OPTIMAL LONG SHORT-TERM MEMORY (LSTM) HYPERPARAMETERS SELECTION USING PASTORALIST OPTIMIZATION ALGORITHM FOR CUSTOMERS SENTIMENT ANALYSIS

By

SHEHU, Safiya Aliyu
MTech/SICT/2018/8598

A THESIS SUBMITTED TO THE POSTGRADUATE SCHOOL FEDERAL UNIVERSITY OF TECHNOLOGY, MINNA, NIGERIA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF MASTER OF TECHNOLOGY IN COMPUTER SCIENCE

JANUARY, 2022

ABSTRACT

Users usually express their sentiments online which has great influence on the product customers buy. Sentiment analysis is the computational study of people’s emotions toward an entity. Sentiment analysis often faces the challenge of insufficient labeled data in Natural Language Processing (NLP) and other related areas. Long Short-Term Memory (LSTM) is one of the deep learning models widely used by researchers in solving sentiment analysis problem. However, they possess some drawbacks such as longer training time, more memory for training, easily overfits, and sensitivity to randomly generated parameters. Hence, there is a need to optimize the LSTM parameters for enhanced sentiment analysis. This research proposes an optimized LSTM approach using a newly developed novel Pastoralist Optimization Algorithm (POA) for enhanced sentiment analysis. The model was used to analyze sentiments of customers retrieved from Amazon product reviews. The performance of the developed POA-LSTM model shows optimal accuracy, precision, recall and F1 measure of 77.36%, 85.06%, 76.29%, and 80.44% respectively when compared with other optimization algorithm which is Biogeography-based optimization algorithm (BBO) with 76.10%, 78.64%, 83.50%, 80.99% and also the LSTM model with 71.62%, 78.26%, 74.23%, and 76.19% respectively. It was also observed that POA with 20 pastoralist population size performs better than other models with 10, 15, 25, and 30 populations.

TABLE OF CONTENTS

Cover Page

Title Page i

Declaration **Error! Bookmark not defined.**

Certification **Error! Bookmark not defined.**

Acknowledgement **Error! Bookmark not defined.**

Abstract ii

Table of Contents iii

List of Tables vi

List of Figures viii

Abbreviations viii

CHAPTER ONE 1

1.0 INTRODUCTION 1

1.1 Background to the Study 1

1.2 Statement of the Research Problem 2

1.3 Aim and Objectives of the Study 3

1.4 Significance of the Study 3

1.5 Scope of the Study 3

CHAPTER TWO **5**

2.0 LITERATURE REVIEW **5**

2.1 Preamble 5

2.2 Sentiment Analysis 6

2.2.1 Rule-based 7

2.2.2 Machine learning-based 7

2.2.3 Lexicon-based 7

2.3 Machine Learning Algorithms 8

2.3.1 Traditional machine learning algorithms 9

2.3.1.1 Naïve bayes (NB) 9

2.3.1.2 Support vector machine (SVM) . 9

2.3.1.3 Logistic regression (LR) 10

2.3.1.4 Random forest (RF) 11

2.3.2 Deep learning (DL) 11

2.4 Pastoralist Optimization Algorithm (POA) 14

2.5 Biogeographic-Based Algorithm 16

CHAPTER THREE **20**

3.0 MATERIALS AND METHODS **20**

3.1 Research Design 20

3.2 Collecting data and Extracting Features 21

3.3 Preprocessing 23

3.4 Problem Formulation and Model Design 24

3.5 Optimized POA-LSTM Model Implementation 25

3.6 Performance Evaluation 26

3.6.1 Accuracy 26

3.6.2 Precision 27

3.6.3 Recall 27

3.6.4 F1 Score 27

CHAPTER FOUR 29

4.0 RESULTS AND DISCUSSION 29

4.1 LSTM Model Results 29

4.2 POA-LSTM Model Results 34

4.3 BBO-LSTM Model Results 39

4.4 Performance Evaluation Results 43

CHAPTER FIVE 46

5.0 CONCLUSION AND RECOMMENDATION 46

5.1 Conclusion 46

5.2 Recommendation 46

5.3 Contribution to Knowledge 47

REFERENCES 48

APPENDIX 53

LIST OF TABLES

Table	Title	Page
2.1	Pastoralist Optimization Algorithm (POA)	16
2.2	Biogeographic-Based Optimization Algorithm	18
3.1	Set-Up Parameters for POA-LSTM and LSTM models	24
3.2	POA-LSTM Implementation steps	26
3.3	Parameters settings for LSTM and POA-LSTM models	28
4.1	POA-LSTM Optimization	38
4.2	BBO-LSTM Optimization	43
4.3	Performance Evaluation	45

LIST OF FIGURES

Figure	Title	Page
2.1	Sentiment Analysis Categorization	7
2.2	LSTM Deep Learning Model	13
2.3	Flowchart of Biogeographic-Based Algorithm	18
3.1	Research Methodology	20
3.2	Methodology	21
3.3	Sample of Dataset	22
4.1	Training Parameters of LSTM with 50 Nodes	30
4.2	Training Progress of LSTM with 50 Nodes	30
4.3	Confusion Matrix of LSTM with 50 Nodes	31
4.4	Training Parameters of LSTM with 100 Nodes	32
4.5	Training Progress of LSTM with 100 Nodes	33
4.6	Confusion Matrix of LSTM with 100 Nodes	34
4.7	Training Parameters of POA-LSTM	35
4.8	Training Progress of POA-LSTM	36
4.9	Convergence Curve of POA-LSTM	37
4.10	Confusion Matrix of POA-LSTM	39
4.11	Training Parameters of BBO-LSTM	40
4.12	Training Progress of BBO-LSTM	41
4.13	Convergence Curve of BBO-LSTM	42
4.14	Confusion Matrix of BBO-LSTM	44
4.15	Accuracy, Precision, Recall and F1-Score Graph	47

ABBREVIATIONS

Abbreviation	**Meaning**
ABC	Artificial Bee Colony
AI	Artificial Intelligence
BA	Bat Algorithm
BBO	Biogeographical-Based Algorithm
CNN	Convolutional Neural Network
DBN	Deep Belief Network
DNN	Deep Neural Network
GA	Genetic Algorithm
GOA	Grasshopper Optimization Algorithm
HIS	Habitat Suitability Index
LR	Logistic Regression
LSTM	Long-Short Term Memory
ML	Machine Learning
NB	Naïve Bayes
NLP	Natural Language Processing
POA	Pastoralist Optimization Algorithm
PSO	Particle Swarm Optimization
RF	Random Forest
RNN	Recurrent Neural Network
SIV	Suitability Index Variable
SVM	Support Vector Machine

CHAPTER ONE

1.0 INTRODUCTION

1.1 Background to the Study

The action of purchasing, selling, transferring, or trading items, services, or even information through computers and networks, most especially the Internet and internet, is referred to as electronic commerce also written as e-commerce. (Turban *et al.,* 2011). Although the terms e-commerce and e-business are frequently used interchangeably, they are not interchangeable. E-business encompasses a broader span of the e-commerce process. E-business includes "customer service, cooperation in collaboration with business associates, online learning, and computerized interaction within a corporation," in addition to e-commerce. Regardless of whatever term you choose, they both involves the purchase online and sale of goods and services. The majority of customers express their opinions on E-commerce websites where they purchase thing such as Amazon, CNET, epinion, zdnet, consumer review, IMDB, and so forth (Krishna *et al.*, 2019).

Companies are advertising their products through Internet media. Customers' Internet reviews might now influence a product's buying choice. Various e-commerce businesses, including Amazon, eBay, Flipkart, and Walmart, analyze their customers' feedback in various ways (Waldron, 2016).

One of the world's largest Internet retailers is Amazon. Before purchasing a purchase on Amazon, many people look at the products and reviews of the product. However, Amazon reviews are not always of items, but rather a combination of product and service evaluations (Bhatt *et al.,* 2015). Purchasing a thing is often a two-way street involving two entities: customers and company owners (Heller & Parasnis, 2011). Consumers frequently utilize reviews to make purchasing decisions, whereas companies want not just to sell their products but also to obtain feedback in the form of consumer reviews. Consumer evaluations regarding purchased items that are published on the Internet have a significant influence (Flanagin *et al.*, 2014). Because others frequently have a large influence on our ideas, behaviors, perception of reality, and the choices customers make, human nature is typically constructed to make decisions based on studying and benefiting from other consumer experience and views (Kim & Srivastava, 2007). As a result, anytime customers make a decision, they solicit opinions from others. This is true not only for customers, but also for businesses and institutions. Consumers' means of expressing their ideas and sentiments have altered in recent years as a result of developments in social networks, virtual communities, and other social media communities (Zhang & Tran, 2019). Discovering large amounts of data from unstructured data on the web has become an important challenge due to its importance in different areas of life (Bolden & Moscarola, 2000).

1.2 Statement of the Research Problem

Already many deep machine learning models, such as the Long Short-Term Memory model (LSTM), have been widely proposed to solve sentiment analysis issues (LSTM). They do, however, have certain limitations, longer training periods, larger training memory, easier overfitting, and vulnerability to randomly produced parameters are all factors that affect their performance. As a result, improving the deep learning model parameters is required for improved sentiment analysis. Pastoralist Optimization Algorithm is a newly created meta-heuristic that has proved effective in tackling complicated optimization problems attributable to the algorithm's advanced search approach (Abdullahi *et al.*, 2018; 2019). The performance of the deep learning model was enhanced using POA in this investigation to enhance the effectiveness of the sentiment analysis model, which has not been done by previous researchers.

1.3 Aim and Objectives of the Study

The aim of this study is to create an optimal consumer sentiment analysis model by combining the pastoralist Optimization Algorithm (POA) with deep learning methods.

The objectives are to:

i. Formulate the sentiment analysis problem and design an LSTM deep learning model for sentiment analysis
ii. Optimize its parameters using POA and implement the model in (i).
iii. Evaluate the model's performance using accuracy, Recall, F1 Score and Precision.

1.4 Significance of the Study

This study will not only improve sentiment analysis task but also presents an opportunity to explore more market research, market analysis, product review, and product feedback, as well as studying the status of a product, market research, market analysis, product review, and product feedback. The important benefit of this system is that it is not domain-specific. As a result, the same model may be trained for both product and service ratings without compromising performance.

1.5 Scope of the Study

This work is limited to analyzing customer’s reviews or post written in English only and the model will only be optimizing two parameters which is the learning rate and number of hidden unit out of the other ten parameters used. The approaches used in this study is limited to MATLAB programming language machine learning environment. The algorithms implemented in this thesis were developed on a 64-bit Windows 10 system with 232 GB ROM, 8GB RAM and 2.50Ghz duo core processor speed.

CHAPTER TWO

2.0 LITERATURE REVIEW

2.1 Preamble

A summary of this chapter related to this thesis is provided. Four different sections were used to present all the articles. An explanation of all the machine learning algorithms utilized in this thesis is provided in the first section, the second section provides a review of existing works in which suitable datasets were created, the third section summarizes the different optimization algorithms to be used, the fourth and final section provides a review of existing works on customer's sentiments.

(Schukla, 2018) introduced a technique that evaluates text quality using annotations from scientific articles. Its methodology uses two ways to obtain annotation sentiments. It determines overall sentiment scores by counting all of the annotations produced by the documents. Its issue is that it indicates a complicated link between annotations. A large query knowledge base including metadata is required for the strategy. For hotel reviews, Kasper and Vela, (2018) presented a "Web Based Opinion Mining method". The research presented an assessment method for online user evaluations and comments in objective is to assist quality controls in hotel management systems. It can recognize and retrieve online reviews and only understands German reviews. It does have a multi-topic area and therefore is based on multi-polarity categorization; the system could detect neutral, such as "don't know," and multi-topic cases detected in their corpus to "classify sentiment polarity as neutral."

According to Jeh and Widom (2014), defined data mining as a strategy used to crawl through various web resources to obtain necessary information, allowing a person or a firm to advertise business, comprehend fresh promotions floating on the Internet, marketing dynamics, and so on. There is a rising tendency amongst businesses, organizations and web data mining which allows companies and individuals to gather information and then use that knowledge to their advantage. Technology for data mining is undergoing rapid change, with new and improved approaches being developed all of the time from acquiring whichever information is required. Web data mining technology is broadening the scope of data collection while also increasing worries about data security. There's also a great deal of personal data that is available, and web data mining has aided in keeping the need to protect such information at the forefront of people's thoughts. There are several online tools for site scraping and data extraction that are really useful. However, the main issue that customers confront with them is that they are incredibly pricey. Aditya and Vikesh, (2015) introduced a Recursive RNN-based intelligent model for movie review classification. This is a system for classifying sentiments in natural language, i.e. text. This approach is a sentence-level extension for sentiment categorization. Jan and Mark, (2016) developed a technique for categorizing tweet sentiment. The deep learning method is extended in this study. In this example, they used 2-layer convolution neural networks. In this scenario, the entire work is divided into three subtasks. Anand and Sachin, (2017) introduced CNN as a novel strategy for achieving the same aim with a different approach. SendicNet 6, which is built on symbolic and sub-symbolic AI, has an ensemble of top down and bottom up learning integrated in it (Cambria *et al.*, 2020).

2.2 Sentiment Analysis

Because of its relevance in many areas of life, sentiment analysis has gained in popularity as a study topic over the years. Techniques employed, dataset structure, and rating level are some of the characteristics that can be used to classify sentiment analysis (Xiang & Gretzel, 2010). The diagram depicts many types of sentiment analysis. Some instances of how sentiment analysis can be applied are as follows:

2.2.1 Rule-based

Attempts to rank information extracted from a dataset based on word polarity. Other rules apply, such as negation words, emoticons, idioms, and dictionary polarity (Lee *et al.*, 2009).

2.2.2 Machine learning-based

This entails using an existing dataset to train a sentiment analysis model before deployment (Xiang & Gretzel, 2010).

2.2.3 Lexicon-based

When sentiment polarity is measured in a review or remark utilizing semantic orientation in the evaluation of opinion and subjectivity, it results in either positive or negative sentiment polarity (Collobert *et al.*, 2011).

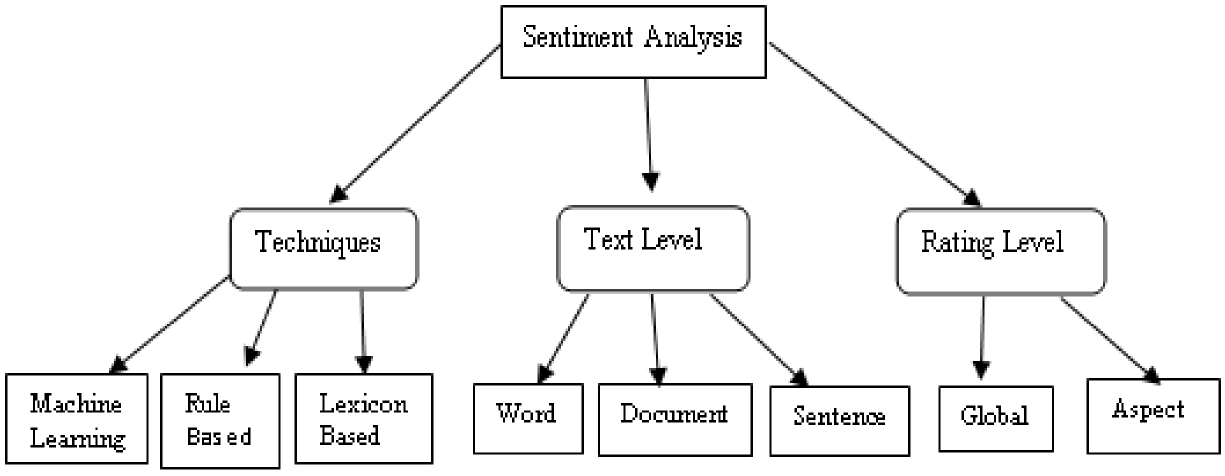


Figure 2.1 Sentiment Analysis Categorization
(Anais *et al.*, 2013)

There are a variety of methodologies used in sentiment classification research projects. This research focuses on reporting on the different sentiment analysis methodologies, with a special deep machine learning approaches. (Wang *et al.*, 2016) built a multilingual attention system model of a network for code translation opinion prediction. The model (LSTM) was utilized to transform every post into a classification purpose by which an ANN model may be derived was employed to extract information from various contexts. (Rashkin *et al.*, 2017) used multilingual connotation frames as the main strategy to forecast attitudes of those social media who use LSTMs utilize social media LSTMs. For successful sentence recognition, an attention-based LSTM was presented by (Wang *et al.*, 2016).

To increase sentiment analysis performance, (Yang *et al.*, 2017) developed a two attention-based two-way LSTM. (Liu & Zhang, 2017) expands the attention model by separating the attention obtained from the left contexts of the target and right contexts of the target. They added numerous gates to better restrict their attention participation. For information gathering, a DNN was proposed (Dou, 2017). Document representation and sentiment analysis were built using a Deep Neural Network cascade with a Long Short Term Neural Network cascade. The investigation of ensemble models for rating the emotional content of financial blog posts and headlines. (Akhtar *et al.*, 2019) was presented. (Chen *et al.*, 2016) product and consumer features and preferences, as well as categorizing their feelings.

2.3 Machine Learning Algorithms

In the ’50s and ’60s, Machine Learning (ML) started as pattern recognition. They got better as more data is increasing by the day so many researchers have used machine-learning algorithms that include deep learning concepts and classical machine learning to detect customer sentiments in messages or posts online. Using deep learning techniques, this part will briefly cover the numerous research linked to sentiment analysis of online material concerning user opinions, feelings, and evaluations toward various subjects and goods.

2.3.1 Traditional machine learning algorithms

2.3.1.1 Naïve bayes (NB)

This algorithm depends on the Bayes Theorem and is particularly well suited to situations with high dimensional input feature space Wikarsa and Thahir, (2015). The theory states that a feature used for classification does not depend on any other features' value. For example, provided with a set of features $X = x_1, x_2, \ldots, x_n$ acquired from review, with the respective target label $Y= y_1, y_2, \ldots, y_k$. Equation 2.1 is assigned by an algorithm to y_i with the maximum posterior probability. The posterior probability is $P(Y|X)$, the likelihood is $P(X|Y)$, Y has the independent probability of $P(y)$, and X has the independent probability of $P(X)$. This algorithm has a limitation in the sense that features being classified are not always independent and is the reason why it is referred to as "Naïve". Given a set of features, the algorithm operates by predicting the classes of those features.

$$P(Y/X) = P(X|Y)) * P(Y)) / (P(X)) \tag{2.1}$$

2.3.1.2 Support Vector Machine (SVM)

This is a well-known method that is best suited for linear classification issues. It is also seen as one of the best supervised algorithms of machine learning Lundeqvist and Svensson, (2017). Originally, it was designed for binary classification problems. With multi-class classification problems, it is extended using a one-against-one or one-against-all strategy by using several binary classifiers on different segments of a problem. Provided with a binary classification problem and a dataset made up of feature vectors $x = (\{x_i\}i=0)^n$ where $x_i \in$

R(N- 1) and their classes $y = (\{y_i\}i=0)^n$ where $y_i \in \{+1, -1\}$, the SVM must complete these two objectives: firstly, a hyperplane must be located along RN-1 separating two subspaces of the input vector. This means that each class has one subspace; The second objective is to increase the margin between the border vectors of the two subspaces and the separating hyperplane. The hyperplane equation is presented in equation 2.2. With w being the vector that defines the hyperplane orientation and b the bias that defines the hyperplanes offset from the origin. SVM is not based on the theory of probability because it defines a clear margin by implicitly mapping to a high-dimensional feature space from input. The equation for SVM is given in equation 2.3.

$$w.x + b = 0 \tag{2.2}$$

$$f(x) = \text{sgn}(w.x + b) \tag{2.3}$$

2.3.1.3 Logistic Regression (LR)

This is a powerful machine learning algorithm for binary classification problems (O'Dea *et al.*, 2015). LR is named after the logistic function because it is based on the theory. A real number is mapped by the logistic or sigmoid function (equation 2.4) to values between 0 and 1 but never exactly at these limits. The real numerical value z is transformed with e representing the base of the natural logarithms. It is extended by collecting binary classifiers through a strategy called "one- vs- all" which estimate the most likely output by analyzing each output separately from other outputs and then selecting the output with the highest probability when dealing with multi-class classification problems. The equation representing LR is given in equation 2.5. The predicted outputs are represented by y, b0 and b1, x being the bias or intercept term and the single input value coefficient (x). Each column of the input data must have a b co - efficient that goes with it (a constant real value) learnt from the training data.

$$1 / (1 + e^{-z}) \tag{2.4}$$

$$y = e^{(b0 + b1 * x)} / 1 + e^{(b0 + b1 * x)} \tag{2.5}$$

2.3.1.4 Random Forest (RF)

This is an ensemble approach in the form of a nearest neighbour predictor. Ensembles improve performance by using a divide-and-conquer strategy. The ensemble methods are based on the idea that a "strong learner" can be obtained by combining a group of "weak learners". The algorithm starts with an equivalent to a weak learner which is a common machine learning approach known as "decision tree" When an entry is entered the data is collected in discrete chunks at the head of that same decision tree as it passes through the tree (Zhang *et al.,* 2018). This means that during training a multitude of decision trees are built because for a classification problem it produces the mode of the classes while for a regression problem the mean prediction of the individual trees. RF makes adjustments accordingly in order to counteract the habit of the decision trees in over-fitting to the training set.

2.3.2 Deep learning (DL)

Deep learning is a more sophisticated Artificial Neural Network (ANN) that uses several deep network layers to learn. Deep machine learning has piqued people's curiosity in recent years because of its potential to resolve issues more quickly and easily than shallow networks. Because of advancements in computers and big data analytics, its adoption is now viable (Zhang *et al.,* 2018). According to (Vateekul & Koomsubha, 2016), deep learning models can solve both unsupervised and supervised issues. Deep Belief Networks (DBN), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN) are three popular models of deep machine learning (Usama *et al.*, 2019). Deep learning has been used to solve a variety of difficult issues, including speech identification, natural language processing (NLP), and computer vision applications such as identification of faces (face recognition). Despite its effectiveness, establishing the proper layers and the number of hidden variables that a hidden layer should have has been one of deep learning's most difficult challenges (Bengio *et al.,* 2013).

Deep learning is being used to handle the difficult challenge of sentiment analysis. Deep learning extracts and transforms features via a cascade of multiple-layer nonlinear processing units. Lower layers learn simple characteristics near to the data input, whereas upper layer learn out more sophisticated characteristics generated from features in the lower layers (Lee *et al.*, 2009). (Wang *et al.,* 2019) described a CNN structured deep network for visual sentiment categorisation dubbed the Deep Coupled Adjective and Noun (DCAN) neural network. The fundamental idea behind DCAN is to use adjective and noun text descriptions as (weak) supervised signals to learn two intermediate emotional expressions. These learnt representations are then concatenated and utilized to classify sentiment. (Guggilla *et al.*, 2016) offer a CNN and LSTM-based deep neural network model that classifies claims using word2vec and language embedding (classifying sentences to be factual or feeling). (Wang *et al.,* 2016) suggested a visual sentiment framework based on a convolutional neural network and tested their model using photographs from Twitter and Flickr.

The Long Short-Term Memory is almost like a form of RNN often used learn long-term dependencies (Zhang *et al.,* 2021). The recurring model of the LSTM, like those of other RNNs, is difficult and challenging. It has four layers that connect in an interesting way, including the hidden state and the cell state. Figure 2.2 illustrates a typical LSTM model.

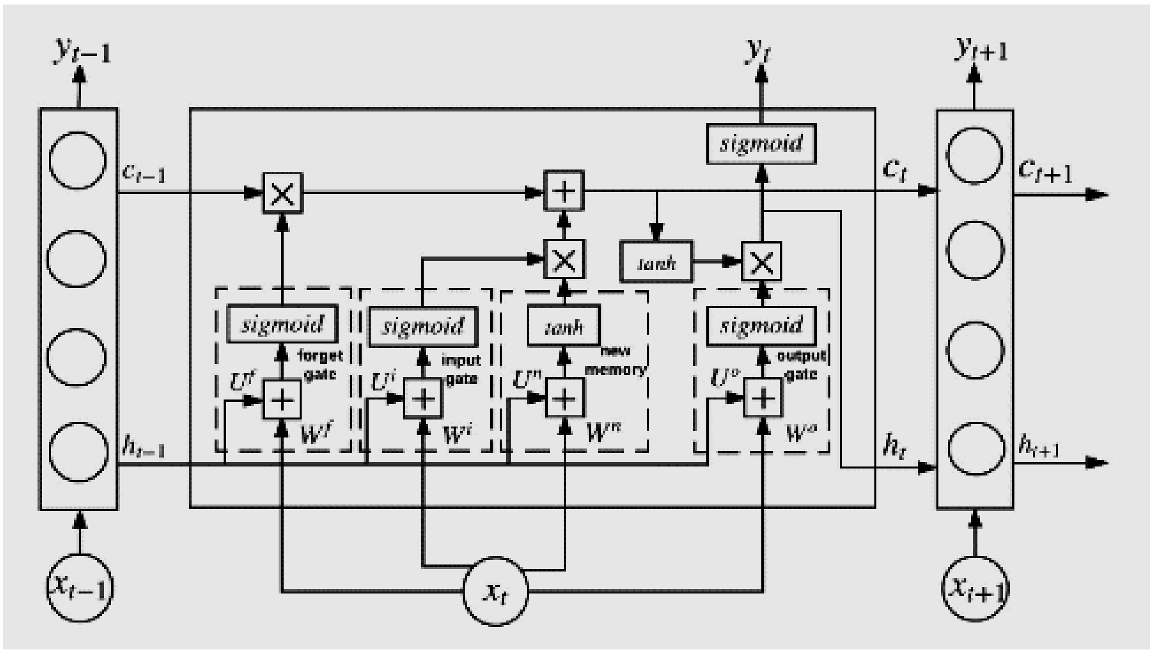


Figure 2.2 LSTM deep learning model

(Zhang *et al.*, 2018)

Mathematically, LSTM are represented mathematically as follows:

$$I_t = \Phi(W_i x_t + V_i H_{t-1} + B_i) \tag{2.6}$$

$$F_t = \Phi\big(W_f x_t + V_f H_{t-1} + B_f\big) \tag{2.7}$$

$$O_t = \Phi(W_o x_t + V_o H_{t-1} + B_o) \tag{2.8}$$

$$C_t = F_t \odot C_{t-1} + I_t \odot tanh(W_c x_t + V_c H_{t-1} + B_c) \tag{2.9}$$

$$H_t = O_t \odot \tanh(C_t) \tag{2.10}$$

W and *V* are the weight parameters, while *B* is the bias vector, x_t and H_t are the input and hidden state vectors of LSTM unit at *t* time, respectively, while, I_t, O_t, and F_t are the activation vectors of the input, output, and forget gate. Finally, sigmoid function and memory cell state vector are denoted by Φ and C_t , respectively.

2.4 Pastoralist Optimization Algorithm (POA)

Pastoralist Optimization Algorithm (POA) is an unique meta-heuristic motivated by nomadic pastoralists' socio-cultural lifestyle (Abdullahi *et al.*, 2018). It tries to emulate the behavior of nomadic pastoralists in their search for suitable pasture, water, and a healthy habitat for their animals. The pastoralist is the search agent in POA, and the ith pastoralist is represented as:

$$P_i = \left[P_{1,1}, P_{1,2}, P_{1,3}, \ldots, P_{i,D}\right] \tag{2.11}$$

Where D is the problem's dimension. The scouting and camping phases are the two fundamental parts of the POA process. Pastoralists travel quicker with larger step sizes during the scouting phase, and the best scout spot is selected as camp. Slower mobility and smaller step sizes define the camping period. To avoid local optima trapping, herding pastoralists split off onto other regions. Equation (2.12) gives the original position of the jth pastoralist (S_j), and Equation (2.13) gives the updated location of the jth pastoralist, and the evaluation continues until the maximum scouting rate is attained.

$$S_j = rand([L_b, U_b]^D) \tag{2.12}$$

$$S'_j = (S_b - S_j) + \varepsilon_j * \eta_j * \zeta \tag{2.13}$$

where S'_j denotes the new scout j position surrounding the best-found spot S_b, j ($\varepsilon \in \{-1,1\}$), is scout ε_j energy, η_j is scout j's step size (0,(0.001*Ub)), and ζ is a positive constant that represents the amount of times scouters move quicker than herders. The camp site is set to the best scout location (S_b). Equation (2.13) was used to separate the camp by herd pastoralist, and Equation (2.14) was used to decrease the camp size after each split (2.14).

$$P_K' = P_b + (rand(0,r) * \varepsilon_k * \eta_k) \tag{2.14}$$

$$r'' = \frac{r'}{nP} \tag{2.15}$$

where P_K' is the kth pastoralist's new position, so far P_b has shown to be the finest/best pastoralist. $rand(0,r)$ is a random integer between 0 and r, $\boldsymbol{r}$ is the camp radius, $\varepsilon_{\boldsymbol{k}}$ is the kth pastoralist energy ($\varepsilon \in \{-1,1\}$) and $\eta_{\boldsymbol{k}}$ is the kth pastoralist step size ($\eta \in \{0,(0.001 *$ Ub)}). Also, r'' is the current iteration's camp radius, while r' is the prior iteration's camp radius. If all locations have been explored, the best camp location is returned as the global ideal solution, as well as the best camp site being returned as the global best solution if all locations have been investigated; otherwise, the procedure is restarted with fresh scout locations identified using Equation (2.15).

$$S''_j = rand(L_b, U_b]^D) - S_b \tag{2.16}$$

where S''_j is the new scout location, $L_b \; and \; U_b$ are the search space's lower and upper bounds respectively. POA was developed utilizing a biological evolution technique and has shown to be particularly effective when used to numerical and issues involving combinatorial optimization. Other POA variants with a cultural evolution approach there has also been discovered (Zhang *et al.*, 2018), Figure 2.2 shows the POA phases.

Table 2.1: Pastoralist Optimization Algorithm (POA)

```
Algorithm 1. Pastoralist Optimization Algorithm
Input: POA Parameters; Search space
Output: Optimal Solution

1. Start
2. Initialize POA parameters
3. Select scout pastoralist randomly and initialize scout location
4. Evaluate the fitness of each scout, update scout location and normalize
   scouts locations within the search space until maximum scouting rate is
   reached
5. Select best scouting location
6. Evaluate fitness of pastoralist and determine best pastoralist within a camp
7. Split pastoralist to different locations within camp and evaluate fitness of
   each pastoralist
8. Repeat step (vii) until maximum splitting rate is reached. For each split,
   divide the current camp radius by the number of pastoralists
9. Update the best camping pastoralist
10. If all regions within the search space have not been explored,
        a. Update scout location
        b. Repeat steps (iv) to (x) and update the global camp best pastoralist
11. Else, return the global best-found pastoralist
12. Stop
```

2.5 Biogeographic-Based Algorithm

Biogeographic Optimization (BBO) is an evolutionary algorithm that optimizes a function by randomly and repeatedly increasing candidate solutions, based on a given quality metric or fitness function. BBO is a meta-heuristic generation stimulated via way of means of the biogeographical conduct of species in nature. It turned into proposed via way of means of Simon (2008) and it's been correctly applied to clear up many optimization issues with inside the energy system (Ma *et al.*, 2017) together with monetary dispatch (Bhattacharya & Chattopadhyay, 2010) and energy management (Bansal *et al.*, 2013). It is commonly used to optimize multi-dimensional real-valued functions, however it does now no longer use the gradient of the feature, this means that that it does now no longer require the feature to be differentiable like classical optimization methods, together with gradient descent and quasi Newton's method.

Every individual is regarded as a habitat in BBO, and their habitat suitability index (HSI) is used to measure their health. In GA, each chromosome is made up of genes, but in BBO, each habitat is defined by Suitability Index Variables (SIVs or habitants). HSI is similar to health feature in optimization issues. A higher HSI approach higher habitat conditions (Kaveh *et al.*, 2019). BBO is a geographical manner of mission of organic species. Each geographical area is represented via way of means of an index referred to as a Habitat Suitability index (HSI). Suitability Index Variable (SIV) is some other index is used to symbolize the place of habitat and livelihood conditions. The health of every habitat has similarities to its HSI cost and quantity of species. To enhance the low HSI solution, it accepts functions from excessive HSI solution. The emigration rate (μ) and immigration rate (λ) of the BBO is expressed with inside the following Equations. (2.17) and (2.18) respectively.

$$\mu = E \times k\ ns \tag{2.17}$$

Where, E is the most emigration rate, k is the quantity of species with inside the habitat and most quantity of species is ns.

$$\lambda = I\ (1 - k\ ns) \tag{2.18}$$

Where, I is the most immigration rate.

In BBO migration, every answer is represented with the aid of using a n size vector referred to as a habitat. Each size with inside the habitat is taken into consideration to be an SIV. The goodness of a habitat is similar to HSI fee and quantity of species. To enhance the answer, low HSI answer stocks facts with excessive HSI answer (comparable as GA and PSO), while sharing is primarily based totally on immigration (λ) and emigration rates (μ). In this process, habitats are selected from the population. Initially, habitat (H_i) is chosen primarily based totally at the immigration rate (λ_i), and different habitat (H_j) is chosen the usage of emigration rate (μ_j). Afterward, the randomly decided on SIVs have migrated from $H_j$solution and seem in Hi.

Table 2.2: Biogeographic-Based Optimization Algorithm

Biogeographic-Based Algorithm

```
Start Procedure Migration mechanism for n habitats
For i=1 to n DO
IF randomly generated number < $\lambda_i$
SELECT: $Hi$ with immigration rate $\lambda_i$
FOR j=1 to n DO
IF randomly generated number < $\mu_j$
SELECT: $Hj$ with emigration rate $\mu_j$
Replace a random SIV of $Hi$ with its corresponding SIV
from $H_j$ as given in
ENDIF
ENDFOR
ENDIF
ENDFOR
```

Figure 2.3 show the flow chart of Geographic-Based Algorithm below

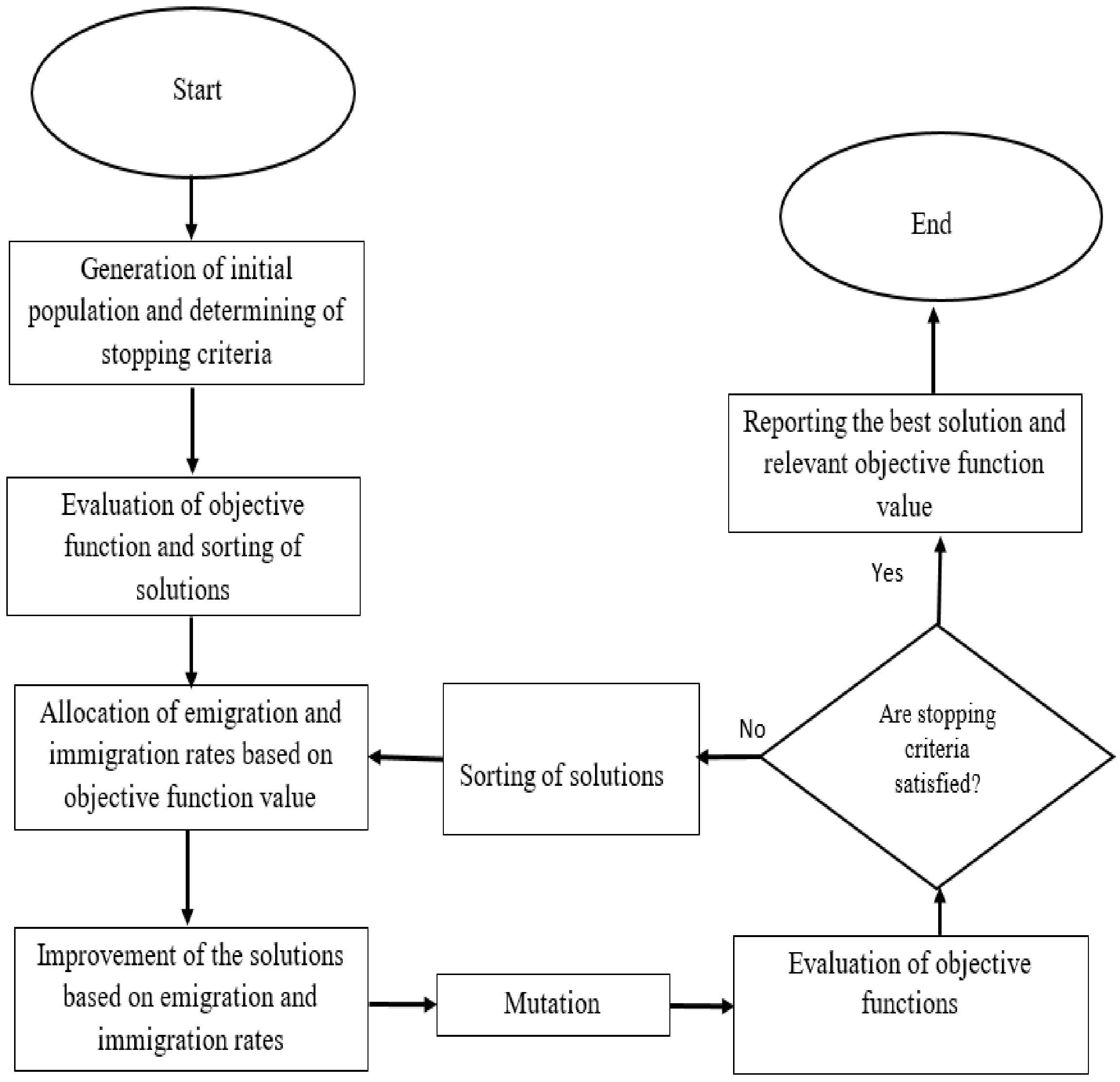

Start
Generation of initial population and determining of stopping criteria
Evaluation of objective function and sorting of solutions
Allocation of emigration and immigration rates based on objective function value
Improvement of the solutions based on emigration and immigration rates
Mutation
Evaluation of objective functions
Are stopping criteria satisfied?
No
Sorting of solutions
Yes
Reporting the best solution and relevant objective function value
End

Figure 2.3 Flowchart of Biogeographic-based Algorithm

(Omid *et al.*, 2016)

CHAPTER THREE

3.0 MATERIALS AND METHODS

3.1 Research Design

In this chapter, the detailed procedures that were followed for the successful completion of this research were discussed. Figure 3.1 depicts the essential tasks that were carried out in order to achieve the study's objectives.

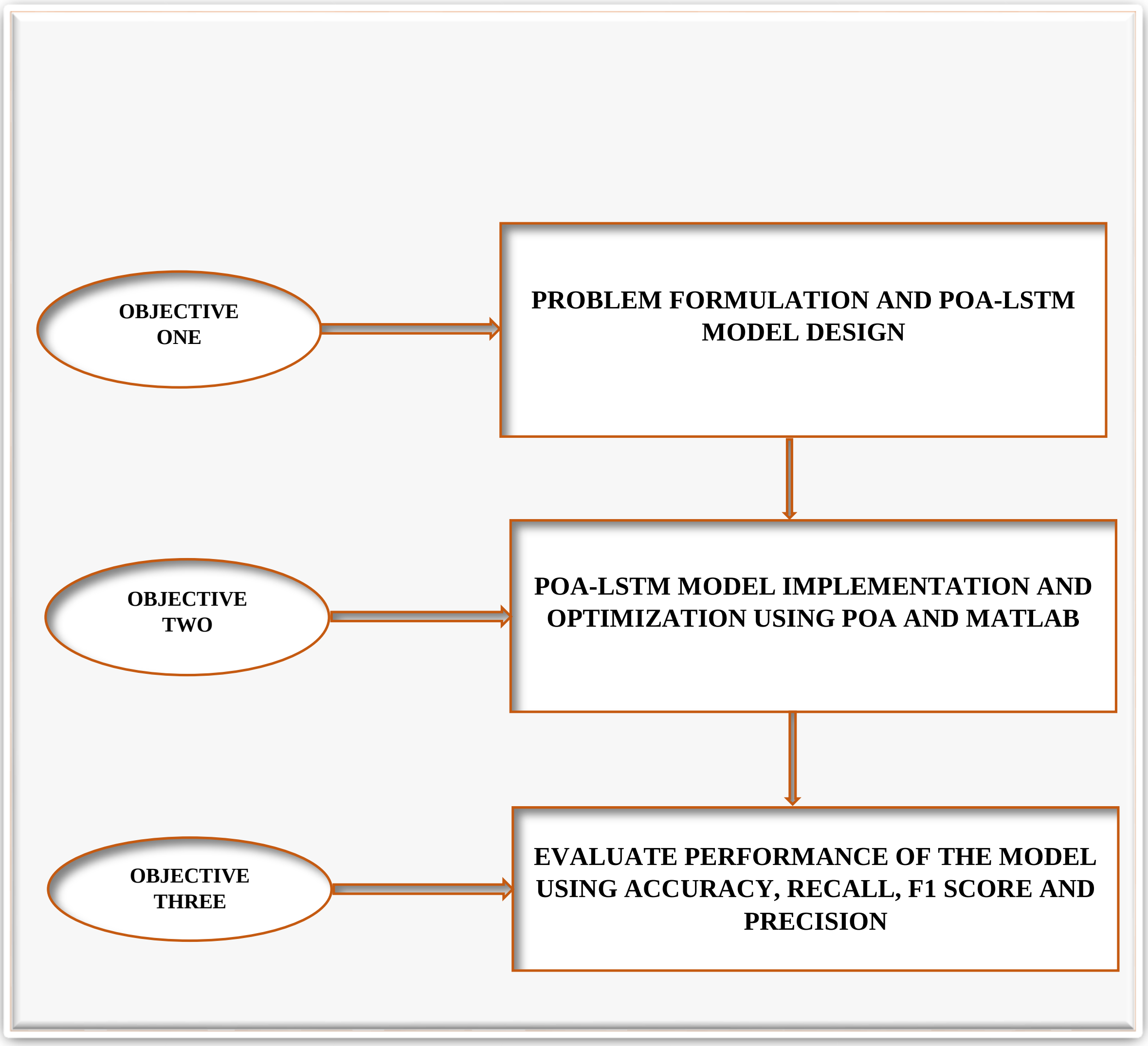

OBJECTIVE ONE
PROBLEM FORMULATION AND POA-LSTM MODEL DESIGN
OBJECTIVE TWO
POA-LSTM MODEL IMPLEMENTATION AND OPTIMIZATION USING POA AND MATLAB
OBJECTIVE THREE
EVALUATE PERFORMANCE OF THE MODEL USING ACCURACY, RECALL, F1 SCORE AND PRECISION

Figure 3.1 Research Methodology

Figure 3.2 depicts the approach used in this study to meet the goals and objectives. Data collection, preprocessing, and feature extraction are the first steps. After that, the optimized POA-LSTM model will be created. After then, the performance will be assessed.

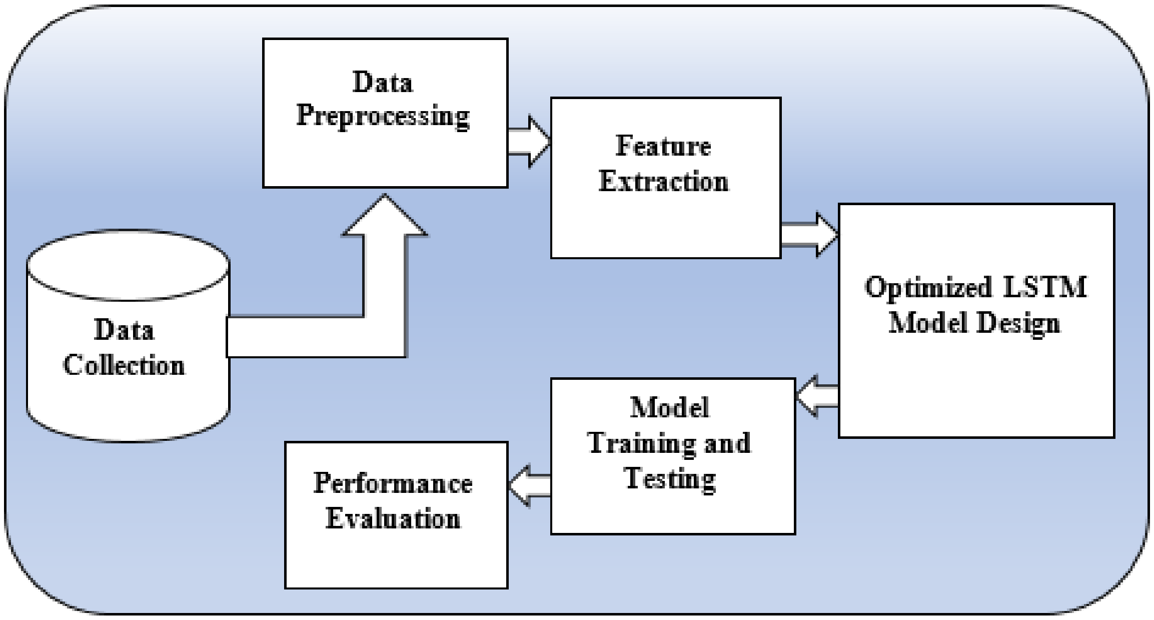
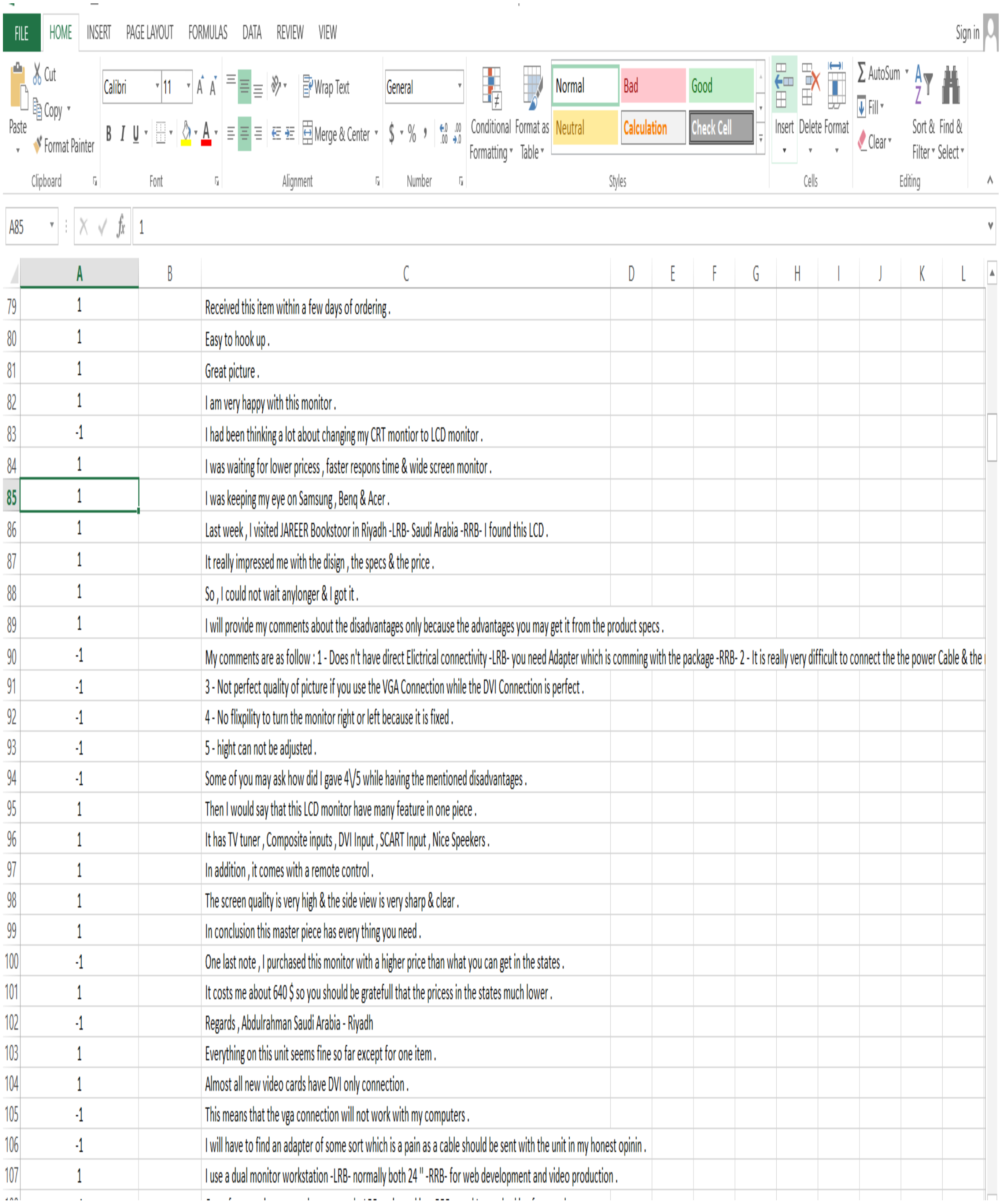


Figure 3.2 Methodology

3.2 Collecting data and Extracting Features

The dataset that was utilized to conduct the experiments in this research, three review datasets for three goods are included in this work: Wireless router, Computer, and Speaker. The Wireless router dataset contains 4512 sentences. There are also 1205 sentences for the computer dataset while the speaker dataset contains 689 sentences which makes a total of 6406 sentences. The dataset's reviews are all from Amazon, which can be obtained at amazon.com. Figure 3.3 depicts a portion of the computer dataset obtained from Amazon.



	A	B	C
79	1		Received this item within a few days of ordering .
80	1		Easy to hook up .
81	1		Great picture .
82	1		I am very happy with this monitor .
83	-1		I had been thinking a lot about changing my CRT montior to LCD monitor .
84	1		I was waiting for lower pricess , faster respons time & wide screen monitor .
85	1		I was keeping my eye on Samsung , Benq & Acer .
86	1		Last week , I visited JAREER Bookstoor in Riyadh -LRB- Saudi Arabia -RRB- I found this LCD .
87	1		It really impressed me with the disign , the specs & the price .
88	1		So , I could not wait anylonger & I got it .
89	1		I will provide my comments about the disadvantages only because the advantages you may get it from the product specs .
90	-1		My comments are as follow : 1 - Does n't have direct Elctrical connectivity -LRB- you need Adapter which is comming with the package -RRB- 2 - It is really very difficult to connect the the power Cable & the
91	-1		3 - Not perfect quality of picture if you use the VGA Connection while the DVI Connection is perfect .
92	-1		4 - No flixpility to turn the monitor right or left because it is fixed .
93	-1		5 - hight can not be adjusted .
94	-1		Some of you may ask how did I gave 4\/5 while having the mentioned disadvantages .
95	1		Then I would say that this LCD monitor have many feature in one piece .
96	1		It has TV tuner , Composite inputs , DVI Input , SCART Input , Nice Speekers .
97	1		In addition , it comes with a remote control .
98	1		The screen quality is very high & the side view is very sharp & clear .
99	1		In conclusion this master piece has every thing you need .
100	-1		One last note , I purchased this monitor with a higher price than what you can get in the states .
101	1		It costs me about 640 $ so you should be gratefull that the pricess in the states much lower .
102	-1		Regards , Abdulrahman Saudi Arabia - Riyadh
103	1		Everything on this unit seems fine so far except for one item .
104	1		Almost all new video cards have DVI only connection .
105	-1		This means that the vga connection will not work with my computers .
106	-1		I will have to find an adapter of some sort which is a pain as a cable should be sent with the unit in my honest opinin .
107	1		I use a dual monitor workstation -LRB- normally both 24 " -RRB- for web development and video production .

Figure 3.3 Sample of dataset

3.3 Preprocessing

Following data collection, the users' opinions were preprocessed to convert them from their raw form to the one that machine learning algorithms could fully understand. Preprocessing also makes it possible for the removal of noise from the data, likely to result in more accurate learning algorithms. The following are the steps in the pre-processing process:

1. Removal of URLs
2. Removal of special symbols/emoticons
3. Removal of stop words from the dataset
4. Convert the dataset (transforming a string sequence into individual tokens\ strings with a defined or given meaning).

Following data preparation, the characteristics necessary for training by the learning algorithm are extracted. There are two ways to text representation: word vector representation, which represents each word in a document as a vector representation and a vector of N dimensions, in which each word in a text is represented as a vector of N dimensions. This method ignores semantic information as well as the order and structure of words in a document. It just cares about the appearance of words. The representation of word embedding is the second strategy used in this work. By grouping comparable words in a text collection in a vector space, word embedding delivers syntactic and semantic information to learning algorithms. Algebraic procedures can be done on the embedding with this characteristic. The vector space must be trained on a set of texts in order to provide correct word embedding.

3.4 Problem Formulation and Model Design

The optimized POA-LSTM and LSTM models were designed by configuring the LSTM's models parameters appropriately such as: The learning rate, number of layers, number of epoch, and number of inputs. Also, the parameter of the POA was also set in the design stage. Table 1 shows the parameters that were selected for the POA-LSTM and LSTM models were optimized.

Table 3.1: Set-Up Parameters for POA-LSTM and LSTM Models

Parameters	**POA-LSTM**	**LSTM**
Maximum Epoch	20	20
Learning Rate	[0.01-0.1]	0.05
Sequence Input	300 Dimensions	300 Dimensions
Fully Connected Layer	2	2
LSTM layer	[1-100] Hidden Units	50-100 Hidden Units
Softmax Layer	Softmax	Softmax
Classification Output Layer	Crossentropyex	Crossentropyex
Scouting Rate	3	-
Splitting Rate	5	-
Number of Pastoralist	20	-

The two important parameters which seems to have an effect on the performance of an LSTM model are learning rate and number of hidden units in the LSTM layer which were optimized using POA. The learning rate is set between 0.01 to 0.1 while the number of hidden units is set between 1 to 100. Other parameters of LSTM include the sequence input which is set at

300 dimensions, whith 2 fully connected layers, and crossentropyex as classification output layer.

The mean squared error is the fitness function (F) utilized by the mechanism to evaluate the optimal settings as shown in equation 3.1

$$F = minimize\left(\frac{1}{M}\sum_{k=1}^{M}(A_k - P_k)^2\right) \tag{3.1}$$

where M is the number of occurrence's, A_k and O_k are denoted the kth sample's actual and output values, respectively.

3.5 Optimized POA-LSTM Model Implementation

Three experiments were performed. For each experiment, the preprocessed dataset were split into training and testing. 70% of the dataset was used to train the LSTM model, while 30% of the untrained data was used to test it. The first experiment was done using the POA-LSTM model while the second and third experiment were carried out using the LSTM model with 50 and 100 hidden units respectively. Also, the LSTM models used a learning rate of 0.05. Table 3.2 shows the implementation steps of the optimized model.

Table 3.2: POA-LSTM implementation steps

```
Algorithm 2: POA-LSTM Sentiment Analysis Algorithm
Input: Sentiment data; LSTM model
Output: Optimal position; Best model

i     Start
ii    Data collection and pre-processing
iii   Feature extraction
iv    LSTM model design
v     POA parameters initialization and initial population generation
vi    Select scout pastoralist randomly and initialize scout location
vii   Train and test LSTM model
viii  Evaluate the fitness of each scout, update scout locations and normalize scouts locations
      within the search space until maximum scouting rate is reached
ix    Select best camping location and initialize pastoralist in the camp
x     Train and test LSTM model using new camp locations
xi    Evaluate fitness of pastoralist and determine best pastoralist within camp
xii   Split pastoralist to different locations within camp
xiii  Train and test LSTM model using new camp locations
xiv   Evaluate fitness of each pastoralist and update local best pastoralist
xv    Repeat step (xii) - (xiv) until maximum splitting rate is reached. For each split, divide the
      current camp radius by the number of pastoralists
xvi   If all regions within the search space have not been explored
          a.  Update scout location
          b.  Repeat steps (vii) – (xv) and update the global camp best pastoralist
xvii      Else, return the global best- found pastoralist
xviii     Return best model
xix   Stop
```

3.6 Performance Evaluation

The improved/optimized POA-LSTM model and LSTM models were evaluated using the accuracy, precision, and F1 score performance metrics. They are stated mathematically as;

3.6.1 Accuracy

The ratio of all properly categorized samples to all samples is the accuracy of a model, and it is expressed as;

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \tag{3.2}$$

3.6.2 Precision

Precision is defined as the percentage of samples properly categorized, and is expressed as;

$$Precision = \frac{TP}{TP+FP} \times 100\% \tag{3.3}$$

3.6.3 Recall

Is the percentage of positive sentiment samples that were properly detected out of the total number of positive samples. It can be expressed as follows:

$$Recall = \frac{TP}{TP+FN} \times 100\% \tag{3.4}$$

3.6.4 F1 score

The F1 score is a balance of precision and recall, and it is calculated as follows:

$$F1 = 2\left(\frac{Precision * Recall}{Precision + Recall}\right) \tag{3.5}$$

Where, TN (True Negative) is appropriately categorized negative sentiments, TP (True Positive) is appropriately categorized positive sentiments, FN (False Negative) is inappropriately categorized positive sentiments and FP (False positive) is inappropriately categorized negative sentiments. N is the number of instances, T_i and P_i are the target and predicted values of the i^{th} sample respectively.

Table 3.3: Parameters Settings for LSTM and POA-LSTM Models

Parameters	**POA-LSTM**	**BBO-LSTM**	**LSTM**
Maximum Epoch	20	20	20
Number of Hidden Nodes	18	26	100
Learning Rate	0.02	0.02	0.05
Scouting Rate	3		-
Splitting Rate	5		-
Number of Pastoralist	10:30	-	-
Number of Habitats (nPop)	-	10:30	-
Keep rate (nKeep)	-	0.2	-
Number of New Habitats	-	NPop-n Keep	-
nKeep	-	Round(keepRate *nPop)	-

CHAPTER FOUR

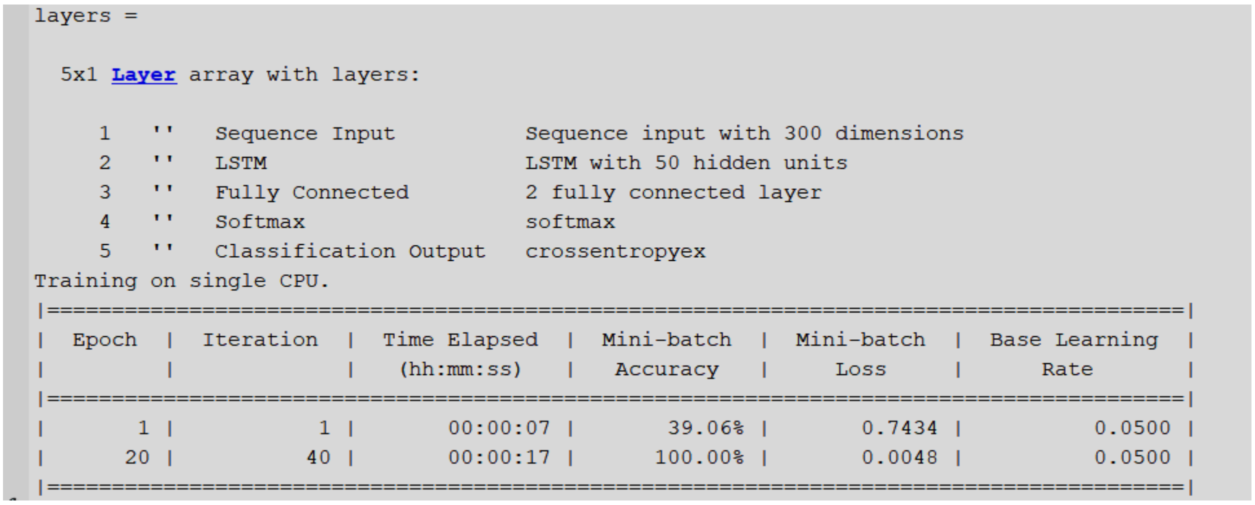
RESULTS AND DISCUSSION

4.1 LSTM Model Results

Two LSTM models were trained with different number of hidden nodes. The first model was trained with 50 nodes while the second model was trained with 100 nodes. The numbers of nodes were manually and randomly selected.

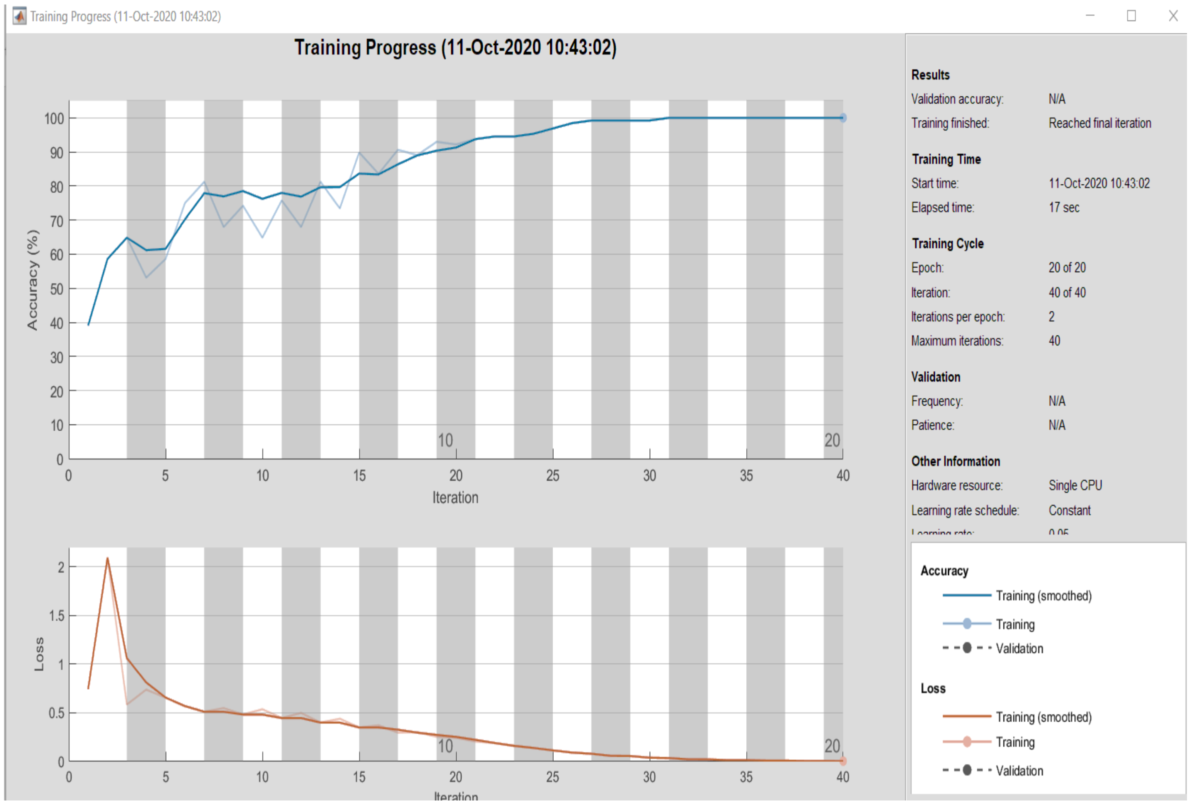
Figure 4.1 shows the training parameters of the LSTM-50 model design. It shows the training parameters for the input layer, the output layer, the LSTM, fully connected and **softmax** layers. The design (model) was configured to have a sequence input with a dimension of 300 pre-trained word embedding vectors, an LSTM layer with 50 hidden layers with a constant learning rate, a softmax activation function and an epoch of 20. The output layer uses the crossentropyex classification activation function. These parameters were tested and shown to produce the training results.

Figure 4.2 shows the training progress and loss curve for the LSTM model. The green curve represents the accuracy against iteration plot while the red curve represents the loss against iteration curve. From both curves, the result shows that the maximum training accuracy was obtained at the 31st iteration while lowest training loss was obtained at the 32nd iteration.



```
layers =

  5x1 Layer array with layers:

     1   ''   Sequence Input          Sequence input with 300 dimensions
     2   ''   LSTM                    LSTM with 50 hidden units
     3   ''   Fully Connected         2 fully connected layer
     4   ''   Softmax                 softmax
     5   ''   Classification Output   crossentropyex
Training on single CPU.
|========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|========================================================================================|
|       1 |           1 |       00:00:07 |       39.06% |       0.7434 |          0.0500 |
|      20 |          40 |       00:00:17 |      100.00% |       0.0048 |          0.0500 |
|========================================================================================|
```

Figure 4.1 Training parameters of LSTM with 50 Nodes



Results
Validation accuracy: N/A
Training finished: Reached final iteration

Training Time
Start time: 11-Oct-2020 10:43:02
Elapsed time: 17 sec

Training Cycle
Epoch: 20 of 20
Iteration: 40 of 40
Iterations per epoch: 2
Maximum iterations: 40

Validation
Frequency: N/A
Patience: N/A

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant

Figure 4.2 Training progress of LSTM with 50 Nodes

The confusion matrix of the LSTM-50 models is shown in Figure 4.3 the matrix is a comparison of actual and expected customers' opinions. The matrix shows the True Negative

(TN), True Positive (TP), False Negative (FN) and False Positive (FP) of the models. These values were used to calculate the performance evaluation metrics described in Chapter three. The matrix shows that the True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP) are; 32, 73, 24,30 for LSTM-50.

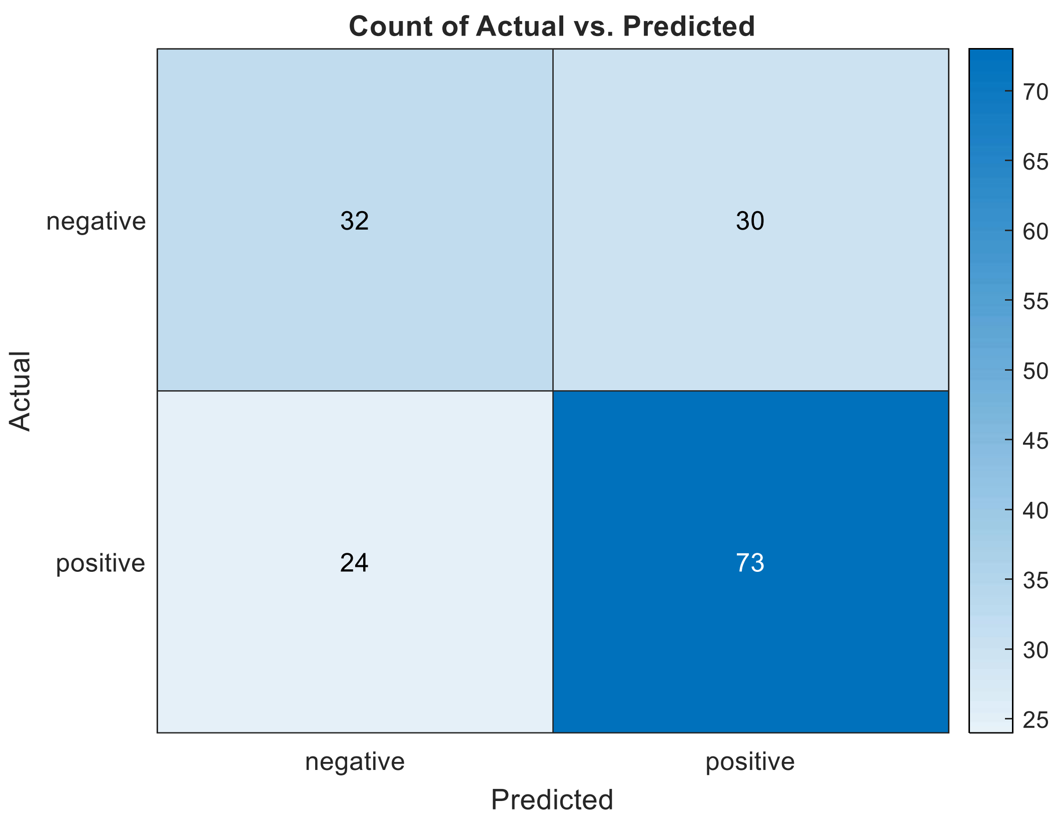
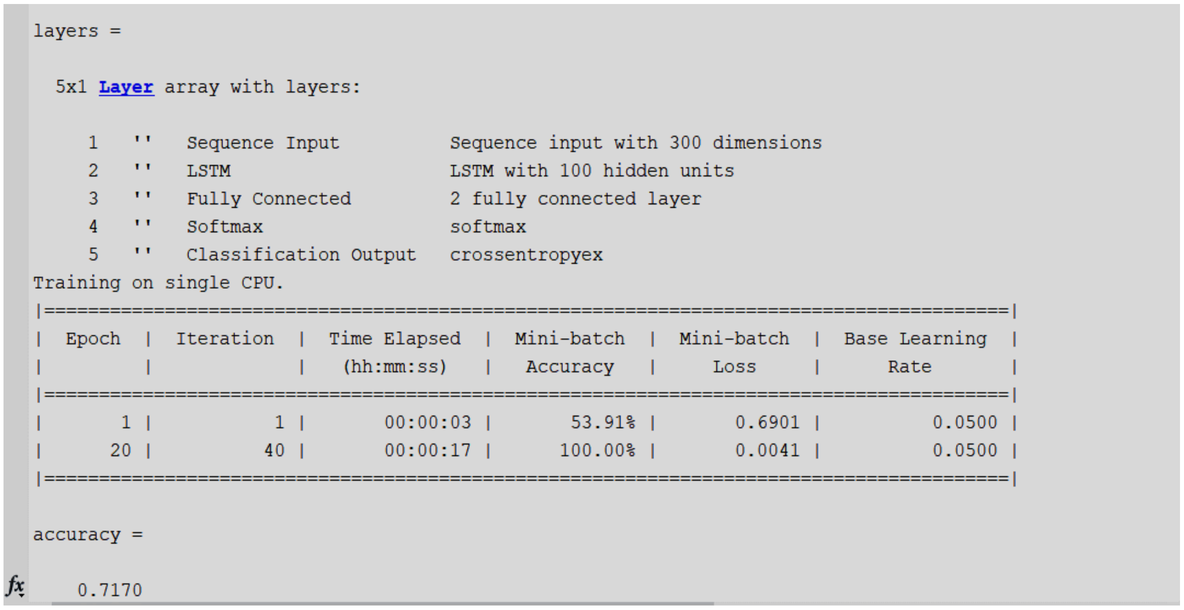


Figure 4.3 Confusion matrix of LSTM with 50 Nodes

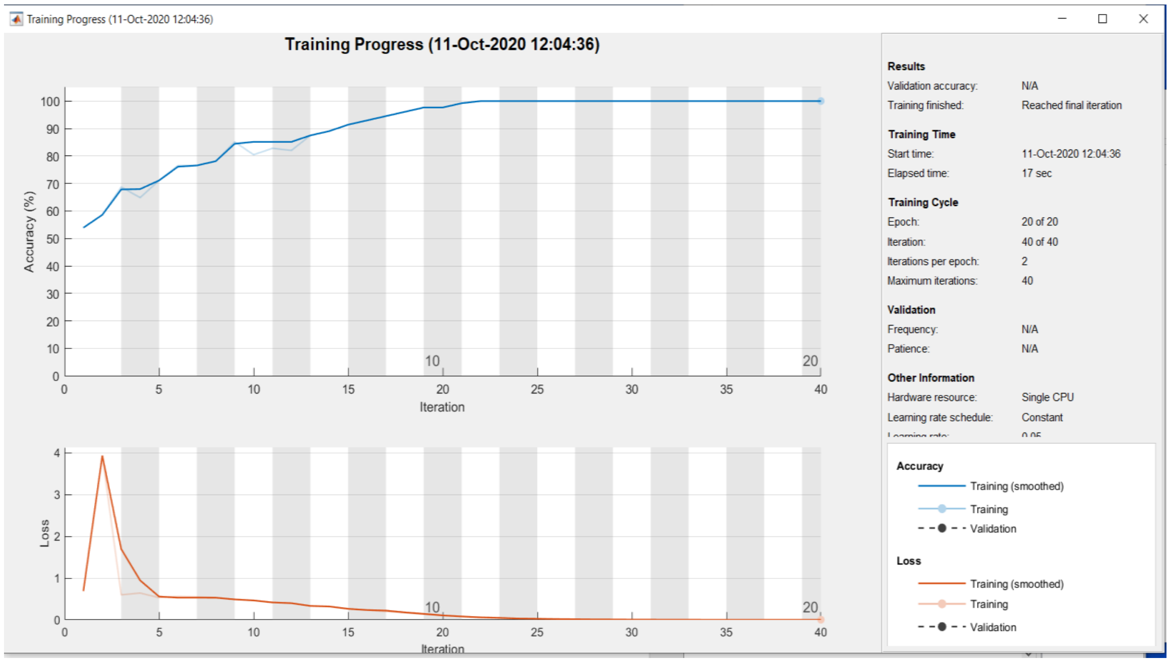
Figure 4.4 shows the training parameters of the LSTM-100 model design. It shows the training parameters for the output and the input layer, the LSTM, fully connected and softmax layers. The model was configured to have a sequence input with a dimension of 300 pre-trained word embedding vectors, an LSTM layer with 50 hidden layers with a constant learning rate, a softmax activation function and an epoch of 20. The output layer uses the crossentropyex classification activation function. These parameters were tested and shown to produce the training results.

Figure 4.5 shows the training progress and loss curve for the LSTM model. The green curve represents the accuracy against iteration plot while the red curve represents the loss against iteration curve. From both curves, the result shows that the maximum training accuracy was obtained at the 22nd iteration while lowest training loss was obtained at the 35th iteration.



```
layers =

  5x1 Layer array with layers:

     1   ''   Sequence Input          Sequence input with 300 dimensions
     2   ''   LSTM                    LSTM with 100 hidden units
     3   ''   Fully Connected         2 fully connected layer
     4   ''   Softmax                 softmax
     5   ''   Classification Output   crossentropyex
Training on single CPU.
|=========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|=========================================================================================|
|       1 |           1 |       00:00:03 |       53.91% |       0.6901 |          0.0500 |
|      20 |          40 |       00:00:17 |      100.00% |       0.0041 |          0.0500 |
|=========================================================================================|

accuracy =

    0.7170
```

Figure 4.4 Training parameters of LSTM with 100 Nodes



Results

Validation accuracy:	N/A
Training finished:	Reached final iteration

Training Time

Start time:	11-Oct-2020 12:04:36
Elapsed time:	17 sec

Training Cycle

Epoch:	20 of 20
Iteration:	40 of 40
Iterations per epoch:	2
Maximum iterations:	40

Validation

Frequency:	N/A
Patience:	N/A

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.05

Figure 4.5 Training progress of LSTM with 100 Nodes

The confusion matrix of the LSTM-100 models is shown in Figure 4.6 the matrix is a comparison of actual and expected customer opinion. The matrix shows the True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP) of the models. These values were used to calculate the performance evaluation metrics described in Chapter three. The matrix shows that the True Negative (TN), True Positive (TP), False Negative (FN), and False Positive (FP) are; 42, 72, 25, 20 for LSTM-100.

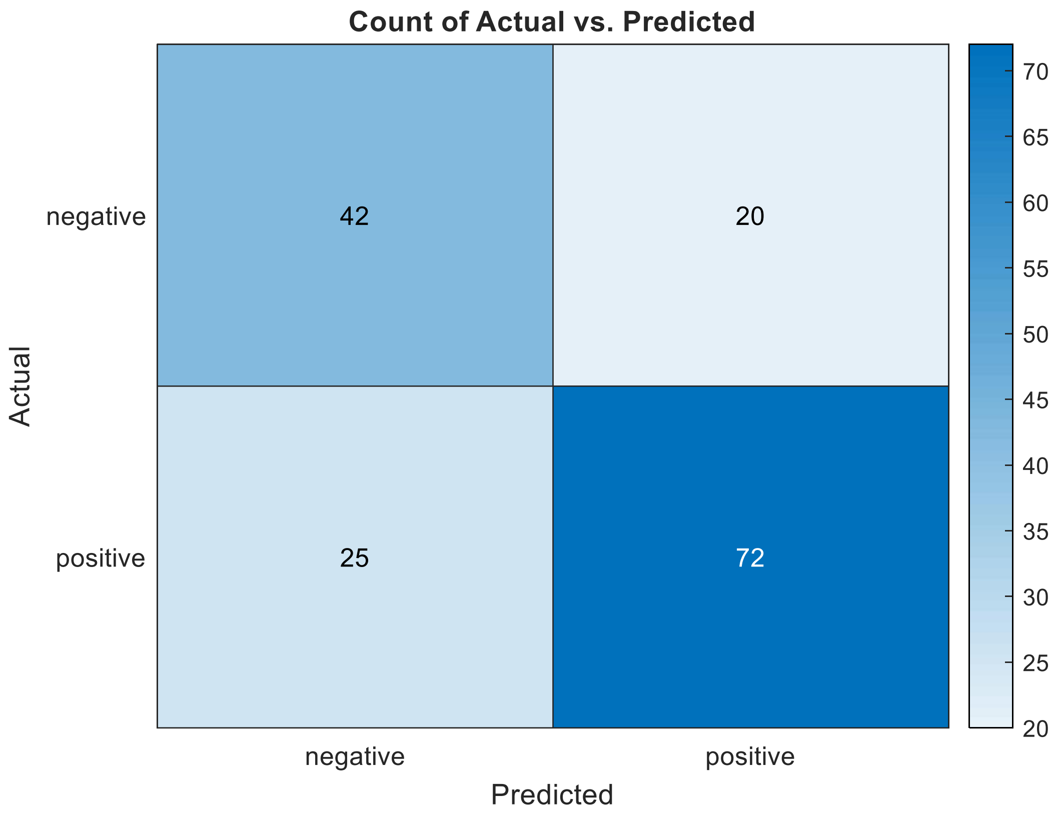
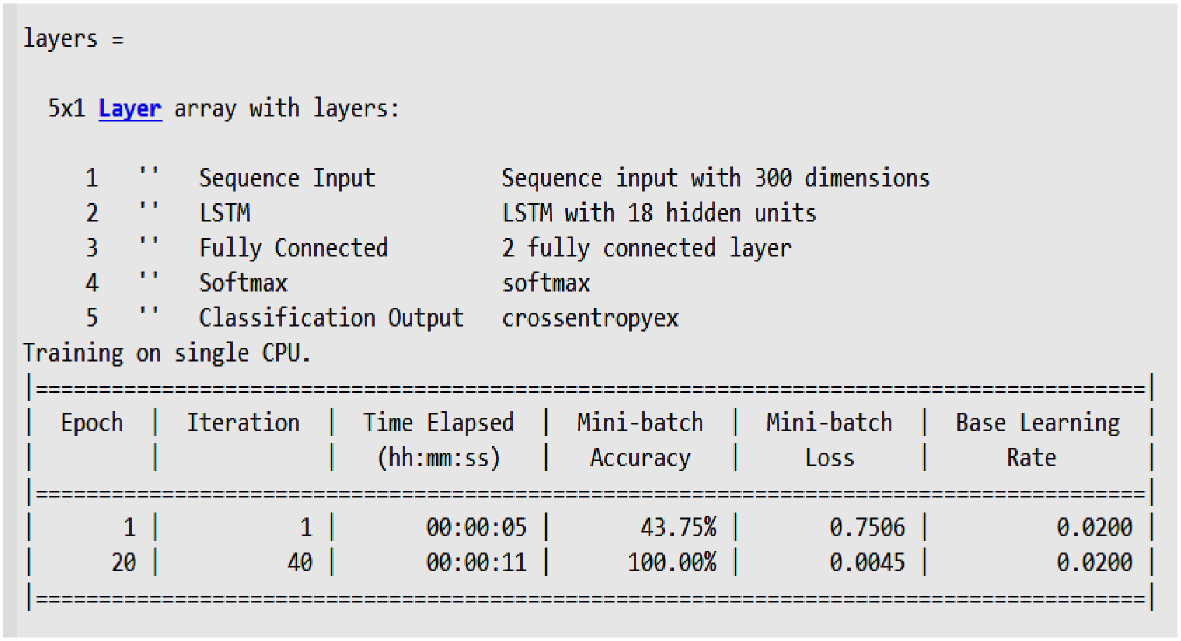


Figure 4.6 Confusion matrix of LSTM with 100 Nodes

4.2 POA-LSTM Model Results

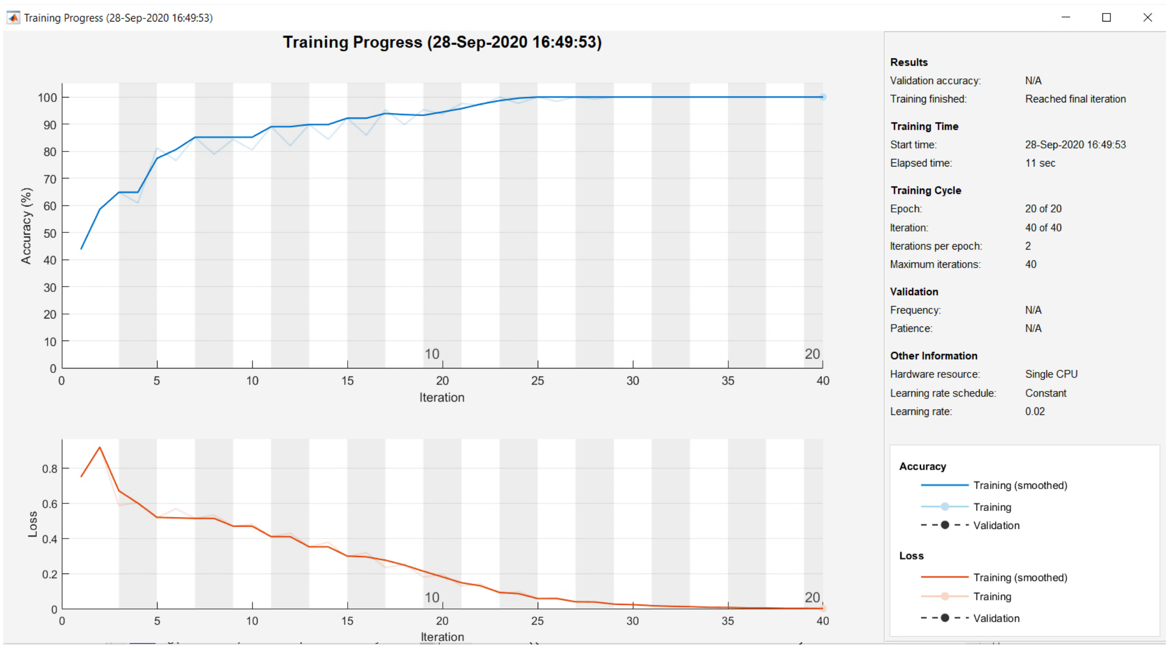
Figure 4.7 shows the training parameters of the POA-LSTM model design. It shows the training parameters for the output and the input layer, the LSTM, fully connected and softmax layers. The model was configured to have a sequence input with a dimension of 300 pre-trained word embedding vectors, an LSTM layer with 18 hidden layers which has 0.02 learning rate, a softmax activation function and an epoch of 20. The output layer uses the crossentropyex classification activation function. These parameters were tested and shown to produce the best training results.

Figure 4.8 shows the training progress and loss curve for the trained POA-LSTM model. The green curve represents the accuracy against iteration plot while the red curve represents the loss against iteration curve. From both curves, the result shows that the maximum training accuracy was obtained at the 25th iteration while lowest training loss was obtained at the 38th iteration.



```
layers =

  5x1 Layer array with layers:

     1   ''   Sequence Input          Sequence input with 300 dimensions
     2   ''   LSTM                    LSTM with 18 hidden units
     3   ''   Fully Connected         2 fully connected layer
     4   ''   Softmax                 softmax
     5   ''   Classification Output   crossentropyex
Training on single CPU.
|========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|========================================================================================|
|       1 |           1 |       00:00:05 |       43.75% |       0.7506 |          0.0200 |
|      20 |          40 |       00:00:11 |      100.00% |       0.0045 |          0.0200 |
|========================================================================================|
```

Figure 4.7 Training parameters of POA-LSTM



Results
Validation accuracy: N/A
Training finished: Reached final iteration

Training Time
Start time: 28-Sep-2020 16:49:53
Elapsed time: 11 sec

Training Cycle
Epoch: 20 of 20
Iteration: 40 of 40
Iterations per epoch: 2
Maximum iterations: 40

Validation
Frequency: N/A
Patience: N/A

Other Information
Hardware resource: Single CPU
Learning rate schedule: Constant
Learning rate: 0.02

Accuracy
Training (smoothed)
Training
Validation

Loss
Training (smoothed)
Training
Validation

Figure 4.8 Training progress of POA-LSTM

Figure 4.9 depicts the POA-LSTM model's convergence curve, which illustrates the lowest fitness value and convergence rate. It depicts the POA-LSTM models' convergence curves for population sizes of 10, 15, 20, 25, and 30. For each population size, the curve shows the optimum fitness value and the rate of convergence. The result reveals that the algorithm converges at a population size of 20, yielding an optimal value of 0.7736, which is the best value when compared to alternative population sizes. The fitness value is 0.761 for population sizes of 10, 15, and 25, and drops to 0.7484 at population sizes of 30. The best learning rate and number of hidden nodes for population sizes of 10, 15, 20, 25, and 30 are 0.01-17, 0.02-18, 0.01-37, 0.01-5, and 0.02-21, Table 4.1 shows the results of these calculations.

Figure 4.10 shows the confusion matrix for the POA-LSTM models. The matrix is a count of real feelings vs anticipated sentiments. The matrix shows the True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP) of the models. These values were used to calculate the performance evaluation metrics described in Chapter three. The matrix shows that the True Negative (TN), True Positive (TP), False Negative (FN) and False Positive (FP) and are; 41, 80, 21, 17 for POA-LSTM (10), 38, 83, 14, 24 for POA-LSTM (15),49 , 74, 23, 13 for POA-LSTM (20), 40, 81, 16, 22 for POA-LSTM (25), 41, 78,19, 21 for POA-LSTM (30).

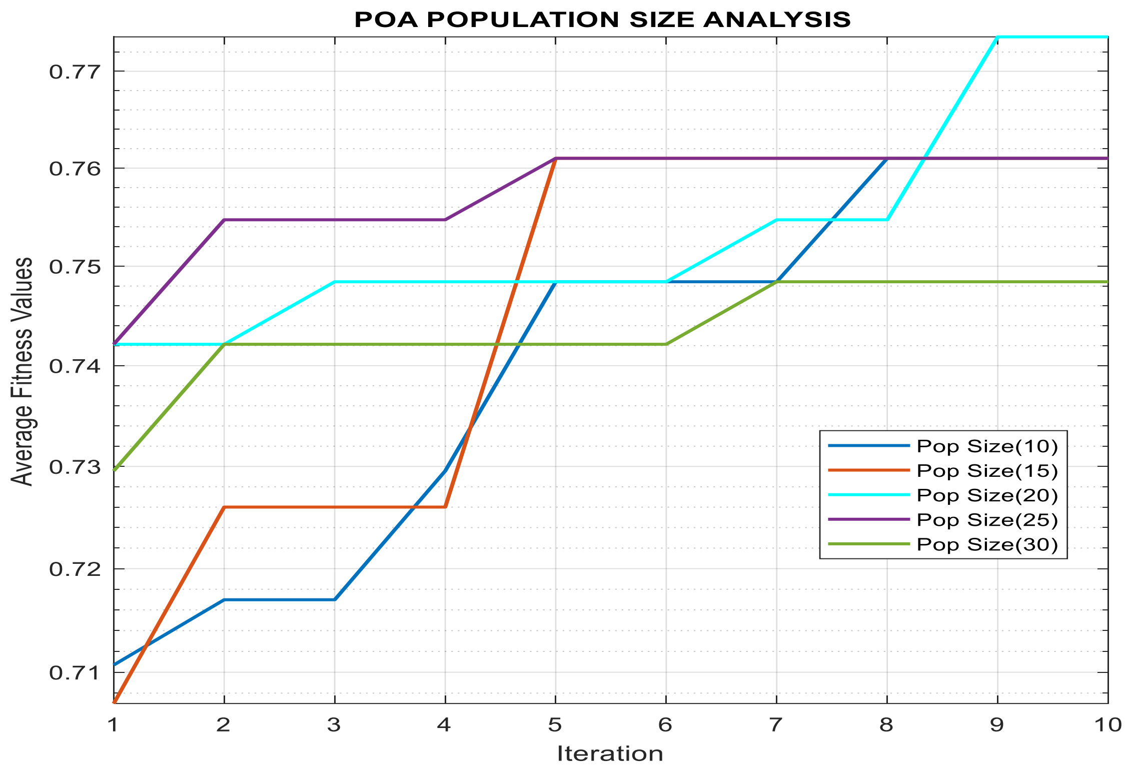
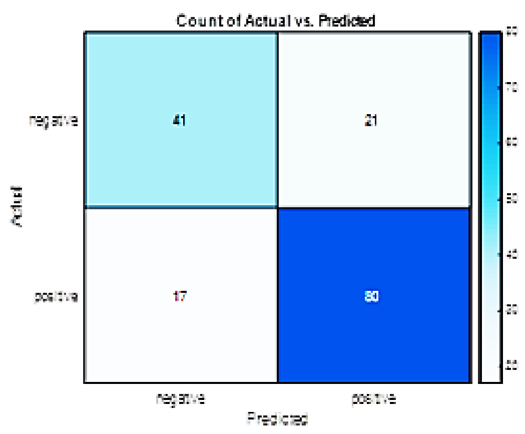


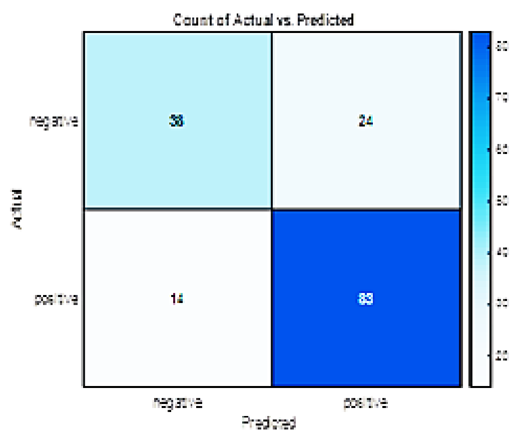
Figure 4.9 Convergence Curve of POA-LSTM

Table 4.1 POA-LSTM optimization

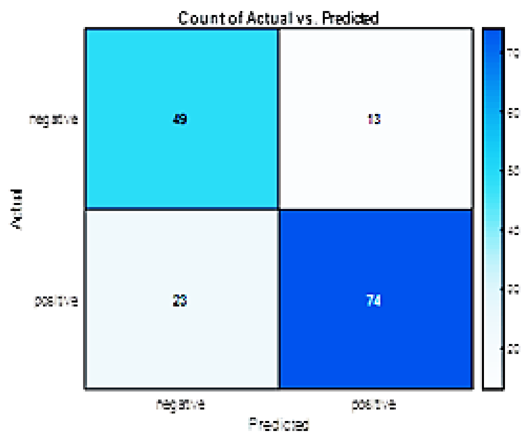
POA Population Size	Optimal Learning Rate	Optimal Hidden Nodes	Optimal Fitness Value
10	0.010	17.0	0.7610
15	0.020	18.0	0.7610
20	0.010	37.0	0.7736
25	0.010	5.0	0.7610
30	0.020	21.0	0.7484



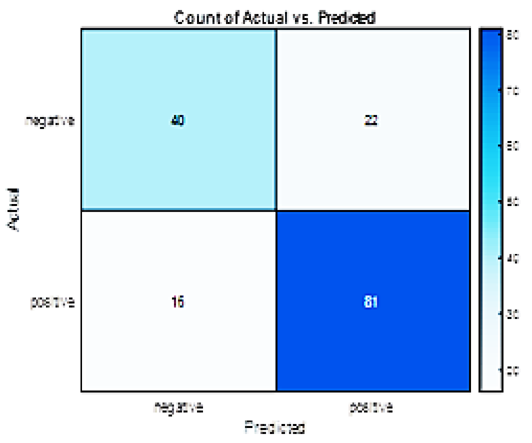
(a) 10 pastoralist



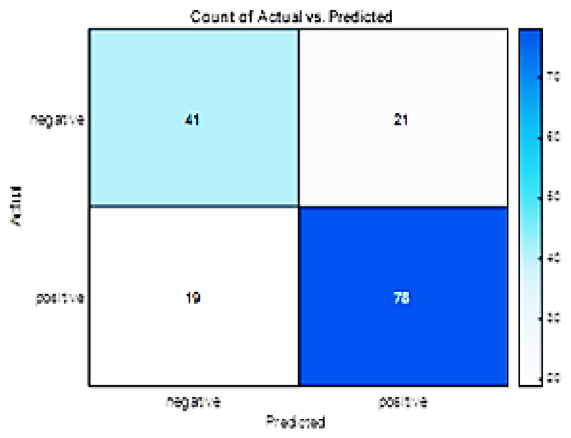
(b) 15 Pastoralist



(c) 20 Pastoralist



(d) 25 Pastoralist



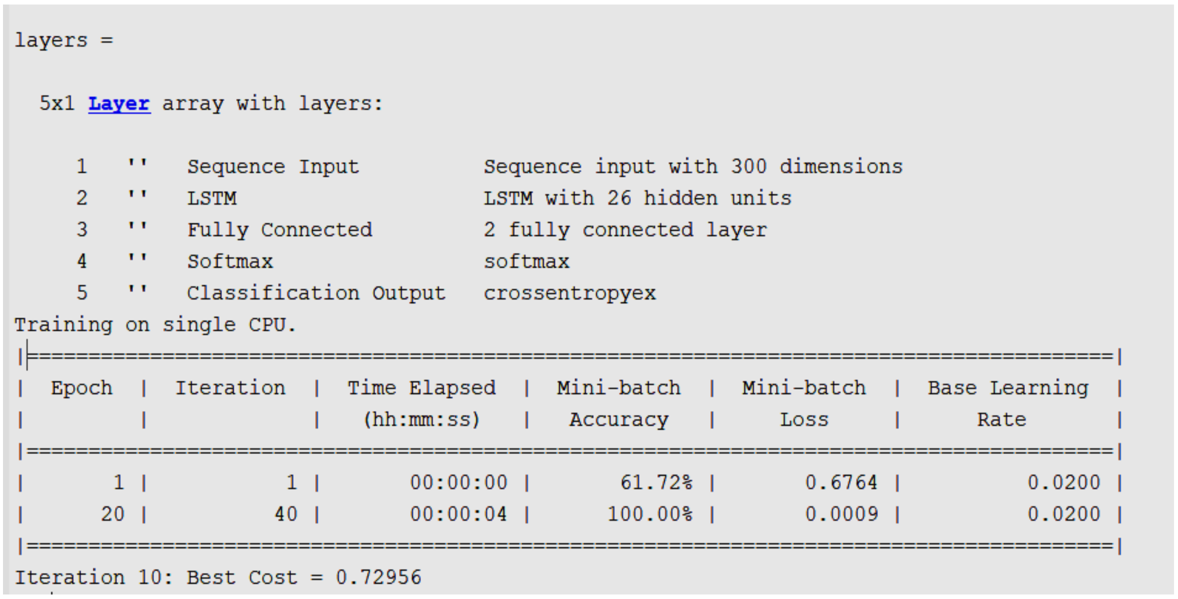
(e) 30 Pastoralist

Figure 4.10 Confusion matrix for POA-LSTM

4.3 BBO-LSTM Model Results

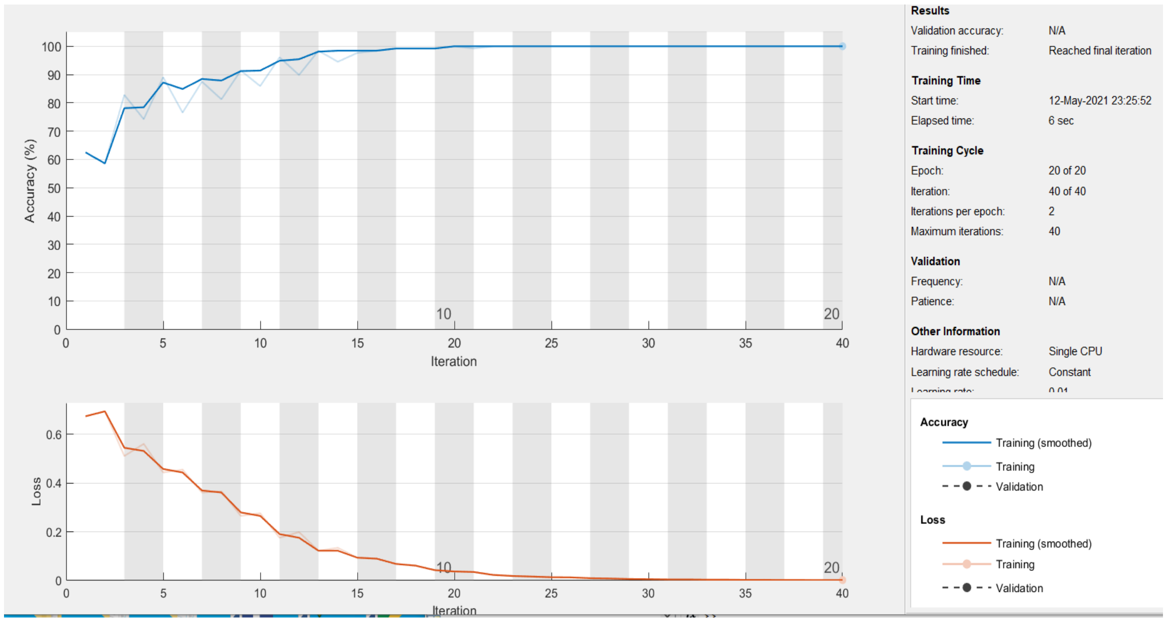
Figure 4.11 shows the training parameters of the BBO-LSTM model design. It shows the training parameters for the input layer, the output layer, the LSTM, fully connected and softmax layers. The model was configured to have a sequence input with a dimension of 300 pre-trained word embedding vectors, an LSTM layer with 26 hidden layers which has 0.02 learning rate, a softmax activation function and an epoch of 20. The output layer uses the crossentropyex classification activation function. These parameters were tested and shown to produce the best training results.

Figure 4.12 shows the training progress and loss curve for the trained BBO-LSTM model. The green curve represents the accuracy against iteration plot while the red curve represents the loss against iteration curve. From both curves, the result shows that the maximum training accuracy was obtained at the 21st iteration while lowest training loss was obtained at the 29th iteration.



```
layers =

  5x1 Layer array with layers:

     1   ''   Sequence Input          Sequence input with 300 dimensions
     2   ''   LSTM                    LSTM with 26 hidden units
     3   ''   Fully Connected         2 fully connected layer
     4   ''   Softmax                 softmax
     5   ''   Classification Output   crossentropyex
Training on single CPU.
|========================================================================================|
|  Epoch  |  Iteration  |  Time Elapsed  |  Mini-batch  |  Mini-batch  |  Base Learning  |
|         |             |   (hh:mm:ss)   |   Accuracy   |     Loss     |      Rate       |
|========================================================================================|
|       1 |           1 |       00:00:00 |       61.72% |       0.6764 |          0.0200 |
|      20 |          40 |       00:00:04 |      100.00% |       0.0009 |          0.0200 |
|========================================================================================|
Iteration 10: Best Cost = 0.72956
```

Figure 4.11 Training parameters of BBO-LSTM



Results

Validation accuracy:	N/A
Training finished:	Reached final iteration

Training Time

Start time:	12-May-2021 23:25:52
Elapsed time:	6 sec

Training Cycle

Epoch:	20 of 20
Iteration:	40 of 40
Iterations per epoch:	2
Maximum iterations:	40

Validation

Frequency:	N/A
Patience:	N/A

Other Information

Hardware resource:	Single CPU
Learning rate schedule:	Constant
Learning rate:	0.01

Figure 4.12 Training progress of BBO-LSTM

Figure 4.13 depicts the BBO-LSTM model's convergence curve, which illustrates the lowest fitness value and the convergence rate. It depicts the BBO-LSTM models' convergence curves for population sizes of 10, 15, 20, 25, and 30. The curve shows the optimum fitness value and the rate of convergence for each population size. The results demonstrate that at a population size of 30, the algorithm converges, yielding an optimal value of 0.7610, the best value when compared to other population sizes. The fitness value was the same in populations of 10 and 15, but climbed to 0.7358 and subsequently decreased to 0.7296 in populations of 20 and 25.The optimal learning rate and number of hidden nodes selected for 10, 15, 20, 25 and 30 population sizes are; 0.01-21, 0.01-26, 0.01-25, 0.100-43 and 0.01-32 respectively as shown in Table 4.2.

The confusion matrix of the BBO-LSTM models is shown in Figure 4.14 The matrix is a comparison of actual and expected customers opinion. The matrix shows the True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) of the models. These values were used to calculate the performance evaluation metrics described in Chapter three. The matrix shows that the True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN) are; 76, 35, 27, 21 for BBO-LSTM (10), 76, 35, 27, 21 for BBO-LSTM (15), 76, 36, 26, 23 for BBO-LSTM (20), 80, 35, 27, 17 for BBO-LSTM (25), 81, 40, 22, 16 for BBO-LSTM (30).

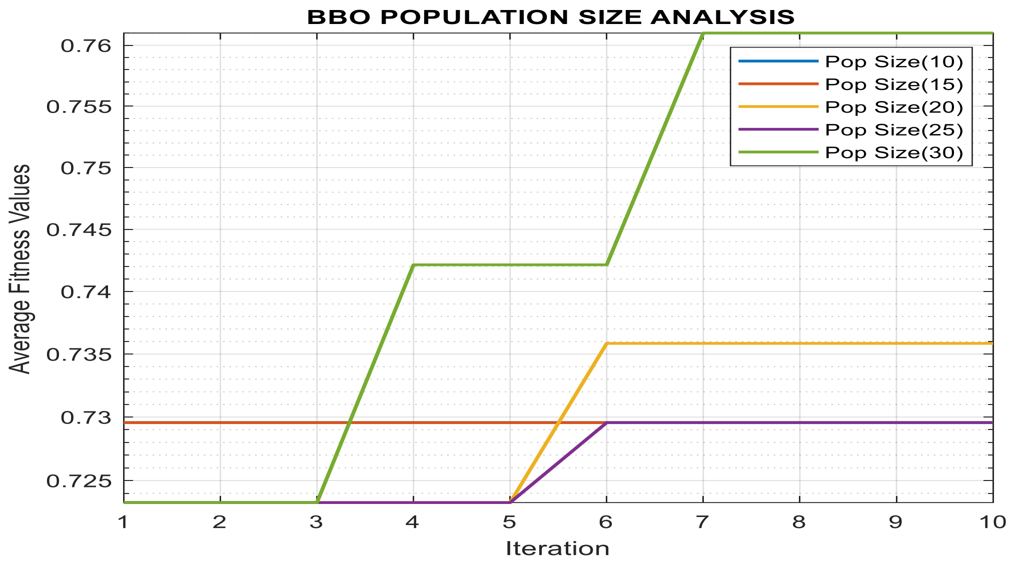
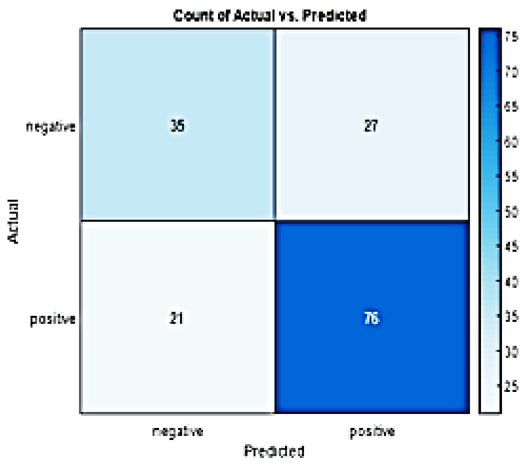


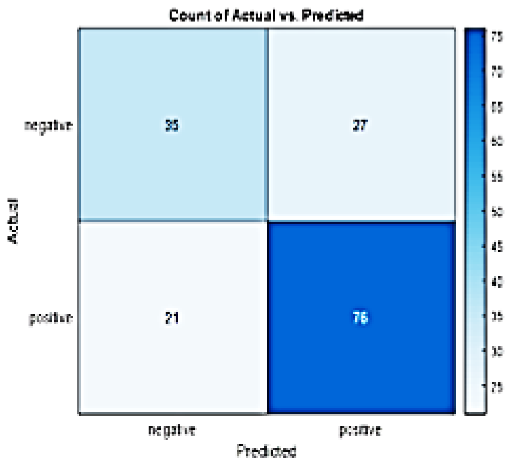
Figure 4.13 Convergence curve of BBO-LSTM

Table 4.2: BBO-LSTM optimization

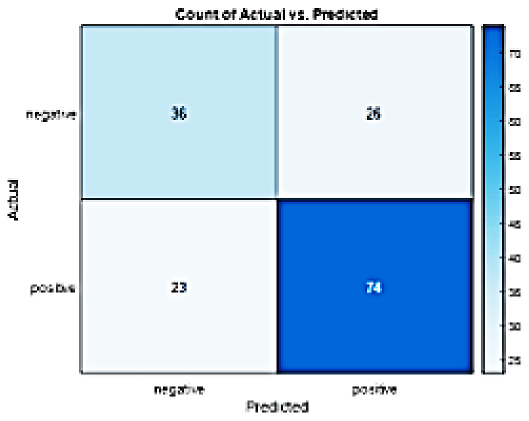
BBO Population Size	Optimal Learning Rate	Optimal Hidden Nodes	Optimal Fitness Value
10	0.01	21	0.7296
15	0.00	26	0.7296
20	0.01	25	0.7358
25	0.100	43	0.7296
30	0.01	32	0.7610



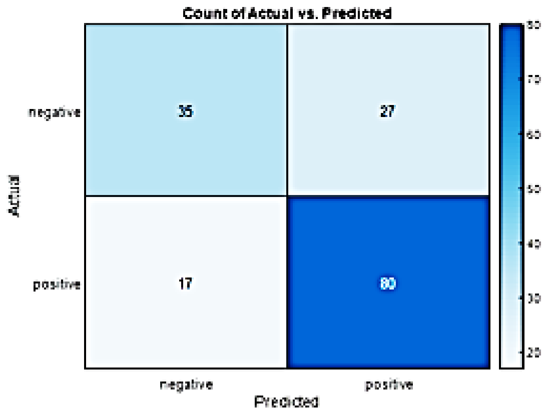
(a) 10 Habitats



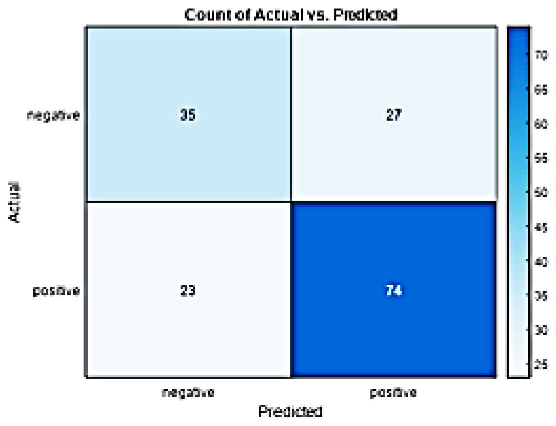
(b) 15 Habitats



(c) 20 Habitats



(d) 25 Habitats



(e) 30 Habitats

Figure 4.14 Confusion matrix of BBO-LSTM

4.4 Performance Evaluation Results

Table 4.3 demonstrates the performance evaluation results for the developed models. It includes the accuracy, precision, recall and F1 score performance metrics. The accuracy, precision, recall and F1-Score obtained by LSTM model trained with 50 nodes are 36.59 percent, 70.87 percent, 75.26 percent, 72.99 percent respectively. Finally, the LSTM model trained with 100 nodes obtains accuracy, precision, recall, and F1-Score of 71.62 percent, 78.26 percent, 74.23 percent, and 76.19 percent, respectively. Similarly, the accuracy, precision, recall and F1-Score of the developed POA-LSTM model are 76.10 percent, 77.57 percent, 85.56 percent and 81.36 percent. The accuracy, precision, recall and F1-Score of the developed BBO-LSTM model are 72.33 percent, 74.77 percent, 82.47 percent and 78.43 percent respectively. In terms of accuracy, recall, and F1-score, the POA-LSTM model beat the other models. This might be due to the POA optimizer selecting the best model parameters. However, it performs slightly less in terms of F1-Score when compared with BBO model. This is because BBO detected more negative cases with less false positive than the other two models as shown in the graph in Figure 4.11.

Table 4.3: Performance Evaluation

Algorithms	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
LSTM-50	36.59	70.89	75.26	72.99
LSTM-100	71.62	78.26	74.23	76.19
POA-LSTM	77.36	85.06	76.29	80.44
BBO-LSTM	76.10	78.64	83.50	80.99

The performance evaluation results of the created models are shown in Table 4.3. It includes the accuracy, precision, recall, and F1 score performance metrics calculated from the TP, TN, FP, and FN values obtained in the confusion matrices in Figure 4.10. The accuracy, precision, recall, and F1-Score for the developed models are as follows; for POA-LSTM (10) model, the figures are 76.10 percent, 79.21 percent, 82.47 percent, and 80.81 percent, respectively, for POA-LSTM (15) model The figures for the POA-LSTM (20) model are 77.36 percent, 85.06 percent, 76.29 percent, and 80.44 percent, respectively, while the figures for the POA-LSTM (15) model are 76.10 percent, 78.64 percent, 83.51 percent, and 81 percent, respectively, and the figures for the POA-LSTM (30) model are 74.84 percent, 78.79 percent, 80.41 percent, and 79.59 percent respectively.

The accuracy, precision, recall, and F1-Score for the existing models are as follows; for BBO-LSTM (10) model, the figures are 69.81 percent, 73.79 percent, 78.35 percent, and 76.00 percent respectively, for BBO-LSTM (15) model are 69.81 percent, 73.79 percent, 78.35 percent, and 76.00 percent respectively. For BBO-LSTM (20) model, the figures are 69.18 percent, 74.00 percent, 76.29 percent, and 75.13 percent respectively, for POA-LSTM (25) model are 72.33 percent, 74.77percent, 82.47percent, and 78.43 percent respectively, while for BBO-LSTM (30) model are 68.55percent, 73.27percent, 76.29percent, and 74.75percent respectively.

The LSTM model trained with 100 nodes obtains accuracy, precision, recall, and F1-Score of 71.62 percent, 78.26 percent, 74.23 percent, and 76.19 percent, respectively. In terms of accuracy and precision, the POA-LSTM (20) model beat the others, while the POA-LSTM (15) model surpassed the others in terms of recall and F1-score. In terms of all the metrics assessed, the optimized models outperform the BBO-LSTM except in terms of Recall, it had a higher value to that of POA-LSTM and also the optimized is far better than the un-optimized LSTM model. This might be due to the LSTM models being trained using the POA optimizer's ideal settings.

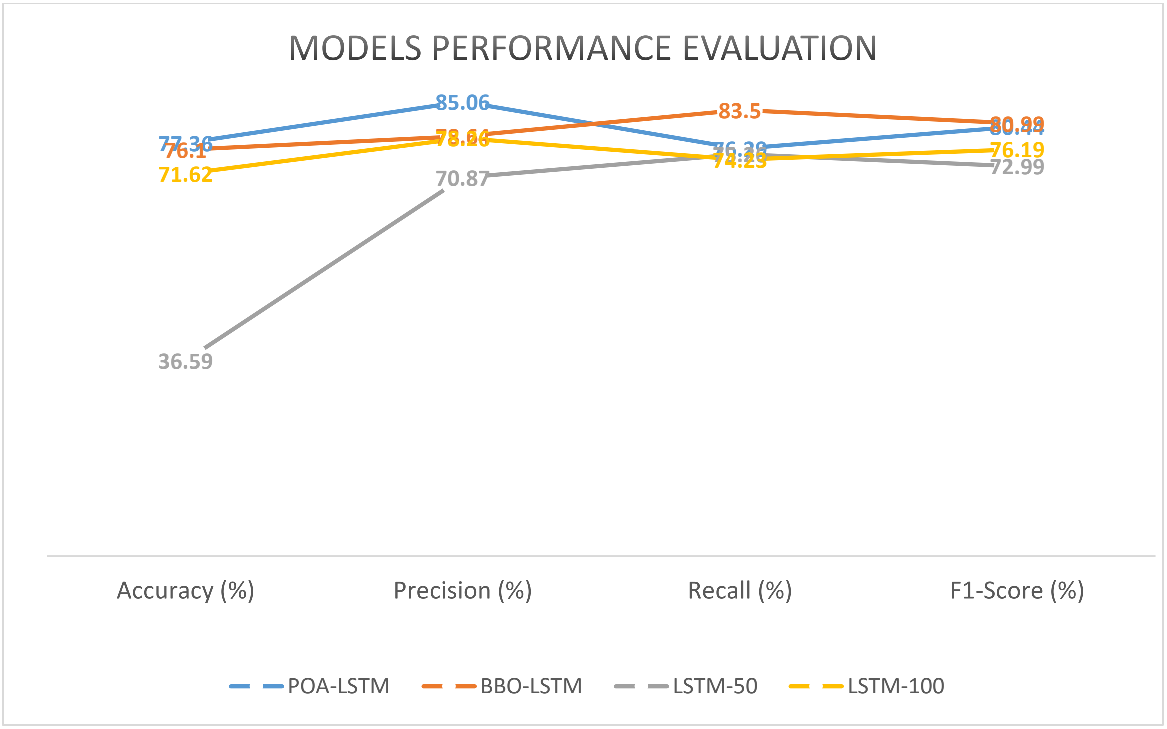


Figure 4.15 Accuracy, Precision, Recall and F1-Score graph of all the models evaluated

CHAPTER FIVE

5.0 CONCLUSION AND RECOMMENDATION

5.1 Conclusion

A customer satisfaction analysis model that is optimized was constructed in this thesis. Customers product review data was collected from Amazon.com and preprocessed after which a recently developed Pastoralist Optimization Algorithm (POA) was applied to optimize some of the parameters of LSTM deep learning model and also compared it with Biogeographic-Based Algorithm (BBO) and the normal LSTM with different nodes. The learning rate and number of hidden units of the LSTM were tuned, and the POA-LSTM algorithm's optimal population size was examined. The results showed that the optimized LSTM model outperforms the other LSTM models in terms of accuracy and precision. Using POA with 20 pastoralists, the optimization yields an ideal learning rate of 0.02 and an optimal node size of 18. This demonstrates that the optimized LSTM with POA outperformed the BBO and not optimized LSTM for analysis of public opinion. In addition, the POA may be utilized as a parameter optimizer.

5.2 Recommendation

This research shows how necessary it is to optimize parameters while using Machine Learning (ML) to learn in order to achieve the best result and design relative to a set of prioritize criteria or constraints. These include maximizing factors such as productivity, strength, reliability longevity, efficiency, and utilization. Based on the results, the following recommendations is therefore made:

i Other sentiment analysis datasets will be investigated in the future to assess the efficiency of the established algorithms.

ii Also, several optimization strategies consisting of Bat Algorithm (BA), Particle Swarm Optimization (PSO), Grasshopper Optimization Alorithm (GOA) and Artificial Bee Colony (ABC) are can be investigated to decide their impact at the LSTM model.

5.3 Contribution to Knowledge

The main Contributions of this thesis is the design of an optimized deep LSTM Model with high performance for the review of customer’s sentiment. This was as a result of the hyperparameter tuning performed which made it possible to identify the optimal settings for the LSTM model.

REFERENCES

Abdullahi I. M., Mu'azu M. B., Olaniyi O. M. & Agajo J. (2018). Pastoralist Optimization Algorithm: A Novel Nature-Inspired Metaheuristic Optimization Algorithm. Proceedings International Conference on Global and Emerging Trends, (ICGET 2018), Baze University, Abuja, pp. 101-105, http:// repository.futminna.edu.ng:8080/jspui/handle/123456789/9088.

Abdullahi I. M**.**, Muázu M. B., Olaniyi O. M., & Agajo, J. (2019). An Investigative Parameter Analysis of Pastoralist Optimization Algorithm (Poa): A Novel Metaheuristic Optimization Algorithm, Journal of Science Technology and Education 7(3), pp. 267-272.availableat:www.atbuftejoste.com. doi:10.1109/cybernigeria51635.2021.9428863

Aditya, T., & Vikesh, K. (2015). Sentiment Analysis on Movie Reviews using Recursive and Recurrent Neural Network Architectures. Department of Electrical Engineering, Stanford University, Stanford, CA - 94305, adityast@stanford.edu.

Akhtar, MS., Kumar, A., Ghosal, D., Ekbal, A., & Bhattacharyya P. (2019). A multilayer perceptron-based ensemble technique for fine-grained financial sentiment analysis. In Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP 2019).

Anand, M. & Sachin, K. (2017). Sentiment Analysis of Tweets in Malayalam Using Long Short-Term Memory Units and Convolutional Neural Nets. Mining Intelligence and Knowledge Exploration: 5th International Conference, MIKE 2017, Hyderabad, India, December 13–15.http// doi: 10.1007/978-3-319-71928-3_31.

Anais, C., Costea, D., Joyeux, O., Hasan, M., & Brunie, L. (2013). A Study and Comparison of Sentiment Analysis Methods for Reputation Evaluation, Available at: *https://liris.cnrs.fr/Documents/Liris-6508.pdf.*

Bansal, A. K., Kumar R., & Gupta, R. (2013). Economic analysis and power management of a small autonomous hybrid power system (SAHPS) using biogeography based optimization (BBO) algorithm," IEEE Trans. Smart Grid 4(1), 638–648. doi: 10.1109/TSG.2012.2236112.

Bengio, S., Deng, L., Larochelle, H., Lee, H., & Salakhutdinov, R. (2013). Guest Editors Introduction: Special Section on Learning Deep Architectures, IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 8, pp. 1795-1797, doi: 10.1109/TPAMI.2013.118.

Bhatt, A., Patel, A., Chheda, H., & Gawande, K. (2015). Amazon Review Classification and Sentiment Analysis. International Journal of Computer Science and Information Technologies, 6(6), 5107–5110, doi: https://doi.org/10.34010/injiiscom.v2i2.7455.

Bhattacharya, A. & Chattopadhyay, P. K. (2010). Biogeography-based optimization for different economic load dispatch problems. IEEE Trans. Power Syst. 25(2), 1064–1077, doi: 10.1109/tpwrs.2009.2034525 Volume: 25, Issue: 2, May 2010).

Bolden, R. & Moscarola, J. (2000). Bridging the quantitative-qualitative divide: the lexical Approach to textual data analysis," Social science computer review, vol. 18, no. 4, pp. 450–460, Volume 18, Issue 4, https://doi.org/10.1177/089443930001800408 (2000).

Cambria, E. Y., Li, F. Z., Xing, S., Poria, & Kwok, K. (2020). Senticnet 6: Ensemble application of symbolic and subsymbolic ai for sentiment analysis, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, pp. 105–114, https://doi.org/10.1145/3340531.3412003.

Chen, H., Sun, M. Tu, C., Lin, Y., and Liu, Z. (2016.) Neural sentiment classification with user and product attention. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2016).

Collobert R., Weston J., Bottou L., Karlen M., Kavukcuoglu K., & Kuksa P. Natural language processing (almost)from scratch. Journal of Machine Learning Research 12(12):2493-2537, 2011.

Dou. Z. Y. (2017). Capturing user and product Information for document level sentiment analysis with deep memory network. In Proceedings of the Conference on Empirical Methods on Natural Language Processing (EMNLP 2017),Copenhagen, Denmark. Association for Computational Linguistics. pg 521-526. https://aclanthology.org/D17-1054, https://doi.org/10.18653/v1/d17-1054.

Flanagin, A. J., Metzger, M. J., Pure, R., Markov, A. & Hartsell, E. (2014). Mitigating risk in ecommerce transactions: perceptions of information credibility and the role of user-generated ratings in product quality and purchase intention," Electronic Commerce Research, vol. 14, no. 1, pp. 1–23.

Guggilla, C., Miller, T. & Gurevych, I. (2016.) CNN-and LSTM-based claim classification in online user comments. In Proceedings of the International Conference on Computational Linguistics (COLING 2016). Technical Papers, pages 2740–2751, Osaka, Japan, https://aclanthology.org/C16-1258.

Heller, B., and Parasnis, G. (2011). From social media to social customer relationship management, Strategy & leadership, vol. 39, no. 5, pp. 30– 37, 2011, ISSN: 1087-8572.

Jan, D., & Mark, C. (2016). Sentiment Analysis using Convolutional Neural Networks with Multi Task Training and Distant Supervision on Italian Tweets. Zurich University of Applied Sciences, Switzerland. deri@zhaw.ch, ciel@zhaw.ch, https://doi.org/10.21256/zhaw-1527.

Jeh, G., & Widom, J. (2014) Mining the Space of Graph Properties, Proceedings of the Tenth {ACM} {SIGKDD} International Conference on Knowledge Discovery and Data Mining, Seattle, Washington, USA.

Kasper, W. & Vela, M. (2018). Sentiment analysis for hotel reviews, proceedings of the computational linguistics-applications, Jacharanka Conference. D-66123 Saarbrücken,Germany. Email: {Walter.Kasper,Mihaela.Vela}@dfki.de.

Kaveh, M., Khishe, M., & Mosavi, M.R. (2019). Designandimplementation of a neighborhood search biogeography-based optimization trainer for classifying sonar dataset using multi-layer perceptron neural network.AnalogIntegr.Circ.Sig.Process.100(2),405–428.

Kim, Y., & Srivastava. J. (2007). Impact of social influence in e-commerce decision making, in Proceedings of the ninth international conference on Electronic commerce, pp. 293–302, ACM, https://dio/10.1145/1282100.1282157.

Krishna, A., Akhilesh, V., Aich, A., & Hegde, C. (2019). Sentiment Analysis of Restaurant Reviews Using Machine Learning Techniques. In Lecture Notes in Electrical Engineering (Vol. 545). Springer Singapore. https://doi.org/10.1007/978-981-13-5802-9_60, (LNEE,volume 545).

Lee, H., Grosse, R., Ranganath, R., & Ng A.Y. (2009). Convolutional deep belief networks for scalable unsupervised learning of hierarchical Machine Learning. Conference: Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, https://doi.org/10.1145/1553374.1553453.

Liu, J., & Zhang Y. (2017). Attention modeling for targeted sentiment. In Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL 2017). DOI: Corpus ID: 2763049

Lundeqvist, E., & Svensson, M. (2017). Author Profiiling: A Machine Learning Approach Towards Detecting Gender, Age, and Native Language of Users in Social Media. UPPSALA Univeristy, Department of Information Technology.

Ma, H., Simon, D., Siarry, P.,Yang, Z., & Fei, M. (2017). Biogeography-based optimization: A 10 year review, IEEE Trans. Emerging Top. Comput. Intell. 1(5), 391–407, https://doi.org/10.1109/TETCI.2017.2739124.

O'Dea, B., Wan, S., Batterham, P., Calear, A., Paris, C., & Christensen, H. (2015). Detecting Suicidality on Twitter. ELSEVIER: Internet Interventions, 183-188. doi:10.1016/j.invent.2015.03.005.

Omid B., Sayed-Mohammad H-M, & Hugo A. L. (2016). Biogeography-Based Optimization Algorithm for Optimal Operation of Reservoir Systems. F.A.S.C.E. Journal of Water Resources Planning and Management/ Volume 142 Issue 1-January 2016.

Rashkin H., Bell E., Choi Y., & Volkova S. (2017). Multilingual connotation frames: a case study on social media for targeted sentiment analysis and forecast. In Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL 2017), https://doi.org/10.18653/v1/P17-2073.

Schukla, A. (2018). Sentiment analysis of document based on annotation", CORR Journal, Vol. abs/1111.1648.

Simon, D. (2008). Biogeography-based optimization" (PDF). IEEE Transactions on Evolutionary Computation. 12 (6): 702–713. doi:10.1109/tevc.2008.919004.

Turban, Efrain, David King, Judy Lange. Introduction to Electronic Commerce. (New Jersey,2011, Prentice Hall).

Usama, M., Xiao, W., Ahmad, B., Wan, J., Hassan, M. M., & Alelaiwi, A. (2019). Deep Learning Based Weighted Feature Fusion Approach for Sentiment Analysis. IEEE Access, 7, pp. 140252-140260, https://doi.org/10.1109/ACCESS.2019.2940051, Corpus ID: 203171877.

Vateekul, P. & Koomsubha, T. (2016). A Study of Sentiment Analysis Using Deep Learning Techniques on Thai Twitter Data. Conference: 2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE). doi:10.1109/JCSSE.2016.7748849.

Waldron, M. (2016). Building a text analysis process for customer reviews in RapidMiner. AYLIEN.http://blog.aylien.com/building-a-text-analysis-process-for-customer-reviews-in-rapidminer.

Wang, J., Fu J., Xu, Y., & Mei T. (2019). Beyond object recognition: visual sentiment analysis with deep coupled adjective and noun neural networks. In Proceedings of the Internal Joint Conference on Artificial Intelligence.

Wang, Y., Huang, M., Zhu, X., & Zhao, L. (2016). Attention-based LSTM for aspect-level sentiment classification. In Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2016), 2016, https://doi.org/10.18653/v1/D16-1058.

Wikarsa, L., & Thahir, S. (2015). A Text Mining Application of Emotion Classification of Twitter Users Using Naive Bayes Method. International Conference on Wireless and Telematics (pp. 1-6). IEEE. doi:10.1109/ICWT.2015.7449218.

Xiang, Z., & Gretzel, U. (2010). Role of social media in online travel information search. T, Researchgate, https://doi.org/10.1016/j.tourman.2009.02.016.

Yang, M., Tu, W., Wang, J., Xu F, & Chen X. (2017). Attention-based LSTM fortarget dependent sentiment classification.In Proceedings of AAAI Conference on Artificial Intelligence (AAAI 2017), https://doi.org/https://doi.org/10.1609/aaai.v31i1.11061.

Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O. (2021). Understanding deep learning requires rethinking generalization. In international Conference on Learning Representations. Oriol Vinyals Communications of the ACM, March 2021, Vol. 64 No. 3, Pages 107-115, https://doi.org/10.1145/3446776.

Zhang, R. & Tran, T. T. (2019). Helping e-commerce consumers make good purchase decisions: a user reviews-based approach," in International Conference on E-Technologies, pp. 1–11,Springer Berlin Heidelberg. doi:10.1007/978-3-642-01187. Corpus ID: 22350244.

Zhang, L., Wang, S., Liu B. (2018). Deep Learning for sentiment Analysis: A Survey, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), pp. 1-34. Huawei Technologies Co. Ltd; National Science Foundation (NSF), Grant/Award numbers: IIS-1650900, IIS1407927. https://doi.org/10.1002/widm.1253. published: 30 March 2018.

APPENDIX

```
clc; clear; close all;
%% PROBLEM PARAMETERS
nVar=2;            % Number of Decision Variables
VarSize=[1 nVar];  % Decision Variables Matrix Size
VarMin1=0.01;      % Decision Variables Lower Bound
VarMax1= 0.1;      % Decision Variables Upper Bound
VarMin2=5;         % Decision Variables Lower Bound
VarMax2= 50;       % Decision Variables Upper Bound
%% BBO Parameters
MaxIt=1;           % Maximum Number of Iterations
nPop=30;           % Number of Habitats (Population Size)
KeepRate=0.2;                  % Keep Rate
nKeep=round(KeepRate*nPop);    % Number of Kept Habitats
nNew=nPop-nKeep;               % Number of New Habitats
% Migration Rates
mu=linspace(1,0,nPop);         % Emmigration Rates
lambda=1-mu;                   % Immigration Rates
alpha=0.9;
pMutation=0.1;
sigma=0.02*(VarMax2-VarMin1);
%% Initialization
% Empty Habitat
habitat.Position=[];
habitat.Cost=[];
% Create Habitats Array
pop=repmat(habitat,nPop,1);
% Initialize Habitats
```

```
tic;
load sentimentData.mat;
load emb.mat
[dataTrain, dataTest,inputSize] = dataPrep(sentimentData, emb);
for i=1:nPop
    a2=randi([VarMin2,VarMax2],1,1);
    a1=(VarMin1+(VarMax1-VarMin1).*rand(1,1));
    a1=str2num(sprintf('%0.2f',a1));
    pop(i).Position=[a1, a2];
    [pop(i).Cost, net, Ypred]=FitFunction(pop(i).Position, dataTrain,dataTest,inputSize);
%     pop(i).Cost=FitFunction(pop(i).Position);
end

% Sort Population
[~, SortOrder]=sort([pop.Cost],'descend');
pop=pop(SortOrder);

% Best Solution Ever Found
BestSol=pop(1);

% Array to Hold Best Costs
BestCost=zeros(MaxIt,1);

%% BBO Main Loop
for it=1:MaxIt
    newpop=pop;
    for i=1:nPop
        for k=1:nVar
        % Migration
        if rand<=lambda(i)
            % Emmigration Probabilities
            EP=mu;
            EP(i)=0;
            EP=EP/sum(EP);
            % Select Source Habitat
            j=RouletteWheelSelection(EP);

            % Migration
            newpop(i).Position(k)=pop(i).Position(k) ...
                +alpha*(pop(j).Position(k)-pop(i).Position(k));

        end

        % Mutation
        if rand<=pMutation
            newpop(i).Position(k)=newpop(i).Position(k)+sigma*randn;
        end
    end

    % Apply Lower and Upper Bound Limits
    newpop(i).Position(1) = max(newpop(i).Position(1), VarMin1);
    newpop(i).Position(1) = min(newpop(i).Position(1), VarMax1);
    newpop(i).Position(2) =round(max(newpop(i).Position(2), VarMin2));
    newpop(i).Position(2) = round(min(newpop(i).Position(2), VarMax2));
    % Evaluation
    [pop(i).Cost, net, Ypred]=FitFunction(pop(i).Position, dataTrain,dataTest,inputSize);
    end

    % Sort New Population
    [~, SortOrder]=sort([newpop.Cost],'descend');
    newpop=newpop(SortOrder);

    % Select Next Iteration Population
    pop=[pop(1:nKeep)
         newpop(1:nNew)];

    % Sort Population
    [~, SortOrder]=sort([pop.Cost],'descend');
    pop=pop(SortOrder);

    % Update Best Solution Ever Found
    BestSol=pop(1);

    % Store Best Cost Ever Found
    BestCost(it)=BestSol.Cost;
    timeSpent=toc;
    % Show Iteration Information
    disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCost(it))]);

end
%% Results
figure;
%plot(BestCost,'LineWidth',2);
semilogy(BestCost,'LineWidth',2);
```

```
xlabel('Iteration');
ylabel('Best Cost');
grid on;
heatmap(table(Ypred,dataTest.Sentiment,'VariableNames',{'Predicted','Actual'}),...
    'Predicted','Actual');

function [dataTrain, dataTest,inputSize] = dataPrep(sentimentData, emb)
doTraining = true; % Training will take ~10 minutes
% doWordEmbeddingTraining = false; % Training a word embedding will take ~10 minute
% load sentimentData.mat;
data=sentimentData;
head(data);
data.Sentiment = categorical(data.Sentiment);
data.Sentiment = renamecats(data.Sentiment,{'1','0'},{'positive','negative'});
data.TextDocuments = preprocessTweets(data.SentimentText);
[~, idx] = removeEmptyDocuments(data.TextDocuments);
data(idx,:) = [];
% load emb.mat
data.Vec = prepData(emb,data.TextDocuments);
[dataTrain,dataTest] = partitionData(data,0.3);
inputSize = emb.Dimension;
%% functions
% function data = readSentimentData(filename,n)
% % Create datastore and read n rows of data
%     ds = datastore(filename,'TextType','string');
%     ds.SelectedVariableNames = {'Sentiment','SentimentText'};
%     ds.SelectedFormats = {'%q','%q'};
%     ds.ReadSize = n;
%     data = read(ds);
% end

function dataReadyForLSTM = prepData(emb,documents)
% Prep the data for the LSTM network

    % Transform tweets for model
    dataReadyForLSTM = doc2sequence(emb,documents);

    % Get the max length and left-pad all sequences with zeros
    sizes = cellfun(@(x) size(x,2),dataReadyForLSTM);
    maxLength = max(sizes);
    parfor i = 1:numel(dataReadyForLSTM)
        dataReadyForLSTM{i} = leftPad(dataReadyForLSTM{i},maxLength);
    end
end

% function tweets = getTweets(predictedSentiment, actualSentiment, predictedData,
testData, tweetData)
% % Get the tweets that were a given predicted sentiment and actual sentiment (ex. tweets
that were predicted negative but were actually positive)
%     predicted = find(predictedData == predictedSentiment);
%     actual = find(testData == actualSentiment);
%     falseActualIndices = predicted(ismember(predicted,actual));
%     tweets = tweetData(falseActualIndices);
% end

% function MPadded = leftPad(M,N)
% % Add padding to left
%     [dimension,sequenceLength] = size(M);
%     paddingLength = N - sequenceLength;
%     MPadded = [zeros(dimension,paddingLength) M];
% end

function [dataTrain, dataTest] = partitionData(data, holdout)
% Partition tweet dataset with a given holdout
   rng(1234)
   % Split the data into training and test sets
   cv = cvpartition(data.Sentiment,'Holdout',holdout);
   dataTrain = data(training(cv),:);
   dataTest = data(test(cv),:);
end

% function s = calculateScore(score)
% % Calculate one overall score for each tweet
%     score(:,1) = -score(:,1);
%     s = sum(score,2,'omitnan');
% end

end

function [accuracy, net, Ypred] = FitFunction(X, dataTrain,dataTest,inputSize)
doTraining = true; % Training will take ~10 minutes

%% Create LSTM Network
numHiddenUnits = X(2);
LearnRate=X(1);
```

```
numClasses = numel(categories(dataTrain.Sentiment));
% numClasses = numel(unique(dataTrain.Sentiment));
maxEpochs = 20;
% miniBatchSize = 27;
layers = [ sequenceInputLayer(inputSize)
   lstmLayer(numHiddenUnits,'OutputMode','last')
   fullyConnectedLayer(numClasses)
   softmaxLayer
   classificationLayer ]
options = trainingOptions('adam',...
   'InitialLearnRate',LearnRate,...
   'SequenceLength','longest', ...
   'Shuffle','never', ...
   'MaxEpochs',maxEpochs, ...
   'GradientThreshold',1,...
   'ExecutionEnvironment','auto',...
   'Plots','training-progress','Verbose',1);
%% Train network
   net = trainNetwork(dataTrain.Vec,dataTrain.Sentiment,layers,options);
%     save LSTMModel.mat net
%% Test Test the Model
[Ypred,scores] = classify(net,dataTest.Vec);
accuracy = sum(Ypred == dataTest.Sentiment)/numel(Ypred);
% Fitness=100-accuracy;
% heatmap(table(YPred,dataTest.Sentiment,'VariableNames',{'Predicted','Actual'}),...
%     'Predicted','Actual');
%% functions
function s = calculateScore(score)
```

```
% Calculate one overall score for each tweet
    score(:,1) = -score(:,1);
    s = sum(score,2,'omitnan');
end

end
```