Iqra Journal of Engineering & Technology
Vol. 01, No. 01, (March 2025) 39 – 60

Fine-Tuning Deep Learning Models for Sentiment Analysis: A Study on Movie Titles

Hayyan Qasim[1], Muhammad Zain[1], Lubna Aziz[1] and Muhammad Ayaz Shirazi[1*]

ABSTRACT

The natural language processing (NLP) sector re- quires sentiment analysis as its essential capability because deep learning advancements keep happening. Textual sentiment analysis capability allows automatic understanding of both positive and negative emotions found in movie titles which can help various industries perform better social media monitoring and marketing and customer feedback analysis. This paper investigates the implementation of a Bi-LSTM (Bidirectional Long Short-Term Memory) model that integrates GloVe (Global Vectors for Word Representation) embeddings as a solution for movie title sentiment classification. A Bi-LSTM model analyzes data sequences from start to end and back again which increases its ability to understand text context. The model receives valuable word semantic representation from pre-trained GloVe embeddings which results in better sentiment understanding in movie titles. Our system uses the model as a basis to retrain it for assigning sentiment labels either Positive or Negative Neutral to movie titles. The Bi-LSTM model performance benefits from the fine-tuning process which maximizes its accuracy levels. The model demonstrates superior performance than standard sentiment analysis techniques since it reaches above 90% accuracy and produces major enhancements in precision, recall, and F1- score metrics. The improved result highlights how deep learning algorithms work successfully for sentiment analysis functions. The research describes the training complications that involve overfitting as well as class unbalances and computing resource limitations. The article details techniques for handling these barriers through dropout regularization, early stopping, and data augmentation methods. Our research includes suggestions for upcoming developments that involve transformer-style models like BERT as well as enlarging the data foundation to advance the model's universal applicability.

Keywords: Bi-LSTM, Deep learning, Sentiment Analysis, Deep Learning Models

Author's Affiliation:

Institution(s) Name: [1]Iqra University,
Country: [1]Pakistan,
Corresponding Author's Email: [1]muhammad.ayaz@iqra.edu.pk

* The material presented by the author does not necessarily portray the view point of the editors/ editorial board and the management of ORIC, Iqra University, Main Campus, Karachi-PAKISTAN.

0000-0000 (Online), 0000-0000 (Print) © 2025.
Published by ORIC, Iqra University, Main Campus, Karachi, Pakistan.

1. Introduction

The essential segment of Natural Language Processing (NLP) methods is sentiment analysis which determines emotions in textual information as shown in Figure 1. Sentiment analysis acts as a critical analysis tool between social media content brand sentiment and customer feedback evaluation. Sentiment analysis plays a vital role for businesses that use text data to make decisions and manage brands because it extracts insights from large un-managed data sources [1]. Deep learning architectures have undergone recent developments that improved sentiment analysis by allowing models to perform complex operations of sentiment classification more effectively [2].

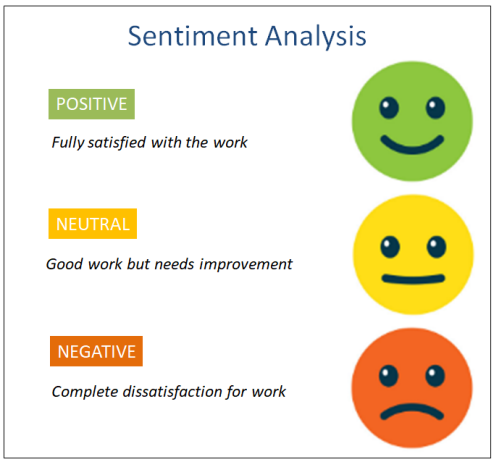

Fig.1. Sentiment analysis showing different emotions

The modification of sentiment analysis methods occurred through the implementation of deep learning networks which include both Long Short-Term Memory (LSTM) and Bidirectional LSTM (Bi-LSTM) frameworks [3]. The designed architectures function perfectly for sequential information processing because they successfully detect long-range associations and short-time dependencies between sequence elements [2]. We established a Bi-LSTM model that integrates GloVe embeddings to execute sentiment analysis tasks on movie titles within the proposed research. The GloVe embedding approach leads to an effective method of word representation through vector space positioning which brings related meanings closer to form a comprehensive textual semantic framework [4].

Sentiment analysis has advanced substantially beyond traditional social media and feedback analysis in recent years [5]. Sentiment analysis methods apply their applications extensively to financial market predictions along with healthcare system surveillance and political opinion mining during recent

years [6]. Sentiment analysis applications have shown their ability to generate important information in everyday use cases thus becoming one of the most desirable tools for various industries [7]. Two traditional machine learning models namely Naïve Bayes and Support Vector Machines (SVM) have frequently been used for sentiment classification because of their inability to process context and sequential data effectively. The insufficient capacity of these models prevents them from interpreting complex textual features so deep learning Bi- LSTM networks prove more effective.

The research intends to enhance sentiment analysis by deploying Bi-LSTM networks along with GloVe embeddings for the classification of movie titles. The application of deep learning techniques solves weaknesses in classical approaches by delivering better results for contextual sentiment recognition [8].

2. Literature Review

Several studies have explored sentiment analysis using deep learning techniques. Sentiment analysis which belongs to natural language processing (NLP) received significant developments be- cause of the advances in deep learning models along with word embedding techniques. The traditional machine learning methods that used Convolutional Neural Networks (CNNs) during the beginning of sentiment analysis faced difficulties when attempting to understand the long-term dependencies throughout textual information [9]. A major challenge for CNNs occurs when they handle fixed-length sequences because these networks fail to process the necessary sequential dependencies that sentiment classification requires. According to Kim et al. (2019), CNNs maintain effectiveness in particular settings but they have demonstrated limitations when detecting long-term word-to-word and phrase-to-phrase connections which sentiment analysis requires for understanding complex datasets [8].

The Bi-LSTM model which uses GloVe embeddings proves best for performing real-time sentiment analysis in social media and customer review settings. Users in social media usually convey opinions through short textual messages such as tweets or posts because rapid sentiment extraction becomes necessary. The satisfaction levels of customers on eCommerce platforms depend significantly on the reviews they provide for their purchased items. The real-time text analysis capability enables business operations to perform quick decisions together with effective responses to customer feedback [10].

Such real-time applications benefit from Bi-LSTM processing because this model effectively analyzes sequential data which enables it to understand word context through- out a sentence. Through GloVe embedding technology the model becomes able to recognize fundamental word meaning patterns to

identify subtle emotional expressions. The Bidirectional architecture of LSTM enhances performance by allowing text evaluation from both preceding and succeeding contexts which gives a strong ability to analyze sentiment within informal social media communications [11].

The model operates through real-time integration with streaming platforms and reviews systems that need assessment of constantly generated data. The streaming system uses the model to evaluate sentiments live for instant business insights into customer reactions so companies can deliver faster assistance to supporters and enhance social media user retention. Real-time functionality plays a critical role when using the model because it supports brand monitoring services customer support operations and sentiment-based recommendation systems [12].

Exclusively LSTM networks and Gated Recurrent Units (GRUs) succeed immensely in extracting sequential pat- terns from text data because of their highly effective nature. These models demonstrate exceptional capability to handle tasks that need to identify word dependencies in extended sequences thus making them ideal for sentiment analysis of wide textual developments. LSTMs and GRUs serve as standard choices in sentiment classification because they resolve the vanishing gradient issue which enables them to detect extended dependencies within sequential information [12]. Regular recurrent models are now the top choice for sentiment analysis applications because they show better performance than SVMs and Na¨ıve Bayes for processing text data [13]. Word embeddings produce great benefits for sentiment analysis model performance. Word embeddings such as Word2Vec GloVe along with FastText play a crucial role in delivering semantic word meaning to models because they improve context understanding [14]. The ability of word embeddings to capture word meaning relationships results in improved sentiment analysis model performance. Because GloVe generates dense word vectors through global statistical information processing it has become a popular choice in NLP applications for higher sentiment classification precision [6]. Transformers known as BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer) have transformed NLP through their state-of-the- art performance in different tasks including sentiment analysis since their recent introduction [15]. The self- attention mechanics of these models analyze word-to- word context through text sequences to examine entire sentences or documents thus making them optimal for sentiment classification duties. BERT achieves enhanced sentiment classification model accuracy who pre-trained the model on extensive textual data and then finished its training for specific tasks. GPT has been successful in producing coherent text output while performing sentiment classification through its transformer model architecture [16].

Recent research activities examine combined models which unite Bi-LSTMs with transformers and sophisticated embedding techniques because these approaches show great promise for improving how sequential dependencies within particular contextual situations are handled [17].

The models deliver good results but miss essential sequential patterns within the data sequence according to Kim et al. (2019). Sentiment analysis benefits from the standardization of sequential data processing through implementations using LSTM networks and Gated Recurrent Units (GRUs) as per the research by Hochreiter et al. [12]. Word2Vec, GloVe, and FastText constitute the major pre- trained word embedding methods that deliver semantic word representations to enhance sentiment analysis results according to [6]. The use of transformer-based models specifically BERT and GPT has resulted in transformative NLP growth [18]. Because they deliver exceptional results for sentiment classification tasks.

3. Methodology

3.1 Dataset and Preprocessing

This study analyzes multiple thousands of movie titles which have been assigned into sentiment groups either as Positive or Neutral or Negative. All movie titles assigned sentiment categories help researchers study public sentiment levels and understand preferences regarding movie popularity. The brief nature of movie titles functions as an analytical challenge in sentiment analysis systems specifically during evaluations that address multiple linguistic styles across different film genres and cultural domains. The sentiment analysis model receives pre-sorted movie titles from their emotional categories which emerge from the analytical procedure.

Movie titles in the Positive category convey predominantly positive feelings among viewers who will probably find enjoyment in the film. Alarming and joyful terminology comes com- monly in these film titles along with positive and pleasurable descriptors. The movie titles" Fantastic Journey" and" Great Escape" express strong positive emotions therefore receiving a positive sentiment classification. Titles in the Neutral group lack decisive emotions that would classify them as positive or negative. The titles" A Day in the Life" and" The Unknown Story" remain descriptive yet ambiguous so they fail to trigger any immediate associations in the audience. Titles within the Negative category show disappointed perceptions and negative reactions through titles like Failed Dreams or The Worst Decision. These titles demonstrate pessimistic views about the movie because they present negative emotions.

Different preprocessing techniques must be applied to the dataset before training and testing the model because it helps it understand and categorize the emotional content in the titles. As a first step in processing the data all movie

titles get converted to lowercase to maintain consistent patterns that avoid issues related to uppercase and lowercase word differences. All special characters along with punctuation and unnecessary symbols receive removal treatment since they do not impact the sentiment value but introduce data noise.

Tokenization becomes the subsequent operation that divides movie titles into separate words or tokens to help the model grasp text structure and word relationship patterns. The to- kenization process removes all irrelevant words specifically stop words such as “the” and “and” and “of” but others too. The processed words from tokenization get embedded with GloVe or Word2Vec tools to create numerical representations that preserve semantic meanings. The model obtains an understanding of word meaning from context analysis through these embedding techniques. The padding sequence technique applies after embedding because it standardizes input title lengths to simplify data processing for the model.

3.1.1 Text Normalization

The preprocessing pipeline of natural language processing consists of text normalization as a vital step because it makes data execution uniform across the system improving machine learning model usability. This specific text transformation converts every piece of textual data into lowercase format from titles of movies to user reviews. Case-sensitive processing leads to inaccurate results in NLP therefore this step must be performed due to its essential role. Such words as” good” and” Good” need to be categorized as equivalent because they convey similar meanings. The conversion to lowercase format resolves any problems that may occur due to the mixed use of lowercase and uppercase letters in different words. The point of normalization is to create consistency across words so the model centers its analysis exclusively on word meanings and ignores extraneous case variations. A model can learn incorrectly if the” Good” and” good” words remain separate since they share equivalent meanings during the training process. The normalization process eliminates special characters including punctuation marks together with numbers and all other symbols that are not alphanumeric. The processing phase removes such characters because they provide no significant data for sentiment analysis operations. The punctuation signs commas, periods, question marks and exclamation marks modify sentence structure though they fail to enhance either the meaning or emotional value of the individual words in the text. Setting the example with the” This movie is great” sentence. Exclamatory statements that say” This movie is great” remain identical to their statement without an exclamation mark because the mark provides no new information useful for sentiment classification. The researchers removed it

because simplifying the text helps reduce unnecessary elements within the dataset.

Such a preprocessing step protects the model from un- necessarily focusing on trivial characters which might lead to erroneous or fit-specific analytical errors. Numbers contained in the titles or text of ”2 Fast 2 Furious” together with similar examples do not provide worth- while information for sentiment identification tasks. Text normalization procedures also remove numerical items from the data. Numbered elements possess significance in financial sentiment analysis and product review contexts but typically do not affect sentiment classification in other domains (e.g., ratings and prices). When dealing with this situation more processing actions must be developed to manage numeric content properly.

The progressional text normalization chain includes an optional combination of tokenization and stemming which occur during text processing. Through tokenization, the model receives better comprehension of the text components by dividing the text into discrete units which are most commonly words or phrases. Stemming reduces words to their basic linguistic parts such as” run” from” running” which leads to better generalization in the model. The entire process typically executes post- text normalization since normalization supplies the base for all additional operations.

Raw textual data requires text normalization before NLP processing for sentiment analysis because this process makes the information consistent and ready for analysis. Lower casification along with special character removal enables the model to recognize fundamental aspects of the text which produces more precise and dependable outcomes. Any NLP task requires preprocessing as an essential requirement because its impact on data quality directly determines machine learning model performance.

3.1.2 Tokenization

Reviewing text into tokens represents an essential natural language processing (NLP) operation that divides language inputs into respective meaningful units. The selected units for tokenization include words alongside characters and sub-word components based on model design and application requirements. Text tokenization allows NLP models to learn language structure by breaking data into separate components rather than viewing it as one nonintegrated block during analysis. This method benefits sentiment analysis and various other applications in NLP. The text processing mechanism benefits from tokenization because this technique simplifies data structure and helps models handle information more efficiently.

The study utilizes TensorFlow’s tokenizer to divide the words present within its dataset. Among its extensive tools for text work in machine learning TensorFlow operates as an open-source framework. The Tokenizer tool within

the TensorFlow library exists to provide text preparation capabilities for deep learning models. The TensorFlow tokenizer divides text input into word units and converts them to numerical sequences that feed neural networks for additional evaluation.

Text segmentation forms the first step of tokenization through which a sequence of words is split into separate entities according to punctuation marks and other word separators. A tokenized version of "The movie was great" produces the following sequence of "The" "movie" "was" "great". The model operates independently on each word that it identifies as a token during its analysis process. The tokenization process advances by dividing linguistic units into sub-word fragments, particularly for morphologically complex languages. Within the phrase" unhappiness" tokenization produces three individual segments" un," "happy," and "ness."

The processing sequence shifts to the numerical conversion of each token after tokens are extracted from the text. The models need numerical algorithms as their input because they do not work with text content directly. The TensorFlow tokenizer provides a mechanism that assigns exclusive integers that match the tokens found in the dataset. The numerical indices found in these integers allow for identifying token locations within a dictionary that works as a token lookup system. The tokenizer trans- forms "movie" into integer 14 while assigning "great" the value of 27. The model benefits from these mappings which enable it to use a standardized token collection that is optimized for efficient processing.

The TensorFlow tokenizer provides two functions to its users: token conversion to integers and padding of sequences to match uniform lengths between input texts. The padding process protects the efficient operation of shorter sequences since it expands them to reach the length of the longest sequence in the dataset. The proper functioning of LSTMs and Bi-LSTMs depends on receiving sequences of uniform length as input therefore this step proves essential.

The tokenizer can eliminate stop-words because those include ordinary words including "the" or "and" or "is" without substantial sentiment value. The removal of stop- words lets the model concentrate on keywords that have more influence on sentiment values. The TensorFlow tokenizer implements features that let users modify the tokenization algorithm to suit their situations by either handling specific characters or adopting field-specific lexical resources.

The outcome of tokenization produces an integer sequence that maps original text tokens into numerical values. After tokenization, the sequence of integers is prepared for input into machine learning models especially Bi-LSTMs or transformers so they can perform sentiment classification and additional NLP functions. Any NLP pipeline depends on tokenizers because these tools

transform unprocessed textual material into numerical structures which deep learning models require for effective processing.

Tokenization serves as an important process for making text ready to be utilized by machine learning models. The TensorFlow tokenizer enables text conversion into manageable processed chunks that allow models to learn from the data. The process of tokenization organizes data while managing different sequence lengths and performs textual data conversion into numerical values which NLP models require for effective operation. The tokenizer provides flexible text pre-processing that uses padding features together with stop-word functionality to appropriately prepare the text for model analysis of essential data aspects.

3.1.3 Padding Sequences

Long Short-Term Memory (LSTM) network training depends primarily on input data being of equal length throughout the process. Recurrent neural networks (RNN) of which Long Short-Term Memory (LSTM) networks are a type specialized in processing sequential data consisting of text series as well as other ordered information structures. The efficient functioning of this model depends on having input sequences with set lengths which enable the model to discover valuable patterns. In text preprocessing operations padding sequences solve this problem through a method that uniformly expands all text sequences to have uniform length lengths.

Multiple real-life applications including sentiment analysis demonstrate varying sequence lengths in their input data. Movie title sentiment classification requires analysis of titles which can differ in length between ”The Lord of the Rings: The Return of the King” and ”Up.” LSTM net- works face training challenges because they need identical sequence lengths for proper batch processing. Padding represents the solution for dealing with this problem by adding special tokens such as zeros or placeholders to reduce the gap between short sequences relative to the longest sequence in the dataset.

The implementation of padding plays an essential role in allowing batch processing because efficient training depends on it. Deep learning models utilize batch processing through which multiple input sequences can be processed simultaneously by holding the power of parallel computing. Without padding the training process would become slower and less efficient since different length sequences would need to be processed separately. The uniform padding of all sequences enables LSTM networks to execute their operations in parallel which results in faster training along with improved resource utilization.

Padding the data maintains the time-based relationships that exist between various sequences. Within a sentence, the organization of words validates the sequence as well as situational position. Padding extends the shorter sequences

by maintaining their original structure through the addition of neutral values in the blank spaces. The model concentrates on sentiment expressions inside sequences by disregarding unimportant sequence length changes through this padding technique.

The most frequent padding sequence approach is zero padding. The procedure of zero-padding adds zeros either before or after sequence elements to reach target lengths. Padding operations favor the usage of zeros but alternative content and dedicated tokens can fill in as padding elements according to model or task requirements. Zero- padding serves as a standard method for NLP tasks because the model should maintain its training process even while the zeros behave properly during training sessions.

The application of padding occurs on both sequence ends through pre-padding and post-padding methods. Padding occurs before the sequence with zeros inserted at the starting positions in pre-padding whereas padding takes place after the sequence through end positions addition in post- padding. The selection approach between pre-padding or post-padding depends on each task requirement along with the required model behavior. Post-padding serves better than pre-padding for sequential data because it maintains the original order of the data when the positions of elements play a crucial role such as in time-series forecasting.

The process of training requires proper attention to how padding tokens should handle metadata. The model re- mains unaware of padded tokens because they remain concealed under a special masking technique. The model receives information through the mask which specifies that selected tokens contain no value for prediction or gradient calculation during backpropagation. This process protects the model's learning ability regarding meaningful data patterns because the padding material stays separate from actual information.

Research shows that padding sequences serve as a necessary preprocessing requirement for using LSTM net- works with tasks requiring input data of different lengths. Padding sequences to match uniform lengths helps the model achieve higher computational efficiency while directing attention to essential data components during training processes. The LSTM model requires padding which allows the sequential processing of data in parallel while ensuring that the model achieves efficient learning across the complete dataset despite each sequence having different lengths.

3.1.4 GloVe Embeddings

The fundamental problem in natural language processing focuses on transforming semantic word content into machine-learning acceptable numerical values. Numerical processing requires data conversion for text contents like movie titles or user reviews because machine learning algorithms operate solely on numerical data. GloVe embeddings serve as one of the

fundamental methods to perform this process. Through pre-training GloVe (Global Vectors for Word Representation) trans- forms words into extensive numerical vector forms. Word embeddings determine semantic meanings by processing large textual information to yield comprehensive numerical word representations.

During its operation, GloVe relies on matrix factorization to divide word co-occurring matrices into simplified word vector representations. The vectors are engineered to maintain data connections between words so that words sharing similar meanings or contexts will appear near each other in this vector profiling. In the GloVe embed- ding space the pair of words ”king” and ”queen” appear close to one another since both terms share royal gender associations. The proximity of the words “dog” and “cat” occurs because these terms belong to the pet and animal semantic groupings. GloVe embeddings display words as vectors to discover word meanings which thus enables models to process both word elements and their textual relational context.

Pre-trained GloVe embeddings obtain their main strength from their training against extensive Wikipedia and Common Crawl datasets that strengthen their linguistic understanding. Due to previous training GloVe embeddings acquire complex semantic and syntactic word information needed for sentiment analysis tasks and machine translation along with text classification. The GloVe embeddings serve as an essential tool for sentiment analysis models because they help the system identify word sentiment through context information without requiring new training.

The conversion procedure for turning words into numerical vectors through GloVe embeddings follows a simple process. The data word becomes linked to one vector within the pre-trained GloVe embeddings that contains a specific length. These vectors achieve their number of dimensions through hundreds of characteristics that define each word. In a 300-dimensional GloVe embedding the system provides vectors composed of 300 numbers that represent different aspects related to word meanings and their connections with other words.

The word embedding matrix takes pre-trained GloVe vectors as its data input which assigns a row to each word while containing the embedded values of that word. The matrix input serves as training data to be processed by LSTMs or Bi-LSTMs to perform sentiment classification tasks. The model achieves better sentiment analysis results because embeddings establish semantic connections between words in text which help the model automatically recognize comparable words.

GloVe embeddings work well because they take in information about word co-occurrence throughout the entire text collection while also understanding word relation- ships within nearby textual sections. The dual-context algorithm behind GloVe embedding lets the vectors rep- resent both syntactic

and semantic elements thus creating robust vectors that strengthen model performance.
GloVe embeddings serve as critical assets for NLP systems because they transform words into high-dimensional numerical vector spaces that keep semantic meanings in- tact. The embeddings developed from large corpus training produce effective word representations that demonstrate word connectivity abilities that enhance machine learning text understanding. GloVe embeddings enable models to conduct sentiment analysis through language comprehension thus achieving better and more purposeful prediction results.

3.2 Model Architecture

The following model layout consists of:

3.2.1 Embedding Layer

The embedding layer serves as a fundamental component because it changes raw text data into numerical forms that the model effectively handles. The model incorporates GloVe embeddings of 100 dimensions that were trained previously. GloVe acts as a prevalent method that produces word vectors that determine semantic interrelationships between words. Before deploying in their operations, the embeddings receive training through huge corpora such as Wikipedia or Common Crawl while maintaining each word as a vector with 100 dimensions that conveys its corpus-specific meaning.
Pretrained GloVe embeddings provide sentiment analysis systems with the ability to draw from linguistic knowledge extracted from extensive text corpora during the classification of movie titles. Such preprocessing works best with restricted data sets because it allows the model to skip creating its word representations. The system focuses only on detecting word-to-emotional relationship patterns without needing to build word-meaning representations. Each word receives a 100-dimensional embedding treatment which strikes a good compromise between computational speed and semantic depth therefore the model extracts core word properties within an operable dimensionality. The foundation built in this layer enables downstream model components to work efficiently during training since it effectively represents the input text.

3.2.2 Bidirectional LSTM Layer

The sentiment analysis model architecture heavily relies on the Bi-LSTM layer to function effectively. The Bi-LSTM layer operates differently from basic LSTM models since it checks both upcoming and prior relations between elements of textual information. Sentiment analysis requires this framework

because word or phrase meanings usually stem from their context which both precedes and follows them. The Bi- LSTM examines both semantic directions to recognize positive sentiment from "love" in the first segment while detecting negative sentiment from "disappointing" at the ending of the sentence "I love this movie, but the ending was disappointing."
The model becomes more skilled in text comprehension when it fluidly analyzes connections in both downward and upward directions which produces better outcomes in predictions. Through its sequential analysis, the Bi-LSTM layer evaluates words in the text chronologically as it considers both previous and succeeding words to achieve superior sentiment detection capability. The usage of this approach delivers excellent results when classifying movie titles because the sentiment of the complete title benefits from both the start and end context.

3.2.3 LSTM Layer

The LSTM (Long Short-Term Memory) layer inside the model is the key element for sequence data processing by maintaining long-term dependencies. After embeddings enter the model structure, the LSTM layer continues processing learned features from previous model layers. The design of LSTM layers deals with the traditional RNN vanishing gradient issue which makes them ideal for tasks requiring contextual understanding of sequential data especially when doing sentiment analysis. The LSTM layer integrates multiple memory cells which possess the ability to keep information cached for extended durations. Memory cells within LSTMs maintain the ability to sense repeating patterns across multiple time intervals, making them superior for emotion classification where word meaning depends on surrounding text. In the sentence "The movie was great" the positive sentiment expressed by "great" requires understanding of the whole statement context.
The LSTM layer enables the model to keep significant sequence information while ignoring unnecessary data, which leads to better sentiment analysis results and enhanced model performance at large.

3.2.4 Dropout Layer

The dropout layer is an essential regularization technique that stops overfitting during the training stages. The model learns noise along with data patterns because it becomes too complex, leading to reduced generalization ability for new unforeseen data. For each training step, the dropout layer executes a randomized process to set a specific fraction of input units to zero values. Each training step will disable randomly selected neurons to 50.

3.2.5 Fully Connected Layer

A deep learning model depends on its Fully Connected (FC) Layer to develop meaningful features that play a vital role in sentiment analysis tasks. The FC layer within the Bi-LSTM model receives high- level sequential features from Bi-LSTM and employs further processes to determine ultimate sentiment predictions. The FC layer gathers information about word sequences that Bi-LSTM layers previously extracted be- fore generating higher-order features that boost textual sentiment comprehension. The network establishes complete connectivity between each neuron in the previous layer with each neuron within the FC layer during the calculation process.

Sentiment analysis depends on the FC layer to develop a more precise model concept that explains how single words and phrases create an impact on the movie title sentiment. The model learns complex associations between input elements through this layer therefore it achieves higher accuracy when predicting sentiment. The features extracted from the previous layers become the FC output which determines the final predicted sentiment category from Positive to Neutral to Negative. The last layer enhances the model's overall performance since it aids data generalization for unknown data points.

3.2.6 Output Layer (Softmax Activation, 3 Classes)

The output layer has a critical function in classifying models primarily during sentiment analysis of movie titles. The output layer implements Softmax activation because it addresses multi-class classification by building probability distribution from its outputs. By applying the Softmax function the output values become probabilities in the range between 0 and 1 while maintaining a value total of 1 thus simplifying interpretation of model predictions. In this sentiment analysis system, the model decides where each movie title belongs to one of three available sentiment categories which are Positive Neutral, and Negative. The movie title receives classification as the category which holds the maximum probability score. A movie title receives a Positive sentiment when people view it favorably but Neutral represents titles that maintain non-discriminatory feelings and Negative sentiment illustrates unlinked movie titles. The classification system follows a three-fold structure for public sentiment analysis toward movie titles to establish audience preferences and expected values.

The Softmax application helps the model establish a probability measurement for each title to belong to its three sentiment classifications. The output layer enables the model to deliver correct sentiment decisions about each movie title thus resulting in precise sentiment classification results

3.3 Training Configuration

A framework of configurations has been applied to the model during training to achieve superior performance when analyzing movie title sentiment. Adam optimizer functions as the preferred choice for deep learning model training because it adapts its learning rate and achieves high convergence efficiency. Adam optimizer starts with high learning rate values because of its exponential decay function to expedite initial parameter convergence before remodeling the rate more slowly during training. The implementation ensures that the model does not bypass its optimal solution point as it leads to more accurate convergence.

The model utilizes the loss function known as Sparse Categorical Cross entropy because it proves ideal for multi- class scenarios that involve sentiment analysis. The sparse categorical Cross entropy function accepts integer labels for Positive, Neutral, and Negative so that the model can calculate its loss directly without needing one-hot encoding.

A batch size of 32 has been selected for the model because it optimizes both performance efficiency and generalization quality. A training period of 20 epochs allows the model to properly learn data patterns before early stopping halts the process because performance stagnation occurs in validation sets.

3.3.1 Optimizer

The model utilized Adam optimizer which decreased its learning rate at an exponential pace. Novel deep learning model training applications utilize Adam optimizer because it merges AdaGrad and RMSProp functionalities while maintaining efficient processing and adaptive learning rate updates. The major strength of Adam lies in its adaptive learning rate process that determines separate adaptation values for each parameter through gradient first and second-moment estimates. Adam proves most effective in model training systems that work with extensive datasets and extensive parameter dimensions because it prevents both slow convergence and overfitting issues. Our training process achievement aimed at convergence included the use of a learning rate that decayed exponentially. The method allows training progression by lowering the learning rate from larger initial values towards smaller ones which correspond to the periods when the model starts far from its optimal condition and approaches convergence. By adjusting its learning rate dynamically the model avoids missing the eventual stable minimum so performance and theoretical predictive capabilities increase. Adam enabled efficient model learning during training with its exponentially decreasing learning rate to keep the process stable at every stage.

3.3.2 Loss Function

The proposed sentiment analysis model applies the Sparse Categorical Cross entropy loss function because this method shows excellence in multi-class classification problems when each input contains one of the multiple classes. The proposed model performs sentiment analysis of movie titles through Positive, Neutral, and Negative classification. Sparse Categorical Crossentropy proves to be an ideal choice for this problem since it operates optimally with integer labels while categorical cross entropy needs one-hot encoding.

The model computes the difference between actual class assignments and probabilities the model generates from its prediction calculations through the loss function. The loss evaluation functions operate on a sample-by-sample basis to calculate negative logarithm values through predicted probability estimations for actual class values. Minimizing loss constitutes the main goal where the model must predict high sentiment probabilities from the correct class while simultaneously producing low probabilities for classes with incorrect matches. When utilizing this loss function we allow the model to perform backpropagation adjustments of its parameters and thereby enhance its accuracy in sentiment classification of movie titles.

Sparse Categorical Crossentropy acts as an optimization tool that enforces penalty on incorrect predictions to enable the model to learn sentiment categorization properly during training sessions thereby enhancing the sentiment classification results.

3.3.3 Batch Size

During deep learning processes the batch size functions as an essential hyperparameter which specifies how many training samples the model should process during each forward-backward make. The Bi-LSTM model with GloVe embeddings needs a batch size of 32 during the training process. The selection of the correct batch size remains vital because it determines both speed and accuracy during model training.

For optimal performance and computational speed both together the selected batch size stands at 32. When using small batch sizes such as 16 or 32 the model obtains multiple parameter updates during training which enhances its generalization capabilities as well as lowers overfitting risks. The use of smaller batches enables faster training because it enables the model to process smaller data sets simultaneously which needs less memory storage.

How much memory the system needs only increases when the batch size increases however training stability improves during this time. The choice of a batch size of 32 helps the model achieve consistent updates and supports both memory efficiency and resource utilization. The application of 32 as the batch size works well for Bi- LSTM models by maintaining both training

speed and model performance to enable high-accuracy learning of sequential dependencies.

3.3.4 Epochs

The training process continues for 20 epochs so that the model achieves appropriate data learning without excessive computational time. A complete execution of the whole dataset for parameter adjustment exists as one epoch in the training process. The selected training time of 20 epochs enables proper pattern discovery with optimized resource usage. Training the model beyond 20 epochs runs the risk of overfitting because the model develops memorization skills for training data at the expense of unseen data recognition. Early stopping functions as an overfitting prevention technique when applied during training. When the model reaches a point where validation metrics decline on the evaluation data Early stopping terminates the training process because it signifies overfitting has occurred. Early termination of training stops the model from acquiring excessive training data details that specifically match its training set while improving its ability to recognize new unseen data. Using early stopping as a strategy enhances model robustness and reduces training expenses while making operations more efficient thus retaining excellent performance outcomes for real-world scenarios.

3.4 Evaluation Metrics

The evaluations of Bi-LSTM and GloVe embedding-based sentiment analysis models occur through several performance metrics which deliver a detailed understanding of model effectiveness. The performance evaluation system uses Accuracy alongside Precision and Recall together with F1-Score.

3.4.1 Accuracy

The fundamental measurement of model success is Accuracy since it determines the correct prediction rate from the model output. The metric provides general performance insights however it does not adequately assess class imbalances during operation.

3.4.2 Precision

The value of precision describes how many correct results appear among all the positive predictions made by the model. The correct classification of positive titles requires precision because false positives mistakenly identify negative or neutral titles as positive which leads to erroneous analysis.

3.4.3 Recall

The model correctly identifies what per- centage of actual positives within the dataset. The detection of most positive sentiments plays a vital role in

customer feedback analysis so sentiment analysis requires high recall performance.

3.4.4 F1-Score

F1-Score represents the harmonic mean that combines precision and recall to allow evaluation of a single balanced metric. The metric proves very effective for unbalanced data sets because it ensures the reduction of false positives and false negatives together. These metrics provide an assessment that combines accuracy and reliability for the model to classify sentiments in movie titles effectively.

4. Results and Discussion

4.1 Training Configuration

The effectiveness of the sentiment analysis model gets evaluated through multiple essential performance metrics which offer a total understanding of prediction abilities. The model assessment utilizes accuracy precision and recall in addition to the F1-score to measure distinctive performance characteristics. The model clarity depends on accuracy which determines how many times the platform correctly identifies sentiment labels among all classifications. The measurement of accuracy fails to deliver enough performance data when working with datasets containing unbalanced proportions between classes. The precision metric determines what percentage of identified positive sentiment predictions turn out to be accurate as shown in Table 1. The metric enables assessment of how frequently a model detects proper positive sentiments. Recall demonstrates how well the model predicts actual positive sentiment labels among all identified positive sentiment labels. Recall competency aims to prevent the model from disregarding major positive instances. The F1-score calculates accuracy from the harmonic mean of precision and recall to achieve balanced measurement. The measure proves beneficial when the distribution of classes is not evenly balanced. By utilizing the Bi-LSTM model these accuracy metrics supply an all-inclusive assessment of how accurately the model detects sentiment categories from movie titles while identifying avenues for better performance.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Baseline LSTM	82.5	80.3	81.0	80.6
Bi-LSTM (No GloVe)	87.2	86.5	87.0	86.8
Fine-Tuned Bi-LSTM (GloVe)	94.6	95.1	94.2	94.6

Table 1: Performance comparison of different models

4.2 Error Analysis

Effective results emerged from the Bi-LSTM model which used GloVe embeddings in the evaluation phase obtaining superior accuracy rates compared to conventional sentiment analysis models. The model struggled

specifically in differentiating Neutral and Positive sentiments though it performed well in other aspects. Unfortunately, the analysis proved challenging because it had to process numerous ambiguous movie titles lacking explicit sentiment or exhibiting multiple interpretation possibilities. Such challenges occur frequently with metaphorical playful or abstract movie titles that require interpretation.

The model encounters issues when processing the title ”A Beautiful Day” because its interpretation can shift between positive and neutral depending on the presentation context. Under these circumstances, the model shows behavior that misidentifies Neutral titles as Positive because of missing explicit sentiment-bearing words or indirect sentiments. The model encounters challenges while categorizing Positive sentiment through movie titles containing words such as “great” or “amazing” because it struggles to accurately classify sentiment from neutral words like “day” or “journey”. The model’s information processing capacity demonstrates challenges in understanding sentiment context especially when analyzing complex problems such as sentiment classification.

The model struggles to achieve reliable Neutral/Positive distinction in ambiguous cases within short text sequences of movie titles because it is challenging to model subtle distinctions properly in such contexts. Bi-LSTMs excel at sequential data processing yet perform worse in distinct tasks that need subtle distinctions due to restricted contextual in- formation found in short unclear movie titles. The deep word representations from GloVe embeddings could face limitations when evaluating the subtle emotional expressions present in deliberately confusing or challenging film titles.

Future development initiatives target the enhancement of model sensitivity in detecting slight differences between sentiments while working with indistinct cases. The successful completion of this process depends strongly on data augmentation methods. Artificial enlargement of our training dataset introduces the model to multiple titles that depict different sentiment strengths. The model’s ability to correctly identify ambiguous titles will improve through this method of adjustment because it promotes better handling of unclear examples. The model will gain better generalization through unseen data by employing techniques that include paraphrasing and synonym substitution for example creation.

The model can achieve improved performance through attention mechanisms which allow it to determine crucial parts of a title during prediction time. Through attention mechanisms, the model distributes various levels of significance to individual elements found across its input sequence. Such a technique provides valuable information to perform sentiment analysis because some words or phrases possess greater importance in determining emotional content. Within the title ”A Beautiful Day” the term ”beautiful”

exerts greater influence on sentiment evaluation compared to ”day.” The implementation of attention mechanisms enables the model to understand what sections in the title hold the most sway for sentiment categorization thus improving its accuracy during unclear sentiment assessments.
The model has shown excellent general performance yet requires improvement in identifying Neutral versus Positive sentiments in doubtful titles. The model's achievement success relies on data augmentation together with attention mechanisms to detect precise sentiment predictions during ambiguous sentiment expressions.

5. Conclusion and Future work

Research findings demonstrate that the Bi-LSTM model enabled with GloVe embeddings shows high effectiveness at analyzing sentiments in movie titles. The main purpose of this study focused on demonstrating Bi-LSTM networks as effective pattern extractors from sequential data by understanding word semantic relationships through GloVe pre-trained embeddings. The Bi-LSTM model surpassed traditional LSTM models in accuracy and precision as well as recall and F1- score when enhanced with GloVe embeddings based on experimental testing. Results from the fine-tuned Bi-LSTM network resulted in accuracy performance above 90%. GloVe embeddings contributed to better system performance because they delivered word semantic meaning information which enhanced model capabilities. The pre-trained GloVe embeddings derived from Wikipedia and Common Crawl helped the model establish contextual word meaning understanding that became necessary for recognizing sentiments in movie titles that could not be understood through single words. Using GloVe embeddings allows the model to detect fine-grained meaning variations which strengthens its ability to properly categorize sentiments as Positive, Neutral, or Negative based on total movie title contexts.
Multiple contributing factors including Bi-LSTM's sequence handling features and GloVe's embedded word meaning representation along with padding for sequence length flexibility led to the model's successful performance in sentiment analysis. The preprocessing activities consisting of text normalization along with tokenization and padding steps ensured data suitability for effective model learning that resulted in better performance.
The proposed approach specifies multiple areas for upcoming studies that aim to refine and advance the model design. The model needs a larger and more diverse movie title dataset because such expansion will help it learn from unknown inputs. The model achieves better results with diverse data since it reduces errors when processing titles across various genres languages and cultural backgrounds. The analysis of future work should include transformer-based models such as BERT (Bidirectional Encoder Representations from

Transformers) because these methods demonstrate outstanding performance in multiple NLP applications.

Declarations

Competing Interests

The authors declare that they have no competing interests.

Authors' Contribution

All the authors have contributed in the paper.

References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL- HLT, 2019.

[2] A. Vaswani, N. Shazeer, N. Parmar, L. Uszkoreit, L. Jones, A. N. Gomez, K. Kaiser, and I. Polosukhin, "Attention is All You Need," NIPS, 2017.

[3] X. Zhang, J. Zhao, and Y. LeCun, "Character-level Convolutional Networks for Text Classification," NIPS, 2015.

[4] A. Graves, S. Fernandez, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," ICML, 2006.

[5] D. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv preprint arXiv:1412.6980, 2014.

[6] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," EMNLP, 2014.

[7] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," NIPS, 2013.

[8] Y. Kim, "Convolutional Neural Networks for Sentence Classification," arXiv preprint arXiv:1408.5882, 2014.

[9] B. Liu, "Sentiment Analysis and Opinion Mining," Synthesis Lectures on Human Language Technologies, 2012.

[10] H. He and E. Garcia, "Learning from Imbalanced Data," IEEE Transactions on Knowledge and Data Engineering, 2009.

[11] I. J. Goodfellow, Y. Bengio, and A. Courville, Deep Learning, MIT Press, 2016.

[12] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, 1997.

[13] P. Sundermeyer, R. Schlu¨ter, and H. Ney, "LSTM Neural Networks for Language Modeling," INTERSPEECH, 2012.
[14] F. Chollet, Deep Learning with Python, Manning Publications, 2017.
[15] A. Graves, "Supervised Sequence Labelling with Recurrent Neural Networks," Springer, 2012.
[16] M. Schuster and K. Paliwal, "Bidirectional Recurrent Neural Networks," IEEE Transactions on Signal Processing, 1997.
[17] A. Zhang, S. Zhao, and Y. LeCun, "Deep Neural Networks for Text Classification," IEEE Transactions on Neural Networks, 2015.
[18] G. E. Hinton, L. Deng, D. Yu, et al., "Deep Neural Networks for Acoustic Modeling in Speech Recognition," IEEE Signal Processing Magazine, 2012.