TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING

Khwopa College Of Engineering
Libali, Bhaktapur
Department of Computer Engineering

A PROPOSAL ON

NEPALI SENTIMENT ANALYSIS USING NEURAL NETWORKS

Submitted in partial fulfillment of the requirements for the degree

BACHELOR OF COMPUTER ENGINEERING

Submitted by

Bikesh Sitikhu	KCE/074/BCT/016
Luja Shakya	KCE/074/BCT/022
Niranjan Bekoju	KCE/074/BCT/025
Sunil Banmala	KCE/074/BCT/045

Under the Supervision of

Er.Dinesh Gothe
Department Of Computer Engineering

Khwopa College Of Engineering
Libali, Bhaktapur
2020-21

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING

Khwopa College Of Engineering
Libali, Bhaktapur
Department of Computer Engineering

A PROPOSAL ON

NEPALI SENTIMENT ANALYSIS USING NEURAL NETWORKS

Submitted in partial fulfillment of the requirements for the degree

BACHELOR OF COMPUTER ENGINEERING

Submitted by

Bikesh Sitikhu	KCE/074/BCT/016
Luja Shakya	KCE/074/BCT/022
Niranjan Bekoju	KCE/074/BCT/025
Sunil Banmala	KCE/074/BCT/045

Under the Supervision of

Er.Dinesh Gothe
Department Of Computer Engineering

Khwopa College Of Engineering
Libali, Bhaktapur
2020-21

Certificate of Approval

The undersigned certify that the final year project entitled **"Nepali Sentiment Classification using Neural Network"** submitted by Dipesh Dulal, Dipesh Shrestha, Gaurab Chhetri and Gaurab Subedi to the Department of Computer Engineering in partial fulfillment of requirement for the degree of Bachelor of Engineering in Computer Engineering. The project was carried out under special supervision and within the time frame prescribed by the syllabus.
We found the students to be hardworking, skilled, bona fide and ready to undertake any commercial and industrial work related to their field of study and hence we recommend the award of Bachelor of Computer Engineering degree.

.....
Er. Dinesh Gothe
(Project Supervisor)

.....
Prof. Dr. Sashidhar Ram Joshi
(External Examiner)
Central Campus Pulchowk, Institute of Engineering
Lalitpur, Nepal

.....
Er. Shiva K. Shrestha
Head of Department
Department of Computer Engineering, KhCE

Copyright

Acknowledgement

We take this opportunity to express our deepest and sincere gratitude to our supervisor Er. Yuba Raj Siwakoti, for his insightful advice, motivating suggestions, invaluable guidance, help and support in successful completion of this project and also for his constant encouragement and advice throughout our Bachelor's program. We are deeply indebted to our teachers Er. Narayan K.C and Er. Anku Jaiswal for boosting our efforts and morale by their valuable advices and suggestion regarding the project and supporting us in tackling various difficulties. Also, we would like to thank Er. Ram Sapkota for providing valuable suggestions and supporting the project. In Addition, we also want to express our gratitude towards Er. Dinesh Man Gothe for providing the most important advice and giving realization of the practical scenario of the project.

Bikesh Sitikhu	KCE/074/BCT/016
Luja Shakya	KCE/074/BCT/022
Niranjan Bekoju	KCE/074/BCT/025
Sunil Banmala	KCE/074/BCT/045

Abstract

There is abundance of Nepali Unicode text in internet due to social media like; Twitter and Nepali news websites. People express their feelings through text by writing reviews about products, movies, news etc. online. Markets like movie industry, tourism industry and several service providers rely on people's opinions about the products and services to get insight about the market. Manually, going through such gigantic reviews and comments is a tedious task. This document shows various approaches taken to build Nepali language sentiment classification system using recurrent neural networks which classifies the sentiment polarity of a given sentence into "positive" and "negative". The system takes input Nepali sentence and outputs the probability of sentiment classes with the best model achieving the accuracy of 82%.

Keywords: *Natural Language Processing, Nepali Language, Neural Network, Machine Learning, Sentiment Classification*

Contents

List of Tables

List of Figures

List of Symbols and Abbreviation

NLP	Natural Language Processing
RNN	Recurrent Neural Network
ARNN	Attention Based Recurrent Neural Network
JS	JavaScript
RegEx	Regular Expressions
JSON	JavaScript Object Notation
KB	Kilo Byte
OOV	Out of Vocab
NELRALEC	Nepali Language Resources and Localization for Education and Communication.
NNC	Nepali National Corpus
POS	Parts of Speech
LSTM	Long Short-Term Memory

Chapter 1

Introduction

1.1 Background Introduction

As motivated by the rapid growth of text data, text mining has been applied to discover hidden knowledge from text in many applications and domains. In business sectors, great efforts have been made to find out customers' sentiments and opinions, often expressed in free text, towards companies' products and services. However, discovering sentiments and opinions through manual analysis of a large volume of textual data is extremely difficult. Hence, in recent years, there have been much interests in the natural language processing community to develop novel text mining techniques with the capability of accurately extracting customers' opinions from large volumes of unstructured text data. Among various opinion mining tasks, one of them is sentiment classification, Sentiment analysis is a type of data mining that measures the inclination of people's opinions through natural language processing (NLP), computational linguistics and text analysis, which are used to extract and analyze subjective information from the web – mostly social media and similar sources [1] . The analyzed data quantifies the general public's sentiments or reactions toward certain products, people or ideas and reveal the contextual polarity of the information. Sentiment analysis is also known as opinion mining. Sentiment analysis carries the basic task of classification of the expressed opinion in a document into "positive", "neutral", "negative".

1.2 Problem Statement

As motivated by the rapid growth of text data, text mining has been applied to discover hidden knowledge from text in many applications and domains. In business sectors, great efforts have been made to find out customers' sentiments and opinions, often expressed in free text, towards companies' products and services. However, discovering sentiments and opinions through manual analysis of a large volume of textual data is extremely difficult. Hence, in recent years, there have been much interests in the natural language processing community to develop novel text mining techniques with the capability of accurately extracting customers' opinions from large volumes of unstructured text data.

Among various opinion mining tasks, one of them is sentiment classification, Sentiment analysis is a type of data mining that measures the inclination of peo-

ple's opinions through natural language processing (NLP), computational linguistics and text analysis, which are used to extract and analyze subjective information from the web - mostly social media and similar sources. The analyzed data quantifies the general public's sentiments or reactions toward certain products, people or ideas and reveal the contextual polarity of the information. Sentiment analysis is also known as opinion mining. Sentiment analysis carries the basic task of classification of the expressed opinion in a document into "positive", "neutral", "negative".

In recent years, many Nepali online news portals like onlinekhabar.com, ratopati.com, setopati.com have been providing news online in Nepali Unicode. Similarly, people use Nepali Unicode to comment, review such online news. Not much work about Nepali Natural Language Processing has been found during our research. Since the Nepali language is morphologically rich and complex the text classifier needs to consider specific language features before classifying the text. Preprocessing of data in Nepali Unicode is at a delicate stage. Very low resources about Nepali Unicode are available. The applications for sentiment analysis are endless. More and more we're seeing it being used in social media monitoring and VOC to track customer reviews, survey responses, competitors, etc. However, it is also practical for use in business analytics and situations in which text needs to be analyzed. Sentiment analysis is mostly used to create recommendation systems which are user specific. The opinion of a user about a specific product is the key factor for determining the likes and dislikes of the user which results in increase of efficient of traditional recommender systems. Today's consumers buy products recommended by Amazon, watch movies recommended by Netflix, and listen to music recommended by Pandora. A research showed that the recommendation system developed using sentiment analysis of user reviews which was used by Amazon, increased their profit boost by 29 percent. Similar results can be achieved in the Nepali market by using a sentiment analyzer for Nepali language.

1.3 Obejctive

The main aim of this project is:

- To test various models for sentiment classification.
- To design and implement a system which can classify sentiment of Devanagari sentences.

Chapter 2

Literature Review

Previously a research paper [2]for sentiment analysis by Peter D. Turney started the classification of sentiment of reviews as recommended (thumbs up) or not recommended (thumbs down). This algorithm achieved an average accuracy of 74Chandan Prasad Gupta and Bal Krishna Bal also proposed [3] the system of Detecting Sentiment in Nepali Texts by using self-developed Nepali Sentiment Corpus and Nepali SentiWordNet. Their system does not use Neural Networks to classify the Nepali texts. The major breakthrough in this field happened when researchers at Stanford University proposed the recurrent neural network system for noise reduction in ASR system [**?**] . This gave a new approach to tackle sequential data in the field of natural language processing and machine learning in general. The recurrent neural network has outperformed different other models in this task of analyzing and processing sequential data such as a sequence of text. The paper [**?**] by Mikolov et. al. discusses the efficient estimation of word embedding using skip gram approach which can be considered as one of the groundbreaking works increasing the efficiency of the classification system. Similarly, other various authors have used various natural language processing techniques to classify sentiment of texts in different languages. Like this paper [**?**] the author discusses the approach taken to analyze Turkish political news. Likewise, in this paper [**?**] author has used a lexicon- based approach for classifying Indian text which is lexically similar to Nepali. Similarly, a latest research paper [**?**] by researchers at Google Brain studies about the attention model for sequence learning tasks where model learns to put attention to certain words than other for output similar to how the brain works.

Chapter 3

Requirement Analysis

3.1 SOFTWARE REQUIREMENT

Our sentiment analysis system requires Python, PHP, MySQL, TensorFlow, Flask, Django, JavaScript, Keras which are described below;

3.1.1 Python

Python is a high-level interpreted programming language for general-purpose programming created by Guido van Rossum which as released in 1991. Python has design philosophy that emphasizes code readability, notably using significant white spaces. It provides constructs that enable clear programming in small and large scales. The programming language features dynamic type system and automatic memory management with support to multiple programming paradigms including Object Oriented Imperative, Functional and Procedural. The programming language has large comprehensive standard library.

3.1.2 PHP

Hypertext Preprocessor (or simply PHP) is a server-side scripting language designed for web development but also used as a general-purpose programming language. The language is used for creating sentiment labeling system as shown in Appendix of this report. Annotator clicks the “positive” or “negative” label which is then saved in JSON format read by data preprocessing system.

3.1.3 MySQL

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of ”My”, the name of co-founder Michael Widenius’s daughter, and ”SQL”, the abbreviation for Structured Query Language. In our project, we use this database to feed into PHP system for data labeling. The database is created from scraping Nepali Sentiment text in python.

3.1.4 Keras

Keras is an open source neural network library written in Python which is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, or Theano. Actual model preparation of different models is done using Keras machine learning framework. We 5use Keras functional API to generate models as well as train the model on different annotated data.

3.1.5 Flask

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. We use flask to create sentiment inference engine that will be created by training the sentiment classification model.

3.1.6 JavaScript

JavaScript often abbreviated as JS, is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm. We use JS to both accompany in sentiment labeling system as well as sentiment inference system which is shown in appendix part of this report.

3.2 FUNCTIONAL REQUIREMENT

3.2.1 Dataset Labeler

Dataset labeler is the labelling system that is used to annotate the texts as "positive", or "negative".

3.2.2 Nepali News Web Scrapper

It is the tool that extracts various Nepali text from twitter as well as various Nepali news portals to fed into Dataset Labeler as well as vector creation process.

3.2.3 Inference System

It is the system formed after training the model using the dataset labeled from dataset labeler and run the prediction model in it.

3.3 NON-FUNCTIONAL REQUIREMENT

These requirements are not needed by the system but are essential for the better performance of sentiment engine. The points below focus on the non-functional requirement of the system.

3.3.1 Reliability

The system is reliable. Sentiment prediction matches 80

3.3.2 Maintainability

A maintainable system is created and Sentiment Analyzer Engine is able to train on new input data and is scalable to millions of data points.

3.3.3 Performance

The forward pass from the neural network is a fast process. For the engine, fast matrix computation occurs.

3.3.4 Portability

Sentiment Analyzer engine is portable and it is easy to integrate into any web application or mobile application imaginable by the use of the REST API's made.

3.4 FEASIBILITY STUDY

The following points describes the feasibility of the project.

3.4.1 Economic Feasibility

The total expenditure of the project is just computational power. The dataset and computational power required for the project are easily available. Dataset is found from the local news site and computational power using our own laptop alongside Google Collaboratory cloud computing service so, the project is economically feasible.

3.4.2 Technical Feasibility

The project is trained with lots of labeled sentiment news data and twitter tweets. Preparing the data has its complexity like scraping from news websites and manually labeling the dataset for sentiment classification. Creating the project will be challenging but is feasible. In terms of availability, the dataset is easily web-scrapped from popular news portals, social networking sites like; twitter and labeled creating a PHP web application. Training huge news dataset takes a lot of computation power. Which is trained in INTEL i5 processor for some hours for about 10K data points. Training the project is also feasible and is better with huge data points and large processing power.

3.4.3 Operational Feasibility

The project can be operational just after training from labeled data using neural network models. After the engine for sentiment analysis for Nepali text is created, the engine can be operated to implement in different recommender systems and text analyzers. Thus, the project is operationally feasible.

Chapter 4

System (or Project) Design and Architecture

We developed a Nepali Language Sentiment Classifier which takes Nepali Unicode sentences as its input and process it to provide sentiment values as its output.

4.1 USE CASE DIAGRAM

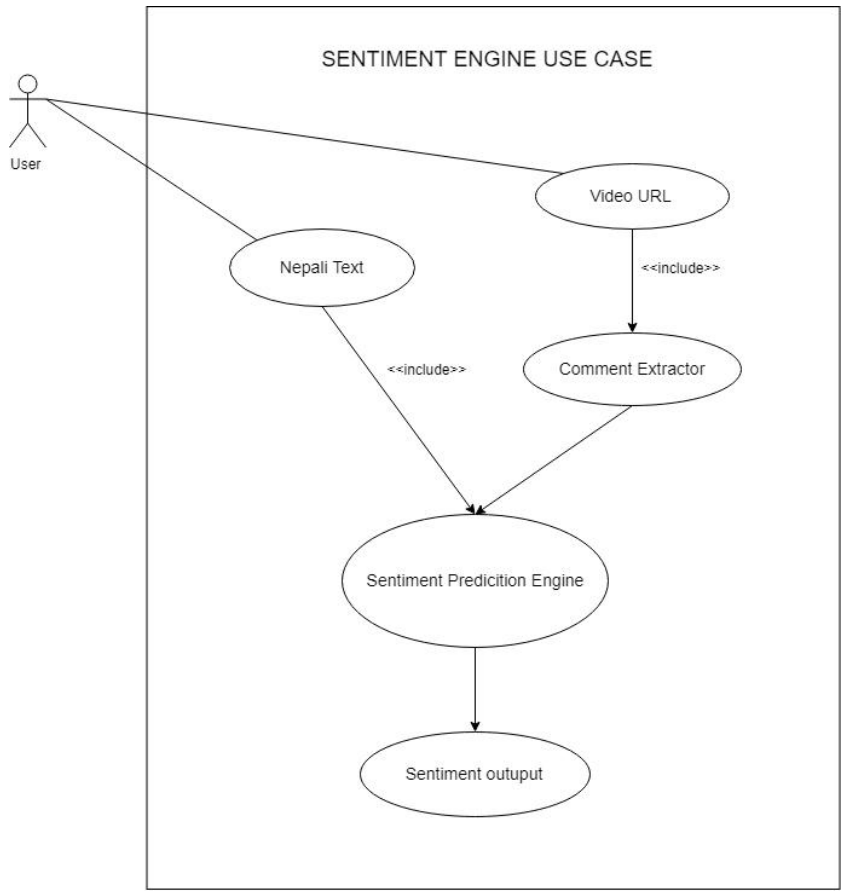

Figure 4.1: System Use Case Diagram

4.2 System Block)

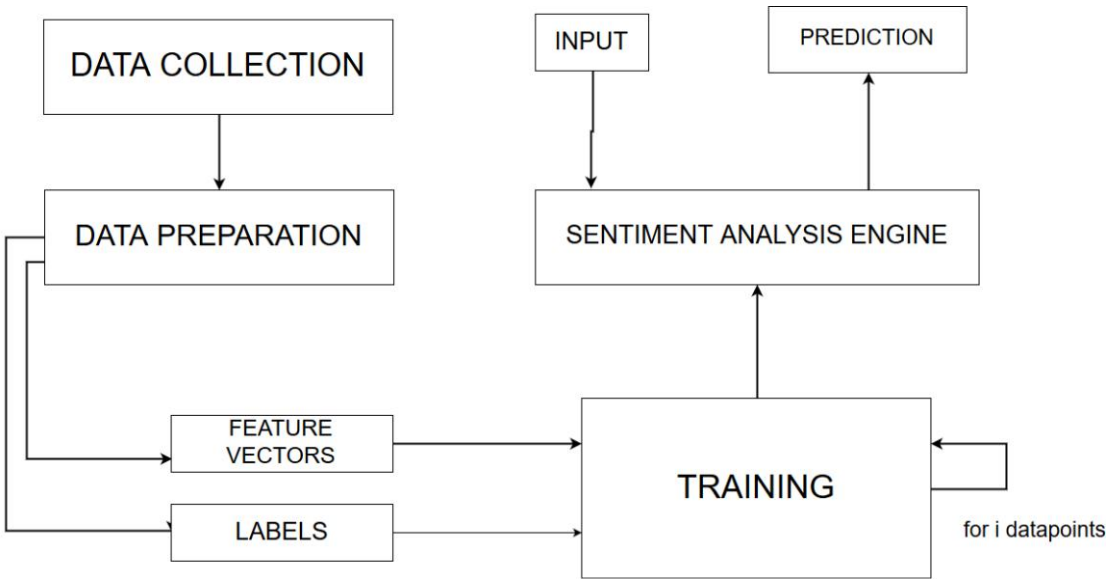

Figure 4.2: System Block Diagram

4.3 Sequence Diagram

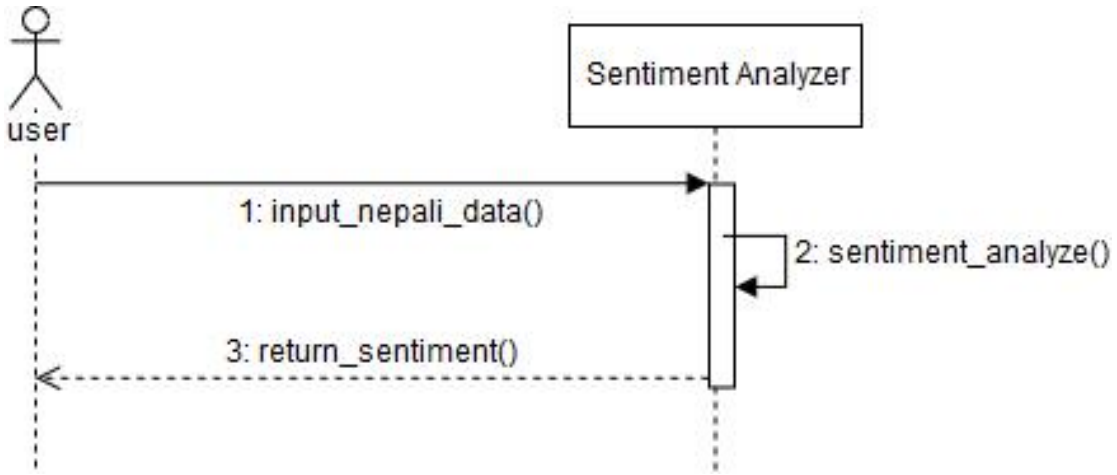

Figure 4.3: Sequence Diagram

Chapter 5

Methodology

5.1 SOFTWARE DEVELOPMENT APPROACH

Prototype model is a software development model where instead of freezing the requirements before design or coding can proceed, a throwaway prototype is built to understand the requirements. The prototype are usually not complete systems and many of the details are not built in the prototype. The goal is to provide a system with overall functionality. In this model, we create the prototype of the actual system, update the requirements and again rebuild the system until the final requirements are met.

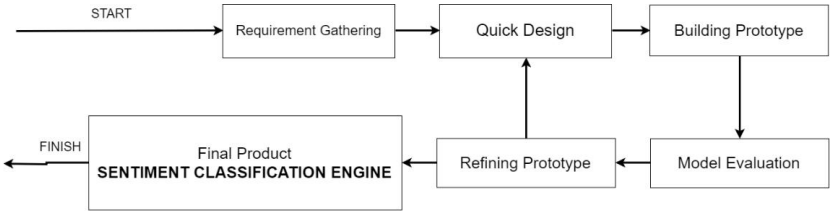

Figure 5.1: Prototype Model for Software Development

5.2 DATA COLLECTION

We have collected the data from popular news websites such as; ratopati.com, ekantipur.com using web-scraping technology also from twitter tweet feed of popular Nepali tweet pages. The data is collected programmatically from the website as different websites have different designs and architectures. The data is mainly collected from the opinions section of the site as the section contains more opinions to mine to give more insightful data and making labeling easy. The API for social networking websites were also used. For POS tagging, the data from NELRALEC [10] The following table shows the amount of data collected from different sources and their usage in our project; 10Table 6.1 Data Sources with numbers

Table 5.1: Data Sources with numbers

Source	Number of Data	Usage
Nepali News Corpus	14364 Articles	Vectorization
Facebook	5021 Comments	Labeling
Twitter	2160 Tweets	Labeling
Annapurnapost.com	400 Articles Vectorization,	Labeling
Nagarik News	4481Articles	Vectorization
National News	7452 Articles	Vectorization
Bsgnews.com	1000 Articles	Vectorization, Labeling
Setopati News	1090 Articles	Vectorization
NELRALEC Corpora	798961 Sentences	POS Tagging

5.2.1 Twitter Tweet Collection

We applied for Twitter Developer Account to get API keys to gather data from twitter. Tweepy [9] which is a python tweet API library was used to make API requests and model the response in appropriate format. We used some of the queries for the API like; 'brb1954', 'hello$_s$*arkar*', '*thapagk*', '*nepalitweet*'*etc.Fromthesequeriesthetweets*

Figure 5.2: Tweet in JSON Format

In this way JSON formatted tweets is passed into Data Labeling System (Section 6.6) and stored in MySQL for easy access from database rather than flat file.

5.3 DATA PREPARATION

After the data has been collected, it was processed to make into correct format so that the neural networks can understand both the inputs and outputs. The block diagram (figure 6.2) shows the data preparation steps of the project.

5.3.1 Tokenization

The unlabeled and unprepared data is fed into the tokenization engine. There are two types of tokenization; one sentence level tokenization and another word level tokenization. Following example shows both the sentence level tokenization as well as world level tokenization.

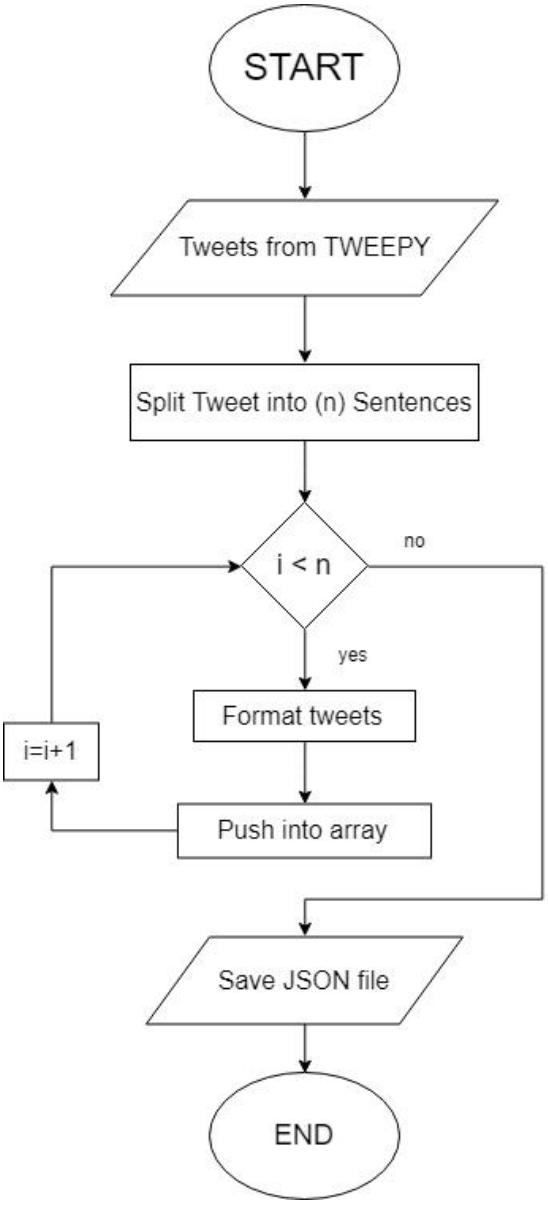

Figure 5.3: Tweet tokenization and storing flow chart

5.3.2 Stop Words Removal

The tokenized Nepali words will now be preprocessed to remove stop words such as; , etc. which has no meaning or participation in sentiment analysis. This stop word removed tokenized data will now be fed into stemming algorithm.

5.3.3 Stemming

We will use snowball [11] rule-based stemming algorithm for removing some of the stems of the Nepali language such as; , , etc.

5.3.4 Word2Vec

As neural networks require numbers as the inputs. The tokenized words need to be converted into feature vectors i.e. list of numbers. Not any random list

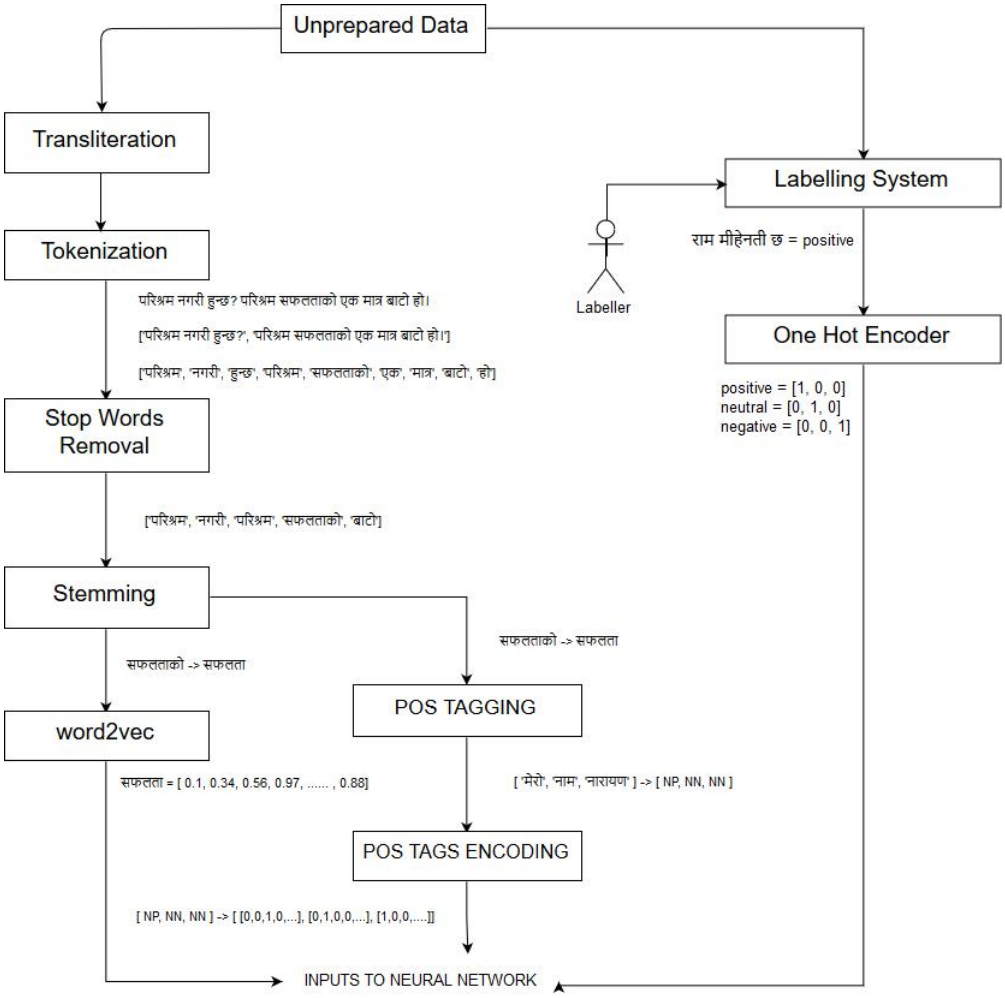

Figure 5.4: Data Preparation Block Diagram

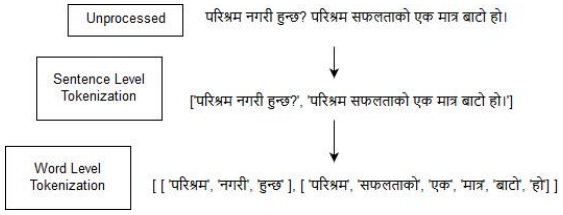

Figure 5.5: Tokenization

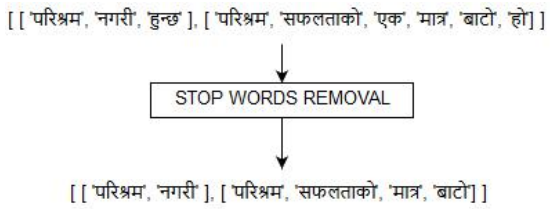

Figure 5.6: Stop Words Removal Process

of number but the related numbers. There must be a correlation between two tokenized words i.e. two vectors must be correlated with each other. Example: In English language, king and boy are more related than queen and girl so king

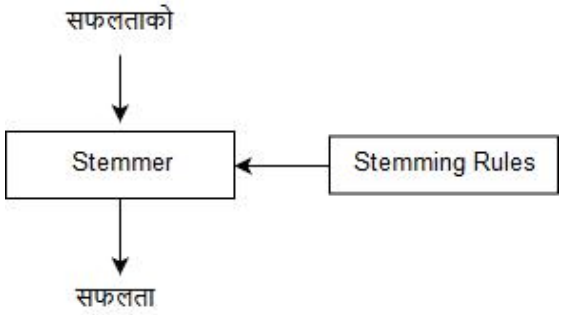

Figure 5.7: Stemming Process

and boy's vector representation should be near to each other than other words. Word2Vec [12] converts those sentences into context linked vector representation as the name suggests.

5.3.5 Labeling System

From the unprocessed data, we need to manually label the data points. For manually labeling the data points we created a system from PHP programming language; reading the data points from MySQL database and then let the user choose between positive or negative sentiment and record it in database all hosted in internet for easy access. The data labeling system is described in detail in section 6.6 of this report.

5.3.6 POS Tagging

POS tagging is a piece of software that assigns parts of speech to each word and some other token. From the word vectors and their respective POS (Parts of Speech) tags need to be placed by the system. POS tagging gives more insight to the data making neural network little bit easier to classify sentiment data. Some of the list of POS tags can be seen in the table below;

Table 5.2: Some of the POS Tags from NELRALEC Tagset

POS Tags	**Description**	**Example**
NN	Common Noun	,,
NP	Proper Noun	,
JM	Masculine Adjective	,
JF	Feminine Adjective	,
VI	Infinite Verb	,
...	...	...

The following figure below shows the POS Tagging system with example in Nepali Language:

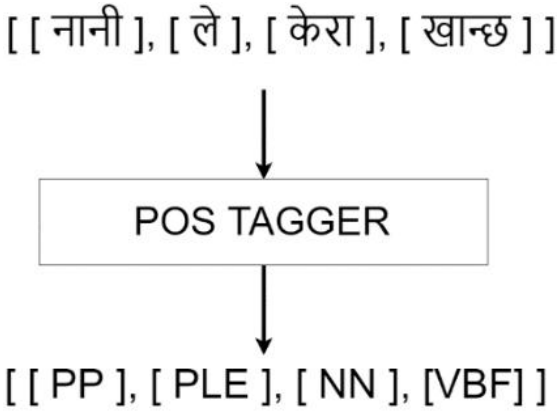

Figure 5.8: POS Tagging System example taken from ANN Based POS Tagging for Nepali Text

5.3.7 One Hot Encoder

One Hot encoding or Binarization is the process of converting the labeled data into binary labels. The encoding scheme of our project is shown in the table below; Table 6.3 One Hot Encoding4

Table 5.3: One Hot Encoding

Labels	Binarization
Positive	[1,0]
Negative	[0,1]

5.4 TRAINING

After data has been prepared the data is fed into various models described in Section 6.5.3 to 6.5.6 in this report. Various models' evaluation was carried out after training the model. The data will be fed into the training system or neural network as shown in the figure below;

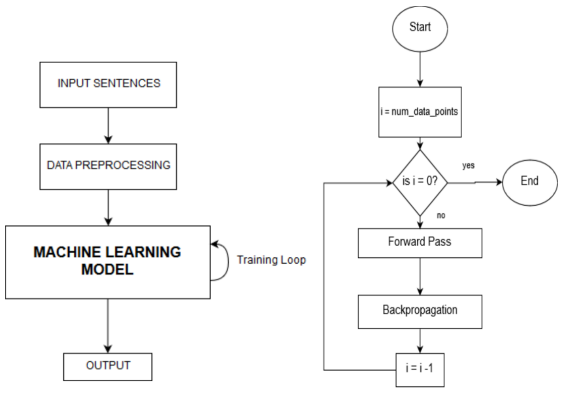

Figure 5.9: Basic Machine Learning Training Process

5.5 ALGORITHMS AND MODELS

As above mentioned, various algorithms and models were developed for building the sentiment classification engine. Following topics describes all the algorithms and models used in the project in detail;

5.5.1 Tokenization Algorithm

There is the use of Regular Expressions also known as RegEx for splitting the documents into tokens. There are two types of tokenization into play namely; sentence level tokenization and word level tokenization. Since, Nepali language sentences are separated with special characters like purnabiram () or question marks (?) it is easy to split them into sentences with the use of RegEx. The algorithm for sentence level tokenization is given below; Sentence Level Tokenization Algorithm

1. Start
2. Input Nepali Sentences
3. RegEx Split Sentences /(?¡=[?!]) +/
4. Output split chunks of Nepali Sentences
5. End

In languages that have words separated by blank spaces, the token boundary for word- level tokenization is the blank/white space. Nepali is one such language so word-level tokenization in Nepali can be achieved by stripping tokens at those white spaces. However, it is not as simple as it looks, especially when dealing with punctuations. Some punctuation is easy to handle. We first need to replace punctuations with white spaces. Similarly hyphen (-) which is used in linking word pairs: opposite, analogy or similar, together. In this case hyphen is considered as the part of token itself. However, hyphen might occur independently in such cases we need to attach the words to make it a single token. Similarly, colon (:) and period (.) can be considered as the part of token itself. The algorithm for word level tokenization can be seen below: *Word Level Tokenization Algorithm*

1. Start
2. Input Sentence
3. Replace punctuations with white spaces
4. Make all words with (:, ., -) symbols a single token
5. Split the formed sentences according to white spaces
6. Output the tokenized sentences
7. End

5.5.2 Stemming Algorithm

The stemming algorithm was used in our project to reduce the redundancy imposed by various words coming from the same stems meaning similar definition for this already implemented algorithm in snowball was used. Snowball is a small string processing language which was designed for creating stemming algorithms that is used in information retrieval tasks. The language was created for creating readily available stemming algorithms and easy information retrieval.

Snowball stemming algorithm was used in our project which is the implementation of Shrestha, I., Dhakal, S.S. (2016) [14] which is suffix stripping in nature. The algorithm was created from 128 suffix rules which are executed step-by-step and in iterative manner to eliminate inflections in Nepali Language. The stemmer was tested in 5000 Nepali words to give overall accuracy around 88.7818shows simplified flowchart about how the snowball stemming algorithm works for three different types of suffixes in the word. *Suffix Category I:*

1. Start
2. Input Words containing category I
3. Strip the suffix
4. Scan for Category II and III
5. If suffix found then go to respective stemming processes else store
6. Stop

Suffix Category II:

1. Start
2. Input words containing Category II
3. Check stripping criteria
4. If satisfied then scan for category II and III and go to respective processes else store the word
5. Stop

Suffix Category III:

1. Start
2. Input words containing category III
3. Strip the suffix
4. Scan for Category II and III
5. If found then go to respective stemming processes else store
6. Stop

5.5.3 Word2Vec Vectorization

Word2Vec are a group of neural network models used to produce word embeddings. The models are shallow two layered neural network as shown in the figure below that are trained to reconstruct the linguistic context of words. The model takes input large corpus of text and produces a vector space, typically a several hundred dimension but in our case 100-dimension vector, with each unique word in the corpus being assigned a corresponding vector in space. Word vectors are positioned in such a way that the words share common contexts in corpus are located in close distance with each other 19in space. We can visualize the vectors in by making cluster of words and know how the word2vec [12] model has trained.

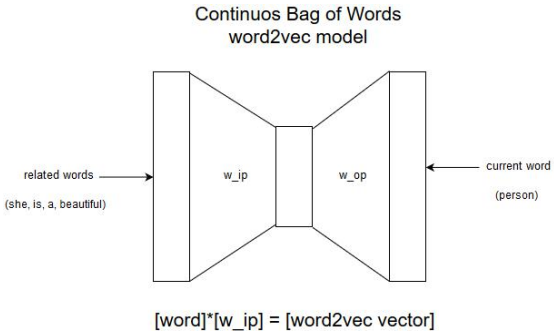

Figure 5.10: Word2Vec Continuous Bag of Words Model

For training Nepali Word2Vec model we first use tokenization and stemming procedures to remove redundancy in the corpus (table 6.1) as much as possible and the fed it into the model for training. Our model was trained in 28787 articles for 5 epochs in 12 core Intel i7 8 th generation processor using multiprocessing which took about 12 hours to train on. Training word2vec model is a one-time process and is used to vectorize for training models described in sections mentioned below. The model learned the representations of 95,818 Nepali words and the graph below shows the distribution of number of words out of vocab.

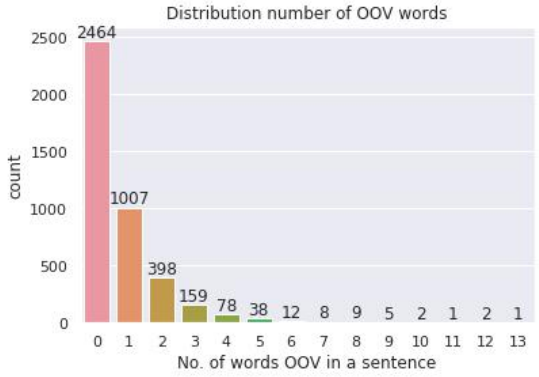

Figure 5.11: Distribution of Number of Words OOV per sentence

5.5.4 Perceptron

Perceptron is an algorithm for learning binary classifier which is also a mathematical model of a biological neuron. While in actual neurons the dendrite receives

electrical signals from the axons of other neurons, in perceptron these electrical signals are represented as numerical values. At the synapses, the electrical signals are modulated in various amounts which is also modeled in the perceptron using weights. Actual neuron fires only when total input crosses a certain threshold which is modeled using a threshold function in our case activation function. The model maps multiple values of input into one output either belonging to some class or not. Perception is the basic building block of a neural network. The mathematical formula for the perceptron looks like the following;

$$Y_i = F($$

Where, Y is output and I is the input. The equation can be seen figuratively as shown below;

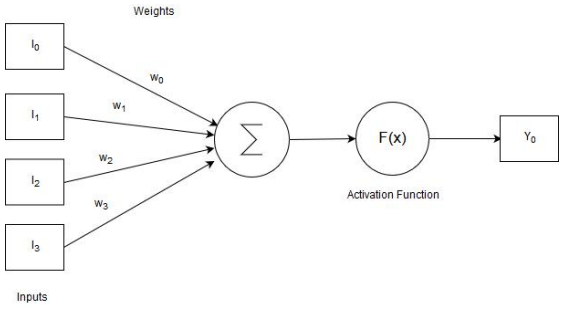

Figure 5.12: Single Perceptron in a Neural

5.5.5 LSTM RNN Model

RNN or Recurrent Neural Network are special types of neural network that is specially developed to learn sequential data. Since, we are trying to classify the sentiment from sequence of tokens this neural network model should perform better than previous dense feed forward model. Unlike, the feedforward model these neural networks can 21use their internal state to process the sequences of inputs because of which they have been used in Language Modeling, Machine Translation, Speech Recognition and various other tasks. But because of major two problems in RNN model called vanishing and exploding gradients problem we use LSTM RNN model for training the classifier as LSTM model uses forget gates to decide when to forget and remember information for long periods of time.

5.5.5.1 LSTM-RNN

LSTM are a special kind of RNN, capable of learning long term dependencies. They were introduced by Hochreiter and Schmidhuber in 1997 [13] , and were refined and popularized by many people. They work tremendously well on large variety of problems, and are now widely used. All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

The LSTM's also have a chain like structure but repeating modules have different structures as shown in the figure below:

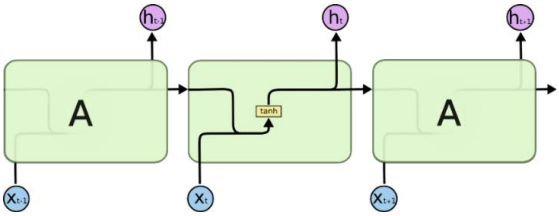

Figure 5.13: Standard RNN Network

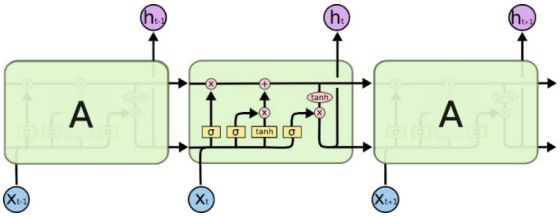

Figure 5.14: RNN-LSTM Network Architecture

The above shown LSTM cell are good at remembering the data is because they can choose to either remember or forger the data based on the weights of the network. This feature of this networks helps to tackle the major problem of general RNN's.

5.5.5.2 Classification Model

For classification, we use two layers of LSTM RNN stacked on top of each other to learn the task. We use the bidirectional LSTM structure for training the sentiment classification model which is a wrapper around LSTM layers that enables bidirectional data flow in the sequences during training process. This model was trained for 3 minutes in Google Colab GPU Kernel for 3 epochs with the testing accuracy of 83accuracy of 80below alongside the hyperparameters taken to train the model;

5.5.6 Attention based LSTM RNN Model

5.5.6.1 Attention Mechanism

In the models for sequence learning, RNN architecture tends to forget the longer sequence of data. Attention is proposed as the solution to this limitation of the architecture basically for encoder-decoder architecture but can be used for any other architectures. Attention is proposed as a method for both alignment and translation. 23Here alignment means which part of the input sequences are relevant to each word in the output. In attention mechanism [8] , instead of encoding the input sequence to single fixed context vector, the model develops a context vector that is filtered specially for each input sequences similar to how humans approach reading longer sequence of texts. Attention model learns to output the weights instead of statistically learning the weights in general neural network architecture. The following figure shows the attention mechanism in context to our project.

In the above figure we can clearly see that word has higher attention in the model than other words meaning the given words has higher probability to affect

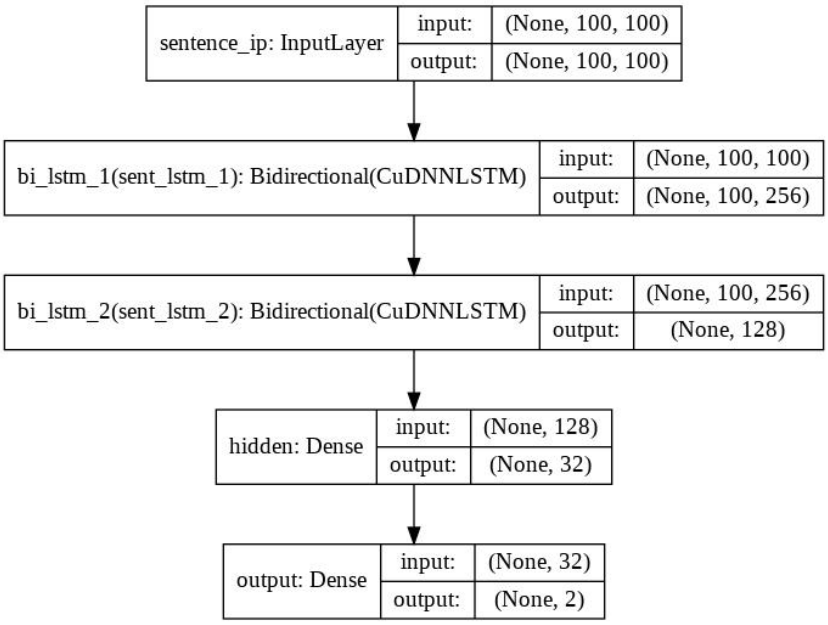

Figure 5.15: LSTM RNN Model

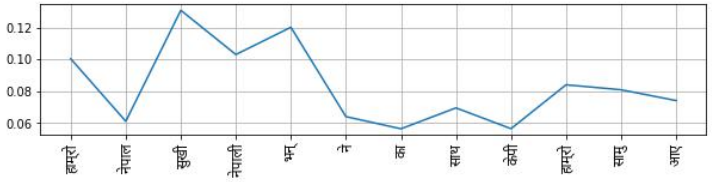

Figure 5.16: Attention Model Visualization

the output of the model than other words.

5.5.6.2 Classification Model

For classification we use slight variation to the model discussed in section 6.5.5.2. Here we add attention mechanism to the layer just after bidirectional LSTM's. The model was trained for 5 minutes in Google Colab GPU achieving validation accuracy of 80and model accuracy of 86parameters are shown below;

5.5.7 POS Integrated LSTM RNN Model

We considered the use of POS tagging system alongside LSTM RNN model to increase the accuracy of the model by increasing the dimension that the data trains on. For this we built separate model for POS and then integrated with LSTM model.

5.5.7.1 POS Tagging Model

Parts-of-Speech tagging model consists of two layered Bidirectional LSTM RNN model which takes Nepali sentence as input and outputs corresponding POS tag for each word in the input sentence. For the training of this model, Nepali National Corpus was used. The Nepali National Corpus (NNC) is a Nepali corpus built

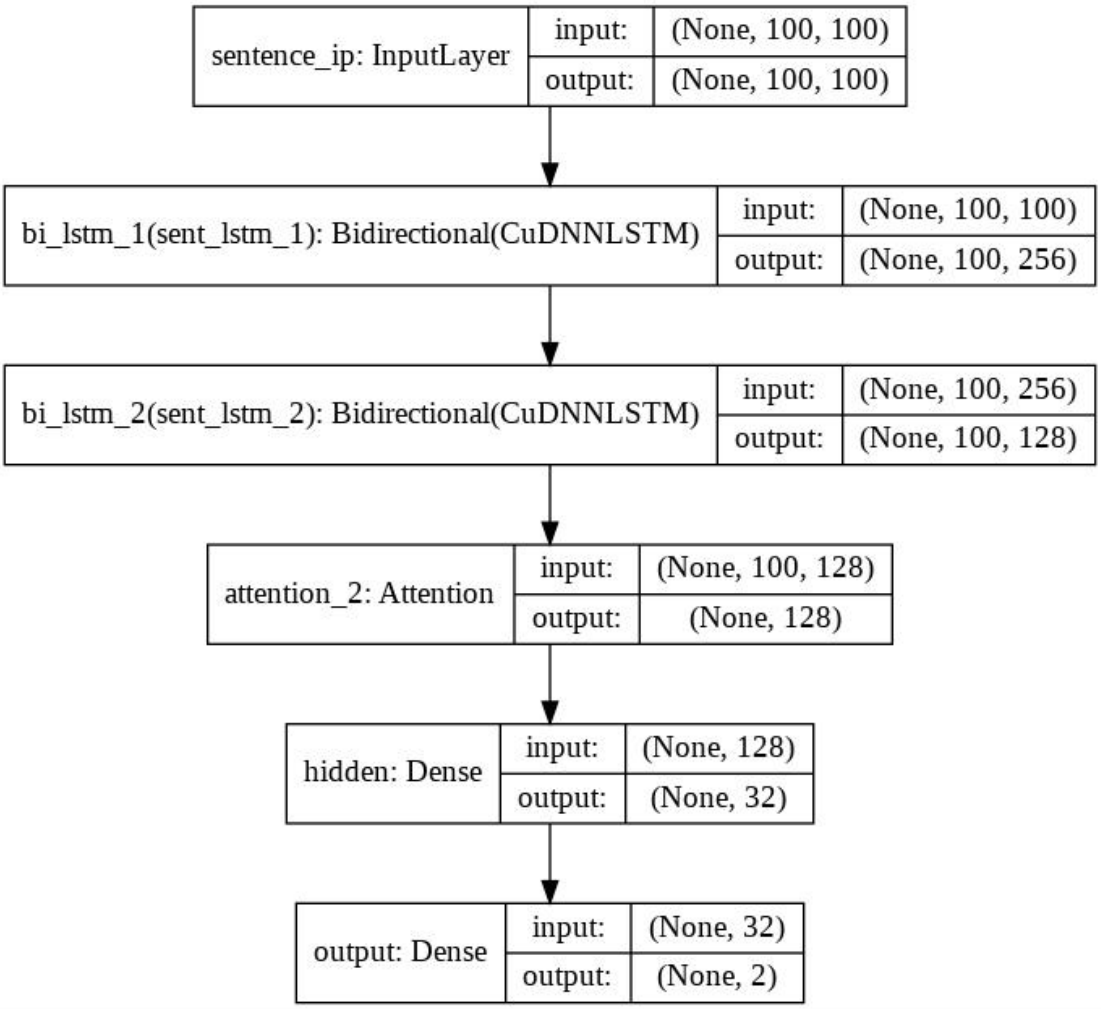

Figure 5.17: Attention Based LSTM RNN model

up 13 million words that are lemmatized and part-of-speech tagged. The corpus was created within the NELRALEC project funded by Asia IT C Programme of the European Commision. The written corpus is a collection of 500 texts of 15 different genres with 2000 words published between 1990 and 1992. The sentences were randomized and split into training and testing with 80-20 split ratio. The sentences were converted to vectors using the pre-trained Word2vec model of 100-dimension matrix. The POS tags are fully hierarchial in accordance to Penn Treebank 25for English language. This model has 114 output nodes for classifying 114 tags that were found in the NNC corpus. The POS tagger LSTM RNN model is shown below:

This model has achieved training accuracy of 98.08% and validation accuracy of 97.42% after training for 8 epochs in NVIDIA 1070Ti GPU.

5.5.7.2 Sentiment with POS

Above figure shows two-input LSTM model where "sentence$_ip$"$isinputforvectorizedtextand\tag$

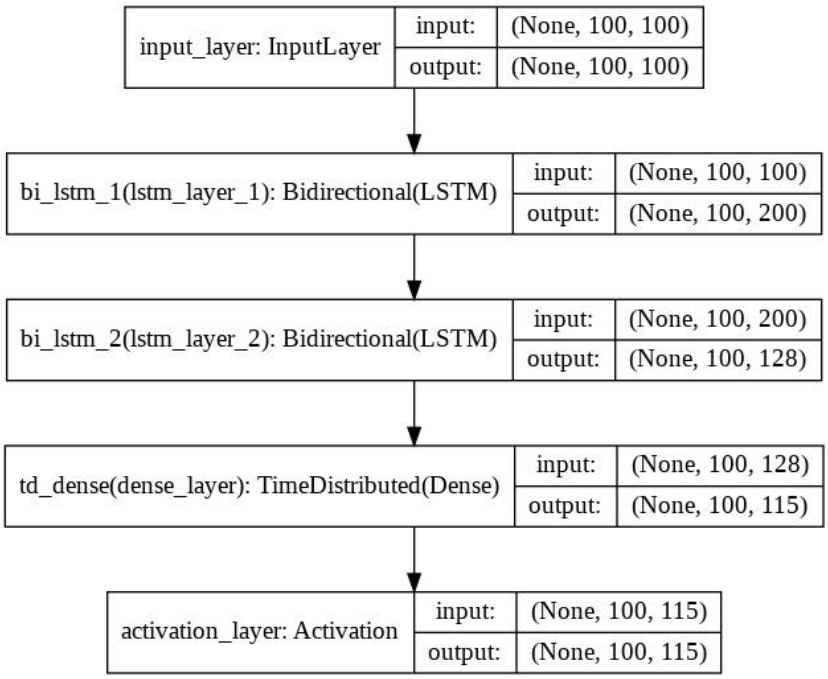

Figure 5.18: POS tagger LSTM RNN model

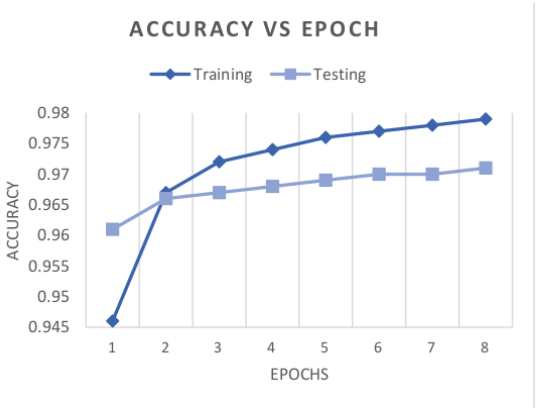

Figure 5.19: Accuracy vs Epoch of POS tagger

5.5.7.3 Sentiment with POS and Attention

The layer is similar to the model described in 6.5.7.2 with addition of attention layer after the LSTM layers.

5.6 DESCRIPTION OF DATA LABELING SYSTEM

It is the system for manually annotating data points scrapped or collected from various sources. Here. Data points are read from MySQL database and PHP application processes the data points according to annotator's annotation and saves them into MySQL database as well as in JSON format which is later serialized by Python script for training the model. The basic block diagram of the system is shown in the figure below and screenshot for labeling system is in appendix

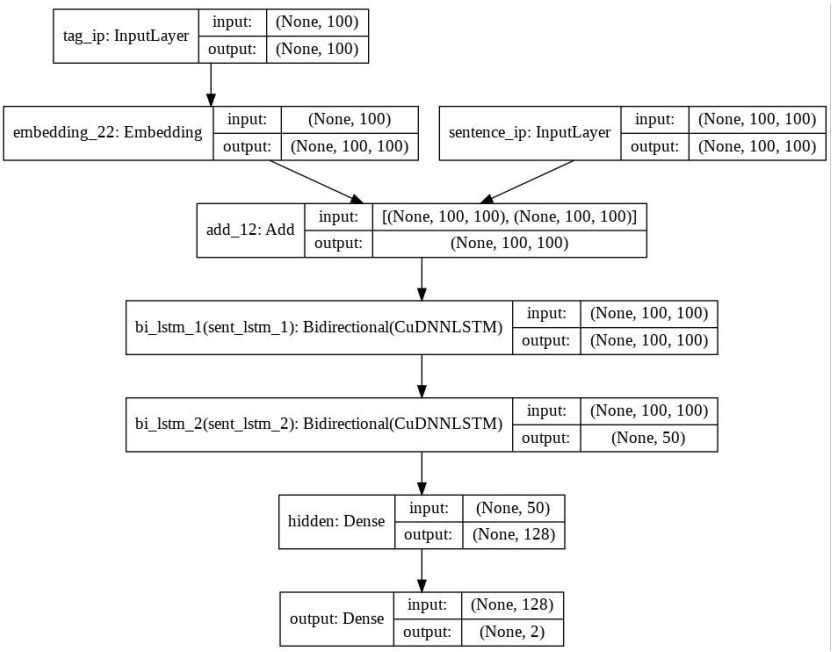

Figure 5.20: Model for Sentiment with POS

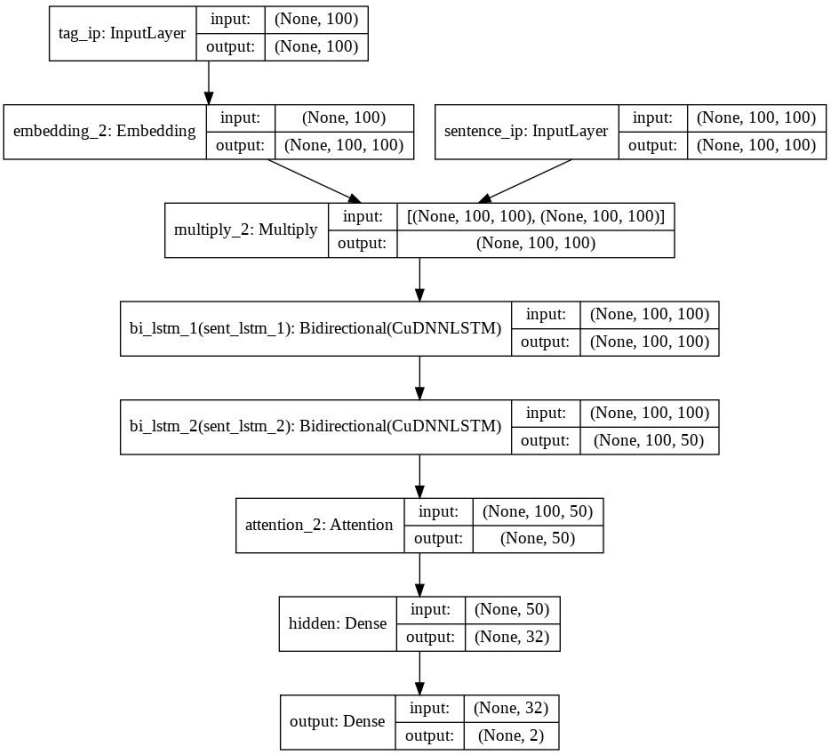

Figure 5.22: Model for Sentiment with POS and Attention

section of this report.

The following table shows data processing steps for the web applications using various RegEx filters to filter out tweets, hash-tags and various other unnecessary tokens for data labeling.

It is one of the most challenging part of the project and using this system we have labeled about 5,680 number of datapoints and description of labelled data

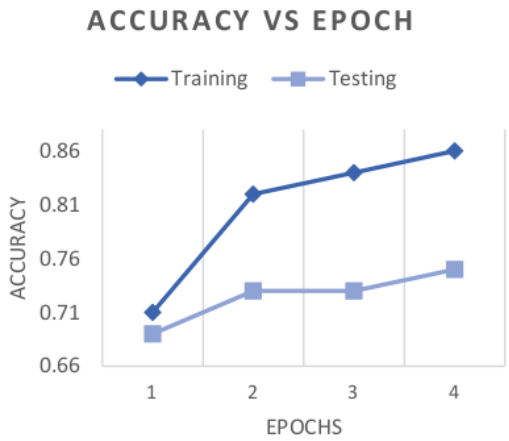

Figure 5.23: Accuracy vs Epoch for Sentiment with POS and Attention

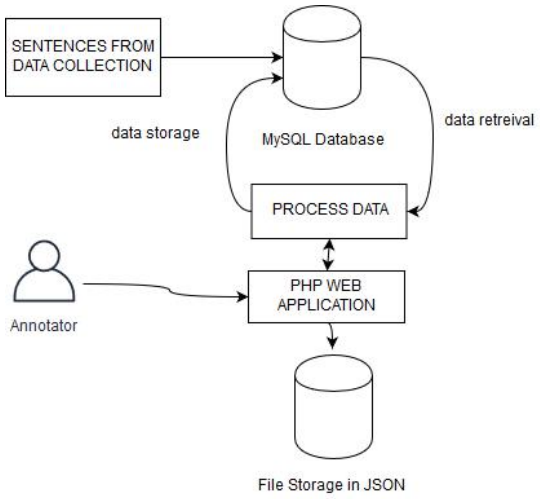

Figure 5.24: Block Diagram of Data Labeling System

into various sentiment classes is shown below in the bar graph.

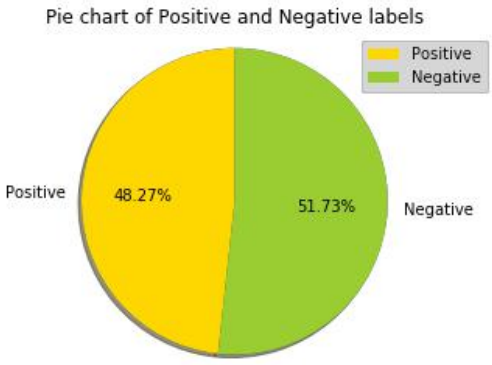

Figure 5.25: Positive and Negative Labels

The screenshot of the labeling system is found in the appendix section of this report.

Table 5.4: Data Processing in Labeling System

S.N.	RegEx	Description
1.	/@[a-zA-Z0-9_] * /*um*	Replace twitter handles with white spaces.
2.	/[—?]*/um	Remove full stops and question marks in Nepali Script because of our system working on sentence level classification.
3.	/[a-zA-z!]/um	Remove all English characters (a-z and A-Z) and exclamations.
4.	/\.\.\./um	Remove trailing full stops at the end of truncated tweets.
5.	/[:] /um	Remove ':' characters present in tweets.
6.	//um	Remove hashtags in tweet.

5.7 DESCRIPTION OF INFERENCE SYSTEM

The system is a simple web browser-based interface used to call backend machine learning pre-trained model. We use the best model found in our analysis to run inference. For this we have also developed modules for preprocessing input Nepali language that contains all the preprocessing algorithm mentioned above. 30Here text input data is taken from the input box when user types Nepali sentences and clicks go button for running the prediction model. The request from user is sent to flask function which will be then sent to our text processing modules responsible for tokenizing, stemming and removing stop words. After this process the text data is sent to embedding model changing all input words tokens into numbers. Similarly, the text is also fed into POS tagging model to output various POS tags. The vectorized numbers will be fed into our pre-trained machine learning models as described in section 6.5 of this document. The model then outputs the probability that the given sentence is either positive or negative. The output is sent to the client through the server and the user gets to know whether the input sentence is either positive or negative. This system aims to show the inference engine that is formed from training the model using manually labelled datapoints. The screenshot of the inference system is shown in the appendix section of this report.

Chapter 6

Expected Outcomes

The labelling system works properly. Sentences from the database were properly displayed in the web application. Labeler successfully labelled about 5,680 data points which was stored in 512KB chunks of files. The system for labelling Nepali sentiment sentences has been completed. The input data from the file was successfully retrieved, then preprocessed and sent to the model for training. Various

models for sentiment classification was trained and the results of the training are shown in the table below:

S.N.	Model	Training Accuracy	Testing Accuracy	Epochs
1.	LSTM RNN Model	80%	82%	3
2.	Attention Based LSTM RNN	88%	80%	5
3.	POS integrated LSTM RNN	86%	74.5%	4
4.	Attention based POS integrated LSTM RNN	89%	75%	4

The best model was found to be LSTM RNN Model with testing accuracy 82%, recall 87%, precision of 76%. Other models performed worse than this model was due to added complexity for low datapoints. After building the basic LSTM model we tried to increase its accuracy by changing various variables such as changing word embedding, adding POS feature to the basic model but, due to low number of datapoints for added complexity in the model, they started overfitting. To solve the problem of overfitting of the model we tried different measures such as dropout, regularization and increasing data. But increasing data was not an option for us due to limited time.

increase its accuracy by changing various variables such as changing word embedding, adding POS feature to the basic model but, due to low number of datapoints for added complexity in the model, they started overfitting. To solve the problem of overfitting of the model we tried different measures such as dropout, regular-

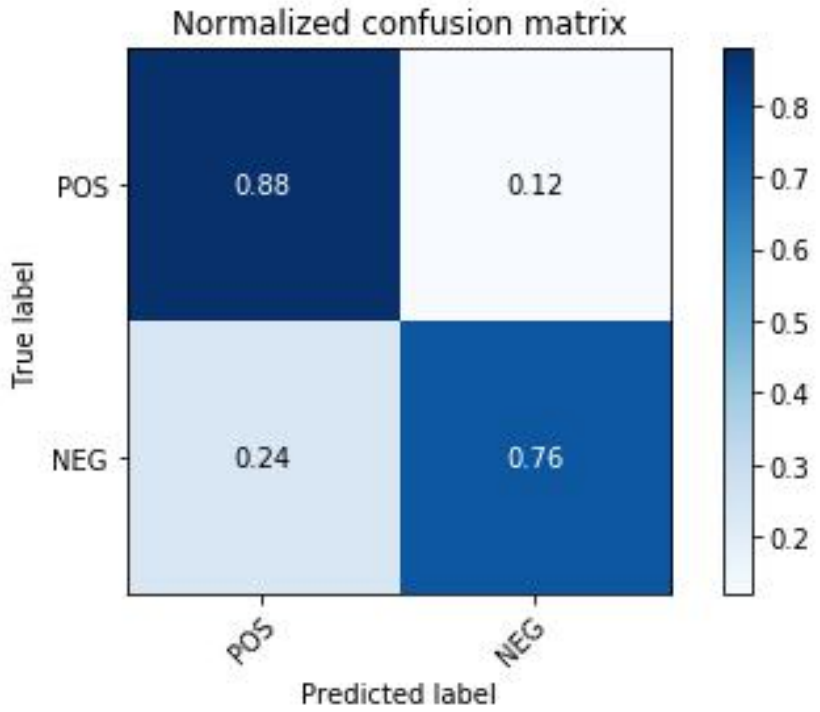

Figure 6.1: Confusion matrix for LSTM RNN Model

ization and increasing data. But increasing data was not an option for us due to limited time.