Tribhuvan University
Institute of Science & Technology

Nepali Text Document classification using Deep Neural Network

Dissertation

Submitted to

Central Department of Computer Science & Information Technology
Kirtipur, Kathmandu, Nepal

In partial fulfillment of the requirements
for Master's Degree in Computer Science & Information Technology

By:
Sanjiv subba limbu
September, 2018

Under the Supervision of :

Mr. Nawaraj Paudel

Co-Supervisor :

Mr. Tej Bahadur Shahi

Central Department of Computer Science and IT
Tribhuvan University, Kritipur
Kathmandu, Nepal

Tribhuvan University
Institute of Science & Technology
Central Department of Computer Science & Information Technology

Student's Declaration

I hereby declare that I am the only author of this work and that no source other than the listed here has been used in this work.

……

Sanjiv subba limbu

Supervisor's Recommendation

I hereby recommend that this dissertation prepared under supervision by **Mr. Sanjiv subba limbu** entitled **"Nepali Text Document classification using Deep Neural Network''** in partial fulfillment of the requirements for the degree of M.Sc. in Computer Science & Information Technology be processed for the evaluation.

……

Mr. Nawaraj Paudel

Central Department of Computer Science and IT
Tribhuvan University, Kritipur
Kathmandu, Nepal

Date: September, 2018

Tribhuvan University
Institute of Science & Technology
Central Department of Computer Science & Information Technology

LETTER OF APPROVAL

We certify we have read this dissertation and in our opinion it is satisfactory in the scope and quality as a dissertation in the partial fulfillment for the requirement of master's degree in Computer Science & Information Technology.

Evaluation Committee

……..
Mr. Nawaraj Paudel

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

(Head)

..……..
Mr. Nawaraj Paudel

Central Department of Computer
Science & Information Tech.
Tribhuvan University, Kritipur,
Kathmandu, Nepal

(Supervisor)

……
(Internal examiner)

……
(External examiner)

Date: September, 2018

AKNOWLEDGEMENTS

First and foremost, I owe my deepest gratitude to my supervisor, Mr. Nawaraj Paudel. I also express my warmest gratitude to my co-supervisor Mr. Tej Bahadur Shahi. Without their assistance, involvement throughout process, encouragement and support this work would hardly have been completed. Thank you very much for all the support and understanding.

I want to express my deepest gratitude to Kathmandu Center for Research and Education, CAS-TU (KCRE) for providing me scholarship which helped me lot to conduct my research work smoothly.

I want to express my gratitude to everyone who supported me throughout the course of M.Sc. CSIT research work.

I thank profusely all the faculty members and staffs of Central Department of Computer Science and Information Technology for their kind help and co-operation throughout my study period.

I want to express my gratitude to everyone who are related directly or indirectly to complete my research work.

Finally, It is my privilege to thank my family and friends, for their constant support, encouragement throughout my research period.

Sanjiv Subba Limbu

September,9,2018

ABSTRACT

Document classification task is to assign a document to one or more classes or categories. This report focuses on classification of Nepali and English Text document with Neural Network like RNN(Recurrent Neural Network) and MNN (Multi Neural Network) and compare the performance of RNN and MNN based on well-known performance evaluation parameter. The result of the Experiment showed RNN outperform MNN.

Keyword: Document classification, RNN, MNN, Neural Network, Artificial Intelligence

Table Of Contents

LIST OF TABLES

TABLE OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

Deep Learning is a new area of Machine Learning research, which has been introduced with the objective of moving Machine Learning closer to one of its original goals: Artificial Intelligence.

Artificial intelligence (AI, also machine intelligence, MI) is intelligence demonstrated by machines, in contrast to the natural intelligence (NI) displayed by humans and other animals. In computer science AI research is defined as the study of "intelligent agents": any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals.[1][2] Artificial intelligence is applied when a machine mimics "cognitive" functions that humans associate with other human minds, such as "learning" and "problem solving". [2] Artificial intelligence (AI) is an area of computer science that emphasizes the creation of intelligent machines that work and react like humans. Artificial intelligence is a branch of computer science that aims to create intelligent machines. It has become an essential part of the technology industry.

Machine learning is a field of computer science that uses statistical techniques to give computer systems the ability to "learn" (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed. [3] Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is sometimes conflated with data mining

Deep learning is part of a broader family of machine learning methods based on learning data representations, as opposed to task-specific algorithms. Learning can be supervised, semi-supervised or unsupervised. [4] Deep learning architectures have been applied to fields including computer vision, speech recognition, natural language processing, audio recognition, social network filtering etc. where they have produced results comparable to and in some cases superior to human experts.[5]

Deep learning is a class of machine learning algorithms that:[6]

- ❖ use a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
- ❖ learn is supervised (e.g., classification) and/or unsupervised (e.g., pattern analysis) manners.
- ❖ learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.

1.1.1 Artificial Neural Network

Artificial neural network (ANN) or neural network are computing systems inspired by the biological neural networks that constitute brains. Such systems "learn" (i.e. progressively improve performance on) tasks by considering examples, generally without task-specific programming. In artificial intelligence (AI), new advances make it possible that artificial neural networks (ANNs). ANN learn to solve complex problems in a reasonable amount of time .ANNs are theoretical vehicles that aid in the understanding of neural information processing .[7]

An ANN is based on a collection of connected units or nodes called artificial neurons. Each connection between artificial neurons can transmit a signal from one to another. The artificial neuron that receives the signal can process it and then signal artificial neurons connected to it.

1.1.1.1 Mathematical model of Artificial Neuron

For the simple understanding of how multi-neural network work can be understood by the single layer neural network. In multi-neural network the computation process are same but more hidden layers are introduced in the network.

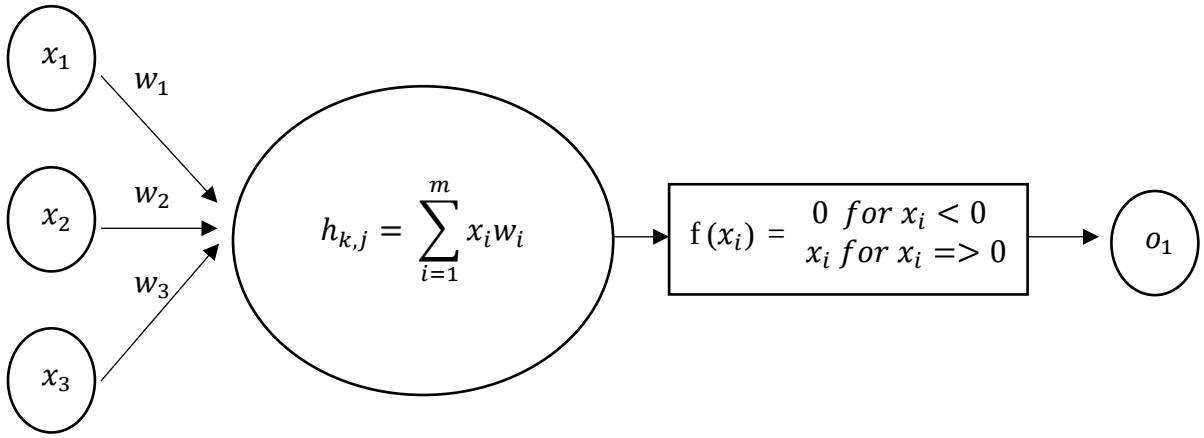

Figure 1:. working of single-layer perceptron network

Where,

x_i= inputs supplied to the neural network.

w_i= weight.

$h_{k,j}$= hidden unit.

f (x_i) = activation function.

o_i= output unit.

Computation in neurons are carried by equation, $h_{k,j} = \sum_{i=1}^{m} x_i w_i$

1.1.2 Recurrent Neural Network (RNN)

A recurrent neural network (RNN) is a class of artificial neural network where connections between units form a directed graph along a sequence. This allows it to exhibit dynamic temporal behavior for a time sequence RNNs can use their internal state (memory) to process sequences of inputs. Recurrent neural networks (RNNs) contain cyclic connections that make them a more powerful tool to model such sequence data. [8]

In a traditional neural network, we assume that all inputs are independent of each other but RNNs hold idea of dependent of inputs in sentences. E.g. To predict the next word in a sentence RNNs store information of word appeared few steps back. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations. In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are limited to looking back only a few steps. Recurrent Neural Network uses backpropagation algorithm, but it is applied for every time stamp. It is commonly known as Backpropagation Through Time (BTT).

1.2 Problem Definition

Ideally, In the text domain, the classifier should classify incoming documents to the right existing classes used in training and also detect those documents that don't belong to any of the existing classes. This problem is called open world classification or open classification.[9]

Unstructured, unorganized data are available in huge amount in digital world. Using the data directly from the source is not possible because of the unstructured data. Without organizing the textual data, benefit from available piles of data is not possible.

For document classification MNN and RNN approach is used. This paper intends to understand the result of different setting of hyper parameter of neural network model and to find MNN and RNN document classification efficiency.

1.3 Objectives

The main objectives of this research work are to classify the textual data in the form of text document using the deep learning algorithm like Recurrent Neural Network (RNN) and Multi Neural Network (MNN). With Deep learning computational techniques we try to discover useful information hidden across data and classify them.

1.4 Report organization

This thesis is organized as: chapter 1 gives introduction to deep learning, artificial neural network, chapter 2 gives background information on components for the implementation process of RNN and Multi Neural Network, chapter 3 gives information on research methodology, chapter 4 gives details on implementation, chapter 5 gives information on result and analysis and chapter 6 concludes thesis.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

2.1 Background

2.1.1 Neural Network Model

$$\sum_{i=1}^{m} x_i w_{n,i}$$

The neurons computed $\sum_{i=1}^{m} x_i w_{n,i}$ and computed value is then passed into activation function to map $h_{k,j}$to corresponding output value. The process of summation of product weight and input is again continued in another layer neurons and so on till the output node reached.

Where,

f = activation function.

m = number of inputs supplied to the neurons.

x_i= inputs.

$w_{n,i}$= weights.

$h_{k,j}$= hidden units.

k = layer number

e.g. 1 for first layer

2 for second layer and so on.

2.1.2 Multi Neural Network

The Multi-layer neural network or multi-layer perceptron (MLP) consists of three or more layers (an input and an output layer with one or more hidden layers) of nonlinearly-activating nodes making it a deep neural network. The multilayer perceptron is the most known and most frequently used type of neural network. On most occasions, the signals are transmitted within the network in one direction: from input to output. There is no loop, the output of each neuron does not affect the neuron itself. This architecture is called feedforward. There are also feed-back networks, which can transmit impulses in both directions, due to reaction connections in the network. These types of networks are very powerful and can be extremely complicated. [10]

Since MLPs are fully connected, each node in one layer connects with a certain weight to every node in the layers. Back propagation algorithm is used to adjust weights in the network. Activation function of a node defines the output of that node.

2.1.3 Activation Function

In computational networks activation function of a node defines the output of that node given an input or set of inputs. A standard computer chip circuit can be seen as a digital network of activation function than can be "ON" (1) or "OFF" (0), depending on input. Activation function is important to map particular output to a particular set of inputs.

2.1.3.1 Rectified Linear Unit (ReLU) Activation Function

Rectified Linear Unit (ReLU) activation function is implemented for the Multi-Neural Network. [11] The range of ReLU is from 0 to 1.The boundary equation for ReLU is shown below

$$\mathrm{f}(x_i) = \begin{matrix} 0 \ for\ x_i < 0 \\ x_i \ for\ x_i => 0 \end{matrix}$$

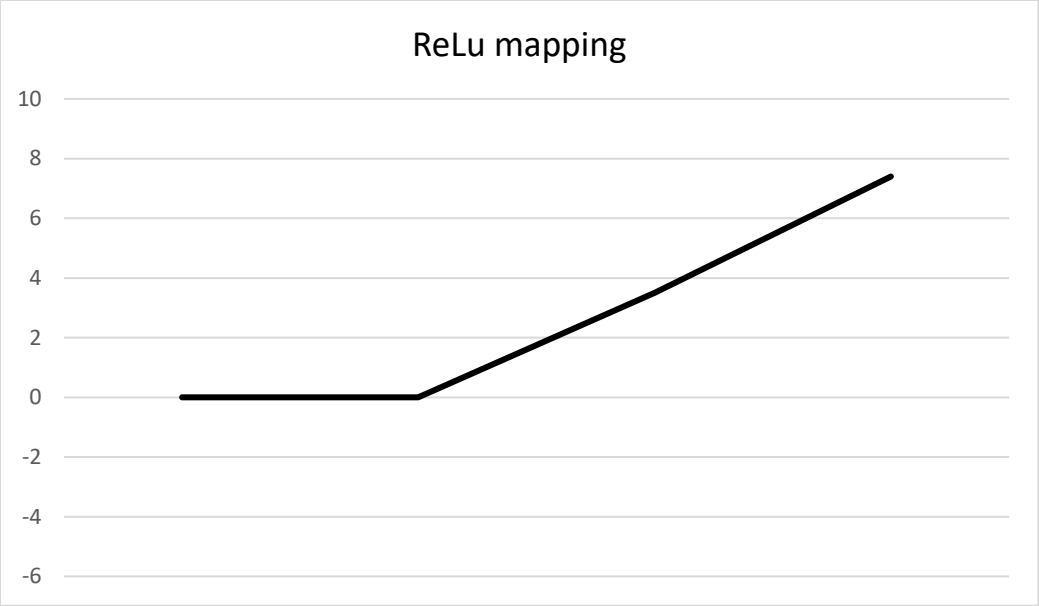

Figure 2:. Graph of ReLU mapping input to corresponding output.

2.1.3.2 Tanh activation function

Tanh activation function is implemented for the Recurrent Neural Network. tanh(x) = $\frac{2}{1+e^{-2x}}$ -1

The range of the tanh function is from -1 to 1

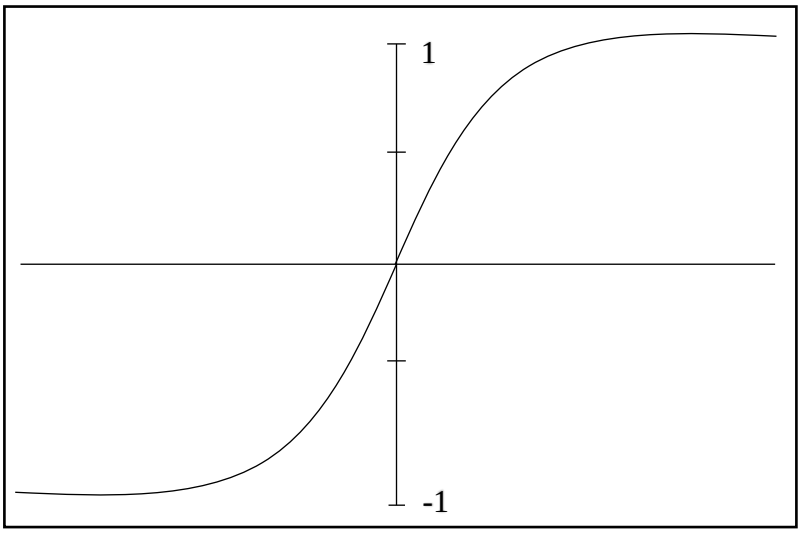

Figure 3:. Graph of Tanh activation function mapping input to corresponding output.

2.1.4 SoftMax Function

SoftMax function takes output of neural network e.g. o_1 (output) from above figure as input and produce the output as probability distribution.

SoftMax function force the outputs of the neural network to sum to 1 so they can represent a probability distribution across discrete output which is mutually exclusive.

SoftMax function is given by, [12]

$$o_i = \frac{e^{z_i}}{\sum_{i=1}^{j} e^{z_i}}$$

Where, o_i = output of SoftMax function for i^{th}number input.

j = total number of input and $z_i = i^{th}$ input for SoftMax function.

2.1.5 Cross-Entropy Loss Function

Cross-entropy is the cost function while training the neural network model. The cross-entropy loss function is the method to measure the distance between predicted probability of the neural network model and the expected label or class. [12]

Cost function is given by,

$$L = -\sum_j t_j \log y_j$$

Where,

j = total number of the predicted output.

t_j = targeted value or class label for j^{th} predicted output.

$y_j = j^{th}$ predicted output of neural network model.

L = loss function

When the value of L (loss function) is low it is considered the low distance between the predicted probability of neural network model and expected result i.e. the error or loss of the neural network model is low and vice versa if value of L (loss function) goes high.

2.1.6 Gradient Descent Algorithm

Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function. To find a local minimum of a function using gradient descent, one takes steps proportional to the negative of the gradient (or approximate gradient) of the function at the current point. If instead one takes steps proportional to the positive of the gradient, one approaches a local maximum of that function; the procedure is then known as gradient descent.

Gradient descent is a way to minimize an objective function J (θ_i) parameterized by a model's parameters (θ_i) by updating the parameters in the opposite direction of the gradient of the objective function with respect to the parameters. The learning rate (α) determines the size of the steps we take to reach a (local) minimum. In other words, we follow the direction of the slope of the surface created by the objective function downhill until we reach a valley.[13]

surface created by the objective function downhill until we reach a valley.5 In gradient descent output value generated by the neural network model is taken into account then little step is taken in direction to the correct value or target value in order to decrease error in neural network model and the step is continued until the output value converge to the local minima or the neural network is considered to be trained.

Convergence function, $\theta_j' := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ [13]

Where, α = learning rate, it tells how big step should be taken in process of convergence.

$J(\theta_0, \theta_1)$ = function to minimize. $\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$ = it calculates slope at the point (θ_0, θ_1)

θ_j = present gradient descent value. θ_j' = new gradient descent value.

$\theta_j \in \mathbb{R}$, where $\mathbb{R}$ = real number

For example, considering the function with only one parameter, J (θ_1)

$\theta_j' := \theta_j - \alpha \frac{d}{d\theta_j} J(\theta_1)$

J (θ_1)

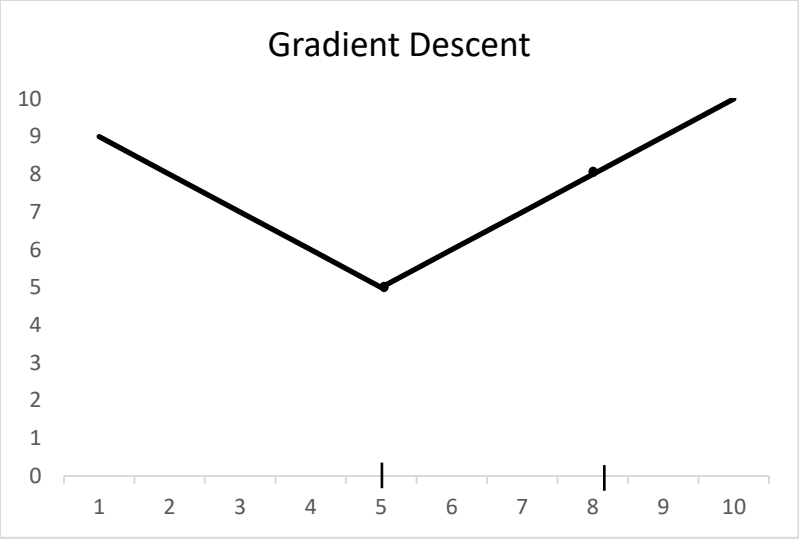

Figure 4:. gradient descent when slope is positive.

When $\theta_1 = 8$

Minimal point is at $\theta_1 = 5$

derivative of cost function, J (θ_1) is calculated to find the slope at point $\theta_1 = 8$ in function curve or graph. $\frac{d}{d\theta_j}$ J (θ_1) = positive value and θ_j':= θ_j – α * (positive value)

It causes the θ_j' value to decrease which is taking the step toward the minimal point for the purpose

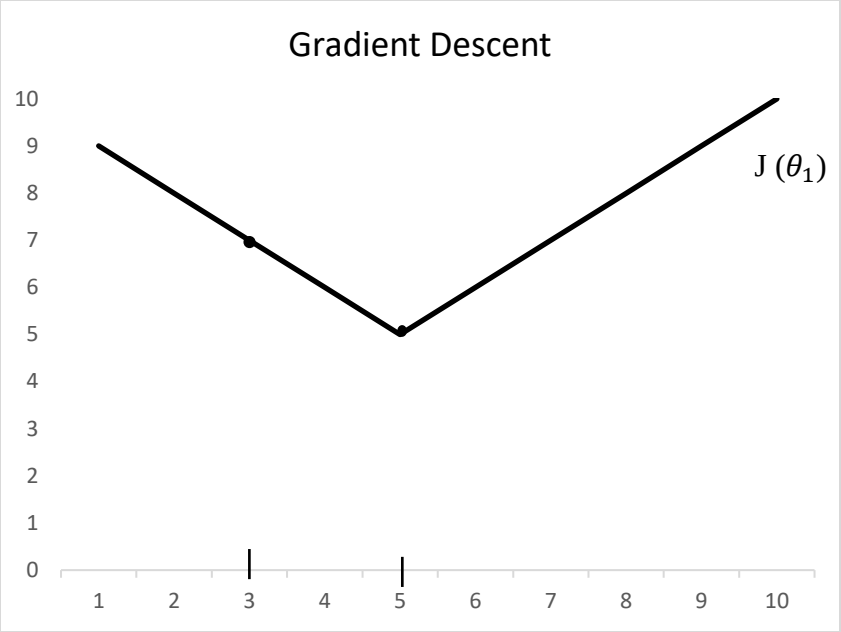

Figure 5:. gradient descent when slope is negative

When $\theta_1 = 3$

Minimal point is at $\theta_1 = 5$

derivative of cost function, J (θ_1) is calculated to find the slope at point $\theta_1 = 3$ in function curve or graph

$\frac{d}{d\theta_j}$ J (θ_1) = negative value

θ_j':= θ_j – α * (negative value)

It causes θ_j'values to increase its new value moving toward the minimal point for the purpose of convergence.

2.1.7 Vanishing/Exploding Gradient problem in Deep Network

When training neural network model vanishing/exploding gradient problem is derivative of cost function sometimes get exponentially big or exponentially small this makes training difficult.

When training neural network model with backpropagation RNN suffer Vanishing/Exploding Gradient problem. When the problem takes place, training takes too long and the accuracy suffers.

While training neural network constantly calculates cost value. Cost value is difference of the predicted output value of neural network and expected value from a set of labelled training data.

Cost is minimized by making adjustments to the weights over and over throughout the training process until the lowest possible cost value is obtained.

Training process utilized a gradient, which measures the rate at which the cost will change with respect to a change in a weight.

2.1.7.1 Vanishing gradient

When the gradient is very small, neural network train slowly. The gradient could potentially vanish or decay back through the neural network as a result, the early layers of the network are the slowest to train.

Vanishing gradient cause network only to remember recent events and forgets more distant previous events. Gradient at any point is the product of the previous gradients up to that point.

2.1.7.2 Exploding gradient

When the gradient is very large, neural network train quickly. Exploding gradients are a problem where large error gradient accumulate and result in very large updates to neural network model weights during training. This has the effect in neural network model, being unstable and unable to learn from training data.

2.1.7 Back propagation

While the network is being train the loss function (cross-entropy) measure the error of the neural network model. The error or cost of the neural network model is the difference between the output value predicted by neural network model and expected output value for neural network model. [14] Neural network model calculates the error to go back through the network adjusting the weights so the next time error will be small. Gradient descent is used to reduce the error of the neural network model. Back propagation algorithm propagates back through the network adjusting weights and process is repeated for all the inputs i.e. training data set. This process is repeated until the error gets small enough that it can consider a trained.

2.1.8.1 Back Propagation Through Time (BPTT)

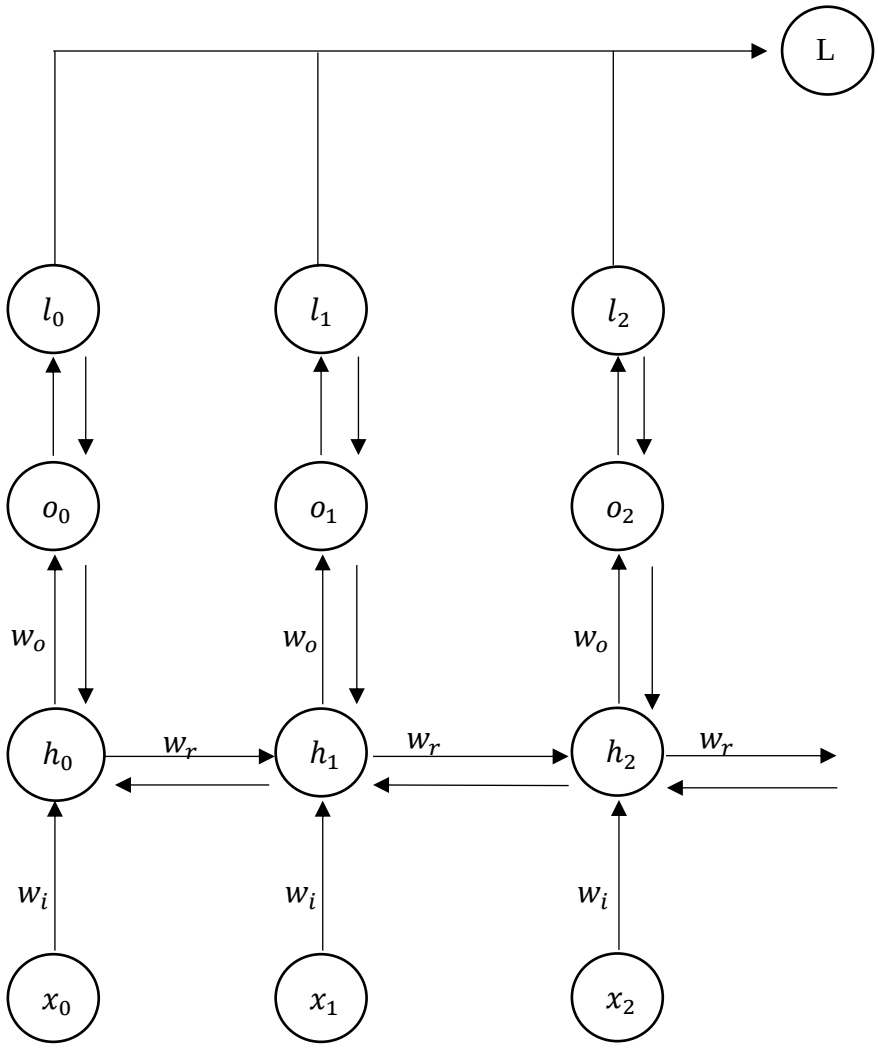

Figure 6:. Simple Recurrent Neural Network (RNN) with BPTT

Back propagation in feedforward networks moves backward from the final error through the outputs, weights and inputs of each hidden layer. The error(loss) is calculated and optimizer is then used to adjust the weight which decreases the error.

In order to compute the backpropagation, the loss function is required.

Where,

l_i= loss of the i^{th} output or element-wise loss.

Back propagation calculation is carried out through the neural network model. To compute back propagation the loss function is required.

At first element wise loss is calculate i.e. loss in every output (l_i)

Cross-entropy loss function shown below,

L_e= - $t_i \log o_i$

Where,

L = over all loss.

i = total number of the predicted output.

t_i = targeted value or class label for i^{th} predicted output.

o_i= i^{th} predicted output of neural network model.

L_e= element wise loss of o_i

Now, overall loss for the entire sequence data is shown below

L = $-\sum_i t_i \log o_i$

Backpropagation computes in opposite directions to all of the forward propagation steps. Backpropagation process allows to compute appropriate quantities and update the parameters of the neural network model so the error generated is minimized.

BPTT utilizes the gradient decent to update the weight and bias of neural network which in turns minimize the error in neural network.

2.1.8 LSTM (long Short-Term Memory)

Long short-term memory (LSTM) units (or blocks) are a building unit for layers of a recurrent neural network (RNN). An RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a cell, an input gate, an output gate and a forget gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three gates can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation (using an activation function) of a weighted sum. Intuitively, they can be thought as regulators of the flow of values that goes through the connections of the LSTM. [8]

The RNN cell is simple unit which compute the inputs and bias.

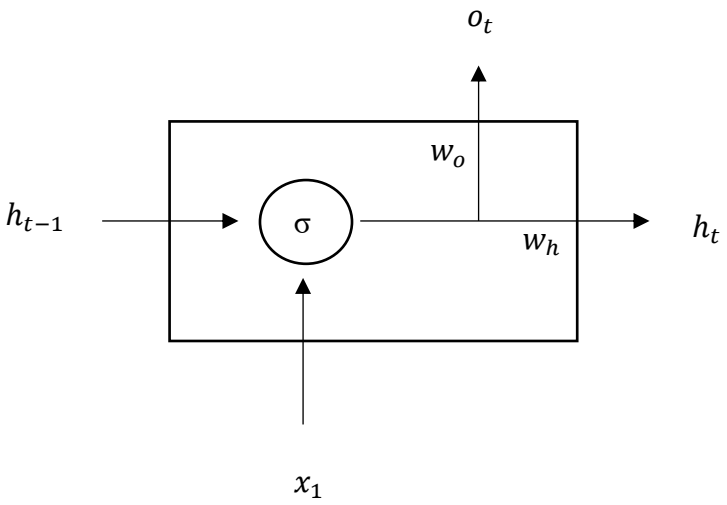

Figure 7:. block diagram of RNN cell.

Where,

h_{t-1}, h_t= hidden state.

h_t= σ (w_a*h_{t-1}+$w_h * x_t$)

x_t= input to the RNN cell.

σ = activation function.

o_t=output value of RNN cell. t = time step and w_a, w_h= weights.

LSTM unit have the memory cell (C) and it provide bit of memory to remember.

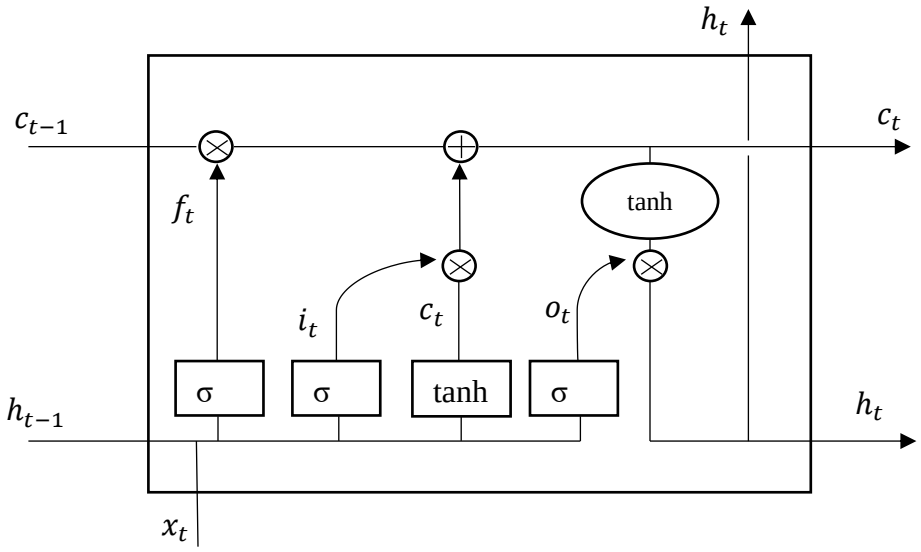

Figure 8:. block diagram of LSTM cell.

Where,

f_t= σ (w_f*[h_{t-1}, x_t]) = forget gate.

i_t= σ (w_i*[h_{t-1}, x_t]) = input gate.

$\hat{c}_t$= σ (w_c*[h_{t-1}, x_t]) = new information to update.

c_t= f_t*c_{t-1} + i_t*$\hat{c}_t$ = new cell state.

o_t= σ (w_0*[h_{t-1}, x_t]) = output gate.

h_t= o_t ∗ tanh (c_t) = hidden state.

c_{t-1}= previous cell state.

c_t= current cell state.

x_t= input value at t^{th}time step.

h_{t-1}= previous hidden state.

First step in LSTM is to decide what information to throw away from cell state. This decision is made by sigmoid layer called “forget gate layer”.

f_t is computed to check if the previous cell state value should forget or not.

It outputs a number between 0 and 1 for each number in the cell state

If output is 1 then the value is kept in cell state.

If output is 0 then forget the value.

If value is 0.7 then keep exactly 70% of the current data. This similar concept is also applied for rest of other gates.

f_t compute how much previous state is going to contribute to cell state based on hidden state and input.

Second step in LSTM is to decide what new information to store in the cell state. It consists of two parts. First sigmoid layer called “input gate layer” decides which values to update. Next, tanh layer created a vector of new candidate values, $\hat{c}_t$that could be added to the state.

i_t is computed to check if update is required if value is 1 then it cleans previous value and update with new information. The new information is $\hat{c}_t$.

$\hat{c}_t$ computes what is new information going to cell state based on hidden state and input.

Third step in LSTM, i_t , $\hat{c}_t$, f_t,c_{t-1} are combined to create an update to the cell state.

To Update the old cell state, c_{t-1} into the new cell state c_t.The new candidate values, scaled by how much we decided to update each state value.

$c_t = f_t * c_{t-1} + i_t * \hat{c}_t$

Finally, we decide what we are going to output. This output will be based on our cell state but will be a filtered version. First, we run sigmoid layer through tanh and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to.

$o_t = \sigma\,(w_0 * [h_{t-1}, x_t])$ and $h_t = o_t * \tanh\,(c_t)$

2.2 Literature Review

In 1943 Warren McCulloch and Walter Pitts created a computational model for neural networks based on mathematics and algorithms called threshold logic. This model paved the way for neural network research. In paper they highlighted how neurons might work. In order to describe how neurons in the brain might work, they modeled a simple neural network using electrical circuits. [15]

In 1949, Donald Hebb wrote The Organization of Behavior, a work which pointed out the fact that neural pathways are strengthened each time they are used, a concept fundamentally essential to the ways in which humans learn. If two nerves fire at the same time, he argued, the connection between them is enhanced.

Rosenblatt (1958) created the perceptron, an algorithm for pattern recognition. With mathematical notation, Rosenblatt described circuitry not in the basic perceptron, such as the exclusive-or circuit that could not be processed by neural networks at the time. [16] This was important in development of the neural network later.

In 1962, Widrow & Hoff developed a learning procedure that examines the value before the weight adjusts it (i.e. 0 or 1) according to the rule:

Weight Change = (Pre-Weight line value) * (Error / (Number of Inputs)).

It was based on the idea that while one active perceptron may have a big error, one can adjust the weight values to distribute it across the network, or at least to adjacent perceptron. Applying this rule still results in an error if the line before the weight is 0, although this will eventually correct itself. If the error is conserved so that all of it is distributed to all of the weights than the error is eliminated. They developed learning procedure that examines the value before the weight adjusts.

Neural network research stagnated after machine learning research by Minsky and Papert (1969) They discovered two key issues with the computational machines that processed neural networks. The first was that basic perceptron were incapable of processing the exclusive-or circuit. The second was that computers didn't have enough processing power to effectively handle the work required by large neural networks. Neural network research slowed until computers achieved far greater processing power.

The first multilayered network was developed in 1975, an unsupervised network.

The term Deep Learning was introduced to the machine learning community by Rina Dechter in 1986, [17] and Artificial Neural Networks by Igor Aizenberg and colleagues in 2000.

Support Vector Machines were applied to text classification in 1998 [18]

AdaBoost was enhanced to handle multi-labels in 2000 by Schapire and Singer. They showed approach where, the task of assigning multi-topics to a text is regarded as a ranking of labels for the text. This ranking-based evaluation was inspired by Information Retrieval.

In 2009, neural network raised again in different concept as deep learning by Hinton.

In 2018, Yu Meng, Jiaming Shen, Chao Zhang, Jiawei Han in paper "Weakly-Supervised Neural Text Classification" compared the classification performance of different neural classifiers . In their experiment they evaluated the empirical performance of their method for weakly supervised text classification. They proposed a weakly-supervised text classification method built upon neural classifiers. This work was significant as result shown that method outperforms baseline methods significantly, and it is quite robust to different settings of hyperparameters and different types of user-provided seed information. They suggested to study on how to effectively integrate different types of seed information to further boost the performance of method. [19]

In 2018, Neel Kant, Raul Puri, Nikolai Yakovenko, Bryan Catanzaro in paper "Practical Text ClassificationWith Large Pre-Trained Language Models" .They compare the deep learning architectures of the Transformer and mLSTM and found that the Transformer outperforms the mLSTM across categories.They demonstrated that unsupervised pretraining and finetuning provides a flexible framework that is effective for difficult text classification tasks. They found that the finetuning was especially effective with the transformer network.[20]

Has¸Im Sak, Andrew Senior, Franc¸Oise Beaufays in paper "Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling" evaluated and compared the performance of LSTM RNN architectures on a large vocabulary speech recognition task – the Google Voice Search task. They used a hybrid approach for acoustic modeling with LSTM RNNs, wherein the neural networks estimate hidden Markov model (HMM) state posteriors. They showed that deep LSTM RNN architectures achieve state-of-the-art performance for large scale acoustic

modeling. The proposed deep LSTMP RNN architecture outperforms standard LSTM networks.[8]

CHAPTER 3

RESEARCH METHODOLOGY

3.1 Data Set Preparation

Data is collected from Mr. Tej Bahadur Shahi ,he is also Co-Supervisor of this thesis work. The data is collection of Nepali news. Data was collected from various online Nepali News portals using web crawler. The news portal namely ratopati.com, setopati.com, onlinekhabar.com, and ekantipur.com were used to gather text related to different news types. There was around 20 different classes of news data but only two class are taken into consideration for this thesis work because, other class had less data so two class chosen was based on the size of the data. Business and Interview data had the largest size compare to other news data.

3.1.1 Preprocessing

Unstructured textual data usually requires some preprocessing before it can be analyzed. This process includes a number of optional text preprocessing and text cleaning steps, such as replacing special characters and punctuation marks with spaces, removing duplicate characters, removing user-defined or built-in stop-words, and word stemming. This process cleans the data and then fed into another step of preprocessing of data.

3.1.2 Data Vectorization

Vectorization is the process to convert the raw data to the data which can be feed into the computer. There are different types of vectorization methods for this research work Bag of Words model approach is used.

3.1.2.1 Bag of Words model

The bag-of-words model is a simplifying representation used in natural language processing and information retrieval (IR). Also known as the vector space model. In this model, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. The bag-of-words model has also been used for computer vision.

The bag-of-words model is commonly used in methods of document classification where the (frequency of) occurrence of each word is used as a feature for training a classifier. Length of each

document that are learned from is non-standardized. All the words in document can't be as an input feature. One approach which analyze all these is called bag of words.

The basic idea is to take the word and count the frequency of the occurrence of those words from document. The word is considered as the feature and it is unique.

For example, consider feature john, likes, watch, movies, football and sentences

Sentence 1: बैंकले फि, कमिसन, अन्य आम्दानी र विदेशी मुद्राको विनिमयबाट २३ करोड रुपैयाँ आर्जन गरेको छ

Sentence 2: बैंकले निक्षेप ३१.३४ प्रतिशतले बढेर ६४ अर्ब ४८ करोड रुपैयाँ पुगेको छ भने कर्जा २८ प्रतिशतले वृद्धि भएर

The vector representation of sentences is shown

Table 1:.Vector representation of sentences

feature	विदेशी	कमिसन	आम्दानी	आर्जन	बैंकले
Sentence 1	1	1	1	1	1
Sentence 2	0	0	0	0	1

The vector representation of sentences is then feed into the neural network to train and test.

3.2 Steps in Text classification

The various steps involving in text classification is shown with the help of block diagram

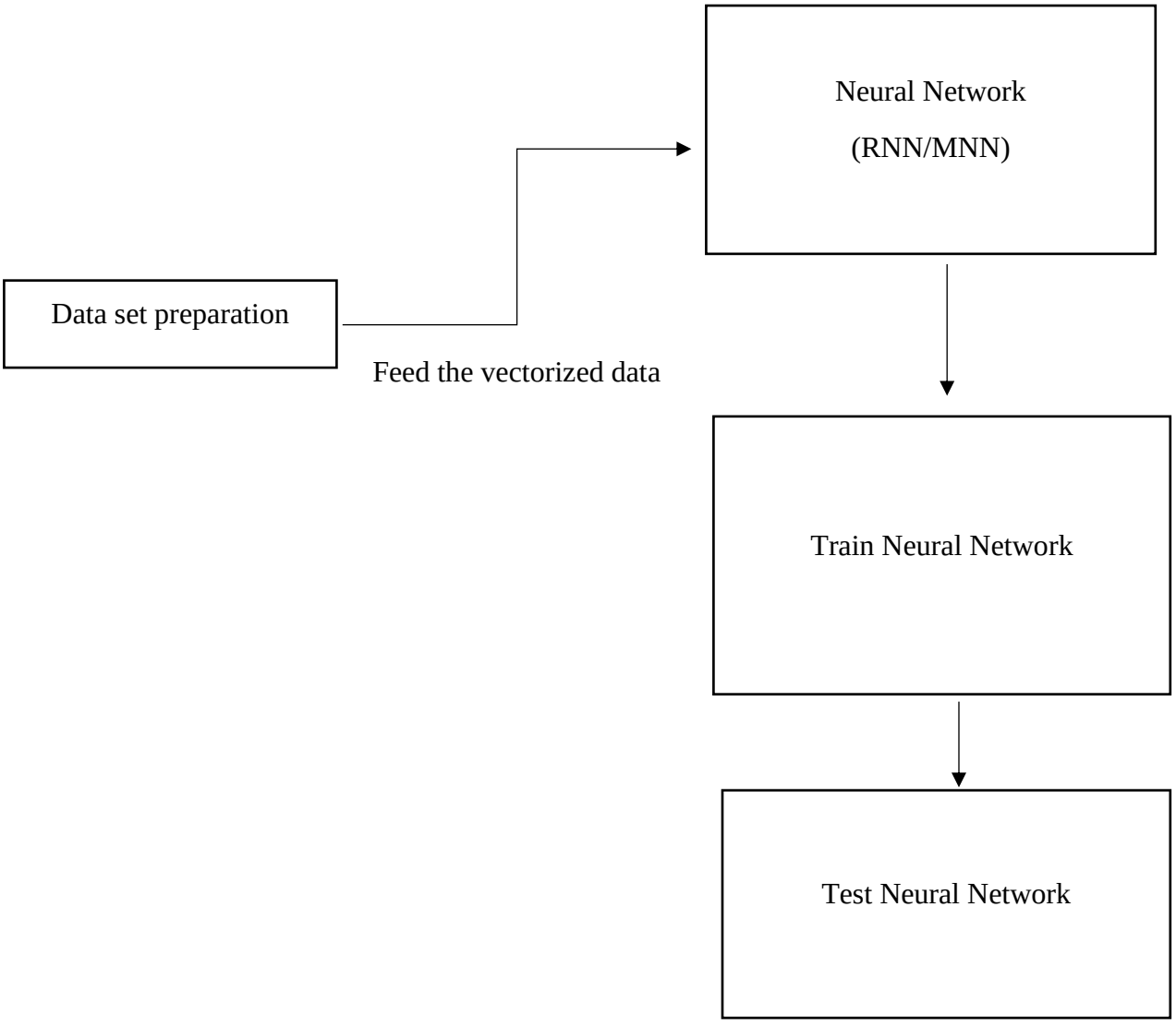

Figure 9:. Block Diagram of Steps in Text Classification

3.2.1 Block diagram of overall classification system design

It provides details of the Programing language used and the API(Application Programming Interface) to create the basic component of the Neural Network like variables, Nodes, Edges, activation functions etc. Implementing Neural Network Model and its training and testing steps.

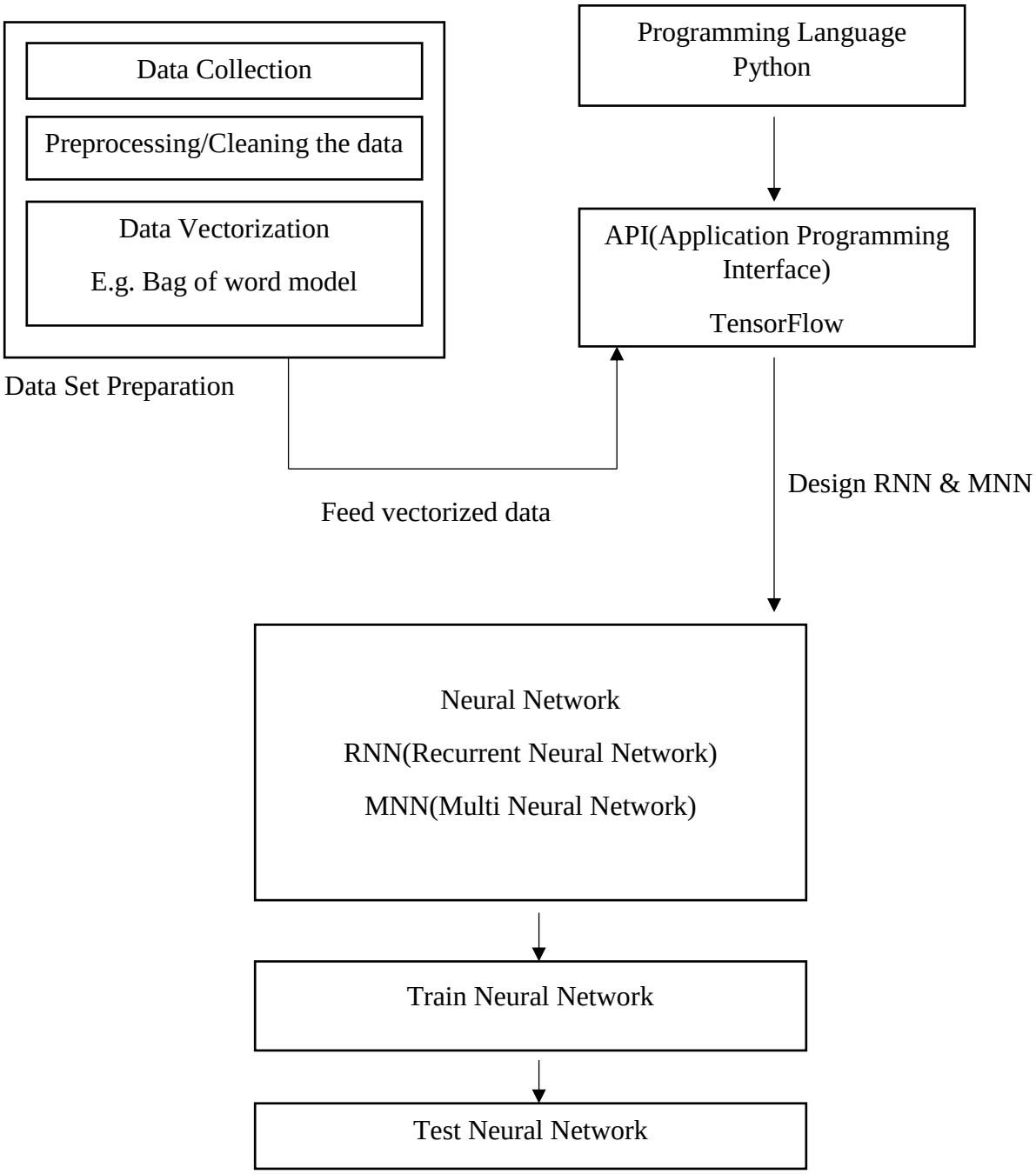

Figure 10:. Block Diagram of Over All Classification System

3.3 Performance Evaluation Parameter

It is used to evaluate effective search results that satisfied the users query. Evaluation metrics are Precision, Recall, F-measure. Evaluation parameter requires the confusion matrix. A confusion matrix is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

Consider the confusion matrix below for classification of Nepali news and English news

Table 2:.Confusion Matrix for the sample

n = total number of sample	P' predicted	N' predicted
P actual	TP	FN
N actual	FP	TN

Where

n = Total number of sample

P = the input data is actually interview.

N = the input data is not actually interview.

P' = system predicted input is interview.

N' = system predicted input is not interview.

TP = It occur when system take sample x and classify it as x

FP = It occur when system take sample of not x and classify it as x

FN = It occur when system take sample of x and classify it as not x

TN = It occur when system take sample of not x and classify it as not x

3.3.1 Recall

It is the fraction of the relevant documents that are successfully selected.

Recall = $\frac{TP}{TP+FN}$

3.3.2 Precision

It is the fraction of retrieved results that are relevant to the query is correct.

Precision = $\frac{TP}{TP+FP}$

3.3.3 Accuracy

the accuracy of a measurement system is the degree of closeness of measurements of a quantity to that quantity's true value.

Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$

3.3.4 F-Measure

It considers both the precision and the recall of the test to compute the score.

F-Measure = $\frac{(\beta^2+1)PR}{\beta^2 P+R}$

Where

P = Precision

R = Recall

β = 1

= control parameter which decides how much influences is put on Precision or Recall

CHAPTER 4

IMPLEMENTATION

4.1 Multi-Neural Network

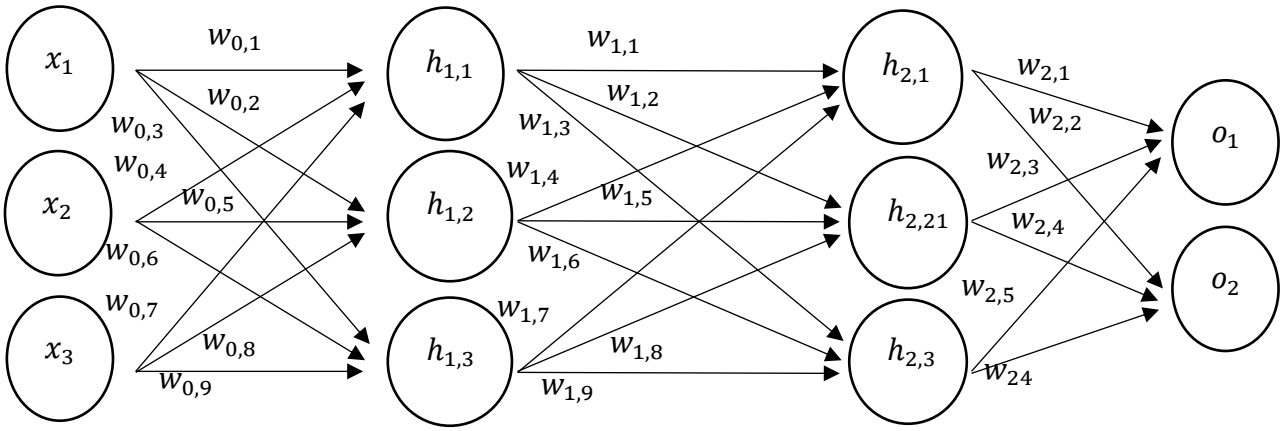

Figure 11. Simple multi-layer neural network

x_i= input at i^{th} node of input layer

$h_{k,j}$= hidden unit at $(k,j)^{th}$ neurons of hidden layer

o_i= output at i^{th} output of output layer

$w_{n,j}$= weight on the $(n,j)^{th}$ directed arrow.

i,j,k,n ∈ $\mathbb{R}$, where $\mathbb{R}$ = real number

neuron is a computational unit that takes inputs x_i, w_i and compute $h_{k,j}$= f $(\sum_{i=1}^{m} x_i w_{n,i})$. Computation on neuron of neural network model is carried according to the neural network model.

4.2 Recurrent Neural Network

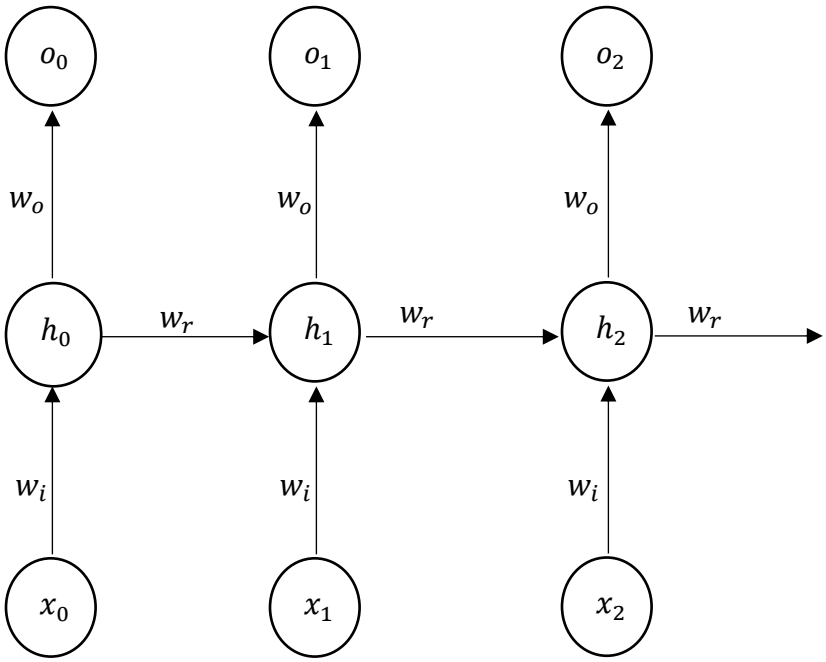

Figure 12. Simple Recurrent Neural Network (RNN)

Where,

f = activation function, t = time step.

x_i= input at i^{th} time.

h_i= hidden layer neuron at i^{th} time.

o_i= output at i^{th} time.

w_i= weight of input node.

w_o= weight of output node.

w_r= weight as the feed from previous time step.

Hidden layer neuron takes input $w_r w_i x_i$ and computes, h_i= f ($w_i x_i$+$w_r h_{i-1}$)

Output node takes input $w_i h_i$and computes, o_i= f ($w_o h_i$)

4.3 Mathematical Implementation of Classifier

It describe the brief information on how mathematically classification is carried out in Classification Model eg. RNN,MNN .

Steps involved are in the process are given below

- feed vectorized input to the network.
- Capture the output of the network
- use of tensorFlow function to compare the correctness of the result of the Neural network.

Consider the sample of input data with 6 output class

[0 2 4 ……..……1 0 1] [1 0 0 0 0 0]

[1 2 4 ……..1 0 1] [0 1 0 0 0 0]

[0 2 1 ……..1 0 1] [0 0 1 0 0 0]

[0 1 4 ……..1 0 1] [0 0 0 1 0 0]

[0 1 3 ……..2 0 1] [0 0 0 0 1 0]

[0 1 4 ……..3 0 2] [0 0 0 0 0 1]

Input vectorized data Class label

Consider the input x = [0 2 4 ……. 1 0 1] [1 0 0 0 0 0]

And label provided for the input x is cL (class label) = [1 0 0 0 0 0]

Consider output of network for input x

y = [90.71244812 20.46666718 -5.11788464 -23.22550774 89.61177826 -74.63144684]

Tensorflow(tf) function is applied to the output(y)

tf.argmax(list,1) = it return the index of the item with greater value

tf.argmax(y,1) = 0 , it is the index of output y which has greatest in list y.

tf.argmax(cL,1) = 0 , index of cL (class label) which has greatest value in list cL(class label).

TensorFlow function is applied to compared correctness of the result . tf.equal(y,cL) it return true if both are match and false if both are not matched.

Eg. Considering the result from above

tf.argmax(y,1) = 0 and tf.argmax(cL,1) = 0

now correct = tf.equal(tf.argmax(y,1), tf.argmax(cL,1))

then, correct = true

This is the result of one data inside the list of data and for the training process all the list of data goes through same process. correct contain the list of true and false from the result of comparison of all input dataset. correct = [true false true …….]

correct contain length that is equal to the input vector supplied to the network.

4.4 Programming Language and TensorFlow

The programming language used is Python. Open source library TensorFlow is integrated which is developed by the Google Brain Team. Originally created for tasks with heavy numerical computations. TensorFlow provides the API (Application programming interface) for python. It supports CPUs, GPUs, and distributed processing. TensorFlow structure is based on the execution of a data flow graph. Data flow graph has two basic units Nodes, Edges. Nodes represent mathematical operation. Edges represent multi-dimensional arrays also known as Tensors. The standard usage is to build a graph and then execute after the session is created. Inputs are fed into nodes through variables or placeholders.

The example of simple graph is shown below

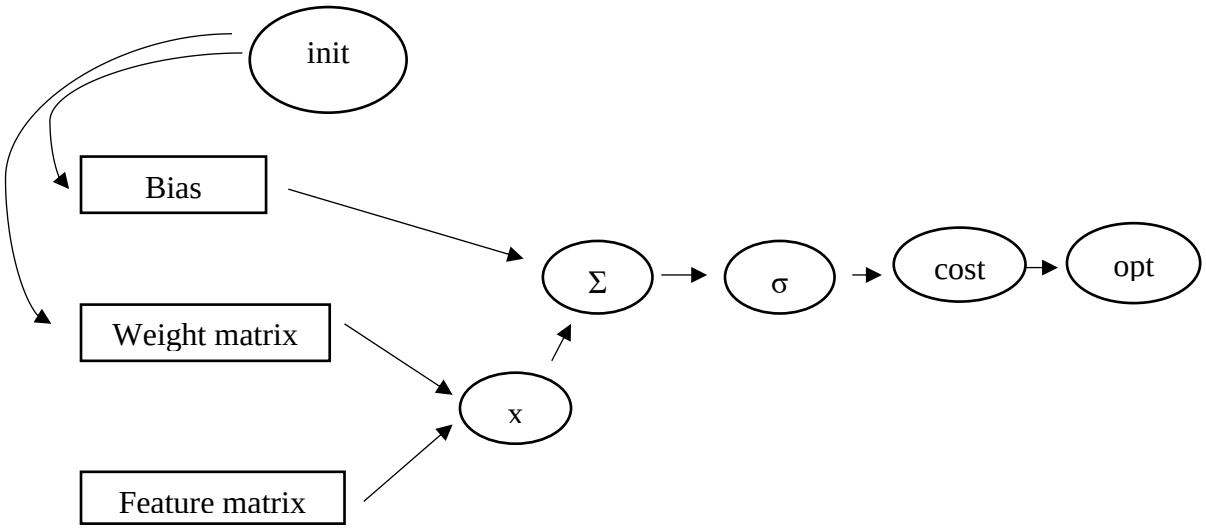

Figure 13:. Block diagram of Data flow graph of TensorFlow

CHAPTER 5

RESULT AND ANALYSIS

5.1 Experimental Setup

5.1.1 Data set

Nepali text data are taken to train the Neural Network. Classes for Nepali data set are Interview and Business. Total sample data for the Nepali data set is 32,862. Hyperparameter like learning rate, epoch , batch size , hidden layer is chosen to conduct the different experiment.

5.1.2 Train and Test set split

80% of the sample data is taken to train Neural Network. To test the classifier experiment are carried out with 20% of Nepali data.

Table 3:.No of sample on Train set

Sample Data	Total Number	Train (80% of sample)
Nepali	41,078	32,862

Table 4:.No of sample on Test set for specific class

Sample Data	Class label	Total Number	Test 2 (20% of sample)
Nepali	Business	16,938	3,388

5.1.3 Experiment 1

MNN tuned with hyperparameter and its final loss is shown below

Table 5:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.01	10	100	3	100	410.62

Taking 20% of Nepali Data set and test is carried on MNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 6:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	194 (TP)	3286 (FN)
	Not- Business	3194 (FP)	102 (TN)

Recall =0.055, Precision = 0.057, Accuracy = 0.043

F-Measure = 0.056 , where $\beta = 1$

5.1.4 Experiment 2

MNN tuned with hyperparameter and its final loss is shown below

Table 7:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.05	15	200	9	200	410.62

Taking 20% of Nepali Data set and test is carried on MNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 8:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	1301 (TP)	1833 (FN)
	Not- Business	2087 (FP)	1555 (TN)

Recall =0.415, Precision = 0.384, Accuracy = 0.421

F-Measure = 0.39 , where $\beta = 1$

5.1.5 Experiment 3

MNN tuned with hyperparameter and its final loss is shown below

Table 9:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.09	20	300	12	300	410.62

Taking 20% of Nepali Data set and test is carried on MNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 10:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	807 (TP)	2755 (FN)
	Not- Business	2581 (FP)	633 (TN)

Recall =0.22, Precision = 0.23, Accuracy = 0.21

F-Measure = 0.23 , where $\beta = 1$

5.1.5 Experiment 4

MNN tuned with hyperparameter and its final loss is shown below

Table 11:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.5	25	400	15	400	NAN

NAN is shown whenever the loss is either too great or too small. Theoretically number is either close to – infinity or + infinity

Taking 20% of Nepali Data set and test is carried on MNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 12:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	2032 (TP)	2142 (FN)
	Not- Business	1356 (FP)	1246 (TN)

Recall =0.48, Precision = 0.59, Accuracy = 0.48

F-Measure = 0.53 , where $\beta = 1$

5.1.5 Experiment 5

MNN tuned with hyperparameter and its final loss is shown below

Table 13:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.9	30	500	18	500	NAN

NAN is shown whenever the loss is either too great or too small. Theoretically number is either close to – infinity or + infinity

Taking 20% of Nepali Data set and test is carried on MNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 14:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	2102 (TP)	2184 (FN)
	Not- Business	1286 (FP)	1204 (TN)

Recall =0.49, Precision = 0.62, Accuracy = 0.48, F-Measure = 0.54 , where $\beta = 1$

5.1.3 Experiment 6

RNN tuned with hyperparameter and its final loss is shown below

Table 15:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.01	10	100	3	100	262.26

Taking 20% of Nepali Data set and test is carried on RNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 16:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	1752 (TP)	1107 (FN)
	Not- Business	1636 (FP)	2281 (TN)

Recall =0.61, Precision = 0.51, Accuracy = 0.59

F-Measure = 0.56 , where $\beta = 1$

5.1.3 Experiment 7

RNN tuned with hyperparameter and its final loss is shown below

Table 17:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.05	15	200	6	200	139.32

Taking 20% of Nepali Data set and test is carried on RNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 18:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	1850 (TP)	1067 (FN)
	Not- Business	1538 (FP)	2321 (TN)

Recall =0.63, Precision = 0.54, Accuracy = 0.61

F-Measure = 0.58 , where $\beta = 1$

5.1.3 Experiment 8

RNN tuned with hyperparameter and its final loss is shown below

Table 19:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.09	20	300	9	300	92.4

Taking 20% of Nepali Data set and test is carried on RNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 20:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	1880 (TP)	987 (FN)
	Not- Business	1508 (FP)	2401 (TN)

Recall =0.65, Precision = 0.55, Accuracy = 0.63

F-Measure = 0.6 , where $\beta = 1$

5.1.3 Experiment 9

RNN tuned with hyperparameter and its final loss is shown below

Table 21:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.5	25	400	12	400	82.01

Taking 20% of Nepali Data set and test is carried on RNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 22:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	1878 (TP)	1008 (FN)
	Not- Business	1510 (FP)	2380 (TN)

Recall =0.65, Precision = 0.55, Accuracy = 0.62

F-Measure = 0.59 , where $\beta = 1$

5.1.3 Experiment 10

RNN tuned with hyperparameter and its final loss is shown below

Table 23:. Hyperparameters and final loss

Learning rate	Epoch	Batch Size	Hidden Layer	Hidden Unit	Final loss
0.9	30	500	18	500	102.01

Taking 20% of Nepali Data set and test is carried on RNN neural network. Not- Business Class is mislabeled class. Business class is the actual labelled class.

Table 24:.Confusion matrix for Nepali Data of MNN

		Predicted Class	
	Sample = 3,388	Business	Not- Business
Actual class	Business	1784 (TP)	1185 (FN)
	Not- Business	1604 (FP)	2203 (TN)

Recall =0.59, Precision = 0.52, Accuracy = 0.58

F-Measure = 0.55 , where $\beta = 1$

5.2 Bar Chart Analysis of Result of Experiment

5.2.1 Nepali Dataset test with 20% of sample data for MNN with all 5 experiment

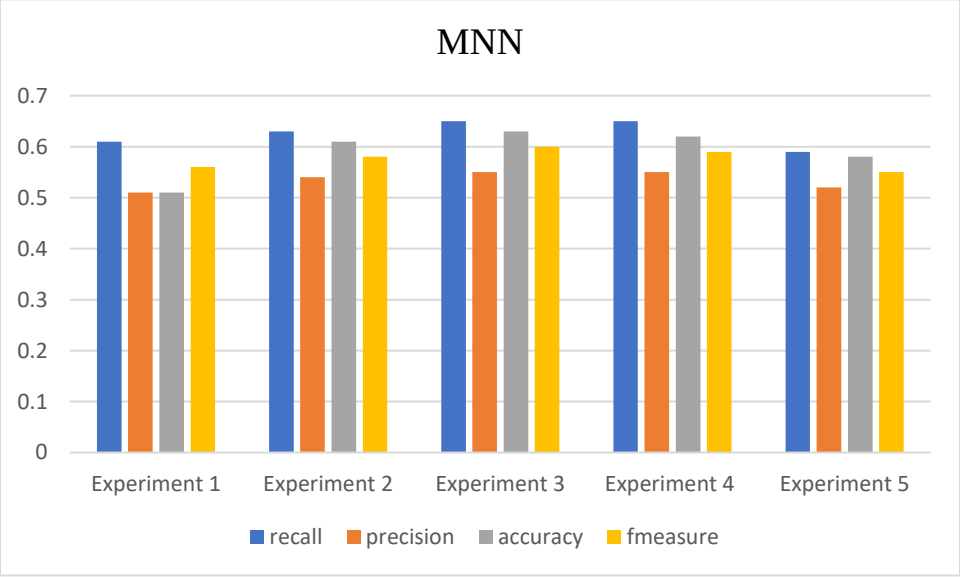

Figure 14:. Bar chart for Nepali Data of MNN

Testing with 20% sample data of Nepali data in MNN shows the highest recall was found in experiment 1 and lowest recall was on experiment 3. Lowest accuracy was in Experiment 1 and highest in Experiment 5. On average, performance evaluation parameter is good in experiment 5.

5.2.2 Nepali Dataset test with 20% of sample data for RNN

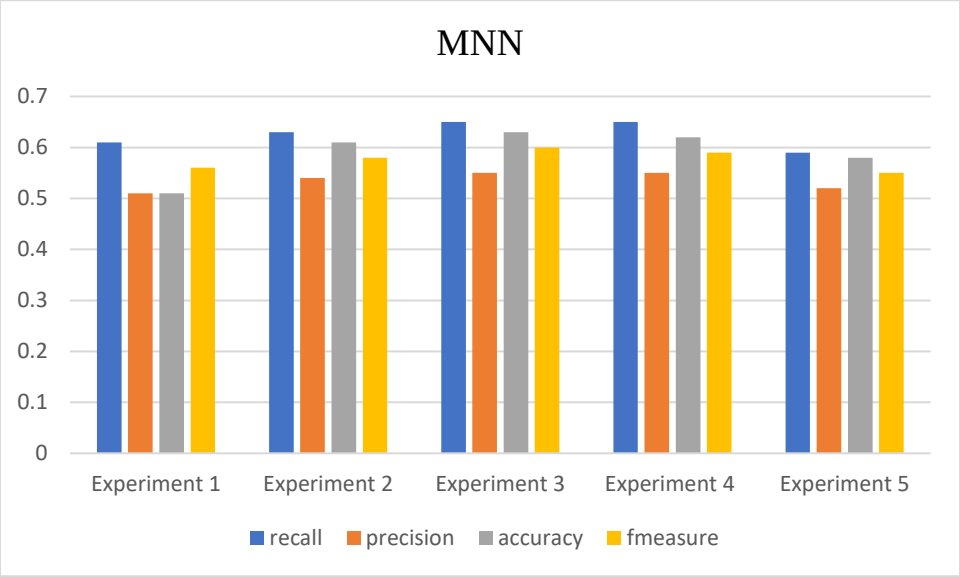

Figure 15:. Bar chart for Nepali Data of RNN

Testing with 20% sample data of Nepali data in RNN shows recall is lowest in experiment 1 and highest in experiment 3. Accuracy is lowest in experiment 1 and highest in experiment 3. On average, performance evaluation parameter is good in experiment 3.

On comparing the result of the MNN and RNN we can conclude that RNN outperform the MNN the highest accuracy achieved by MNN is 48 % and highest accuracy achieved by RNN is 63%.

CHAPTER 6

CONCLUSION AND RECOMMENDATION

5.1 Conclusion

Text Document classification was carried out with RNN and MNN neural network. The classifier performance are examined on Nepali News Text Data. The experiment results showed the accuracy of RNN outperform the accuracy of MNN. Nepali data set test carried on 20% of sample test data showed slight difference in the accuracy on both neural network model. The highest accuracy of MNN was 48% and the highest accuracy of RNN was 63%. The lowest accuracy of MNN was 5.7% and lowest accuracy of RNN was 59%.

Nepali data set test sample could not achieve the high accuracy in RNN and MNN because of the word embedding problem.

5.2 Recommendation

The data collection should be large so the neural network model can train properly. Nepali data set lack accuracy because of word embedding problem. Lack of proper stemming and lemmatization technique in Nepali data set cause neural network model to fail to achieve high accuracy. Proper library should introduce to carry the word embedding for Nepali data.

REFERENCES

[1] P. David, A. K. Mackworth, and R. Goebel, “Computational Intelligence and Knowledge 1.1 What Is Computational Intelligence?,” Comput. Intell. A Log. Approach, no. Ci, pp. 1–22, 1998.

[2] G. Brewka, Artificial intelligence—a modern approach by Stuart Russell and Peter Norvig, Prentice Hall. Series in Artificial Intelligence, Englewood Cliffs, NJ., vol. 11, no. 01. 2009.

[3] A. L. Samuel, “Some Studies in Machine Learning Using the Game of Checkers Some Studies in Machine Learning Using the Game of Checkers,” IBM J., vol. 3, no. 3, pp. 535–554, 1969.

[4] J. Schmidhuber, “Deep Learning in neural networks: An overview,” Neural Networks, vol. 61, pp. 85–117, 2015.

[5] D. Ciregan, U. Meier, and J. Schmidhuber, “Multi-column deep neural networks for image classification,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 3642–3649, 2012.

[6] D. Y. Li Deng, “Deep Learning Methods and Applications,” Found. Trends Signal Process., pp. 199–200, 2013.

[7] M. van Gerven and S. Bohte, “Artificial Neural Networks as Models of Neural Information Processing,” Front. Res. Top., pp. 1–2, 2017.

[8] H. Sak, A. Senior, and F. Beaufays, “Long Short-Term Memory Recurrent Neural Network Architecturesfor Large Scale Acoustic Modeling,” 2018.

[9] G. Fei and B. Liu, “Social Media Text Classification under Negative Covariate Shift,” no. September, pp. 2347–2356, 2015.

[10] M. Popescu, L. persescu Popescu, V. E. Balas, and N. Mastorakis, “Multilayer Perceptron and Neural Networks,” vol. 8, no. 7, p. 579, 2009.

[11] G. E. Hinton, “(n.d.) Vinod Nair, Georffrey E. Hinton, Rectified Linear Units Improve Restricted Boltzmann Machines,” no. 3.

[12] Z. Zhang and M. R. Sabuncu, “Generalized Cross Entropy Loss for Training Deep Neural Networks with Noisy Labels,” no. NeurIPS, 2018.

[13] S. Ruder, “An overview of gradient descent optimization algorithms,” pp. 1–14, 2016.

[14] D. E. Rumelhart, G. E. Hinton, and W. J. Ronald, LEARNING INTERNAL REPRESENTATIONS BY ERROR PROPAGATION, no. V. 1985.

[15] W. S. Mcculloch and W. Pitts, “A logical calculus nervous activity*,” vol. 52, no. l, pp. 99–115, 1990.

[16] P. J. Werbos, “BEYOND REGRESSION NEW TOOLS FOR PREDICTION AND ANALYSIS IN THE BEHAVIORAL SCIENCES,” 1974.

[17] R. Dechter, “LEARNING WHILE SEARCHING IN CONSTRAINT-SATISFACTION-PROBLEMS*,” pp. 178–183, 1986.

[18] T. Joachims, “Introduction 2 Text Categorization 3 Support Vector Machines.”

[19] Y. Meng, J. Shen, C. Zhang, and J. Han, “Weakly-Supervised Neural Text Classification.”

[20] N. Kant, R. Puri, S. C. Ca, N. Yakovenko, B. Catanzaro, and S. C. Ca, “Practical Text Classification With Large Pre-Trained Language Models.”