU-based model proves to be a powerful tool for handling sequential data, making it highly suitable for sentiment analysis and other natural language processing applications.

6.2 Future Recommendation

To enhance future performance, consider optimizing hyperparameters by experimenting with different numbers of GRU units and learning rates. Diversify the training data through data augmentation techniques. Integrate attention mechanisms for more nuanced text analysis. Utilize transfer learning with BERT to better handle complex language structures. Combine multiple models using ensemble methods to boost performance. Test the model's adaptability across different domains through domain adaptation. Assess the model’s effectiveness in real time scenarios like social media monitoring. Develop methods to explain the model’s predictions, enhancing trust and transparency. These approaches can help create a more robust, accurate, and reliable sentiment analysis model.

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] 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.

[4] P. Acharya and B. K. Bal, *A Comparative Study of SMT and NMT: Case Study of English-Nepali Language Pair*. 2018. doi: 10.21437/SLTU.2018-19.

[5] B. Krishnamurthy, S. Senthilkumar, A. Singh, and P. Sharma, *Sentiment Analysis with NLP on Twitter Data*. 2022. doi: 10.1109/SMARTGENCON56628.2022.10084036.

[6] S. Saini, R. Punhani, R. Bathla, and V. Shukla, *Sentiment Analysis on Twitter Data using R*. 2019. doi: 10.1109/ICACTM.2019.8776685.