Democratic and Popular Republic of Algeria
Ministry of Higher Education and Scientific Research
University Centre of Illizi
Institute of Science

Department of Computer Science

Artificial Intelligence & its Applications Master's Thesis

Text-Based Emotion Detection using RNN

Authors:

- CHEKAIEM Abdelfattah
- MAATALLAH Abdallah

Advisors:

- Mr. Messikh Chouaib
  messikh.chouaib@cuillizi.dz

Academic Year: 2025

Abstract

The availability of vast amounts of multimedia data and its widespread applications, along with automatic sentiment analysis and sentiment classification in text, has become a compelling research topic among researchers. The state of the text and the contextual state between adjacent phrases play a significant role in analyzing and detecting sentiments in text. In this study, a method based on Recurrent Neural Networks (RNNs) was developed to capture the textual state and contextual relationships between phrases, employing layers such as LSTM, GRU, and an attention mechanism. First, textual data is passed to an LSTM layer, then to an attention mechanism and a GRU layer. Finally, their outputs are integrated to extract sentiments. The proposed model is evaluated using two metrics: Accuracy and F1-score. The numerical results demonstrate that the proposed method achieves outstanding performance in detecting sentiments from text, with a detection rate of 97.22% and a classification accuracy of 98.02%.

Key words: Emotion detection , Machine Learning, Sentiment Analysis, RNN, LSTM, GRU, Attention.

Acknowledgements

Indeed, all merit, praise, and gratitude are due to Allah Almighty for granting us success in completing this work. We ask Allah, Glorified is He, to make this knowledge beneficial in this world and for us in the Hereafter.
We send prayers and peace upon the noblest of Messengers, our master and beloved Muhammad (peace and blessings be upon him), and upon his family and all his companions.
Indeed, acknowledging kindness is but a small part of repaying it.
We specifically mention our supervisor Mr. Messaikh Chouaib, who generously provided his advice and guidance to facilitate our work within the framework of this research, as words are all we possess to express our gratitude towards those who have showered us with kindness. To you, we offer our highest expressions of appreciation and respect.
We also do not forget to extend our sincerest expressions of gratitude to all our companions on this journey, including professors and fellow students, especially those in the master's program in informatics.
We extend our heartfelt gratitude to everyone who helped us complete this work, whether it was with a prayer or a word of encouragement. We also extend our gratitude to everyone who wished to see our work become tangible. We also extend our gratitude to everyone who expressed satisfaction with our achievements.
And all thanks are due to Allah, from the beginning to the end.

Table of Contents

List of Figures

List of Tables

Chapter 1

General Introduction

In today's digital age, the textual data available online has become vast and diverse, leading to the emergence of new challenges and opportunities in the field of Natural Language Processing (NLP). Sentiment analysis through text is one of the important applications that helps in understanding users' opinions and attitudes towards various topics.
The Recurrent Neural Network (RNN) model is considered one of the effective tools in this field due to its capabilities in processing sequential data and understanding temporal contexts within texts. [1]

1.1 Problem Statement

There are some challenges that need to be overcome to improve the accuracy of sentiment analysis, such as understanding the context of the text, dealing with complex expressions, and neutralizing linguistic noise.
Through this study, we seek to propose a model based on RNN that is capable of extracting sentiments more accurately and analyzing the factors influencing the improvement of the model's performance.

1.2 Objectives and Contributions

In this study, we aspire to achieve the following objectives:

- Propose a model based on RNN for sentiment analysis through texts.
- Improve the model's performance through advanced techniques such as LSTM, GRU, and Attention.
- Develop a practical framework that can be used in real-world applications such as analyzing social media tweets or customer reviews.

1.3 Methodology

This study relies on an experimental methodology through which we go through several stages. First we need to do data collection and processing. Afterwards, we build a RNN model. Then, we test it after training it with multiple datasets. Finally, the model's performance is evaluated using several metrics such as classification accuracy and the F1-score.

1.4 Thesis Structure

The rest of the report is divided into 5 chapters. In the first chapter, we review the concept of emotion and how to detect and analyze it from text. In the second chapter, we focus on Recurrent Neural Networks and how they generally work. As for the third chapter, we address some of the studies conducted on sentiment analysis from text and review the most important results they reached. In the fourth chapter, we discuss the steps taken to complete the RNN-based sentiment analysis model. Finally, in the fifth chapter, we discuss the most important results obtained during the training and testing of the proposed model.

Chapter 2

Emotion Detection From Text

2.1 Introduction

Affective computing is a branch of artificial intelligence that focuses on understanding and managing human emotions through various types of data. This chapter specifically concentrates on textual data as a primary medium for emotion analysis.[2]
The chapter explores several aspects of textual emotion recognition. It begins by defining the concepts of emotion and emotion detection, followed by an overview of models and techniques used for sentiment analysis in text.
Finally, it discusses key challenges associated with emotion detection in textual content.

2.2 Emotions

To this day, there is no precise, universally agreed-upon definition of emotion. However, several definitions have been proposed, including:

- Emotion is a subjective feeling state characterized by the emergence of intense and focused feelings often directed toward a specific object or situation. [3]
- Emotion is a complex psychological state that involves three distinct components: a subjective experience, a physiological response, and a behavioral or expressive response. [4] ‘
- Emotions are brief, involuntary, patterned responses to external or internal events that are relevant to the goals of the organism.[5]

The term ”emotion” refers to a series of responses triggered by specific brain regions, involving communication between the brain and the body as well as interactions among different brain areas. These responses engage both neural and humoral systems. The culmination of these processes results in an emotional state characterized by physiological and behavioral changes. [6]

2.3 Emotion Detection

Emotions are a means of expression for humans. A significant NLP challenge is emotion detection and recognition. Closely related to Sentiment Analysis, text-based emotion detection (ED) is a relatively new area of study. While Emotion Analysis seeks to identify and acknowledge kinds of emotions through the expression of text—such as anger, contempt, fear, happiness, sadness, and surprise—Sentiment Analysis seeks to identify positive, neutral, or negative emotions.[7]

2.4 Emotion Models

In order to build ED systems, a knowledge of what and how many types of emotions are there is required. This knowledge is referred as *the emotion model*. Many models have been suggested in order to explain how we express feelings. Discrete emotion models and dimensional emotion models are two major types of emotion modeling.

2.4.1 Discrete Emotion Models

The discrete model of emotions is based on distinct emotion classes or categories, among these models:

- The Paul Ekman model classifies emotions into six fundamental groups. It argues that six (06) fundamental emotions emerge from distinct brain systems, shaped by an individual’s perception of a situation, rendering emotions autonomous. The fundamental emotions include: Joy, sadness, anger, disgust, surprise, and fear. The amalgamation of these sentiments can

yield more intricate emotions, including guilt, shame, pride, lust, and greed.[8]

- Similar to Paul Ekman's theory, Robert Plutchik's model proposes that a limited number of basic emotions exist in opposing pairs and can blend to form more complex emotional states. In addition to Ekman's six basic emotions, Plutchik identified eight primary emotions: joy, sadness, anger, fear, trust, disgust, anticipation, and surprise. These emotions are organized into four opposing pairs: joy versus sadness, trust versus disgust, anger versus fear, and surprise versus anticipation. Plutchik also suggests that each emotion can vary in intensity, depending on an individual's interpretation of events or experiences.[9]
- The model proposed by Ortony, Clore, and Collins challenges the traditional views of basic emotions as presented by Ekman and Plutchik. However, it concurs with them that emotions arise based on individuals' interpretations of events and vary in intensity. This model offers a broader and more comprehensive classification, identifying 22 distinct types of emotions, thereby extending and complementing the concept of basic emotions. These emotions include relief, envy, blame, self-blame, appreciation, shame, compassion, disappointment, admiration, hope, fear-confirmed, sadness, satisfaction, gloating, and hatred. [10]

2.4.2 Dimensional Emotion Models

The dimensional model holds that emotions are interrelated, not independent, therefore the need to organize them in a spatial realm.
Therefore, dimensional models depict the two main fundamental behavioral states of good and bad by placing emotions on a dimensional space, either an uni-dimensional (i.e., 1-D), or multidimensional (i.e., 2-D and 3-D), to indicate how related emotions are. The degrees of occurrence, from low to high, for both the uni-dimentional and multidimensional spaces are affected. Though seldom used, the fundamental idea of the uni-dimensional models permeates most multidimensional models.

- **Russell's Circumplex Model :**Significant in dimensional emotion representation, Russell presents the circumplex of affect, a circular two-dimensional framework. The paradigm separates emotions in the Arousal and Valence domains; Arousal does so by Activation and Deactivation, while Valence does so by Pleasantness and Unpleasantness.Emotions,

says the Circumplex model of Affect, are linked rather than separate. Rusell's model is shown in Figure 2.1 [11]

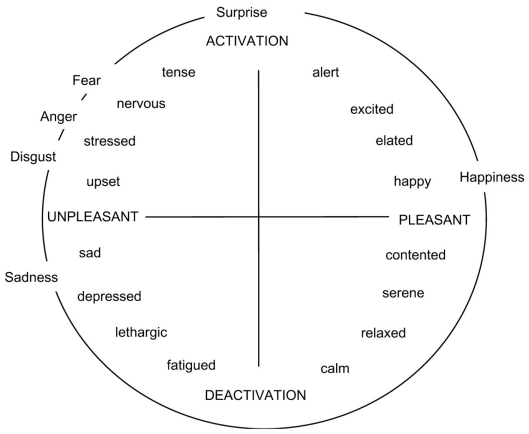


Figure 2.1: Russel's emotion model

- **PAD Model:** Based on three primary dimensions—pleasure, arousal, and dominance—the PAD (Pleasure-Arousal-Dominance) model is a psychological tool for describing emotional states. Created in 1974 by Albert Mehrabian and James Russell, it finds application in nonverbal communication, environmental psychology, and marketing. Used to gauge people's emotional reactions in many contexts, the PAD model helps to clarify how outside elements affect emotions and actions. . [12]

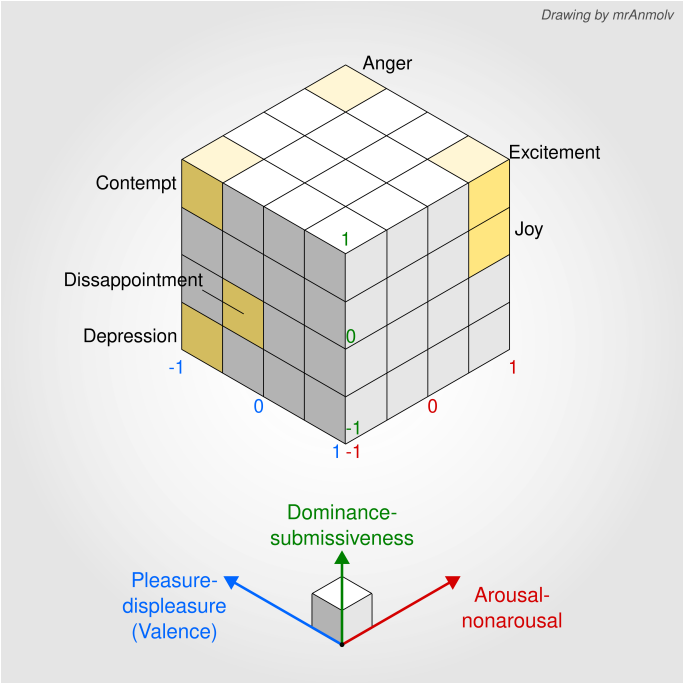


Figure 2.2: Visual representation of the PAD model

- **Core Affect Model :** A psychological tool called the Core Affect Model explains the fundamental emotions people feel devoid of clear outside influence. Two primary dimensions—pleasure and arousal—define this model. The Circumplex Model, a circular system, illustrates these dimensions and distributes distinct emotions based on their interaction. For instance, while tranquillity is a mix of high pleasure and low arousal, happiness might be a mix of high pleasure and moderate arousal.Created by James Russell, this model is applied in areas such as affective psychology, marketing, and human-machine interactions to grasp how emotions shape behaviour and decision-making.[13]

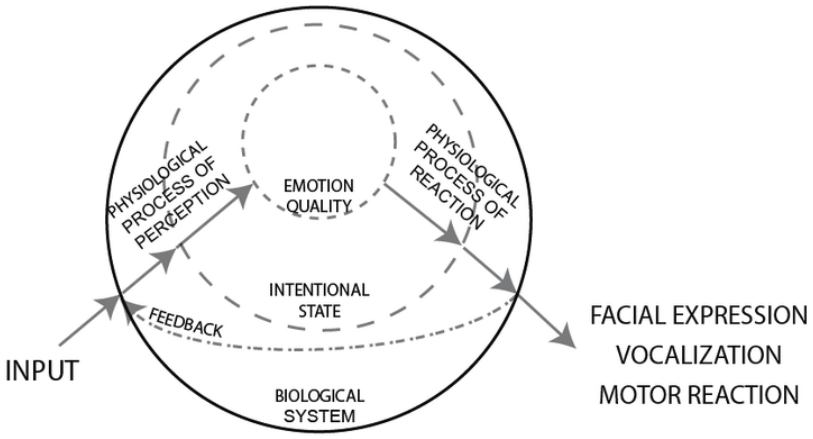


Figure 2.3: Visual representation of the Core Affect Model

2.5 Emotion Detection From Text

Currently, individuals prefer expressing their thoughts, opinions, sentiments, and emotions publicly. Social media provide a platform for expressing opinions through concise prose, such as tweets or posts. This paragraph may have indicators of emotion. Social media serves as a valuable repository of emotional text data, making it an excellent resource for behavioral studies concerning individual and collective emotions.
Emotion detection systems in social media will prove effective in various applications. ED technologies will significantly influence individuals and societal issues, such as mental health counseling via social media and polling organizations that utilize emotional data to gauge public sentiment on topics, enabling prompt responses. [14]

2.5.1 Challenges of Emotion Detection From Text

In the realm of emotion recognition in text, it is imperative to highlight several obstacles that arise during the process. All these issues pertain to the characteristics of textual data. If there were words that clearly convey the emotion, it would be straightforward; however, emotions often express themselves indirectly and ambiguously. The majority of text postings and messages are composed in an informal manner. Certain textual data encompass multiple emotions, while others exhibit spelling inaccuracies, grammatical flaws, dialectal variations, or incorporate more than one language inside a single phrase. The multitude of themes and emotional states complicates the manual creation of a tagged corpus that encompasses all ED cases. [15]

If we want to use a supervised method for detecting emotions by training a computer program with data that has been labelled by hand, this process will be difficult because of the many different themes and emotions involved. A last problem is the constancy of emotional annotation, as human annotators exhibit unreliability due to subjective judgement. Consequently, many annotators will categorise emotions into distinct emotion classes..
[15]

2.5.2 Approaches Used for Emotion Detection From Text

Textual emotions can be detected in many different ways. In fact, the computational techniques used in this field are part of affective computing, of which emotion detection is a subcategory.
Many studies have divided the field into many categories. The approaches of these studies have been specified and might be categorized into four categories: keyword-based techniques, n-based techniques, machine learning techniques, and the hybrid approach. [16]

Machine Learning-Based Methods

Both supervised and unsupervised machine learning techniques are used for textual emotion recognition; a model meant to train a classifier on a subset of the dataset then tests the classifier using the rest of the dataset. The supervised approach trains and tests the supervised classifier using an annotated emotion dataset.
Naive Bayes, Support Vector Machine, and Decision Tree are the three most commonly used classifiers. According to the definition, the unsupervised classification uses unlabeled data with the classes. Starting with many seed words for each emotion, the classifier then connects them to sentences.
In this way, sentences are assigned to emotions. This trains the classifier model, which is then used to label the test data. Although the unsupervised approach is more generalized, supervised classification often produces greater accuracy. [17]

Lexicon-Based Methods

Among the semantic analysis techniques is the lexicon-based approach. The semantic orientation of lexicons determines the emotional orientation of a whole paper or collection of sentences. Lexicon dictionaries can be created manually or automatically. Many studies make use of the WorldNet dictionary. Lexicons are found all over the paper to start with; then WorldNet or another kind of online thesaurus can be used to find antonyms and synonyms and enlarge that vocabulary. Among these methods, we may discover:

1. **keyword-based methods** The keyword-based emotion detection method is the simplest and most intuitive. Finding and matching patterns that are similar to emotion keywords is the aim. Finding the word in a sentence that best expresses the emotion is the first objective.
   This is often achieved by first tagging the words in a phrase with a parts-of-speech tagger and then extracting the Noun, Verb, Adjective, and Adverb (NAVA) keywords. These are the most likely words to convey emotion, according to the majority of linguistic and emotion-based research. FNext, we compare these items to a list of terms that, based on a specific emotion model, represent various emotions.Whichever emotion best fits the term is the emotion of that particular sentence.
   Different strategies could be applied when a word fits several feelings on the list. Every word in certain keyword dictionaries has a likelihood score for every emotion; the emotion with the highest probability score is selected as the word's emotion. Other studies select the first emotion

associated with the word as the dominant one. The keyword dictionary or keyword reference list differs depending on the researcher. Usually, researchers create keyword dictionaries depending on emotions and the related terms. [18]

2. **Corpus-based methods** This determines the sentiment direction of context-specific terms. This strategy has two approaches:
   - The statistical approach determines the positive polarity of phrases that display unpredictable behaviour in positive contexts. They have negative polarity if their negative text shows negative recurrence. The term has neutral polarity if the frequency is the same in both positive and negative text.
   - The semantic method gives words and words semantically close to those words sentiment values; this can be done by finding antonyms and synonyms for that word.
3. **Hybrid based methods** This approach integrates machine learning-based implementation with keyword-based execution. By teaching a mix of classifiers and using detailed language information from dictionaries and thesauri, this method can produce more accurate results. The method has the benefit of balancing the great cost of employing human indexers for information retrieval tasks and reducing the complexity experienced when merging several lexical resources.s. [19]

2.6 Conclusion

This chapter covered text data-based emotion detection (ED). We began by clarifying the notions of emotion and ED and their respective models. Furthermore, we discussed text-based emotion recognition, apart from certain difficulties for ED. We also examined various methods for text-based emotion detection (ED).

Chapter 3

Recurrent neural networks (RNN).

3.1 Introduction

The field of artificial intelligence has witnessed remarkable developments in recent decades, surpassing the traditional boundaries of computing. Among the most prominent of these developments is the emergence of recurrent neural networks, one of the cornerstones of artificial intelligence and deep learning. Their superior ability to process sequential data, whether text, audio, or organized images, has made them important in a number of advanced applications such as machine translation, text analysis, speech recognition, and other fields.
In this chapter, we will review what a neural network is, how they work, the types of neural networks, their most prominent applications, and the benefits and challenges associated with them. [20]

3.2 Definition

Recurrent neural networks (RNNs) have been a major focus of research and development since the last century. They were designed to learn sequential or time-varying patterns. They are neural networks with feedback connections. Examples include BAM networks, Hopfield machines, Boltzmann machines, and recurrent backpropagation networks. RNN techniques have been applied to a wide range of problems, such as those related to dynamical systems with time sequences of events. [21]
Recurrent neural networks (RNNs) are used to predict daily flood levels based on past daily flood, tide, and meteorological data. They are also used to solve sequential or temporal problems such as language translation, natural language processing (NLP), sentiment analysis, speech recognition, image translation, and more. [22]
Another definition of a recurrent neural network (RNN) is that it is a neural network that simulates a discrete-time dynamic system, with an input x_t, an output y_t, and a hidden state h_t, where the subscript t represents time. The dynamic system is defined by [23] :

$$\begin{aligned}\mathbf{h}_t &= f_h(\mathbf{x}_t, \mathbf{h}_{t-1}) \\ \mathbf{y}_t &= f_o(\mathbf{h}_t)\end{aligned}$$

- f_h: state transition function
- f_o: output function

The parameters of each function are defined by a set of parameters: θ_h and θ_o.

3.3 Traditional Recurrent Neural Network

A traditional RNN is built by defining a transfer function and an output function, as in the following equations [24]:

$$\begin{aligned}\mathbf{h}_t &= f_h(\mathbf{x}_t, \mathbf{h}_{t-1}) = \phi_h(\mathbf{W}^\top \mathbf{h}_{t-1} + \mathbf{U}^\top \mathbf{x}_t) \\ \mathbf{y}_t &= f_o(\mathbf{h}_t, \mathbf{x}_t) = \phi_o(\mathbf{V}^\top \mathbf{h}_t)\end{aligned}$$

- W is the transition matrices.
- U is the input matrix.
- V is the output matrix.
- ϕ_h and ϕ_o are nonlinear functions applied element-wise.

An illustrative example of this recurrent neural network is shown in Figure 3.1

(a) A conventional RNN.

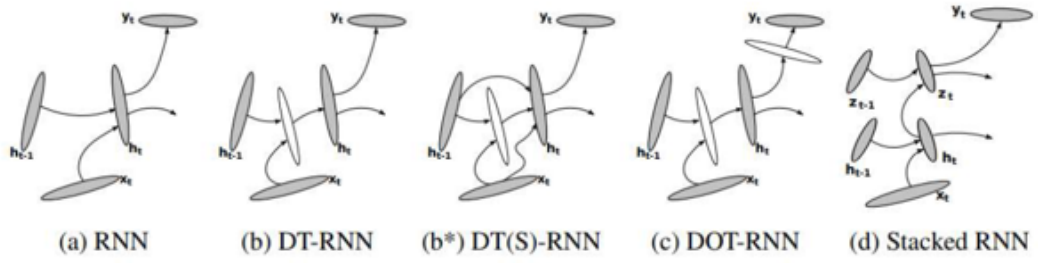


Figure 3.1: Illustrations of different recurrent neural networks (RNN)

(b) Deep Transition Recurrent Neural Network (DT-RNN).
(b*) Deep Transition Recurrent Neural Network with Shortcut Connections (DTS-RNN).
(c) Deep Transition, Deep Output Recurrent Neural Network (DOT-RNN).
(d) Stacked RNN.

3.4 Recurrent Neural Network Architectures

There are different types of architectures, such as fully connected and partially connected networks, including multilayer-feedforward networks with distinct input and output layers.[25]

- **Fully Connected Networks :** model describes an architecture in which every neurone in a given layer links to every neurone in the following layer. For an RNN, this kind of connection allows information to be transported from one time step to the next by means of reverse or temporal connections within the same layer as well as forward layers (like the output layer). Although this design incurs a significant computational cost and increases

the likelihood of overfitting without appropriate regularisation methods, it enhances the model's ability to learn complex patterns in sequential data.[26]

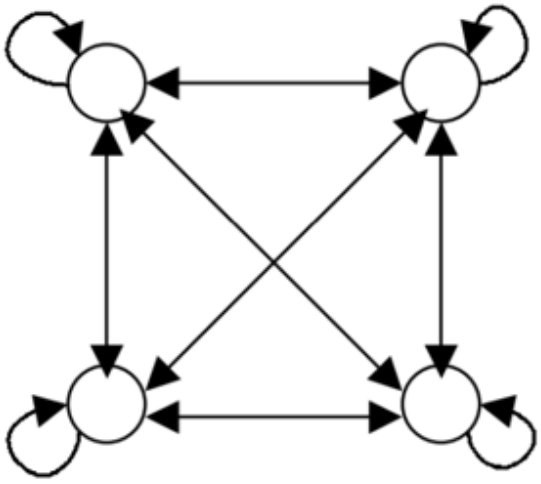


Figure 3.2: Fully connected networks

- **Partially Connected Networks:**
  Unlike the first model, this architecture lacks complete connectivity among all neurones. Rather, some neurones' connections are limited depending on particular qualities, such as temporal distance or statistical relevance. Aiming to lower the number of learnable parameters, this approach helps to lower computational complexity and enhance general performance, particularly under enormous data loads or in the presence of noisy data.[26]

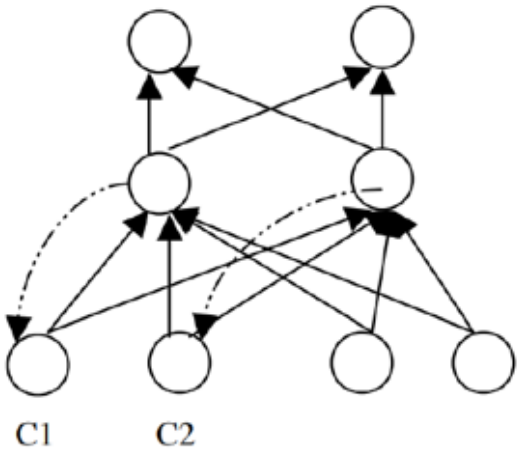


Figure 3.3: Partially Connected Networks

Simple partially recurrent neural networks are used to learn character strings. The weights of the context units (C1 and C2) are processed in the same way as the input units. The context units receive time-delayed feedback from the second-layer units. The training data consists of the inputs and their subsequent desired outputs [26]

3.5 Learning in Recurrent Neural Network

Learning is a fundamental aspect of neural networks, and it is a major feature that makes the neural network approach so attractive for applications that have been an elusive goal for artificial intelligence from the beginning. Learning algorithms have long been a focus of research.
Hebbian learning : This concept in neuroscience explains how coupled neurones that are frequently triggered strengthen their connections. Often referred to as the "neurones that fire together, wire together" principle, this idea is critical for understanding how the brain acquires knowledge and creates memories.[27] **Gradient Descent :**Used in machine learning, Gradient Descent is an optimisation method that updates parameters and progressively reduces the cost function. This approach, which helps models perform better, depends on computing derivatives to find the best direction for changing values.[28]
Hebbian learning and gradient descent learning are key concepts upon which neural network techniques have been based. A popular manifestation of gradient descent is back-error propagation introduced by [28][27]. While backpropagation is relatively simple to implement, several problems can occur in its use in practical applications, including the difficulty of avoiding entrapment in local minima. The added complexity of the dynamical processing in recurrent neural networks from the time-delayed updating of the input data requires more complex algorithms for representing the learning
Researchers have developed a variety of schemes that extend gradient methods, particularly backpropagation learning, to recurrent neural networks. In 1990, Werbus introduced the time-regression backpropagation approach, which models the time evolution of a recurrent neural network as a series of static networks using gradient methods. In 1986, Lapides and Farber used another model that deploys a second main neural network to perform the computations necessary to program the attractors of the original dynamic subnetwork. [26]

3.6 Types of Recurrent Neural Networks

Different types of recurrent neural networks are used for different use cases, such as sentiment classification. Common variations of recurrent neural network architectures include:

3.6.1 Standard Recurrent Neural Networks

These are good for tasks that process sequential data in real time and have difficulty learning long-term dependencies. The network consists of neurons that operate over time. At each point in time, the network receives new inputs, modifies the hidden state, and predicts the outcome or makes a decision at each point in time using this hidden state.[29]

If x_t is the input at time step t, and h_t is the hidden state, the update is computed as follows:

$$h_t = \tanh(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$
$$y_t = W_{hy}h_t + b_y$$

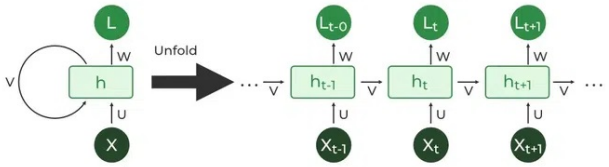


Figure 3.4: Standard Recurrent Neural Networks

3.6.2 Bidirectional Recurrent Neural Networks (BRRNs)

An extension of conventional RNNs meant to handle sequential data in both forward and backward directions, a Bidirectional Recurrent Neural Network (BRNN) allows the network to use both past and future context when generating forecasts. [30]
Moving in a forward direction, it updates the hidden state dependent on the current input and the prior hidden state at each time step, functioning like a standard RNN. On the other hand, the backward hidden layer examines the input sequence in the reverse direction, updating the hidden state depending on the present input and the hidden state of the next time step. [30]
BRNN's accuracy is better than that of traditional unidirectional recurrent neural networks since it can handle information both ways and consider both past and future settings. PPredictions are based on the combined outputs of the two hidden layers; therefore, these layers can complement each other. layers [30]
For instance, I enjoy apples. It is excellent for you.

Here, if we utilised a standard RNN, the first line would confuse what we are referring to, fruit or company. But, as it operates by moving in both ways, BRNN will find its context readily.

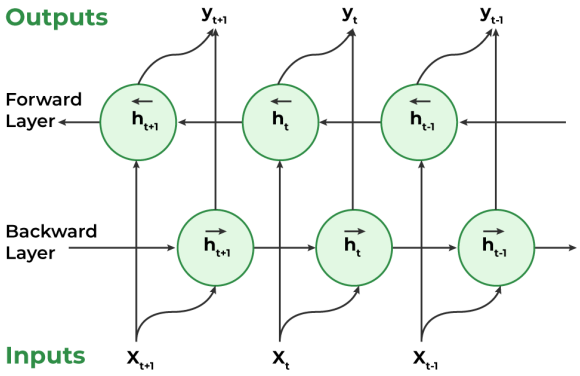


Figure 3.5: Bi-directional Recurrent Neural Network

Working of Bidirectional Recurrent Neural Network

- **Inputting a Sequence:** A sequence of data points, each represented as a vector with the same dimensionality, is fed into a BRNN. The sequence may vary in length.
- **Dual Processing:** Both the forward and backward directions are used to process the data. The hidden state at time step t is computed as follows:
  - **Forward direction:** Based on the input at step t and the hidden state at step $t-1$.
  - **Backward direction:** Based on the input at step t and the hidden state at step $t+1$.
- **Computing the Hidden State:** A non-linear activation function (such as `tanh` or `ReLU`) is applied to the weighted sum of the input and the previous hidden state. This acts as a memory mechanism, enabling the network to retain information from earlier steps in the sequence.
- **Determining the Output:** A non-linear activation function is applied to the weighted sum of the hidden state and output weights to compute the output at each step. This output can be:
  - The final output of the network.
  - An input to another layer for further processing.

3.6.3 Long Short-Term Memory (LSTM)

This is a recurrent neural network architecture introduced by Sepp Hochreiter and Juergen Schmidhuber as a solution to the vanishing gradient problem. This work addressed the problem of long-term dependencies. LSTMs contain ”cells” in the hidden layers of the neural network, which contain three gates: an input gate, an output gate, and a forget gate. These gates control the flow of information needed to predict the output in the network.[31]

Working of LSTM :

LSTM architecture has a chain structure that contains four neural networks and different memory blocks called cells.

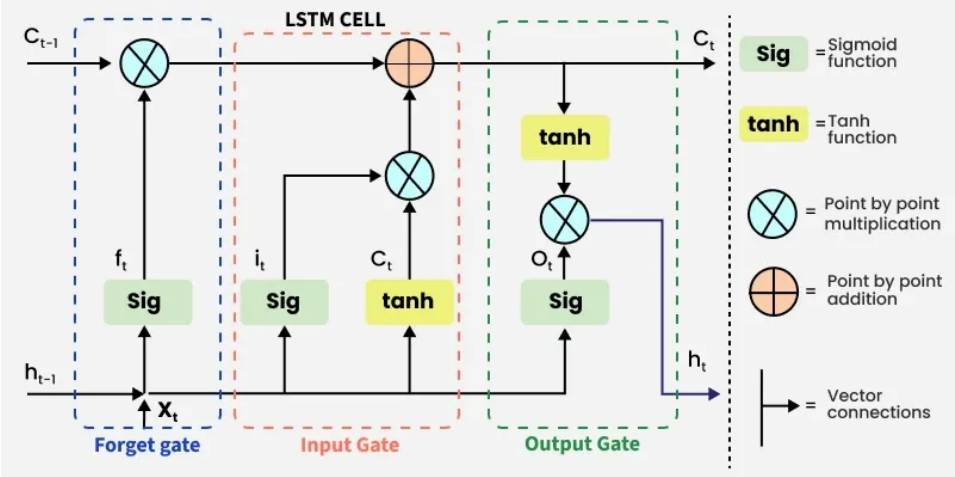


Figure 3.6: LSTM Model

Cells store information, while gates process memory. There are three gates: the input gate, the output gate, and the forget gate.[31]

3.6.4 Gated Recurrent Units (GRUs)

Similar to LSTMs, they use hidden states instead of "cell states" to organize information. They have two gates: a reset gate and an update gate. Like the gates within LSTMs, the reset and update gates control how much information is retained. GRUs are computationally more efficient and require fewer parameters compared to LSTMs due to their simpler architecture, making them faster to train. [32]
These gates allow the GRU to control the flow of information in a more efficient manner compared to traditional Recurrent Neural Networks (RNNs), which solely rely on the hidden state without such gating mechanisms.[32]

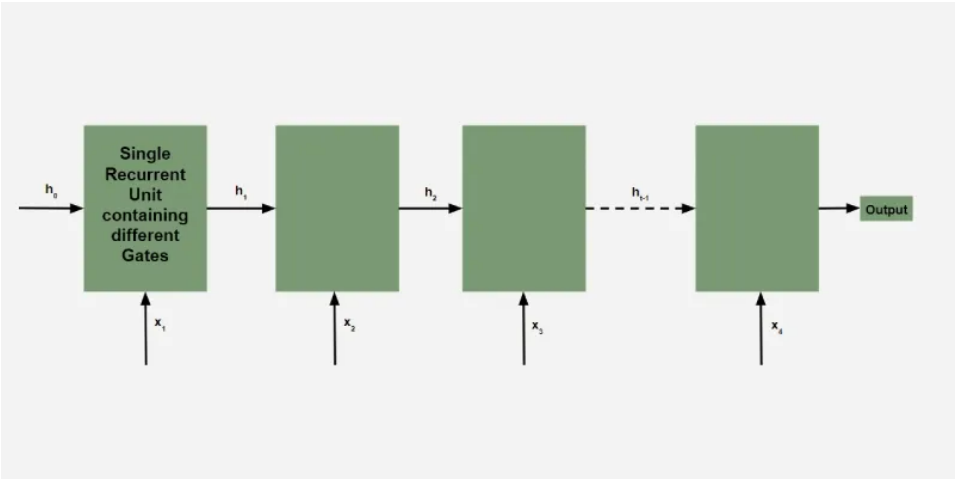


Figure 3.7: GRUs Model

The Gated Recurrent Unit (GRU) consists of two main gates:

- **Update Gate** (z_t): This gate decides how much information from the previous hidden state should be retained for the next time step.

- **Reset Gate** (r_t): This gate determines how much of the past hidden state should be forgotten.[33]

3.7 Limitations of Recurrent Neural Networks

The use of recurrent neural networks in artificial intelligence has declined in favor of architectures such as transformer models (BERT and GPT). These were popular for processing sequential data due to their ability to handle temporal dependencies.
However, recurrent neural networks are still used in specific contexts where their sequential nature and memory mechanism can be useful, especially in smaller, resource-constrained environments or for tasks where data processing benefits from incremental iteration.
For those who wish to define their own recurrent neural network cell layer with custom behavior, Keras, an open-source library now integrated into TensorFlow, provides a Python interface for recurrent neural networks. The API is designed for ease of use and customization. [34]

3.8 Conclusion

Recurrent Neural Networks (RNNs) are a cornerstone of deep learning, specifically engineered for sequential data. Their recurrent architecture enables them to retain and utilize past information, making them crucial for natural language processing, time series analysis, and speech recognition. RNN structures have evolved from basic forms to Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), which addressed the vanishing gradient problem and demonstrated a strong ability to capture long-range dependencies, leading to significant progress in various applications.

Chapter 4

Related Work

4.1 Introduction

In this chapter, we present and analyze previous studies related to our topic, with the aim of understanding the research efforts that have addressed this field and identifying research gaps that can be leveraged to make new contributions.
By examining the available literature, we will focus on the most prominent theories and concepts related to the topic, in addition to the research methods used and the most important findings of previous studies.

4.2 Attention-based Multi-modal Sentiment Analysis and Emotion Detection in Conversation using RNN

This study [35] presents a model based on RNN and attention mechanisms for sentiment analysis and emotion recognition in conversations using multi-modal data such as text (using CNN and GloVe). Experiments were conducted on three standard datasets: IEMOCAP, CMU-MOSEI, and CMU-MOSI. A binary attention-based technique was used to extract context between phrases and understand the significance of the constituent media before merging.
A recurrent neural network model, specifically the Gated Recurrent Unit (GRU), was used to capture the interlocutor's state and extract context.
By incorporating contextual information, interlocutor state, and prior emotional state, the proposed model achieves better performance than standard baselines in terms of classification accuracy, reaching up to 76%.

4.3 Deep Learning Approach to Text Analysis for Human Emotion Detection from Big Data

The study [36] proposes a model known as "Deep Learning-Powered Semantic Text Analysis" (DLSTA) to detect human emotions by analyzing massive texts using natural language processing and deep learning techniques.
The model relied on word embeddings such as Word2Vec to capture linguistic meanings and contexts, enhancing classification accuracy. The DLSTA model achieved a sentiment detection rate of 97.22% and a classification accuracy of 98.02%.
The study also highlighted the importance of incorporating semantic and syntactic features of texts to improve the performance of machine learning-based models.

4.4 Transformer Models for Text-based Emotion Detection

This comprehensive review explores the development of transformer models , focusing on BERT (Bidirectional Encoder Representations from Transformers) and its applications in text emotion detection.
It highlights the importance of contextual information in natural language processing and discusses models such as GPT, Transformer-XL, XLM, and BERT, highlighting the advantages and disadvantages of each. The article provides an analysis of recent work using BERT models in emotion detection, reviewing the datasets used and the results achieved.
It also highlights current challenges and suggests future research directions to enhance the effectiveness of these models in understanding human emotions from text.

4.5 Emotion Detection for Social Robots Based on NLP Transformers and an Emotion Ontology

The paper discusses the development of a framework to enable social robots to recognise human emotions through text analysis and store this information in a semantic repository using an emotion ontology. The framework relies on converting speech to text using Google's API, then using natural language processing (NLP) transformer models to classify emotions in texts. This system was tested in a museum guidance scenario, where robots can record the emotions evoked by artworks in visitors. The results demonstrated acceptable performance compared to state-of-the-art models, with a clear plan for improvement.
This research contributes to improving human-robot interaction by enabling robots to understand and store human emotions in an organised manner, paving the way for the development of intelligent applications based on this information.

4.6 Amazon Fine Food Reviews with BERT Model

This study [37] aims to develop a model based on BERT to analyze texts and predict review ratings based on their textual descriptions.
The model achieved 79% accuracy and 54% loss in prediction on the test set. The study demonstrated that using BERT to analyze textual reviews is an effective and accurate option, especially when dealing with semantically rich texts such as food reviews.
It also demonstrated the importance of using modern deep learning techniques instead of traditional models in text-based sentiment classification tasks.

4.7 Multimodal Hierarchical Approach to Speech Emotion Recognition from Audio and Text

The study [38] aims to develop an advanced speech emotion recognition (SER) system by integrating audio and text signals using a deep learning-based architecture.
Text representations extracted using the ELMo v2 model were used to capture the emotional context in texts, along with audio features.
The proposed approach focuses on mimicking the human way of understanding emotions, which is done hierarchically, analyzing emotions across multiple levels of information. The IEMOCAP model achieved a classification accuracy of 74.5%.

4.8 A Review of Different Approaches for Detecting Emotion from Text

The study [39] provides a comprehensive review of the various methods used to detect emotions from texts, focusing on models, datasets, emotion dictionaries, metrics used, and associated challenges.
Algorithms such as Naive Bayes and SVM and models such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) were used to analyse texts. The results showed excellent results.

4.9 A Deep Neural Network Model for the Detection and Classification of Emotions from Textual Content

In this study [40], a deep machine learning model was developed that can detect and classify emotions in written texts, particularly those extracted from social media.
A deep neural network model known as BiLSTM (Bidirectional Long Short-Term Memory) was used to analyze the texts. The model focuses on five major emotions: joy, sadness, fear, shame, and guilt. The proposed model demonstrated outstanding performance.

4.10 Conclusion

After reviewing previous studies, it becomes clear that there are many research efforts that have addressed the topic of discovering and analyzing emotions from text from different perspectives, using various deep learning models. This demonstrates the importance of research in this field.
This serves as a starting point for our study, which aims to create a recurrent neural network (RNN) model for detecting emotions in textual data. The model recognizes linguistic patterns and temporal contexts, accurately classifying emotions such as happiness, sadness, or anger. It also evaluates its effectiveness in understanding word sequences and semantic dependencies. The model's applicability in real-world contexts includes social media analysis.

Chapter 5

Methodology

5.1 Introduction

Our approach, designed to classify emotions into six categories, uses a machine learning approach. The proposed system consists of various components:
(a)collection and segmentation of data sets.
(b)structure of the model.
(c) design of the model.
(d) mathematical implementation of the model.

5.2 Dataset collection and preparation

This module is comprised of two steps.

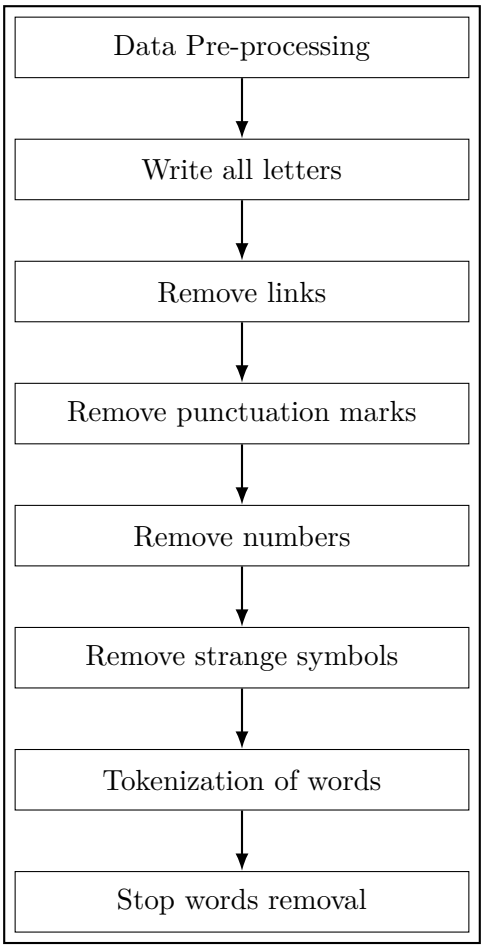
5.2.1 Dataset Collection

First, we obtained the required dataset from Kaggle. The dataset consists of 21460 texts classified into six emotional categories (sadness, anger, love, surprise, fear, and happiness).

5.2.2 Text Preprocessing

It is important to preprocess data before using it for analysis or prediction. We preprocess texts to prepare them for model building, which is the first step in natural language processing projects. All preprocessing steps are illustrated in Figure.

Data Pre-processing Steps



Removing Stop Words

Stop words are typically removed from text before training machine learning models because they occur frequently and provide no unique information for clustering. All of these words were removed from the entire dataset. Examples of some stop words in English include: (I, we, you, you, to, in, or)

Removing Punctuation

Removing punctuation from text data is a common method for preprocessing text. This method helps process each text individually. When punctuation is removed, the words ”data?” and ”data” are treated equally. Text data contains a lot of punctuation, and in this step, we remove all punctuation from tweets as follows: `'() * +, -. /:;? @ |`

Numbers removal

Because we're dealing with text, the number may not be very useful in text processing. As a consequence, numbers in text can be removed. Removing Special Characters Special characters are non-alphanumeric characters. They are commonly used in comments, hints, and monetary figures. They do not contribute to text understanding and confuse algorithms. We can remove these characters and numbers. For example, @sami, @marouane.

Tokenization of words

We use the method (Tokenization of words) to split a sentence into words. The output of word tokenization can be converted to Data Frame for better text understanding in machine learning applications. Machine learning models need numeric for clustring. Word tokenization becomes a crucial part of the text (string) to numeric data conversion. Tokenization is done as shown in Figure

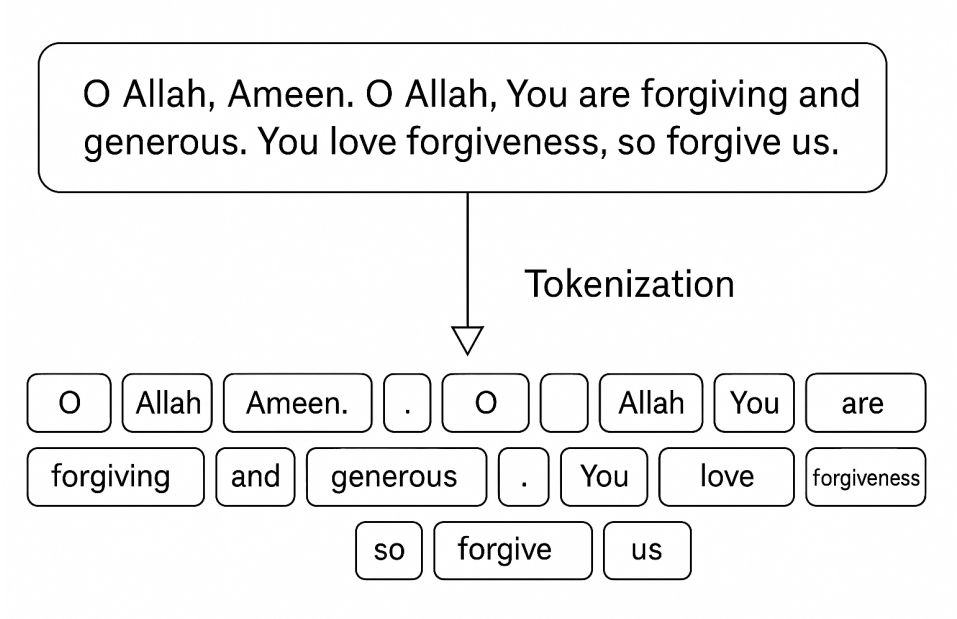


Figure 5.1: Tokenization example

5.2.3 Split training and testing

In this step, we divide the dataset into two sets; (i) Train dataset, and (ii) Test dataset by applying the train-test split method of `SKlearn`[1]. The original data set is divided into train and test set by 80%-20% ratio, respectively, where the training data are used for the model building and the effectiveness of the model is evaluated on the test set.

Original set: 21 460 (100%)		
Train set 15 022 (70%)	Validation set 2 146 (10%)	Train set: 4 292 (20%)

Table 5.1: Dataset Statistics

No	Customer Reviews	Sentiment
1	Edouard looked at him , and felt a sickening dism,fear	Happy
2	i just know that im feeling so hot now,love i just know that im feeling so hot now	love
4	i would also hate for you to feel i was selfish in my decision	anger
5	i even feel weird living with lay people again	surprise
6	i feel weird	surprise
7	im feeling kinda homesick	sadness
8	i want to feel affectionate	love
10	i feel more lively	Happy

Table 5.2: Training set sample

No	Customer Reviews	Sentiment
3	i was feeling lethargic hahaha	sadness
9	i feel like i m trying to be that guy who hangs out with curious george	surprise

Table 5.3: Test set sample

The statistics of the dataset, shown in Table dataset_stats, depict that in the first step, we used an 80:20 split to divide the dataset by applying the *train-test split* method of `Scikit-learn` .
In the second round, a 70:10 split is applied, in which 10% is used as a validation set. The validation chunk aims at hyperparameter tuning and model evaluation.
A sample of the dataset used for model training is represented in Table 5.2 , where the model learns from the training data.
In Table 5.3 , a sample of the test set is provided, which gives an unbiased evaluation of the model.

[1]https://scikit-learn.org/stable/index.html

5.3 Model structure and Design

The proposed method consists of different components Figure , described briefly as follows:

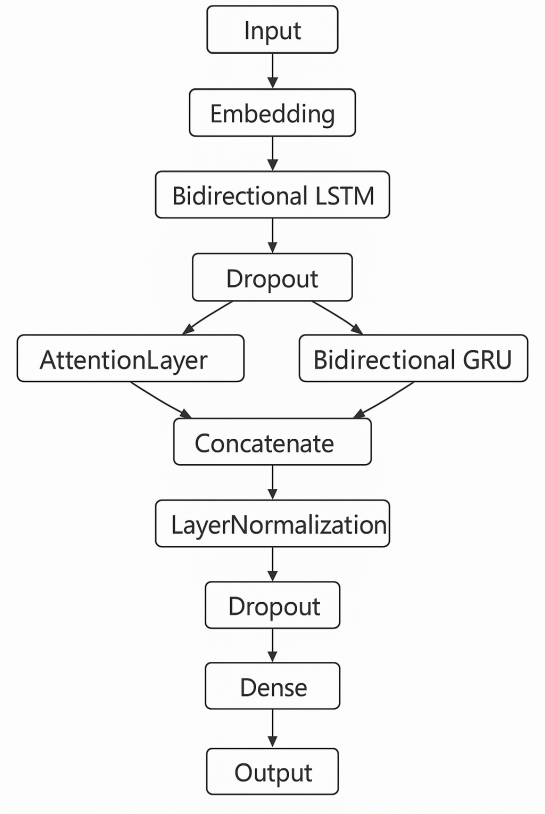


Figure 5.2: Model structure

The BiLSTMATTN model is proposed for the emotion classification task. The proposed model includes five layers: the embedding layer, the dropout layer, the BiLSTM layer, the GRU layer, the attention layer, and the output layer. Details of each layer are described below.

Processing data and input preparation

Beforewe input the text data into the ML model,we pre-process the. For this purpose, first,we convert all entire textwords into lowercase. After that, tokenization is performed using keras tokenizer, which breaks the text into smaller subtext chunks (words). This determines the vocabulary in which each word is assigned a unique number (index). Resultantly, a sequence of indexes is created.

Formation of the feature vector

After the transformation of input text into a sequence of the indexes (numbers), sequence of indexes is transformed to a feature vector through the Keras embedding layer. It takes word index as an input and generates its corresponding real-valued vector as an output.

Feature encoding with BiLSTM

The BiLSTMlayer acts as an encoder, such that for every word in the input text, it not only preserves the past information but also the future information. Therefore, the BiLSTM outputs a more enrich representation by exploiting the forward contextual information as well as the backward contextual information.

Attention vector

Classical methods like RNN cannot capture significant words from the text data. To overcome this problem, we used an attention mechanism in the proposed technique to focus on the salient words in the dataset.

GRU

A Gated Recurrent Unit (GRU) is a type of recurrent neural network (RNN) designed to overcome the long-term forgetting problem that plagues traditional RNNs. Proposed in 2014, GRUs are very similar to LSTMs, but their architecture is simpler, making them faster to train and perform well in many applications, such as natural language processing. GRUs aim to control the flow of information within a cell using only two "gates," rather than three as in LSTMs:

Output layer

Finally, the classification is performed at the output layer by applying the sigmoid function.

5.3.1 Embedding layer

Consider an input text comprised of n tokens: $X = [x_1, x_2, x_3, \ldots\ldots x_n]$. Each token x_i is transformed into continuous valued vector $x_i \in R_w$, where w represent the dimension of the word embedding. In the proposed work, we used Keras embedding layer to generate the word embedding vector. Now the embedding layer produces a Feature matrix $F \in R^{w.k}$, where k is the length of the input text. This input representation is passed to the next layer.

5.3.2 Dropout layer

The objective of the dropout layer is to control the overfitting phenomenon. The dropout layer's range lies within 0-1. In our work, we specified the value of 0.5. This layer is incorporated after the embedding layer, to randomly turn off the activation of neurons in the embedding layer.3 The dropout layer randomly deactivates certain values in the embedding layer by changing the value to 0.

5.3.3 Bidirectional layer

The BiLSTM is an extension of the unidirectional LSTM networks. The LSTM processes information only from the past context by ignoring the future context, because, for most of the sequence classification tasks, it is useful to capture both the past and the future context. The BiLSTM model is an extension of the LSTM model in which two LSTM layers are used.35The model is capable of accessing the future as well as past information. The BiLSTM operation is modeled using the following equations.

Forward LSTM	**Backward LSTM**
Forget gate	
$f_t = \sigma(W_f \cdot x_t + U_f \cdot h_{t-1} + b_f)$ (1)	$f_t = \sigma(W_f \cdot x_t + U_f \cdot h_{t+1} + b_f)$ (7)
Input gate	
$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t-1} + b_i)$ (2)	$i_t = \sigma(W_i \cdot x_t + U_i \cdot h_{t+1} + b_i)$ (8)
Input activation	
$a_t = \tanh(W_a \cdot x_t + U_a \cdot h_{t-1} + b_a)$ (3)	$a_t = \tanh(W_a \cdot x_t + U_a \cdot h_{t+1} + b_a)$ (9)
Output gate	
$o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t-1} + b_o)$ (4)	$o_t = \sigma(W_o \cdot x_t + U_o \cdot h_{t+1} + b_o)$ (10)

Table 5.4: BiLSTM gates

Internal state	
$state_t = i_t \otimes a_t + f_t \otimes state_{t-1}$ (5)	$state_t = i_t \otimes a_t + f_t \otimes state_{t+1}$ (11)
Output	
$h_t = \tanh(state_t) \otimes o_t$ (6)	$h_t = \tanh(state_t) \otimes o_t$ (12)

Table 5.5: Internal states and output equations

Term	**Meaning**
x_t	Input vector at time t
h_t	Hidden state at time t
f_t	Forget gate
i_t	Input gate
o_t	Output gate
a_t	Input activation
W, U, b	Weight matrices and biases
σ	Sigmoid activation function
tanh	Hyperbolic tangent activation function
$\otimes$	Element-wise multiplication
$\overrightarrow{h_t}$	Hidden state from forward LSTM
$\overleftarrow{h_t}$	Hidden state from backward LSTM
$[\cdot;\cdot]$	Concatenation of forward and backward outputs

Table 5.6: Mathematical terms for BiLSTM

The BiLSTM gates are presented in Table 5, depicting both Forward LSTM and Backward LSTM. Whereas internal states and the output equations are listed in Table 6. The weight matrices are represented as U and W, bias term is denoted as b, input, forget, output gates are represented by i, f, and o respectively, and h_t is the output that is passed to the next layer of the network.
So the forward LSTM layer encodes the information, which is denoted as h_t. Similarly, the backward LSTM layer encodes the contextual information denoted by $\overleftarrow{h_t}$. Finally, the elementwise summation is performed to combine the forward and backward LSTM, shown as follows:

$$\overleftrightarrow{h_t} = \overleftarrow{h_t} \oplus \overrightarrow{h_t} \tag{1}$$

The output produced by the BiLSTM layer is given as a matrix H, which consists of vectors $[h_1, h_2, \ldots, h_n]$, which is further made input to the Attention layer for further processing.
Table 7 presents all of the mathematical terms and their definitions used in the BiLSTM layer.

5.3.4 Attention layer

We introduced the attention mechanism to take advantage of its ability to automatically focus on the words that have decisive effects. The computations of the attention layer are described as follows:

$$m_i = \tanh(W h_i + b) \tag{2}$$

$$\alpha_i = \text{softmax}(m_i) \tag{3}$$

$$r = \sum_{i=1}^{n} \alpha_i h_i \tag{4}$$

where h_i is the output from the previous layer, W is the weight matrix, b is the bias vector, α denotes the attention weight for every word in the sentence, and r is the resulting attention vector.

5.3.5 BiLSTM layer

BiLSTM is the third layer in the proposed technique that performs its computation on the input of the dropout layer and finally generates a new encoding. We compute the Forward LSTM (left-to-right) using Equations (1)–(6) and the Backward LSTM (right-to-left) using Equations (7)–(12). The computation of the forward LSTM is described as follows:
First hidden layer - Forward LSTM: The forward LSTM includes the present input i_t, past state h_{t-1}, then using Equations (6)–(11), some computation is executed, and lastly the hidden state $\vec{h}$ is produced as a result of the first hidden layer (forward LSTM):

$$\vec{h} = o_t \odot \tau(c_t)$$

$$\vec{h} = \begin{bmatrix} 0.3 \\ 0.6 \\ 0.5 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \odot \tau \left(\begin{bmatrix} 0.4 \\ 0.7 \\ 0.6 \end{bmatrix} \right)$$

Second hidden layer - Backward LSTM: The Backward LSTM includes the present input i_t, future state h_{t+1}, then using Equations (12)–(17), some computation is executed, and finally the hidden state $\vec{h}$ is produced as a result of the first hidden layer (backward LSTM).

5.3.6 Output layer

The sigmoid function at the output layer predicts the class label, that is, positive class or negative class using the following equation:

$$y = \text{sigmoid}(z) \tag{5}$$

5.4 Conclusion

We examined the suggested method for examining and categorising emotions from text in this chapter, stressing the main stages the system passes through to complete this work effectively. Relevant data is first gathered; then it is processed and the classification run using suitable models.
The following chapter will present the tools utilised in the development environment, and we will use and assess the method we adopted for emotion recognition from text. This useful application will enable us to evaluate the performance of the model and examine its findings.

Chapter 6

Results

6.1 Introduction

In this chapter, we process English texts for sentiment analysis. This processing extracts the polarity of opinions expressed by a set of emotions (joy, sadness, happiness, anger, etc.). We present the most important results we obtained through training our combined LSTM+GRU model with an attention mechanism.
In this chapter, we aim to evaluate the performance of our proposed model from various aspects based on the study's methodology. We present the results in tables and graphs for ease of understanding and discussion.

6.2 Development Environment

6.2.1 Google Colab

Colaboratory or 'Colab'. Lets you run and write the Python code of your choice using your browser. Based on Jupyter Notebook and meant for machine learning training and research, Google offers this for free. You may run machine learning models straight in the cloud with this tool.

- Improve coding skills in the Python programming language.
- Develop deep learning applications using popular Python libraries such as Keras, TensorFlow, PyTorch, and OpenCV.
- Choose a development environment (Jupyter Notebook) that needs no setup. Access to a GPU graphics processor, totally free, distinguishes Colab from other services, nevertheless.

6.2.2 Anaconda

Anaconda is a scientific distribution of Python. It allows you to write and run your choice of Python code through the browser. Offered by Anaconda Enterprise (free), it is used in Jupyter Notebook and is intended for training and research in machine learning (https://www.anaconda.com/about-us)

6.3 Programming language and libraries

6.3.1 Python

In recent years, Python has become the most widely used programming language among computer scientists. It has propelled itself to the forefront of infrastructure management, data analysis, and software development.
Python allows developers to focus on what they do rather than how they do it. It has freed developers from the formal constraints that consumed their time with older languages. As a result, developing code with Python is faster than with other languages . (https://scikit-learn.org/stable/index.html)

6.3.2 Libraries used

- **TensorFlow:** We used this library to define the basic components of the CNN-LSTM architecture. This library is designed for implementing machine and deep learning algorithms, and it also offers great flexibility when used for developing a neural network.(https://www.tensorflow.org/?hl=f
- **Keras:** Allows libraries used with TensorFlow and Keras, we used this library to implement the different layers, activation functions and preparation of the learning base . (https://www.tensorflow.org/
- **NumPy:** We used this library to adapt input types based on the configuration of the models used, designed to handle multidimensional matrices or arrays, as well as mathematical functions operating on these arrays. We used this library specifically for image scanning and window

extraction. (https://numpy.org/)

- **Pandas:** is a fast, powerful, flexible, and easy-to-use open-source data analysis and manipulation tool built on the Python programming language [41]. (https://pandas.pydata.org/)
- **Sklearn :** Among the most useful libraries for machine learning in Python is Scikit-learn. Among the many powerful statistical modeling and machine learning techniques provided by the sklearn package are classification, regression, clustering, and dimensionality reduction.
- **NLTK :** Natural Language Toolkit is a top platform for creating Python applications to interact with human language data. Its active discussion forum, wrappers for industrial-strength NLP libraries, and suite of text processing libraries for classification, tokenisation, stemming, tagging, parsing, and semantic reasoning complement over 50 corpora and lexical resources, including WordNet by offering simple-to-use interfaces.
  NLTK is appropriate for linguists, engineers, students, teachers, researchers, industry users, and others thanks to a hands-on guide presenting programming principles alongside subjects in computational linguistics as well as thorough API documentation. NLTK runs on Windows, Mac OS X, and Linux. Best of all, NLTK is a free, open-source, community-driven initiative. "An incredible library to play with natural language", NLTK has been described as "a great tool for teaching and working in computational linguistics using Python." (https://www.nltk.org/)
- **Matplotlib:** Matplotlib is a complete library for producing static, animated, and interactive visualisations in Python. Matplotlib simplifies basic tasks and makes complex tasks achievable. Produce plots of publishing quality. Create dynamic graphics that can zoom, pan, and refresh. Design layout and visual style. Export to several file types. Include in graphical user interfaces and JupyterLab. Draw on a wide range of third-party Matplotlib-based programs.(https://matplotlib.org/)
- **Re:** The re library in Python is a built-in module used to analyse regular expressions, a powerful tool for searching and changing text based on specific patterns. This library offers a collection of tools to verify text for conformity to particular patterns, search, replace, and split. (https://docs.python.org/3/library/re.html)

6.4 Results

6.4.1 Distribution of categories

After loading the text dataset and cleaning the texts, we printed a plot showing the distribution of categories in our data before entering it into the model for training and testing.

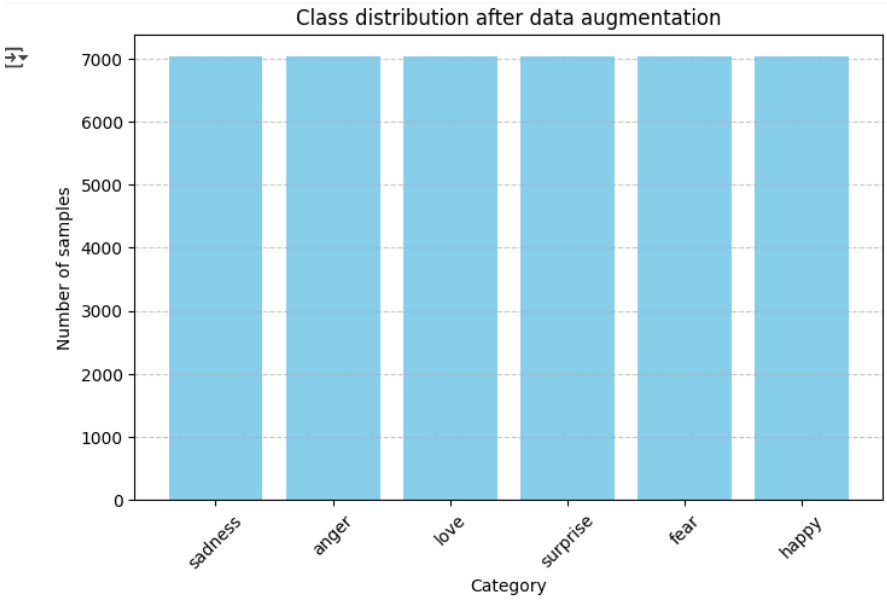


Figure 6.1: Class distribution diagram

After applying data augmentation to the dataset, the plot shows a balanced distribution among the categories, with approximately 7,000 samples per category. This balanced distribution helps mitigate the issue of class imbalance, leading to improved model accuracy and fair evaluation across all categories. Additionally, it reduces the likelihood of bias toward a specific category, making the model's predictions more reliable when analyzing users' emotions.

6.4.2 Performance analysis for each epoch

Epoch	Accuracy	Loss	Validation Accuracy	Validation Loss
1	0.3313	1.7634	0.9032	0.7400
2	0.9225	0.7218	0.9618	0.5646
3	0.9771	0.5540	0.9638	0.5338
4	0.9848	0.5069	0.9713	0.5024
5	0.9900	0.4808	0.9717	0.5019
6	0.9899	0.4717	0.9758	0.4892
7	0.9911	0.4667	0.9772	0.4838
8	0.9930	0.4593	0.9723	0.4942
9	0.9926	0.4595	0.9765	0.4862
10	0.9949	0.4550	0.9765	0.4863

Table 6.1: Table showing the model training results

The table illustrates the evolution of the model's performance over 10 training epochs, showing:

- The model's accuracy started at around 33% in the first epoch and then significantly increased, surpassing 92% in the second epoch, indicating that the model quickly learned the fundamental patterns. The performance continued to improve, reaching 99% in the tenth epoch, suggesting that the model effectively learned from the data.

- A gradual decline in loss values was observed, starting at 1.76 in the first epoch and continuously decreasing to 0.45 in the tenth epoch, demonstrating that the model progressively reduced errors during training.
- Validation accuracy improved, beginning at 90% and rising to 97% in the final epoch, meaning the model performed well on unseen data. Validation loss decreased from 0.74 to 0.48, but its decline was not as sharp compared to training loss, suggesting that the model has stabilized and is no longer improving significantly.

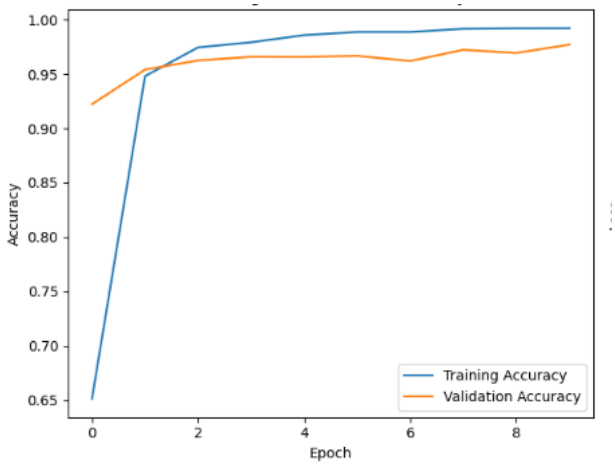


Figure 6.2: Training and Validation Accuracy

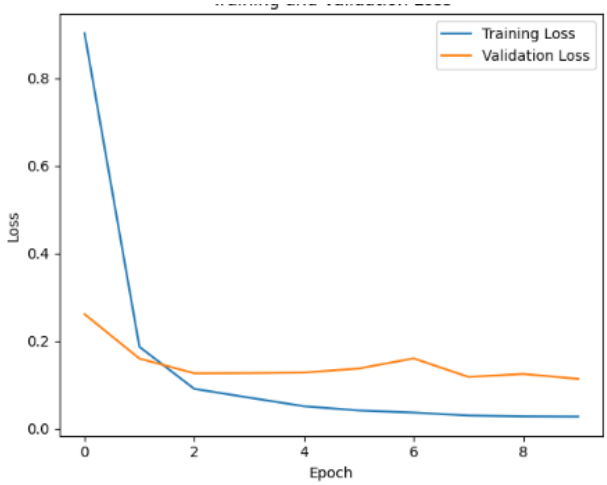


Figure 6.3: Training and Validation Loss

6.4.3 Model evaluation

Table showing the results of model evaluation based on Precision, Recall, F1-score, and number of samples (Support) for each class:

Class	Precision	Recall	F1-Score	Support
anger	0.99	0.97	0.98	1015
fear	0.97	0.98	0.97	1015
happy	0.96	0.97	0.96	1016
love	0.98	0.99	0.98	1016
sadness	0.98	0.97	0.97	1016
surprise	0.99	0.98	0.99	1015
Overall Accuracy	**0.98**	-	-	**6093**
Macro Average	**0.98**	**0.98**	**0.98**	**6093**
Weighted Average	**0.98**	**0.98**	**0.98**	**6093**

Table 6.2: Classification report showing precision, recall, F1-score, and support for each emotion class

The table presents the evaluation results of the model's performance in terms of precision, recall, and F1-score for each category, helping to assess how effectively the model classifies data. The key points are summarized as follows:

- The numbers indicate that the model can effectively predict different categories while maintaining a good balance between precision and recall, with F1-score values ranging between 96% and 99%.
- Category 3 (*love*) achieved the highest performance, with 98% precision and 99% recall, suggesting that the model classifies this category with perfect accuracy.
- Categories 1 (*happy*) have relatively lower precision (96%), which may indicate some misclassification in these categories.
- There is slight variation in the performance of some categories, such as category 4 (*sadness*), where precision is 98% and recall is 97%. This may require further analysis to determine the cause of the difference.
- The overall accuracy is 98%, meaning the model is highly reliable and produces strong results across all categories.
- The macro average and weighted average are both 98%, confirming that the model maintains consistent performance across different categories.

6.4.4 Confusion Matrix

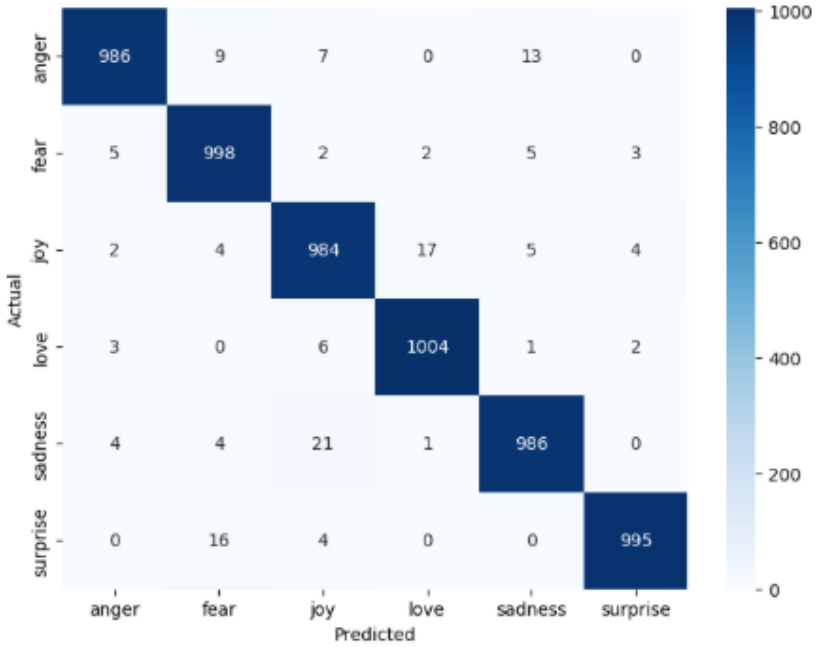


Figure 6.4: Confusion Matrix

The displayed **Confusion Matrix** illustrates the model's performance in classifying six emotional categories. The following table shows the most prominent confusions between emotional categories in the model.
The table highlights the most frequently confused emotion categories based on the model's predictions. Several key observations can be drawn from the confusion patterns:

Original Category	Most Confused Category	Number of Cases	Observations
Joy	Love	17	Two positive emotions that are linguistically close.
Sadness	Joy	21	Likely due to mixed expressions in sentences.
Surprise	Fear	16	Understandable, as surprise and fear sometimes share triggers.
Anger	Sadness	13	Sadness and anger can overlap in tone.
Fear	Sadness	5	A minor overlap, but still expected.

- **Joy vs. Love:** There were 17 cases where the model confused joy with love. This is understandable, as both are positive emotions and may have overlapping linguistic features.
- **Sadness vs. Joy:** The most frequent confusion occurred between sadness and joy (21 cases), possibly due to the presence of mixed emotional expressions in the same sentence or subtle context-dependent cues.
- **Surprise vs. Fear:** With 16 misclassified instances, surprise is often mistaken for fear. This overlap can be attributed to the fact that both emotions may be triggered by unexpected events.
- **Anger vs. Sadness:** The model confused anger with sadness in 13 cases. This could be due to similarities in tone or emotional intensity, making them difficult to differentiate in textual data.
- **Fear vs. Sadness:** Although less frequent (5 cases), fear was also confused with sadness. This minor overlap suggests a level of emotional nuance that might require additional linguistic context to resolve.

Approximate measures (for performance of each category):
The table represents approximate measures of the model's performance in classifying emotional categories:

Category	True Positives (TP)	Misclassified Cases	Approximate Precision
Anger	986	14	$\sim 98.6\%$
Fear	998	2	$\sim 99.8\%$
Joy	984	16	$\sim 98.4\%$
Love	1004	~ 26 (?)	$\sim 97.4\%$ (Approx.)
Sadness	986	14	$\sim 98.6\%$
Surprise	995	5	$\sim 99.5\%$

Table Analysis:

- The model's accuracy is generally high, with all categories achieving over 97%, indicating strong capability in distinguishing emotions.
- The lowest accuracy was recorded for the *Love* category, which was misclassified in approximately 26 cases. This might be due to its linguistic and emotional similarity to *Joy*.
- *Fear* achieved the highest accuracy at approximately 99.8%, with the fewest errors (only 2 cases), suggesting that the model classifies this category with clear distinction.
- Error rates are very low, demonstrating the model's efficiency. However, further improvement can be made by analyzing the causes of confusion, particularly between closely related emotions like *Love* and *Joy*.

6.5 Conclusion

This chapter presented and examined the main results of testing our LSTM+GRU model. The tables and graphs provide a clear picture of the model's effectiveness in detecting sentiment from text. The results demonstrated that the model achieved acceptable classification accuracy.
Through analyzing these results, we highlighted the importance of using LSTM and GRU, along with attentional mechanisms, to capture and identify sentiment from text.

Chapter 7

General Conclusion

This study looked at how to find and analyse feelings in texts using recurrent neural network (RNN) models and ended with a system that can identify human emotions through text analysis from large amounts of data. This paper primarily aimed to explore how RNN models, thanks to their sequential nature and ability to handle contextual data, can understand and extract the emotions inherent in textual content. During this work, an RNN model was built and trained to classify texts according to their emotions (joy, anger, happiness, sadness, love, and surprise), taking into account the challenges associated with natural language processing.

During our study, we faced several challenges, including:

- The need for large and diverse data to ensure the accuracy of the model.
- Processing natural language and dealing with complex or indirect expressions.
- Making the model capable of dealing with and detecting various emotions.
- Reducing errors and improving the accuracy rate.

Our proposed model, which uses architectures or layers like LSTM, GRU, and Attention, has demonstrated promising results. The findings indicate that this approach is a highly viable option for sentiment recognition due to its capability to directly learn features from raw data. The results indicate that our proposed model clearly achieves a top detection rate of 97.22% and a classification accuracy of 98.02% using various emotional term embedding methods.

RNN models, including sophisticated architectures like LSTM and GRU, have shown remarkable efficiency in handling these activities. Their capacity to catch long-range text dependencies—a key factor for properly grasping context and, hence, feelings—explains this.
These models, however, are not without difficulties. Amongst these, the vanishing/exploding gradients issue stands out as particularly important since it might hinder efficient training. Handling the subtleties of human language, like irony, ambiguity, and phrases strongly dependent on cultural background, is very challenging. Furthermore, the development of Natural Language Processing and the arrival of newer, more potent designs such as Transformers and Large Language Models (LLMs) provide fresh standards and difficulties for the performance of conventional RNN models.
Considering the above, we suggest the following future study paths:

- Enhancing RNN model performance.
- Dealing with low-resource languages.
- Comparing RNNs to more recent models.
- Developing methods to grasp and explain how RNN models reach their categorization choices,

Bibliography

[1] Z. C. Lipton, "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[2] MIT Sloan Management Review, "Emotion ai, explained," 2023. Accessed: 2025-05-17.

[3] S. N. Shivhare and S. Khethawat, "Emotion detection from text," 2012.

[4] D. Hockenbury and S. E. Hockenbury, "Discovering psychology," 2010.

[5] D. Keltner, K. Oatley, and J. M. Jenkins, "Understanding emotions," 2013.

[6] E. A. Phelps, "Emotion and cognition: insights from studies of the human amygdala," *Nature Reviews Neuroscience*, vol. 7, no. 2, pp. 148–158, 2006.

[7] H. Alhuzali and S. Ananiadou, "Spanemo: Casting multi-label emotion classification as span-prediction," pp. 1572–1584, 2021.

[8] P. Ekman, "Emotions revealed: Recognizing faces and feelings to improve communication and emotional life," 2003.

[9] R. Plutchik, "Emotion: A psychoevolutionary synthesis," 1980.

[10] A. Ortony, G. L. Clore, and A. Collins, "The cognitive structure of emotions," 1988.

[11] R. Plutchik, "A general psychoevolutionary theory of emotion," 1980.

[12] A. Mehrabian and J. A. Russell, "Approach to environmental psychology," 1974.

[13] J. A. Russell, "Core affect and the psychological construction of emotion," *Psychological Review*, vol. 110, no. 1, pp. 145–172, 2003.

[14] M. Hasan, E. Rundensteiner, and E. Agu, "Automatic emotion detection in text streams by analyzing twitter data," *International Journal of Data Science and Analytics*, vol. 7, no. 1, pp. 35–51, 2019.

[15] Y. Mehta, A. Arya, and N. Rakesh, "Emotion detection from text: a survey," *Multimedia Tools and Applications*, vol. 80, pp. 1–39, 2021.

[16] A. Mitra, "Sentiment analysis using machine learning approaches (lexicon based on movie review dataset)," *Journal of Ubiquitous Computing and Communication Technologies (UCCT)*, vol. 2, no. 03, pp. 145–152, 2020.

[17] M. O. Hegazi, Y. Al-Dossari, A. Al-Yahy, A. Al-Sumari, and A. Hilal, "Preprocessing arabic text on social media," *Heliyon*, vol. 7, no. 2, p. e06191, 2021.

[18] O. Rabie and C. Sturm, "Feel the heat: Emotion detection in arabic social media content," pp. 37–49, 2014.

[19] N. Alswaidan and M. E. B. Menai, "Hybrid feature model for emotion recognition in arabic text," *IEEE Access*, vol. 8, pp. 37843–37854, 2020.

[20] "Recurrent neural networks: A comprehensive review of architectures, variants, and applications," *Information*, vol. 15, no. 9, p. 517, 2024.

[21] L. Medsker and L. C. Jain, "Recurrent neural networks: Design and applications," 1999.

[22] IBM, "Recurrent neural networks (rnns)," n.d. Accessed: 2025-05-13.

[23] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," 2013. [Online]. Available: https://arxiv.org/abs/1312.6026.

[24] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *arXiv preprint arXiv:1312.6026*, 2013. [Online; accessed 13-May-2025].

[25] L. Medsker and L. C. Jain, "Recurrent neural networks: Design and applications," 1999.

[26] L. Medsker and L. C. Jain, "Recurrent neural networks: design and applications," 1999.

[27] P. J. Werbos, "Backpropagation through time: what it does and how to do it," *Proceedings of the IEEE*, vol. 78, no. 10, pp. 1550–1560, 1990.

[28] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[29] H. Salehinejad, S. Sankar, J. Barfett, E. Colak, and S. Valaee, "Recent advances in recurrent neural networks," *arXiv preprint arXiv:1801.01078*, 2017.

[30] GeeksforGeeks contributors, "Bidirectional recurrent neural network," 2025. Accessed: 2025-05-17.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[33] GeeksforGeeks, "Gated recurrent unit (gru) networks," 2023. Accessed: 2025-05-17.

[34] C. Stryker, "Què és una xarxa neuronal recurrent (rnn)?," *ibm.com*, october 2024.

[35] M. G. Huddar, S. S. Sannakki, and V. S. Rajpurohit, "Attention-based multi-modal sentiment analysis and emotion detection in conversation using rnn," 2021.

[36] J. Guo, "Deep learning approach to text analysis for human emotion detection from big data," *Journal of Intelligent Systems*, vol. 31, no. 1, pp. 113–126, 2022.

[37] X. Zhao and Y. Sun, "Amazon fine food reviews with bert model," *Procedia Computer Science*, vol. 208, pp. 401–406, 2022.

[38] P. Singh, R. Srivastava, K. Rana, and V. Kumar, "A multimodal hierarchical approach to speech emotion recognition from audio and text," *Knowledge-Based Systems*, vol. 229, p. 107316, 2021.

[39] A. R. Murthy and K. A. Kumar, "A review of different approaches for detecting emotion from text," vol. 1110, no. 1, p. 012009, 2021.

[40] M. Z. Asghar, A. Lajis, M. M. Alam, M. K. Rahmat, H. M. Nasir, H. Ahmad, M. S. Al-Rakhami, A. Al-Amri, and F. R. Albogamy, "A deep neural network model for the detection and classification of emotions from textual content," *Complexity*, vol. 2022, no. 1, p. 8221121, 2022.

[41] A. R. Damasio, "Emotion in the perspective of an integrated nervous system," *Brain Research Reviews*, vol. 26, no. 2-3, pp. 83–86, 1998.