A COMPARATIVE STUDY OF MACHINE-LEARNING CLASSIFIERS USING WORD2VEC FOR MYANMAR SOCIAL MEDIA COMMENTS

HAY MAR SU AUNG

M.C.Sc. **JUNE 2022**

A COMPARATIVE STUDY OF MACHINE-LEARNING CLASSIFIERS USING WORD2VEC FOR MYANMAR SOCIAL MEDIA COMMENTS

By

HAY MAR SU AUNG

B.C.Sc(Hons:)

A dissertation submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

M.C.Sc.

University of Computer Studies, Yangon
JUNE 2022

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my gratitude to those who helped me with various aspects of conducting research and writing this thesis.

Firstly, I would like to express my deepest gratitude to **Dr. Mie Mie Khin**, Rector, the University of Computer Studies, Yangon, for her kind permission to submit this thesis.

I am grateful to my thesis supervisor **Dr. Win Pa Pa**, Professor, NLP Lab, for her suggestions and encouragement to do this thesis and also for her close supervision, invaluable suggestions, kind guidance and constant encouragement during the course of this work.

I also like to express my thanks to our dean **Dr. Si Si Mar Win** and **Dr. Tin Zar Thaw**, Professor and Head of Faculty of Computer Science, for their kind help in this work. I would like to thank **Dr. Yi Yi Mon**, the Principal of Government Technical High School (Mawlamyine) for administrative supports during my academic study.

I would like to extend my deepest gratitude to **U Myo Myint Naing**, Assistant Lecturer, English Department, for his advice and language editing.

I heartily appreciate the suggestions and recommendations of the teachers who attended all my seminars.

I also like to acknowledge **all my teachers** who taught me throughout the master's degree course and **my friends** for their corporation, and wish to express my gratitude to **my beloved parents** for their invaluable support and encouragement to fulfill my wish. I also would like to thank all of my colleges from IT Department of GTHS for their supports during my academic study.

Without their support, my seminar would never have been completed in such a successfully manner.

ABSTRACT

Sentiment Analysis (SA) is also known as opinion mining. The fundamental task of Sentiment Analysis (SA) is to extract, identify and determine the subjective information that is social sentiment in the source text. The application of Sentiment Analysis found in analyzing opinion to classify the class of a document or collection of documents, like blog posts, reviews, news articles and social media feeds like tweets and status updates. This study presents the idea of Sentiment Analysis in term of comparison on three different machine learning techniques, and illustrates how Myanmar Facebook comments are classified by applying word vector representation techniques. In this work, Facebook comments are used for model training and testing. They are collected from Myanmar Celebrity page. These comments are transformed as word vector by using word vector representation techniques – TF-IDF, Word2Vec and Pre-trained Word2Vec. Then, the word vector are trained with three machine learning classifiers for sentiment identification. Three machine learning classifiers are Logistic Regression, Support Vector Machine (SVM) and Random Forest. According to the experimental results of Pre-trained Word2Vec outperformed other two methods of word vector representation technique.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

CHAPTER 1
INTRODUCTION

In the distinctive events of technology, the data are being stored in the internet which are growing day by day and the amount of data stored up to date is enormous. But some very vital information are carried by these data carry about the opinions of different types of people around the world, so it has become very necessary to summarize these huge amount of data with some automated systems. In these days, many people use social media around the world like Twitter, Facebook and so on. Among them, Facebook is a popular free social networking website. In Myanmar, most people use Facebook to express their opinions and feelings.

Sentiment analysis is a text analysis technique that is used to detect the polarity within the text, whether a whole document, paragraph, sentence or clause. Sentiment analysis is also known as opinion mining. It uses data mining techniques to extract and capture analyzed data to classify the opinion of a document or collection of documents, like blog posts, reviews of product and social media feeds like Tweets, status updates and comments.

In these days, sentiment analysis has been used in many businesses such as social media monitoring, brand monitoring, voice of customer, customer service and market research. With the help of sentiment analysis, these tasks can be done in minimal amount of time, and they can determine the opinion about different areas of business.

There are a large number of vector representation methods to achieve word vector. Vector representation techniques are Word2Vec, Pre-trained Word2Vec and TF-IDF (Term Frequency- Inverse Document Frequency) Vectorizier. In classification problem, there are a number of Machine Learning algorithms, Logistic Regression, Support Vector Machine and Random Forest and so on. Not only vector representation method but also machine learning algorithm becomes more and more popular in these days.

In this study, vector representation techniques are used to convert word vector. And then, Machine Learning algorithm is used to train these word vectors to classify the polarity of comments from Myanmar Celebrity Page which were written in Myanmar Language.

1.1 Motivation of the Study

Most people express their opinion and feelings in social media. There are some limitations, facing by examining a large volume of text from various sources, to learn a user's or audience's opinion on a target object. Most existing algorithms for continuous word representation generally model the syntactic context of words but ignore the sentiment of text. The opinion are generally classified by feature word extraction method. There are some challenges.

Some of these limitations are:

- being problematic for sentiment analysis as usually mapping words with similar syntactic context.
- analyzing huge amounts of data in a short time.
- analyzing more relevant and specific results.

Due to these limitations, an extreme need for syntactic context appeared to enhance the sentiment analysis performance.

1.2 Objectives of the Study

The main objectives of this study is to test the results of word vector using Machine Learning classifiers. The specific objectives are as follows:

(i) to apply word embedding vector for classifying the sentiment polarity
(ii) to do the optimal analysis by analyzing the system with different machine learning algorithms
(iii) to encode sentiment knowledge into pre-trained word vectors to improve the performance of sentiment analysis

1.3 Contributions of the Study

The system develops domain independent social media comment, and classify opinion based on pre-trained Word2Vec models. The proposed system is very useful in Facebook social media domain area. The contributions of the study are as follows:

(i) Pre-trained Word2vec model for Myanmar language is developed.
(ii) Word vector for social media comment based on word vector representation model(Pre-trained Word2Vec, Word2Vec, TF-IDF) which

is combined with the Machine Learning Classifier models(Logistic Regression, SVM, Random Forest) is proposed to classify the sentiment classes.

1.4 Organization of the Study

This study is composed of five chapters, abstract, acknowledgment and references.

Chapter one presents the introduction and objectives of the study. This chapter also describes the motivation, contribution and objectives of this work. Chapter two shows background theory of sentiment analysis, word embeddings and machine learning classifiers. Chapter three describes the details of Sentiment Analysis of Myanmar Facebook page's comments. Firstly, it is explained in detailed of preprocessing of comments, how to generate word vector by using word vector representation model, and then classified these vector. Chapter four explains the design and implementation of programming modules for Graphical User Interfaces which have been used to explain the proposed system. Furthermore, charts and tables are used to display the experimental results. Chapter five presents conclusion including challenges, benefit and limitation of the proposed system.

CHAPTER 2
SENTIMENT ANALYSIS AND CLASSIFICATION

The theoretical backgrounds of sentiment analysis, word vector representation methods, and classification algorithms in machine learnings were described in the following subsections.

2.1 Sentiment Analysis

Sentiment analysis, often known as opinion mining, is a natural language processing (NLP) technique for determining the emotional tone of a body of text. This is a common method for business to determine and categorize customer feedback on a product, service, or concept. It implies mining text for sentiment and sources of published information using data analysis, machine learning (ML), and artificial intelligence (AI).

Sentiment analysis tools assist business in extracting information from unstructured and unorganized language found online such as emails, blog posts, support tickets, web chats, social media channels, forums, and comments. By using rule-based, automatic, or hybrid methods, algorithms can replace manual data processing. Rule-based systems analyze sentiment using predetermined lexicon-based rules, whereas automatic systems use machine learning techniques to learn features from data. The two methodologies are combined in a hybrid sentiment analysis.

In addition to identifying sentiment, opinion mining can extract the polarity (or the amount of positivity and negativity), subject and opinion holder within the text. Furthermore, sentiment analysis can be applied to varying scopes such as document, paragraph, sentence and sub-sentence levels.

2.2 Word2Vec

Word embeddings is a technique where individual words are transformed into a numerical representation of the word (a vector). Where each word is mapped to one vector, this vector is then learned in a way which resembles a neural network. The vectors try to capture various characteristics of that word with regard to the overall text. These characteristics can include the semantic relationship of the word, definitions, context, etc. With these numerical representations, many things like identify similarity or dissimilarity between words can be done. It is neutral network model that attempts

to explain the word embeddings based on a text corpus. Initially, the each word in the corpus is assigned a vector of random numbers. Then, it can be iterated through each document word grab the vectors of the nearest n-words on either side of the target word, concatenate these vectors, propagate them forward through a linear layer + softmax function, and try to predict what the target word was. The error between the estimate and the actual target word is then backpropagated and not only the weights of the linear layer but also the vectors (or embeddings) of the neighbor's words are modified. At the end, the weights from the hidden layer are extracted and these weights encode the meaning of words in the vocabulary.

There are two different model architectures which can be leveraged by Word2Vec to create these word embedding representations. These include,

- The Continuous Bag of Words (CBOW) Model
- The Skip-gram Model

2.2.1 CBOW Model

The CBOW model learns to predict a target word in its neighborhood, using all words. To predict the target word, the sum of the background vectors is used. A pre-defined window size surrounding the target word defines the neighboring terms that are taken into account.

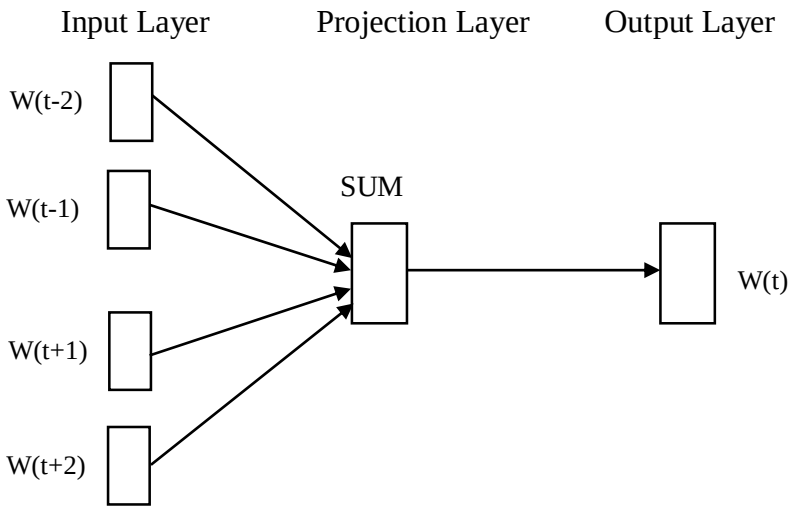

Figure 2.1 CBOW Model Architecture

2.2.2 Skip-gram Model

Skip-gram model predicts the context for given a word. The skip-gram model is the exact opposite of the CBOW model. In this case, the target word is fed at the input, the hidden layer remains the same, and the output layer of the neural network is replicated multiple times to accommodate the chosen number of context words. This model works well with small amount of the training data, represents well even rare words or phrases.

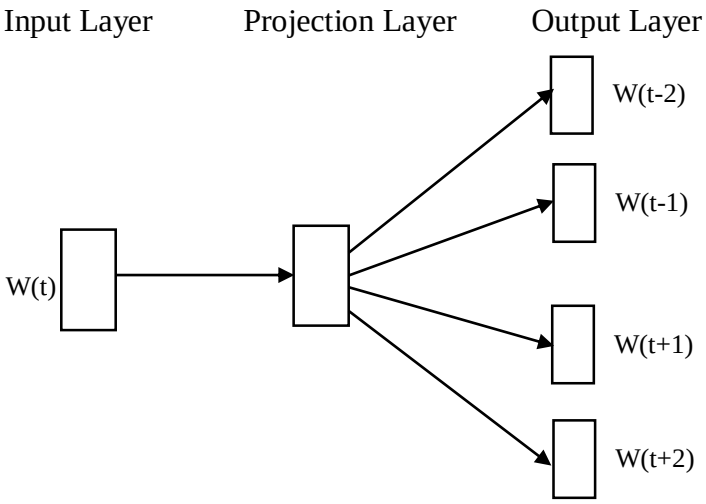

Figure 2.2 Skip-gram Model Architecture

The Word2Vec model is used to extract the notion of relatedness across words or products such as semantic relatedness, synonym detection, concept categorization, selectional preferences, and analogy. A Word2Vec model learns meaningful relations and encodes the relatedness into vector similarity. The main applications of Word2Vec can be summarized in knowledge discovery and recommender systems.

2.3 TF-IDF

TF-IDF short term for term frequency –inverse document frequency is generally a content descriptive mechanism for the documents. It is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in information retrieval, text mining, and user modeling. The TF-IDF value increases proportionally the number of times a word appears in the document, but is often offset by the frequency of the word in the corpus,

which helps to adjust for the fact that some words appear more frequently in general. The term frequency will increase if the frequency of the word increases in the corpus. Variations of the TF-IDF weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query. One of the simplest ranking functions is computed by summing the TF-IDF for each query term; many more sophisticated ranking functions are variants of this simple model. TF-IDF can be successfully used for stop-words filtering in various subject fields including text summarization and classification.

The term frequency (TF) is the frequency of word appears in a specific document. The calculation equation is as:

$$TF = \frac{occurencies\ of\ a\ word\ in\ the\ document}{total\ words\ in\ the\ document} \quad (2.1)$$

Inverse Document frequency (IDF) for a specific word measures the log of the division of the sum of all document numbers and the document numbers with concluding the particular word in it. It is calculated as the following:

$$IDF = log\left(\frac{total\ number\ of\ document}{number\ of\ document \in the\ word}\right) \quad (2.2)$$

The concepts of term frequency and inverse document frequency are combined to produce a composite weight for each term in each document.

$$TF - IDF = TF * IDF \quad (2.3)$$

2.4 Classification Algorithms in Machine Learning

Classification is one of the most significant roles of a supervised learning approach. Machine Learning algorithm and information techniques are applied in this area to solve real world problem. The classification can be performed on structured or un-structured data. It identified the classification algorithm used to identify the target to which new data will observe on the training data. The classification process includes learning from the given dataset or the training corpus and then classifying the new observation into a number of targets or groups.

The amount of information available on the internet is growing with unstoppable rate. The demand for asking for tools to analyze or organize these

enormous data resources was also increased. Classification plays the important role in many information management tasks. Due to these facts, classifying process needs to not only improve performance but also maintain accuracy rate are highly desired.

Automatic classification of text documents is useful in the process of dealing with the massive textual data such as library management system, spam filtering system, classifying news sources and analyzing organizational data as most of data input were in textual data. Many theories and machine learning techniques have been applied in classification approach.

The classification of documents can be done by human experts, however, human cannot be handling these growing dataset and it is time consuming task and human errors in the classification task are also obstacle in the process and these processes are also expensive. Therefore, automatic text classification is the powerful tools for organizing documents and an essential technology for intelligent information system which gains many attentions in recent years. If the need for text classification increases, the business value of the text classification will increase and become the important technique in digital world. This technique has been applied in classifying new electronic documents, finding interesting information web and guiding user's search through hypertext. Accuracy and efficiency of text classification model is important. The accuracy of the model depends on the number of training set data and its fitness. With many classification algorithms, the suitability of an algorithm depends on ease of understanding, ease of model building, computation cost and result accuracy measures. [12]

Classification is the exciting research area in many information retrieval problems such as filtering, routing or searching for relevant information, benefit from the classification research.

2.4.1 Support Vector Machine

In a nutshell, a support vector machine (or SVM) is an algorithm that works as follows: it uses a nonlinear mapping to transform the original training data into a higher dimension. Within this new dimension, it searches for the linear optimal separating hyperplane (that is, a "decision boundary "separating the tuples of one class from another). With an appropriate nonlinear mapping to a sufficiently high dimension, data from two classes can always be separated by a hyperplane. The SVM finds this hyperplane using support vectors ("essential" training tuples) and margins (defined by

the support vectors). Although the training time of even the fastest SVMs can be extremely slow, they are highly accurate, owing to their ability to model complex nonlinear decision boundaries. They are much less prone to overfitting than other methods. The support vector found also provides a compact description of the learned model. SVMs can be used for prediction as well as classification. They have been applied to number of areas, including handwritten digit recognition, object recognition, and speaker identification, as well as benchmark time-series perdition tests.

2.4.2 Logistic Regression

This type of statistical model (also known as logit model) is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables. Since the outcome is a probability, the dependent variable is bounded between 0 and 1. In logistic regression, a logit transformation is applied on the odds—that is, the probability of success divided by the probability of failure. This is also commonly known as the log odds, or the natural logarithm of odds. There are three types of logistic regression models, which are defined based on categorical response.

- In **Binary logistic regression** approach, the response or dependent variable is dichotomous in nature—i.e. it has only two possible outcomes (e.g. 0 or 1). This approach has been used in predicting if an e-mail is spam or not spam or if a tumor is malignant or not malignant. Within logistic regression, this is the most commonly used approach, and more generally, it is one of the most common classifiers for binary classification.
- In **Multinomial logistic regression** model, the dependent variable has three or more possible outcomes; however, these values have no specified order. This approach has been used in movie studios want to predict what genre of film a moviegoer is likely to see to market films more effectively. A multinomial logistic regression model can help the studio to determine the strength of influence a person's age, gender, and dating status may have on the type of film that they prefer. The studio can then orient an advertising campaign of a specific movie toward a group of people likely to go see it.
- In **Ordinal logistic regression** model is leveraged when the response variable has three or more possible outcome, but in this case, these values do have a

defined order. Examples of ordinal responses include grading scales from A to F or rating scales from 1 to 5.

2.4.3 Random Forest

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression. One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression and categorical variables as in the case of classification. It performs better results for classification problems. That's because it consists of multiple decision trees just as a forest has many trees. On top of that, it uses randomness to enhance its accuracy and combat overfitting, which can be a huge issue for such a sophisticated algorithm. These algorithms make decision trees based on a random selection of data samples and get predictions from every tree. After that, they select the best viable solution through votes. It has numerous applications in daily lives such as feature selectors, recommender systems, and image classifiers. Some of its real-life applications include fraud detection, classification of loan applications, and disease prediction.

CHAPTER 3
SENTIMENT ANALYSIS USING WORD VECTOR REPRESENTATION METHODS AND MACHINE LEARNING CLASSIFIERS

The activities that constitute semantic analysis for Myanmar social media comments are described in this chapter. It contains information on the nature of Myanmar language, as well as preprocessing approaches and applicable theories in this system.

3.1 Sentiment Analysis

Nowadays, the inexorable rapid growth of information from the internet or social media determines the emotional tone in order to organize and access these data in a more convenient manner. Sentiment classification is one of the most common methods in resolving these issues since it can handle and organize large amounts of data. Sentiment analysis can be defined as the process of mining text for contextual information in order to detect and retrieve subjective information, such as assisting a corporate organization in understanding the social sentiment of particular product or service information. Text mining frequently includes sentiment analysis.

Sentiment analysis, often known as opinion mining, is a technique for automatically identifying and extracting opinions from data. It allows clients' feelings to be extracted, such as what they like and dislike, or if they are satisfied or unsatisfied. It can handle a big number of page comments, e - mails, and surveys in a matter of minutes, with real-time results. Another option is to undertake sentiment analysis over time to see how sentiment classification changes over time. Sentiment analysis focuses on a text's polarity (positive, negative, or neutral), but it can also detect specific moods and emotions (angry, glad, sad, etc.), urgency (urgent, not urgent), and even intents (interested v. not interested).

There are four types of sentiment analysis. The following are described in detail:

1. Fine-grained sentiment analysis is used to evaluate polarity into separate groups, usually highly positive to very negative, to provide a more exact level of polarity. That it can be thought of as a set of categories that encompass various levels of good and negative:
    - Highly positive

- Positive
- Neutral
- Negative
- Highly negative

This is recognized as graded or fine-grained sentiment analysis, and it can be used to decode 5-star ratings in comments, for example:

- Highly Positive = 5 stars
- Highly Negative = 1 star

2. Rather than identifying positive and negative emotions, emotion detection recognizes specific emotions. Happiness, frustration, shock, anger, and grief are just a few examples. Many emotion recognition systems depend on lexicons (lists of words and the feelings they bring to mind) or sophisticated machine learning techniques. People communicate emotions in a variety of ways, which is one of the drawbacks of employing lexicons. Some words that are commonly used to communicate anger, such as bad or kill (for example, your product is so horrible or your customer service is killing me), can also be used to express satisfaction (e.g. this is bad ass or you are killing it).
3. Intent-based analysis distinguishes between acts and opinions in a text. An online comment indicating dissatisfaction with changing a battery, for example, can motivate customer service to contact you to remedy the problem.
4. Aspect-based analysis collects the particular component that is being mentioned positively or negatively. For instance, a customer might write a review on a product complaining about how short the battery life was. Then, the system will respond that the negative feedback is about the battery life, not the product as a whole.

3.2 Text Classifier

Text classification has been used in a variety of applications, including text filtering, document organization, news story classification, web search for interesting information, spam e-mail filtering, and so forth. Many of these systems were built around a certain language, with the majority of these languages being English,

European, and Asian languages, with only a few works specialized to Myanmar languages. As a result the classification of Myanmar papers has remained challenging due to the morphological richness and security of language resources such as automatic tokenization, feature selection, and stemming.

In general, text can be represented in two separate ways. The first way is as a bag of words, in which a document is represented as a set of words, together with their associated frequency in the documents. Such a representation is essentially independence of the sequence of words in the collection. The second method is to represent text directly as strings, in which each document is a sequence of words. Most text classification methods use the bag-of-words representation because of its simplicity for classification purpose.

3.3 Data Collection

The initial step in the sentiment analysis procedure is to collect data from public Facebook pages. At the time of data collection, there was not previously created dataset for public Facebook page comments in Myanmar Language. To build the own dataset, the dataset was crawled from public comments on Facebook pages related to "Myanmar Celebrity" Page in Myanmar. Social media data is collected through data crawling using the Facepager tool.

3.4 Preprocessing In Myanmar Language

The official language of the Republic of Union of Myanmar is Myanmar Language. It belongs to the Sino-Tibetan language family, which is part of the Tibetan-Myanmar (Tibeto-Burman) subfamily. Myanmar text is written in the Myanmar script, which is derived from a Brahmi-related system imported from South India for the Mon language around the 8th century. The first Burmese inscription comes from the following years and is written in an alphabet that is nearly identical to that of Mon inscriptions. Although casual writing frequently includes spaces after each clause, it is typed from left to right and without any spaces between words. Myanmar language speakers frequently use as much or little space as they see fit, and many speakers even write without any space at all.

Since there are no clear restrictions about spaces in sentences, most people have written spacing between words as they want to appropriate, or none at all. Myanmar is

a free-word-order and verb-final language that normally includes the subject-object-verb sequence. Sentences also contain a large number of preposition adjuncts that appear in several places.

The Myanmar script is composed of 33 consonants, 11 basic vowels, 11 combination symbols and extension vowels, vowel symbols, de-vowelizing consonants, diacritic marks, specified symbols and punctuation marks. [5]. Myanmar has mainly 9 parts of speech: noun, pronoun, verb, adjective, adverb, particle, conjunction, post-positional marker and interjection. [5]

Before using any sentiment analysis technique, it is important to preprocess comments from the Facebook page. Tokenization or text normalization are terms used to describe the process of preprocessing text. The preprocessing stage includes data conversion, data cleaning, and word segmentation. A preprocessing stage was implemented based on the type and requirements of something like the applications. Due to the nature of the language and a lack of resources, preprocessing in Myanmar is still a challenge. Many researchers have tried both traditional and statistical ways to preprocess raw text.

3.4.1 Font conversion

During the font conversion process, the majority of Myanmar-Facebook users utilizes the Zawgyi font. Unicode is used in technological applications. As a result, the data must be converted to Unicode utilizing an online Zawgyi-Unicode converter [36].

Table 3.1 A Sample Font Conversation Sentence

Zawgyi	Unicode
တစ္ဘူးေလာက္လုိခ်င္လုိ႔ပါ	တစ်ဘူးလောက်လိုချင်လို့ပါ

3.4.2 Data Cleaning

The next step is to clean up the data. The text comment data contains several types of noise. Before starting the training and testing, all unwanted punctuation, hash signs, tag names, additional spaces, unknown characters, and numbers are deleted as shown in the Table 3.2.

Table 3.2 A Sample Cleaning Dataset

Raw Sentences	After Cleaning Sentences
"အားပေးမယ်ညေး"	"အားပေးမယ်ညီ"
Happy Birthdaypar Hdgkxjcdg	Happy Birthdaypar
Myo Htet Aung နားထောင်ကြည့်😄😄	နားထောင်ကြည့်😄😄
အိုက် စ် စ် စ် .... မူး လက် ထှာ	အိုက် .... မူး လိုက် တာ
ယေး	ယေး
#Maskလဲမတပ်ကြဘူး 09882612796, □□	Maskလဲမတပ်ကြဘူး

3.4.3 Word Segmentation

The final stage of preprocessing process in this system is the words segmentation stage. In most Natural Language Processing applications, word tokenization is an essential step in the operation of word segmentation. However, word segmentation is done manually in the implemented system. In the preprocessing procedure, the system only accepts words that have already been segmented. Words are segmented throughout the word segmentation process. For example, a sample corpus can be seen as follow.

ချစ်ခဲ့ချစ်ဆဲ ဆက်၍ ချစ် နေ ပါအုံးမည်❤

Happy Birthdaypar

ချစ်တာထက်ထက်ရယ်

ဒီရုပ်က Miss Universe ဖြစ်ချင်သေးတာလား ဆုတောင်းနေလိုက်

Figure 3.1 A sample datatset

Sentences from figure 3.1 are segmented and Figure 3.2 shows the segmentation of the sentences.

ချစ် ခဲ့ ချစ် ဆဲ ဆက် ၍ ချစ် နေ ပါ အုံး မည် ❤

Happy Birthday par

ချစ် တာ ထက်ထက် ရယ်

ဒီ ရုပ် က Miss Universe ဖြစ်ချင် သေးတာ လား ဆုတောင်း နေလိုက်

Figure 3.2 Segmented sample sentences

3.5 Word vector Representation

In analyzing associations among words, phrases, and documents, the word vector representation technique plays an important part. Nowadays, modern technology provides machines with more information about words than previously possible with traditional word representations. Word vectors are numbers that represent the meaning of a word. Word embedding is a sort of word representation that allows machine learning algorithms to understand words with similar meanings.

It is a mapping of words into real-number vectors using a neural network, probabilistic model, or dimension reduction on the word co-occurrence matrix in technical terms. It's an approach for language modeling and feature learning. Word embedding is a neural network-based mapping technique. Word embedding models such as word2vec (Google), Glove (Stanford), and fastest (Facebook). Individual words are represented as real-valued vectors in a predetermined vector space in word embeddings, which is a class of approaches. Because each word is mapped to a single vector and the vector values are acquired in a manner that resembles that of a neural network, the technique is frequently grouped with deep learning.

Feature generation, document clustering, text classification, and natural language processing problems all benefit from word embedding. Word embedding are used in the following applications.

- **Compute related words:** term embedding is used to generate words that are similar to the word being predicted. It also proposes terms that are dissimilar to each other, as well as the most common words.

- **Generate a set of similar words:** This is employed for semantic grouping, which groups items that have similar characteristics together and those that are dissimilar far apart.
- **Feature for text classification:** Text is converted into vector arrays, which are supplied into the model for training and prediction. Because text-based classifier models cannot be trained on strings, this transforms the text into a machine-trainable format. It also has the capability of providing semantic assistance in text-based classification.
- **Document clustering:** Word embedding is very commonly employed in this application.
- **Natural language processing:** Many applications, such as parts of speech tagging, emotional analysis, and syntactic analysis, benefit from embeddings and win over extraction of features phases.

In this system, the three word vector representation methods are used to extract the input words into vectors: pre-trained Word2Vec, Word2Vec, and TF-IDF Vectorizier.

3.5.1 Pre-trained Word2Vec

For pre-trained Word2Vec, Pre-trained Word Embeddings are embeddings that have been learned in one challenge and can be utilized to solve a comparable assigned task. Such embeddings are created by training them on big datasets, saving them, and then using them to solve other problems. Pre-trained word embeddings are a type of Transfer Learning. As they are trained on huge datasets, pre-trained word embeddings capture the semantic and pragmatic meaning of a word. They have the ability to improve a Natural Language Processing (NLP) model's performance. The majority of real-world problems includes a dataset with a substantial number of unusual words. These datasets' embeddings are unable to produce the correct representation of the word. To accomplish this, the dataset needs to have a large number of vocabularies. Frequently recurring words help to create a diverse vocabulary.

3.5.2 Word2Vec Model

In building word2vec model, obtaining the proper feature for the classification process is critical due to the large dimensionality of text features and the presence of

irrelevant (noisy) features. The CBOW model of the Word2Vec method was used to investigate this system. It tries to deduce the output (target word) from its surroundings (context words). The following steps are used to break down the content:

(i) **Data Preparation:** Tokenizing text to define the corpus. The text are converted to a list of tokenized words after pre-processing. The following figure show the tokenized words.

'သေးတာ', 'ဆက်', '၍', 'par', 'က', 'ထက်ထက်', 'ရယ်', 'Birthday', 'ဆဲ', 'မည်',
'Happy', 'Miss', 'ခဲ့', 'ပါ', 'ဖြစ်ချင်', 'ရုပ်', 'အုံး', 'ဆုတောင်း', '❤', 'တာ', 'နေ', 'ချစ်', 'ဒီ',
'Universe', 'လား', 'နေလိုက်'

Figure 3.3 Tokenized words

(ii) **Create Data for Building Word Vector:** the dataset is used to build a word vocabulary dictionary, use one-hot encoding, and create a word index. This step builds unique word vocabulary, prepare single-word training data, and then define "target word" in the CBOW model as the word that comes after a particular term in the text (which will be the "context word"). This means that, for a given word, the next word will be predicted.

The training examples which are scanned through the text with a window size will prepare a context word and a target word. The Table 3.3 will show the target word and context word with window 2.

Table 3.3 Target Word and Context Word Sample

Target word	Context word
ချစ်	'ခဲ့', 'ချစ်'
ခဲ့	'ချစ်', 'ချစ်', 'ဆဲ'
ချစ်	'ခဲ့', 'ချစ်', 'ဆဲ', 'ဆက်'
ဆဲ	'ချစ်', 'ခဲ့', 'ဆက်', '၍'
ဆက်	'ဆဲ', 'ချစ်', '၍', 'ချစ်'
၍	'ဆက်', 'ဆဲ', 'ချစ်', 'နေ'
ချစ်	'၍', 'ဆက်', 'နေ', 'ပါ'
နေ	'ချစ်', '၍', 'ပါ', 'အုံး'
ပါ	'နေ', 'ချစ်', 'အုံး', 'မည်'
အုံး	'ပါ', 'နေ', 'မည်', '❤'
မည်	'အုံး', 'ပါ', '❤', 'happy'
❤	'မည်', 'အုံး', 'happy', 'birthday'
happy	❤', 'မည်', 'birthday', 'par'
birthday	'happy', '❤', 'par', 'ချစ်'
par	'birthday', 'happy', 'ချစ်', 'တာ'
ချစ်	'par', 'birthday', 'တာ', 'ထက်ထက်'
တာ	'ချစ်', 'par', 'ထက်ထက်', 'ရယ်'
ထက်ထက်	'တာ', 'ချစ်', 'ရယ်', 'ဒီ'
ရယ်	'ထက်ထက်', 'တာ', 'ဒီ', 'ရုပ်'
ဒီ	'ရယ်', 'ထက်ထက်', 'ရုပ်', 'က'
ရုပ်	'ဒီ', 'ရယ်', 'က', 'miss'
က	'ရုပ်', 'ဒီ', 'miss', 'universe'
miss	'က', 'ရုပ်', 'universe', 'ဖြစ်ချင်'
universe	'miss', 'က', 'ဖြစ်ချင်', 'သေးတာ'
ဖြစ်ချင်	'universe', 'miss', 'သေးတာ', 'လား'
သေးတာ	'ဖြစ်ချင်', 'universe', 'လား', 'ဆုတောင်း'
လား	'သေးတာ', 'ဖြစ်ချင်', 'ဆုတောင်း', 'နေလိုက်'
ဆုတောင်း	'လား', 'သေးတာ', 'နေလိုက်'
နေလိုက်	'ဆုတောင်း', 'လား'

Because the system can only read numeric values, these target and context words transform text to one-hot encoding. For example, the vector [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0] represents the encoded value of the word "ချစ်" which appears first in the vocabulary. The second word in the vocabulary, "ခဲ့" will be encoded as the vector [0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]. The details example is shown in Appendix 1.

The next stage is showed the overall set of context-target words in one hot encoded form: Target word for 'ချစ်' is set target vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]. Context word for 'ချစ်' is ['ခဲ့', 'ချစ်']. Context word encoded form is [1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

The above table shows the final training data, where the encoded target word is the Y variable and the encoded context word is the X variable.

(iii) Model of Train: Forward pass one hot encoded word, compute loss to calculate error rate, and back propagate weights to change weights. It is a dimension of word embedding. Embedding can be thought of as a collection of traits or entities such as an organization, a name, or a gender. It could be ten, twenty, or one hundred. An increasing number of embedding layers will provide a more detailed explanation of a word or symbol. For example, the trained word2vec on this sample dataset has a dimension of 5. Assume that the first training data point are used to train the CBOW model, and the context word is 'ခဲ့', 'ချစ်' and the target word is " ချစ်" . The weights should be set so that when the model is given the input x= [1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.] – which is the word "'ခဲ့', 'ချစ်' " – it produces output that is close to y = [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.] – which is the word "ချစ်".

The weight matrix (w) for input and the weight matrix for hidden (w_2) is randomly assigned [VxN] and [NxV] respectively where V is number of unique word in vocabulary and N is the number of embedding layer.

W1= [[0.54340494 0.27836939]

[0.42451759 0.84477613]

……..

[0.28589569 0.85239509]

[0.97500649 0.88485329]] [VxN]

W2 = [[0.35950784 0.59885895 ….. 0.59797368 0.73130075 0.34038522]

[0.0920556 0.46349802 …… 0.7795984 0.61032815 0.30900035]] [NxV]

Weight matrix for hidden layer is calculated using Eq(3.1):

$$h = w^T . x \tag{3.1}$$

where as h is weight matrix for hidden, w^T is the transpose of weight matrix for input and x is the input. The calculation example is shown in the Figure 3.4.

$h = w^T . x$

h = [[0.54340494 0.42451759 …….. 0.28589569 0.97500649] *[0.5 0.5 ….. 0.]

[0.27836939 0.84477613 …… 0.85239509 0.88485329]]

= [[0.48396127] [0.56157276]]

Figure 3.4 Hidden Layer Calculation

$$u = w_2^T . h \tag{3.2}$$

For calculation the (3.2) is used the output layer. The calculation example is shown in Figure 3.5.

$u = w_2^T . h$

= [[0.22568379]

[0.55011239]

…….

[0.72719732]

[0.6966649]

[0.33825944]][Vx1]

Figure 3.5 Output Layer Calculation

For each target word, softmax calculates the probability from the values of u. Softmax function uses exponential to get output from softmax in a range between 0 to 1.

$$y_i = \frac{e^j}{\sum_{j=1}^{V} e^j} \quad (3.3)$$

The Eq(3.3) is used to calculate softmax layer. The example calculation of softmax layer is shown in the Figure 3.6.

$$y_i = \frac{e^j}{\sum_{j=1}^{V} e^j}$$

y_i= [[0.02880002]
[0.03983734]
……..
[0.0461251]
[0.03223174]]

Figure 3.6 Softmax Layer Calculation

The next step is to calculate error to see how well the model is working and adjust the weights (w and w_2) if needed. The error values are generated via the following equation.

$$E = -u_i + log \sum_{j=1}^{V} e^{u_j} \quad (3.4)$$

Where i is the index of the actual word (target word) in the output layer. For update weight using back- propagation, the error calculation is calculate by using (3.4). The Figure 3.7 shows the example calculation of error.

$$E = -u_i + log \sum_{j=1}^{V} e^{u_j}$$

= [[-0.97119998]
[0.03983734]
……
[0.0461251]
[0.03223174]]

Figure 3.7 Error Calculation

All of the weight from the hidden layer to the output will be updated during this back propagation phase.

$$new(w) = w_{old} - era * \frac{dE}{dw} \quad (3.5)$$

Where era is learning rate and $\frac{dE}{dw}$ is gradient of E with respect to w. The examples calculation for new_W2 and new_W1 are shown in Figure 3.8 and Figure 3.9 respectively.

```
new_W2 = W1- 0.01*(E*h)
       = [[0.5361149  0.2571294 ]
          [0.41722755 0.82353614]
          [0.00471886 0.12156912]
                ……
          [0.28589569 0.85239509]
          [0.97500649 0.88485329]]  [VxN]
```

Figure 3.8 new_W2 Calculation

```
new_W1 = W1- (E*W2)*x
       = [[0.40651016 0.59693097 …..  0.72906848
          0.33882533]
          [0.14659555 0.46126086 …… 0.60773789
           0.3071903 ]]  [NxV]
```

Figure 3.9 new_W1 Calculation

After numerous iterations of the foregoing training procedure, the final trained model comes with calibrated and correct weights. The initial weight matrix (W1) used as the vector for supplied text at the end of the entire training procedure.

(iv) Output: Using the trained model, calculate the word vector and located words that are comparable. The generated model is stored for further use.

Table 3.4 Word Vector Model Sample

Word	Vector
ချစ်	-0.32560984 1.11867797 -0.09987494 -0.8209551 -1.36967413
ခဲ့	-1.51370949 0.92668339 0.38922248 0.15598285 -0.28219419
ချစ်	-1.11045266 0.30133868 -0.07981049 -0.57563652 -1.576609
ဆဲ	-1.60189689 0.44186286 -0.0440396 0.10373312 -0.59121693
ဆက်	-0.50387149 1.26229947 -0.26336448 -0.5968973 -0.89291625
၍	-0.13901276 1.26913614 -1.10976729 0.06576435 -0.67299234
ချစ်	0.15791282 0.54127007 -1.39351359 0.4788503 -0.84090546
နေ	1.30776807 0.53134837 -0.92318887 -0.46437172 0.67842788
ပါ	0.7751386 0.60217019 -1.44338888 1.05625398 0.11118094
အုံး	0.01089158 0.04114116 -1.68917179 1.03375032 -0.34060809
မည်	0.3606493 -0.17061748 -0.05760646 1.46025745 -0.70067848
❤	-0.73451341 0.43140325 -0.58264506 1.57482271 0.13525838
happy	-1.08864612 0.04636047 -1.35070921 0.43803952 -0.76256094
birthday	-0.71689551 0.99674791 1.45580112 0.19111717 -0.93922606
par	-1.39014332 0.18661155 1.2713963 -0.1763288 -0.47301657
ချစ်	0.35225273 -0.01043937 1.22812116 -1.03713063 -0.60544426
တာ	0.02571915 -0.537943 0.746659 -1.34287798 0.06585692
ထက်ထက်	-0.6059665 -0.45823819 1.11768115 -0.43438305 0.53603584
ရယ်	0.14052032 -0.96470037 1.8020154 -0.49735982 0.29979796
ဒီ	0.58417849 -2.10416926 0.09336293 -0.06481671 -0.4740455
ရုပ်	0.30346595 -1.08649841 -0.06990328 -0.43729342 1.25114143
က	-0.94170679 -1.21463701 0.73184266 0.86321476 1.00437288
miss	0.17359791 -1.03943477 0.15382227 1.28033501 0.13356479
universe	1.20292087 -0.20893494 -0.30961637 -0.19594815 1.27150476
ဖြစ်ချင်	-0.36402664 -0.20201846 0.40126172 0.94269663 1.06146667
သေးတာ	0.14292913 0.17475731 0.43877767 0.48618855 1.01731504

3.5.3 TF-IDF (Term Frequency –Inverse Document Frequency)

The feature selection algorithm TF-IDF is used to remove superfluous words and increase the system's performance. TF-IDF's theoretical background is already covered in Chapter 2. The suggested system classified words that appeared twice in a phrase as feature words and assigned them a predetermined value. The TF-IDF weight typically consists of two terms: the first computes the normalized Term Frequency (TF) and the Inverse Document Frequency (IDF). The Inverse Document Frequency is computed as the logarithm of the number of documents in the corpus divided by the number of documents where the specific term appears; the first term is the number of times a word appears in a document which is divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF), which is computed as the logarithm of the number of documents in the corpus divided by the number of documents where the specific term appears. [25]

The applied system uses labeled documents whose category was already known to use for the training process, then calculate the weight of each word. First of all, to calculate the Term Frequency (TF), the system counts the number of the words and total number of words in the same documents. In the case of the term frequency TF, the simplest choice is to use the raw count of a term in a document. Then, it calculates how the term in its documents is important. Secondly, it measures how the importance of the term in the whole corpus. Finally, the system calculates the weight of the term by combining above two measures and store for the later use.

3.6 Classification

The goal of supervised learning is to train the process data on a specific pattern in order to identify the test part. The classification method is important in the field of sentiment analysis for training data on a pattern that may analyze for either a positive, negative, or neutral view. Logistic Regression (LR), Support Vector Machine (SVM), and Random Forest (RF) are the three Machine Learning classification approaches used in this system.

3.7 Overview Design of System

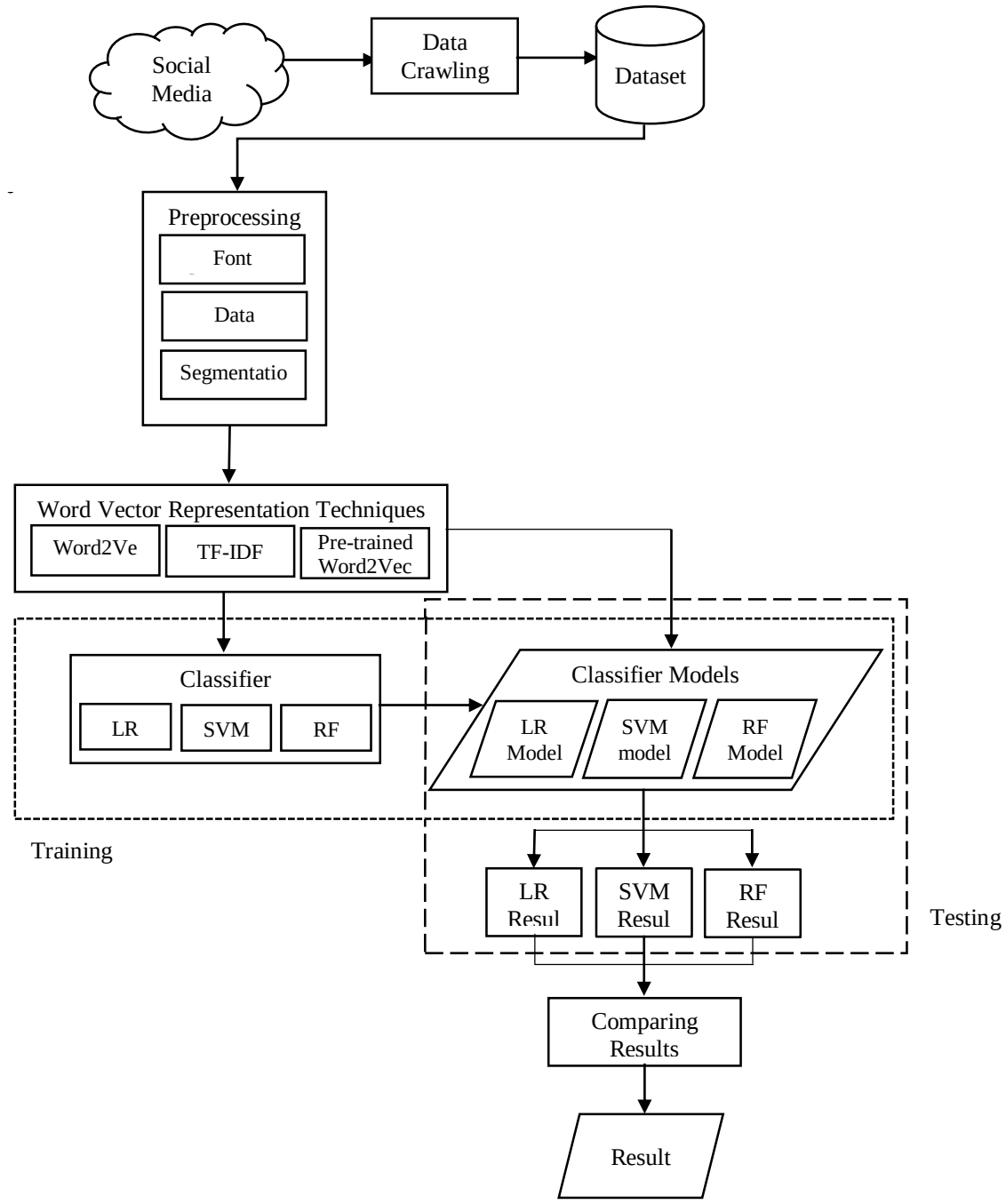

Figure 3.10 Overview Design of the System

In this Figure 3.10, the first step is data crawling. Facepager tool is used to collect data from social media. And then, these data contain various kind of text and noise. So it is need to preprocess. The details of preprocessing step are explained in chapter 3. The next step is to transform word vector by using word vector representation

techniques- Pre-trained Word2Vec, Word2Vec and TF-IDF for both training set and test set.

In training phase, Machine Learning classifiers are used. These Machine Learning classifiers are Logistic Regression, SVM and Random Forest. The classifier are used to train the result of word embedding which are generated from word vector representation techniques. The classifier models are generated.

In testing phase, the generated classifiers models are used to classify the test set's word vector. Finally, the best vector representation is chosen by comparing the performance results of three multiclass classification techniques.

3.8 Example of Sentiment Analysis

This section shows how to calculate the algorithm of a text classification system step by step using a small dataset as an example.

3.8.1 Data Preparation

In Myanmar's language, there has never been a dataset for public Facebook page comments. As a result, the dataset was created by crawling public comments on a Facebook page relating to "Myanmar Celebrity" in Myanmar. Data from social media (Facebook comments) is collected using the Facepager application.

Table 3.5 Statistics of Dataset

Data	Size (no. of sentences)	Type	Source
Training Data	31768		Myanmar Celebrity Facebook comments
Testset 1	2000	Close domain, Open Data	Myanmar Celebrity Facebook comments
Testset 2	2000	Open domain Open Data	Cosmetics Product review

3.8.2 Nature of Dataset

There is a total of 4 sentences in an example dataset and will be used as training documents and these dataset are collected from Myanmar Celebrity social media comments written in Myanmar or English Language and emotional sign. In preprocessing stage, the sentences are preprocessed explained in the section 3.3. The propose system only takes already segmented text as an input. The example dataset with label polarity shown in table 3.11 is used to explain the preprocessing stage. The labels of these 4 sentences are already known and after the calculation of the weight, the words are stored as weight or predefined value for their related category. The proposed system is going to classify documents into three categories such as Positive, Neutral and Negative. In the following example, the sentences class is shown and the calculated weight will be stored as vector for these classes.

s1.txt

Raw Sentences	**Segmentation Sentences**	**Polarity**
ချစ်ခဲ့ချစ် ဆဲ ဆက်၍ ချစ် နေ ပါ အုံးမည်❤	ချစ် ခဲ့ ချစ် ဆဲ ဆက် ၍ ချစ် နေ ပါ အုံး မည် ❤	Positive
Happy Birthdaypar	Happy Birthday par	Positive
ချစ်တာထက်ထက်ရယ်	ချစ် တာ ထက်ထက် ရယ်	Positive
ဒီရုပ်က Miss Universe ဖြစ်ချင်သေးတာ လား ဆုတောင်း နေလိုက်	ဒီ ရုပ် က Miss Universe ဖြစ်ချင် သေးတာ လား ဆုတောင်း နေလိုက်	Negative

Figure 3.11 Example Dataset with Label Polarity

3.8.3 Building Word Vector Representation

The sample dataset s1.txt from the first column of figure 3.11 are segmented manually as to prepare for the input into the sentences and segmented according to segment manually. The s1.txt is segmented into the second column of this figure. Within these words, the users predefines sentences as feature vectors to train with the word vector representation algorithm.

Table 3.6 Dataset for pre-trained Word2Vec

Data	Sentences
Training Set	33,568
Testing 1	2,000
Testing 2	2,000
UCSY Corpus [14]	52,016
	89,584

In this system, Word2Vec's pre-trained CBOW model was previously built using a dataset including 89,584 sentences. Among of them, 52016 sentences are taken to train from UCSY corpus [14]. In this enormous dataset, the left sentences represent the training and testing data. The pre-trained Word2Vec model is similarly created with the Word2Vec algorithm. To train, several hyper-parameters are required—the 300 dimensions of this model have been assigned. For CBOW models with ten workers, a context window of five was employed (the number of processor). The produced pre-trained word vector model has a vocabulary of 53,432 distinct words and 300 dimensions.

The proposed system uses CBOW model of the Word2Vec algorithm. The implementation of Word2Vec in the Gensim python library [35] was used to train on the treated data. For achieving the better implementation of Word Embeddings, most consideration is needed to give certain hyper-parameters. These parameters are: training dataset, dimensionality, context-window, minimum word count, sub-sampling and iteration. The training dataset trained 33,568 sentences of comments (training data). Negative sampling was set because it proved to be efficiently computational relative to hierarchical softmax. For resulting in the better Word Embeddings, the projection layer of the neural network was assigned 300. The context-window size was used 50 to prescribe context-window size for CBOW model. 1e-3 was used the sub-sampling rate to counter the imbalance dataset of frequent words. 1 set as the minimum count for considering each word in the set of sentences during processing of training. It trained Word2Vec model on the treated data with the above mentioned settings of hyper-parameter and stored in a data file format. The word embedding model produced d*p dimension word vectors where d is the dictionary size and p is the projection layer (hidden layer) size. In this study, the size of the dictionary for the training set is 13,395

vocabulary in the dictionary. The detail calculation of this study discussed in above section.

The last vectorization model is TF-IDF vectorizer. In this algorithm, the words are used as input to vector feature with minimum occurrence is 1.

Terms
birthday
happy
miss
par
universe
က
ခဲ့
ချစ်
ဆက်
ဆုတောင်း
ဆဲ
တာ
ထက်ထက်
ဒီ
နေ
နေလိုက်
ပါ
ဖြစ်ချင်
မည်
ရယ်
ရုပ်
လား
သေးတာ
အုံး
၍
❤

Figure 3.12 Collected Terms from Sample Document

The term frequency (TF) is the number of times and a term (word) occurs in a document.

$$tf(t) = \frac{\text{occurencies of a word in the document}}{\text{total words in the document}} \quad (2.1)$$

Inverse Document frequency (IDF) measures how important a term is for the corpus.

$$idf(t) = log\frac{n+1}{\text{df(t)}+1} + 1 \quad (2.2)$$

The concepts of term frequency and inverse document frequency are combined to produce a composite weight for each term in each document.

$$\text{TF-IDF} = \text{TF*IDF} \quad (2.3)$$

The following calculation shows each feature TF-IDF value is calculated:

Table 3.7 Term Frequency Table for Instance Dataset

Terms	D1	D2	D3	D4
birthday	0	0.333333	0	0
happy	0	0.333333	0	0
miss	0	0	0	0.1
par	0	0.333333	0	0
universe	0	0	0	0.1
က	0	0	0	0.1
ခဲ့	0.1	0	0	0
ချစ်	0.3	0	0.25	0
ဆက်	0.1	0	0	0
ဆုတောင်း	0	0	0	0.1
ဆဲ	0.1	0	0	0
တာ	0	0	0.25	0
ထက်ထက်	0	0	0.25	0
ဒီ	0	0	0	0.1
နေ	0.1	0	0	0
နေလိုက်	0	0	0	0.1
ပါ	0.1	0	0	0
ဖြစ်ချင်	0	0	0	0.1
မည်	0.1	0	0	0
ရယ်	0	0	0.25	0
ရုပ်	0	0	0	0.1
လား	0	0	0	0.1
သေးတာ	0	0	0	0.1
အုံး	0.1	0	0	0
၍	0.1	0	0	0
❤	0.1	0	0	0

The Table 3.8 shows the term frequency (TF) - the occurrence of the sample dataset.

Total no of sentences (all document which were used in training) – 4

Total no of term in a document – 26

Number of times terms t appear in a sentence – 1

Calculation of TF-IDF for words- birthday

TF-IDF (birthday),

- TF =1/3=0.333333

IDF = $\log_e (5/(1+1)) +1 = 1.916290$

TF-IDF =TF*IDF =0.333333*1.916290= 0.63

Table 3.8 Calculation of IDF

Term	IDF	Term	IDF
birthday	1.916291	ဒီ	1.916291
happy	1.916291	နေ	1.916291
miss	1.916291	နေလိုက်	1.916291
par	1.916291	ပါ	1.916291
universe	1.916291	ဖြစ်ချင်	1.916291
က	1.916291	မည်	1.916291
ခဲ့	1.916291	ရယ်	1.916291
ချစ်	1.510826	ရှပ်	1.916291
ဆက်	1.916291	လား	1.916291
ဆုတောင်း	1.916291	သေးတာ	1.916291
ဆဲ	1.916291	အုံး	1.916291
တာ	1.916291	၍	1.916291
ထက်ထက်	1.916291	❤	1.916291

Table 3.9 Calculation of TF*IDF

Terms	TF				IDF
	D1	D2	D3	D4	1.91629
birthday	0	0.33333	0	0	1.91629
happy	0	0.33333	0	0	1.91629
miss	0	0	0	0.1	1.91629
par	0	0.33333	0	0	1.91629
universe	0	0	0	0.1	1.91629
က	0	0	0	0.1	1.91629
ခဲ့	0.1	0	0	0	1.51083
ချစ်	0.3	0	0.25	0	1.91629
ဆက်	0.1	0	0	0	1.91629
ဆုတောင်း	0	0	0	0.1	1.91629
ဆဲ	0.1	0	0	0	1.91629
တာ	0	0	0.25	0	1.91629
ထက်ထက်	0	0	0.25	0	1.91629
ဒီ	0	0	0	0.1	1.91629
နေ	0.1	0	0	0	1.91629
နေလိုက်	0	0	0	0.1	1.91629
ပါ	0.1	0	0	0	1.91629
ဖြစ်ချင်	0	0	0	0.1	1.91629
မည်	0.1	0	0	0	1.91629
ရယ်	0	0	0.25	0	1.91629
ရုပ်	0	0	0	0.1	1.91629
လား	0	0	0	0.1	1.91629
သေးတာ	0	0	0	0.1	1.91629
အုံး	0.1	0	0	0	1.91629
၍	0.1	0	0	0	1.91629
❤	0.1	0	0	0	1.91629

Table 3.10 Result of TF-IDF

Terms	TF-IDF			
	D1	D2	D3	D4
birthday	0	0.63876	0	0
happy	0	0.63876	0	0
miss	0	0	0	0.19163
par	0	0.63876	0	0
universe	0	0	0	0.19163
က	0	0	0	0.19163
ခဲ့	0.19163	0	0	0
ချစ်	0.45325	0	0.37771	0
ဆက်	0.19163	0	0	0
ဆုတောင်း	0	0	0	0.19163
ဆဲ	0.19163	0	0	0
တာ	0	0	0.47907	0
ထက်ထက်	0	0	0.47907	0
ဒီ	0	0	0	0.19163
နေ	0.19163	0	0	0
နေလိုက်	0	0	0	0.19163
ပါ	0.19163	0	0	0
ဖြစ်ချင်	0	0	0	0.19163
မည်	0.19163	0	0	0
ရယ်	0	0	0.25	0
ရုပ်	0	0	0	0.19163
လား	0	0	0	0.19163
သေးတာ	0	0	0	0.19163
အုံး	0.19163	0	0	0
၍	0.19163	0	0	0
❤	0.19163	0	0	0

After calculating the TF-IDF method for predefined process, terms are restored by Machine learning in classification stage.

After calculating the weight of each term for each feature, the feature with the highest weight is chosen as the feature's weight or predetermined value. For the classification stage, these values are saved.

3.8.4 Classification of Sentence

During the classification stage, unknown sentence with unknown label are used to classify into the already predefined category. The input sentence is "အရမ်းချစ် idol".

This unknown sentence is segmented into following and considered as an input into the system.

အရမ်းချစ် idol

Figure 3.13 Segmentation of Input Sentence

This sentence is transformed into word vector by using the previous section of word vector representation method. Earlier in the process, predetermined document words are collected, and these words are employed as the beginning population.

Table 3.11 the Collected Sentence with Predefined Polarity

Segmentation Sentences	Polarity
ချစ် ခဲ့ ချစ် ဆဲ ဆက် ၍ ချစ် နေ ပါ အုံး မည် ❤	Positive
Happy Birthday par	Positive
ချစ် တာ ထက်ထက် ရယ်	Positive
ဒီ ရုပ် က Miss Universe ဖြစ်ချင် သေးတာ လား ဆုတောင်း နေလိုက်	Negative
အိုက် စ် . . . . မူး လိုက် တာ	Negative
ယေး	Neutral

This sentences are trained as the form of word vectors by using the three word vector representation techniques. The resulting vectors are also trained for classifying with the machine learning classifiers- Logistic Regression, SVM and Random Forest. The three machine learning classifiers are used to train the vector from resulting of Pre-trained Word2Vec, Word2Vec and TF-IDF vectorizer and stored the generating classification models for the further classification task.

CHAPTER 4

SYSTEM DESIGN AND IMPLEMENTATION

This chapter describes the system's design, implementation, and experimental results in details. The system is written in the Python programming languages, and the textual data is written in the Myanmar 3 font. The algorithm is developed on a dataset gathered from Myanmar's social media.

4.1 Design of the System

The proposed algorithm will categorize comments from Myanmar Celebrity Facebook page on Myanmar language into three categories: positive, neutral, or negative. The system is divided into two phases: training and testing.

For both training and testing stages, social media comments are collected using the Facepager application and manually preprocessed. The data are then tagged and saved as a training data set. And, pre-trained Word2Vec, Word2Vec, and TF-IDF vectorizer are used to transform word vector from the corpus. The real numbers of vector related to words are generated and stored the vector of each word for subsequent usage. The generated vector is then trained with three Machine Learning Classifiers (Logistic Regression, SVM, Random Forest) for classifying predefined polarities, and the classifier model is saved for later use.

In testing stage, the system can accept the news sentence that contains either text data or emotion. After accepting the input, the system performs the preprocessing step. The preprocessing consists of three steps including font conversion, data cleaning and word segmentation done manually. The segmented sentences are used to test. Word Vector Representation Algorithm is used to transform the vector related to each segmented word. And then, these vector, of each word are used to classify sentences with the stored training model of 3 classifiers which will generate the possible polarity of these sentences.

4.2 Implementation of the System

This section describes how to implement a comparative study of machine learning classifiers using Word2Vec for Myanmar social media comments starting from the stage of data collection to measurement of performance of the system.

4.2.1 Data Collection

For the implementation of this system, data resources are collected from Myanmar Celebrity Facebook's page and the comments of cosmetic page. This study will classify based on 3 categories for the research and they are listed as below.

i. Positive
ii. Neutral
iii. Negative

The experiment is conducted using data which are collected from Myanmar public Facebook page which contain comments for all pre-defined categories. The training set consists of over 33568 comments and test sets contain 2000 comments for each of close domain open data at the same Facebook page and open domain open data in the other cosmetic Facebook page. Both training and test data include Myanmar comments which are composed of pure text data with speech transcriptions and emotion.

Table 4.1 Document Collection for Training and Testing Data

Polarity	Training Data (# of sentences)	Testing1 Data (# of sentences)	Testing2 Data (# of sentences)
negative	8079	411	138
neutral	7535	321	1105
positive	17954	1268	757
Total	**33568**	**2000**	**2000**

The collected sentences include posts' comments in Myanmar and English Language which are composed of both pure text data and emotion and speech transcriptions.

4.2.2 Preparation for Training Dataset

First of all, the system uses to crawl the data from social media by using Facepager Tools. In Facepager Tools, the page ID add as a node to crawl the social media page. Then, the access token is filled by login in FB account. The data are fetched as a node from the Facebook page and export the node to the file. As three stages of the preprocessing, the system uses Myanmar Unicode encoding system. So, it is important to make sure that all the collected sentences are in Myanmar 3 Unicode font by using online Zawgyi-Unicode converter. Then, collected sentences are needed to preprocess for clean dataset. The last stage of the preprocessing stage is word segmentation. The proposed system will accept the input as the already segmented words. In this case, the word segmentation is done by the outside of the system and the sentences are segmented according to segment manually. And segmentation rule has been described in chapter 3.

4.2.3 User Interfaces Design of the System

The user interfaces of the applicable system "A Comparative Study of Machine-Learning Classifiers Using Word2Vec for Myanmar Social Media Comments" are shown in the following figure 4.1. The system's user interface is divided into two sections: training and testing. The system's Main Page is depicted in Figure 4.1.

Figure 4.1 Main Page of the System

In training phase, the training dataset is firstly transformed into vector related to the word. The system needs the label of the documents and documents to train in order to collect the feature vector according to their predefined categories. Figure 4.3 shows the training page of the system.

Figure 4.2 Training page of the system

In training process, the page will display train file selector to select text file to train. It converts vectors by using a selected the vector representation technique. These generated vectors are used to train classifier a selected classification method.

Figure 4.3 Training with File path

After choosing the text document file, transformation of vector with selected classifier method is used to train by using train button. The system will use the one of the selected word vector representation algorithm to transform the related word vector features and stored the related word vector feature model. These feature vector are used to train for classification text based on the predefined category. Back button will allow the user to go back to the Home page of the system.

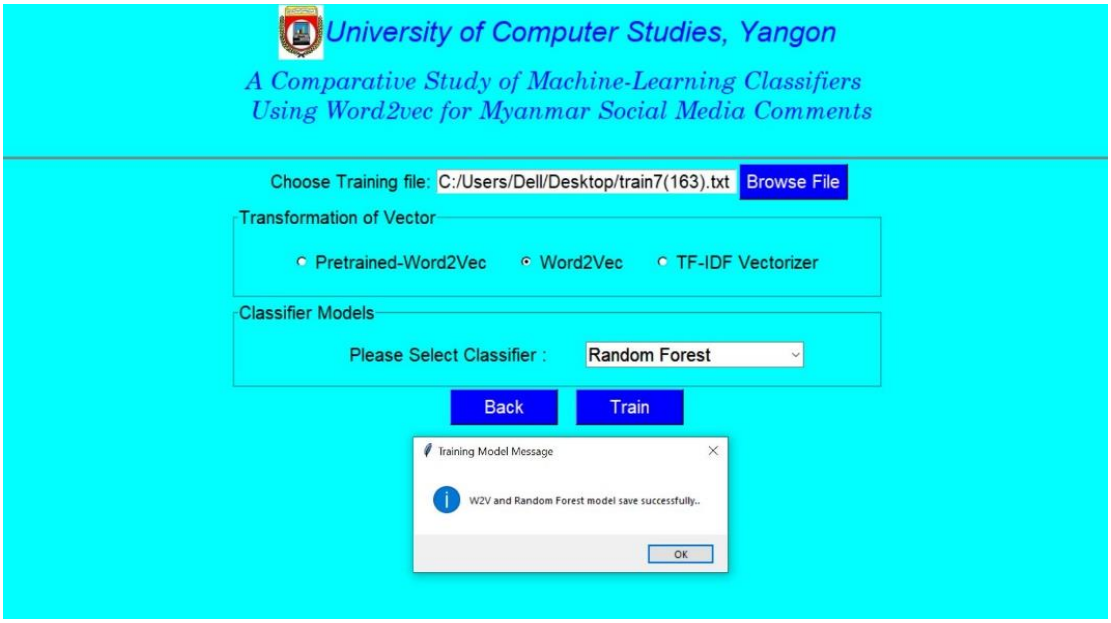

Figure 4.4 Page View after Training Documents with message box

After training the document file, it will show the message about the selected word vector transformation method and classifier models successfully train with the message box.

In classification phase, there are two ways – input file which contains the predefined labels is used for the evolution for comparing performance of the system, and the textbox which is used for the sentence of unknown label are utilized to classify.

Figure 4.5 shows the classification page of the applied system. The page contains the file selector to choose text file to classify and Back button will go back to the Main page of the applied system.

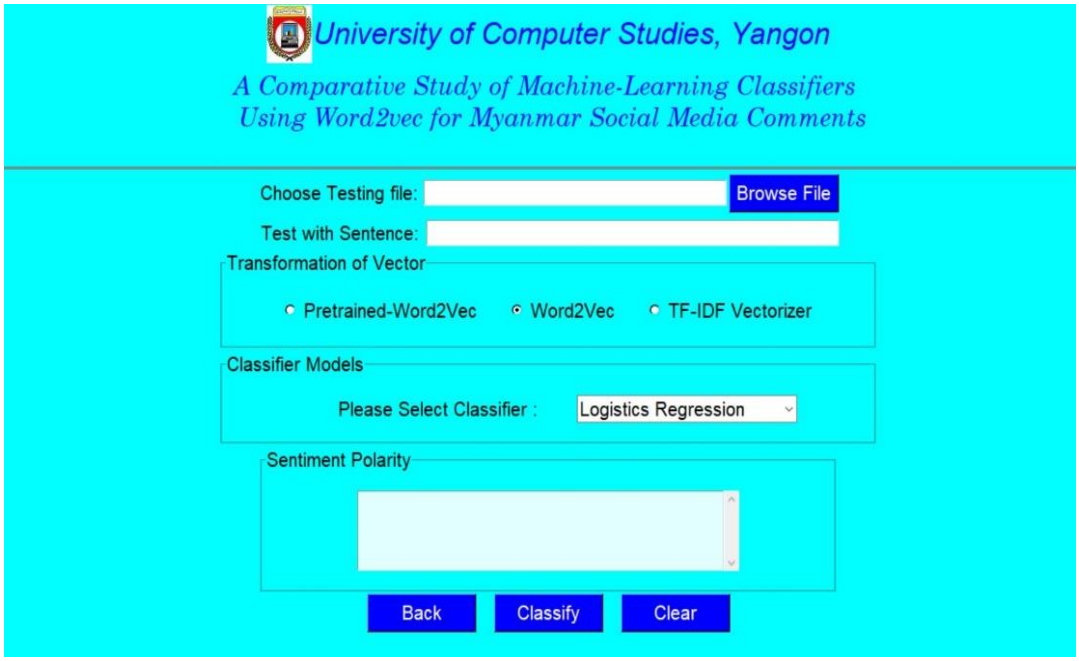

Figure 4.5 Classification Page

For the first way, the input file is selected from the browse file button. And then, the sentences from the input file will be transformed as vectors by one vectorizer, and the selected classifier is classified the polarity from the resulting vectors. The performance of this selected word embedding method and classifier is display as Figure 4.6.

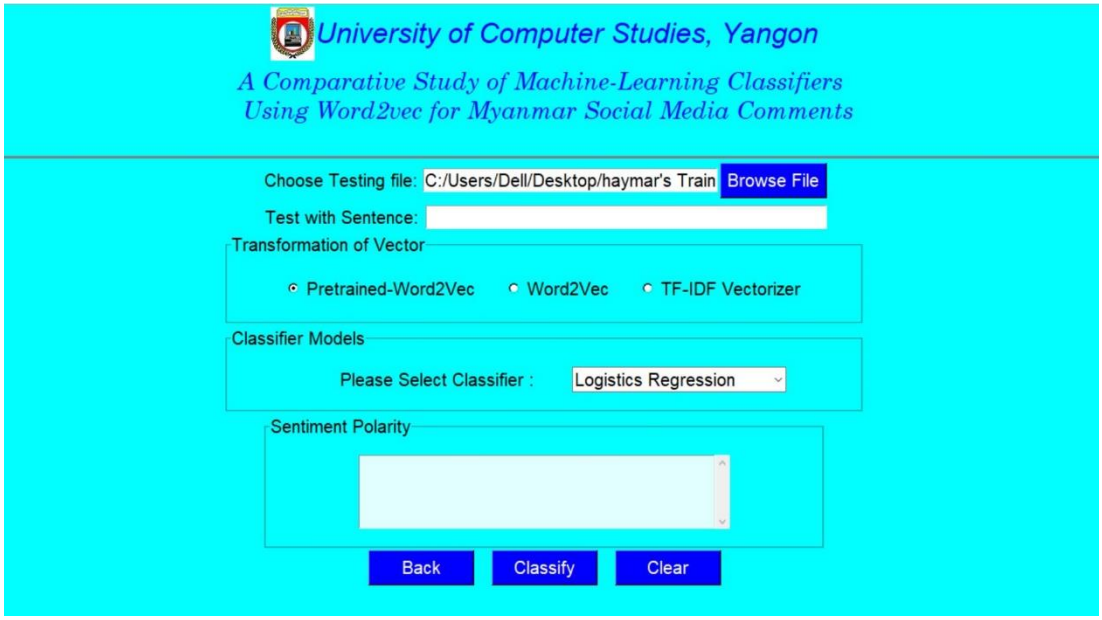

Figure 4.6 Classification Page for input file

After testing the document file, the result will show the message about the selected word vector transformation method and classifier models successfully test with message box. In figure 4.7 is shown.

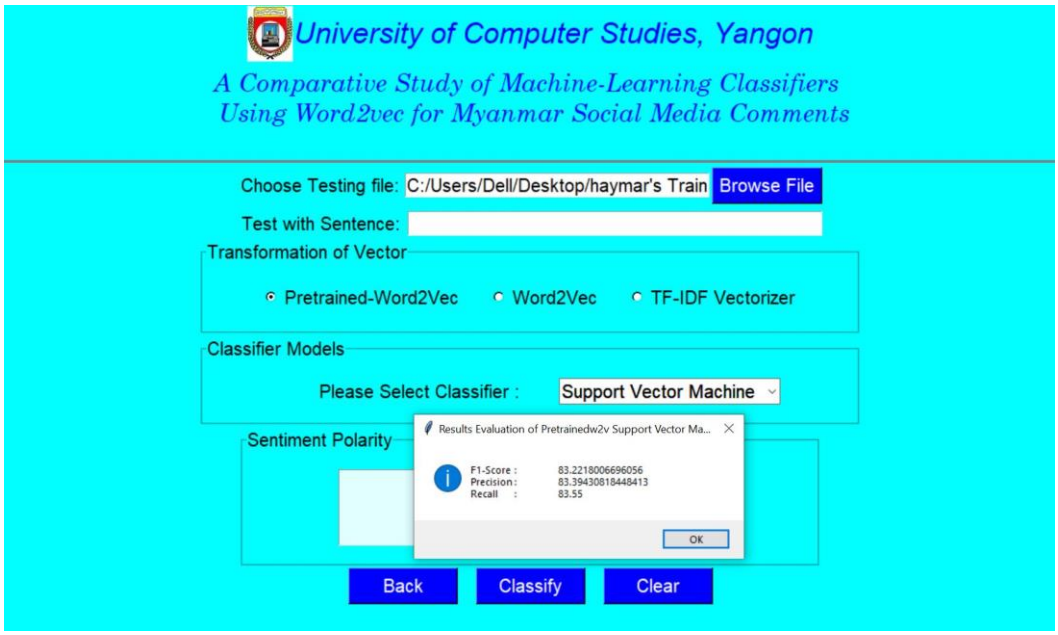

Figure 4.7 Displaying the Result for Classifying

After testing for three word vector representation techniques with the 3 classifiers, the comparative results will be displayed with bar chat bar by using view chat button. The result displays as the following figure 4.8.

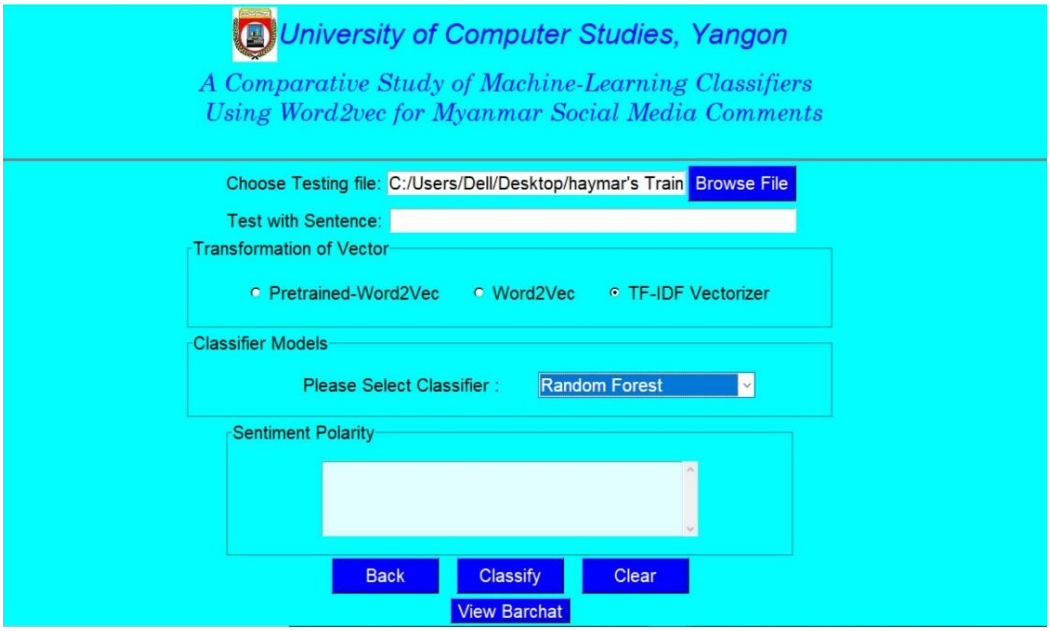

Figure 4.8 View Chats Page for Classifying

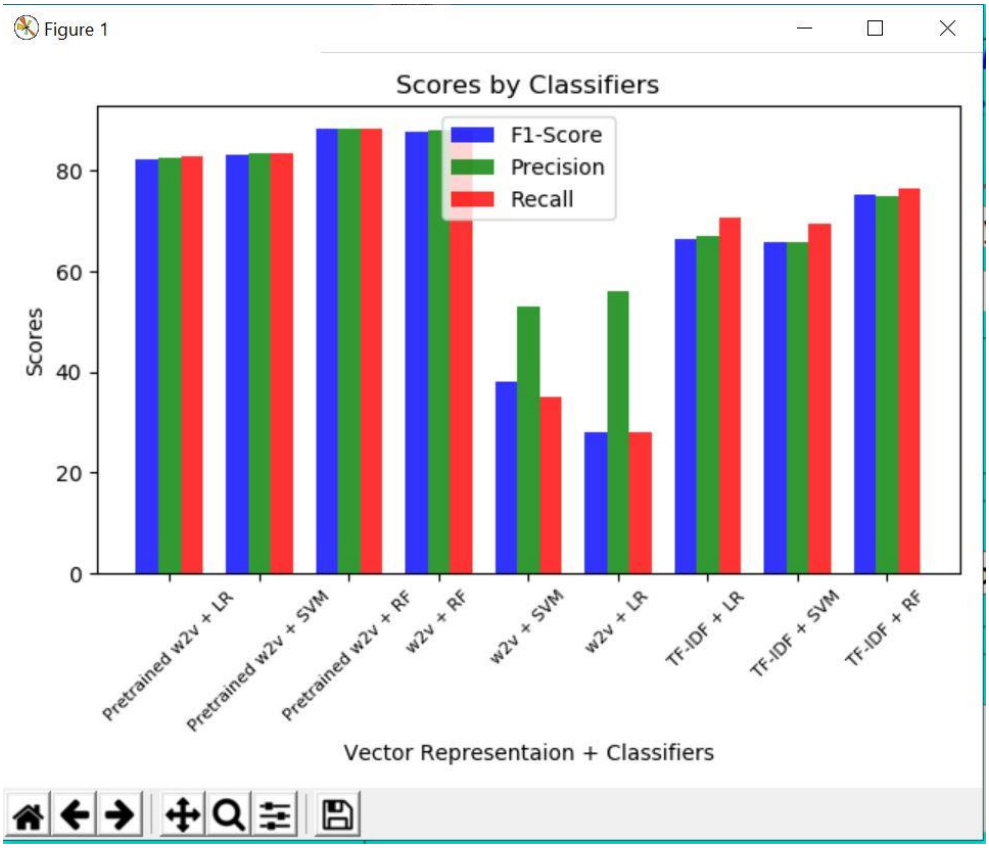

Figure 4.9 Displaying the Chat Page for Classification Performance

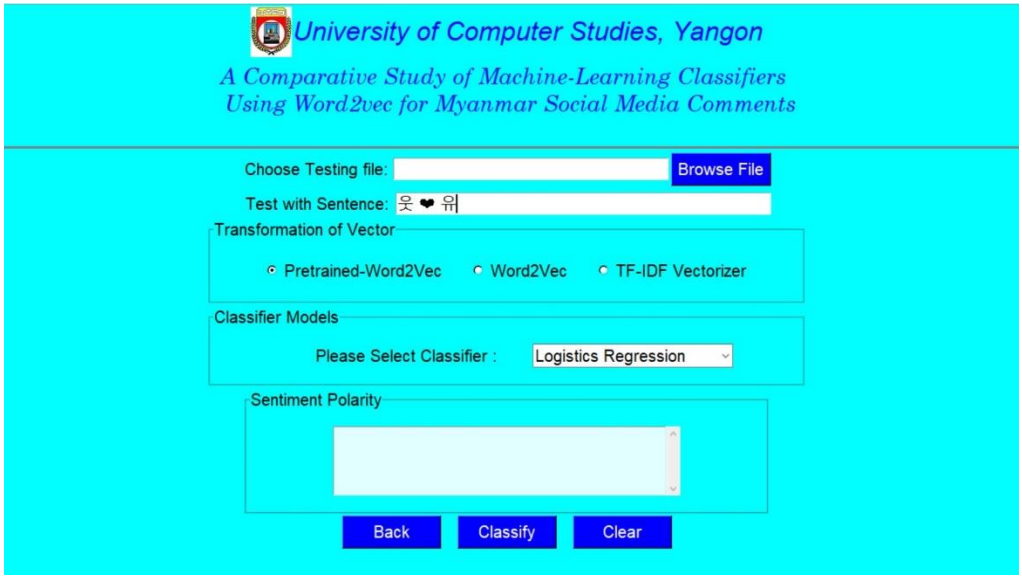

Figure 4.10 Classifying Sentence

In classification stage for sentence input, the system will classify the class of this sentence and then the classification sentence polarity is shown in the same frame

in its page as see in Figure 4.11. The system needs to provide text sentence to train and Back button can take back to the main page.

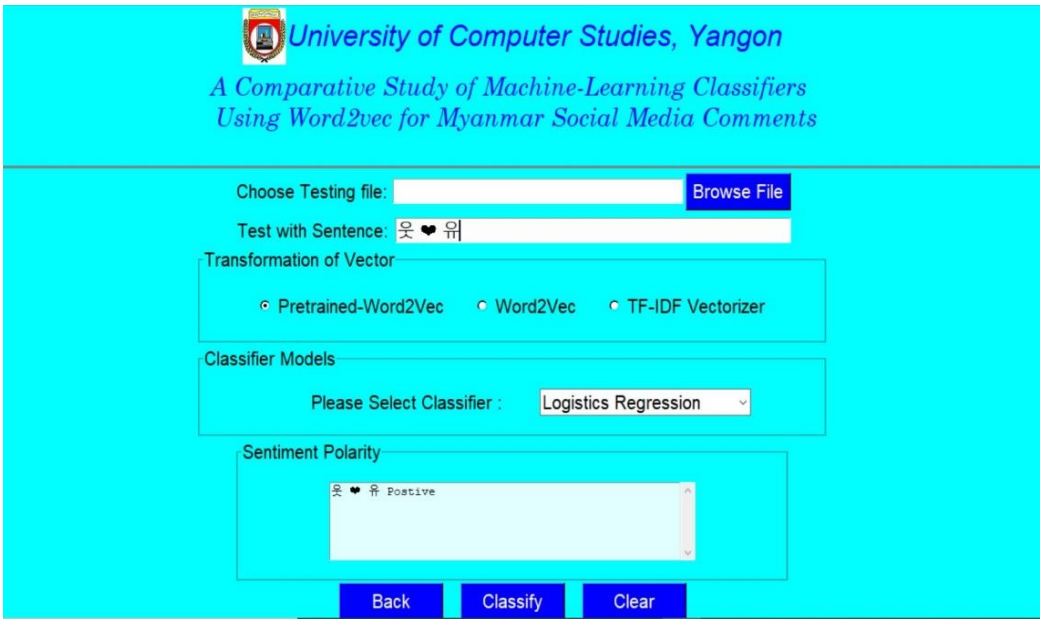

Figure 4.11 Display Label of the Input Sentence

Use Classify button will display the predicted category label in text area by using transformation of vector and classifier model to classify and Back button will take back to main page. Figure 4.10 shows the category of the provided sentence.

The context of text frame from sentiment polarity frame are cleared by using clear button. The following figure 4.12 shows the action clear button.

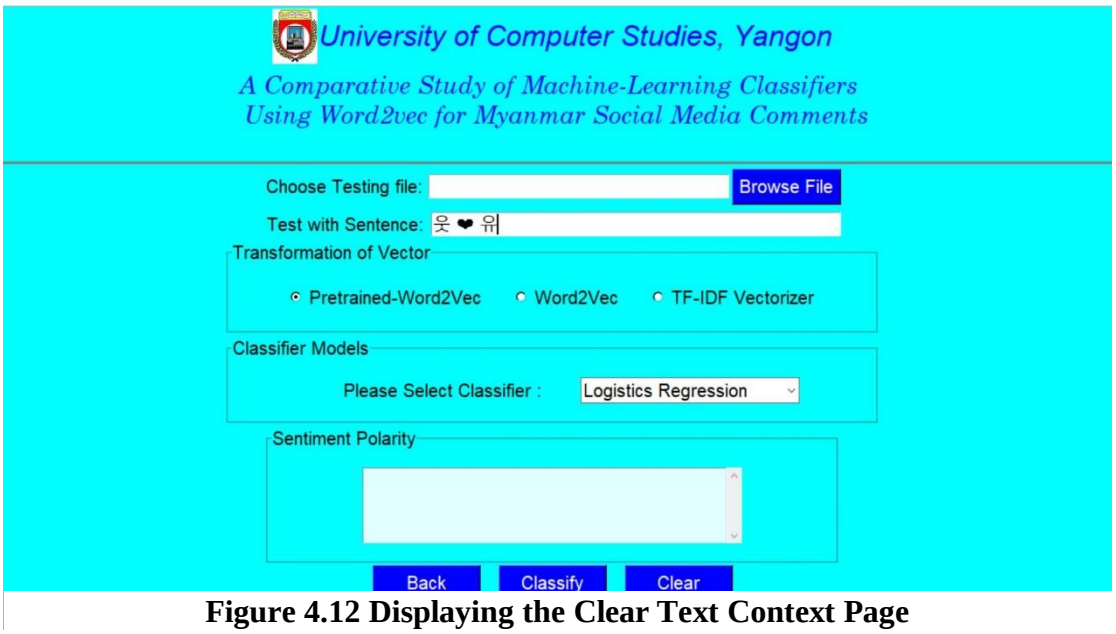

Figure 4.12 Displaying the Clear Text Context Page

4.2.4 Performance Criteria

Precision, recall, and F-measure are utilized as performance measurements for the test set in this system. Precision, recall, and F-measure for each class are calculated when measuring performance. The ratio of the number of successfully classified documents is the precision of a category for a test set in the text classification process. The number of correctly classified documents divided by the number of documents in that category in the training set is known as recall. The F1 score is a weighted average of the precision and recall scores. The following formulae are used to calculate them:

$$Recall = \frac{TP}{(TP+FN)} \quad (4.1)$$

$$Precision = \frac{TP}{(TP+FP)} \quad (4.2)$$

$$F1 - score = \frac{2*(Precision*Recall)}{(Precision+Recall)} \quad (4.3)$$

Where,

TP = true positive (the occurrences of targets actually identified sentences)

FP = false positive (the occurrences of targets that were not correctly identified sentences)

FN = false negative ((the occurrences of sentences which were not identified at all)

4.2.5 Experimental Result and Discussion

The experimental results are presented from word vector representation method with all three different techniques (Logistic Regression, Support Vector Machine, Random Forest) to classify three sentiment polarities of social media dataset. The data of testset2 is collected from the open domain open data – cosmetic pages' comments.

4.2.6 Experimental Results in Test set 1

The experiment performance in test set1 is shown in this section. The data of test set 1 are collected from the same domain open data – Myanmar Celebrities Facebook page's comments are same with training set domain but different from context data.

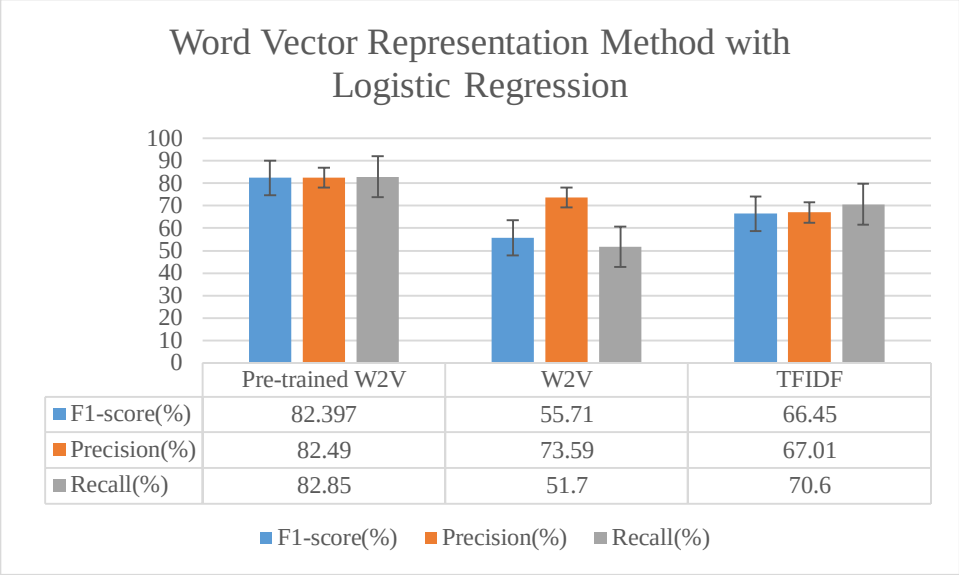

Figure 4.13 Experimental Result of Vector Representation with LG

In Figure 4.11, the result of three word vector representation techniques with the same Logistic Regression shows that the Pre-trained Word2Vec with Logistic Regression is the best performance in the other vector representation methods with the same classifier.

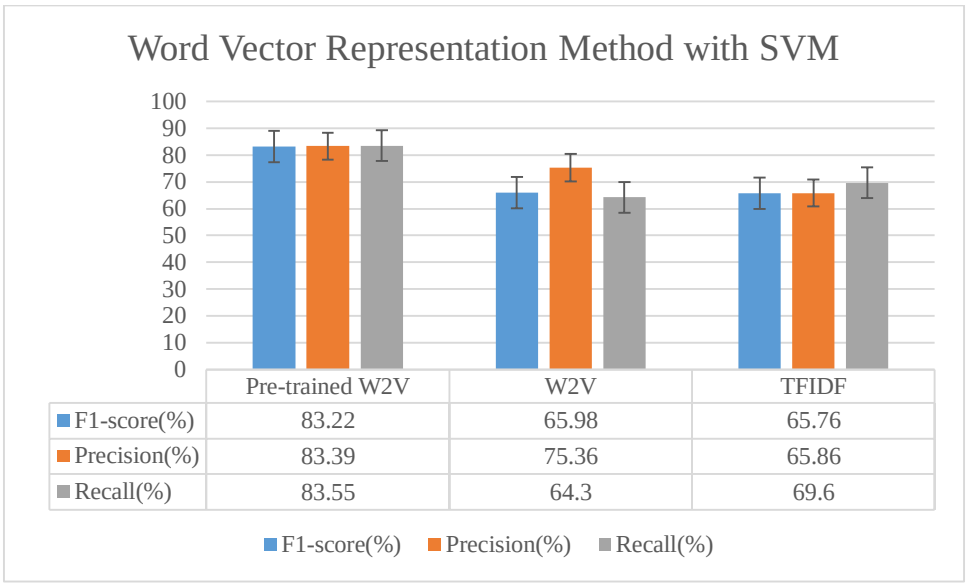

Figure 4.14 Experimental Result of Vector Representation with SVM

The second classifier in this experiment is SVM. The Figure 4.12 shows that the Pre-trained Word2Vec is the best result among other models (Word2Vec and TF-IDF vectorizer) with 83.22% F1-score, 83.39% precision and 83.55% Recall.

In Figure 4.13, the performance result displays that the pre-trained Word2Vec is the better performance than the remaining two vector representation techniques with 88.25% of F1-score, 88.9% of precision and 88.4% of recall.

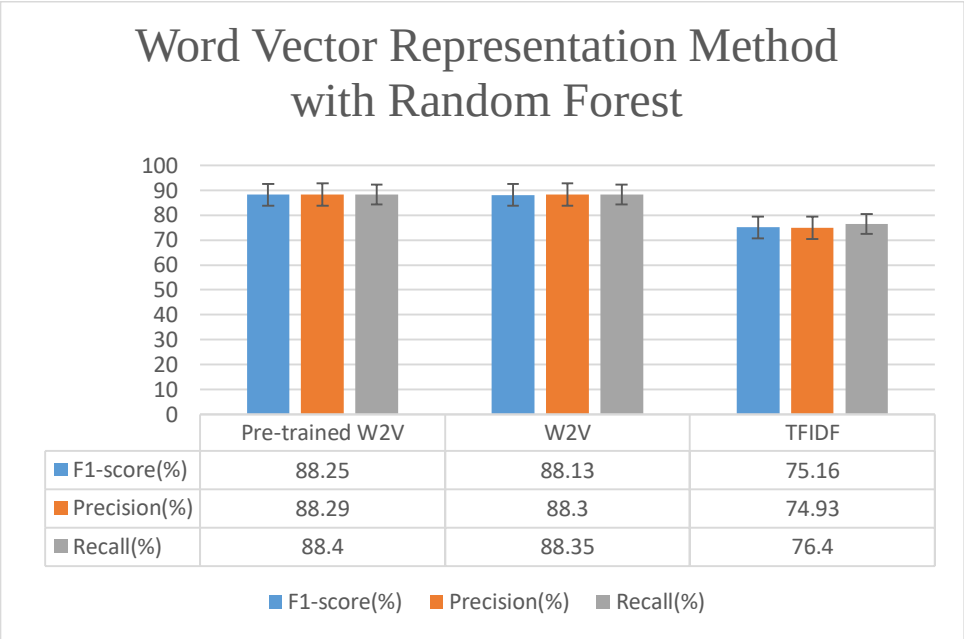

Figure 4.15 Experimental Result of Vector Representation with RF

Therefore, the best vectorization method in this experiment is the pre-trained Word2Vec in test set1.

4.2.7 Experimental Results in Test set 2

The experiment performance in test set1 is shown in this section. The data of test set 2 is collected from the open domain open data – cosmetic page's comments.

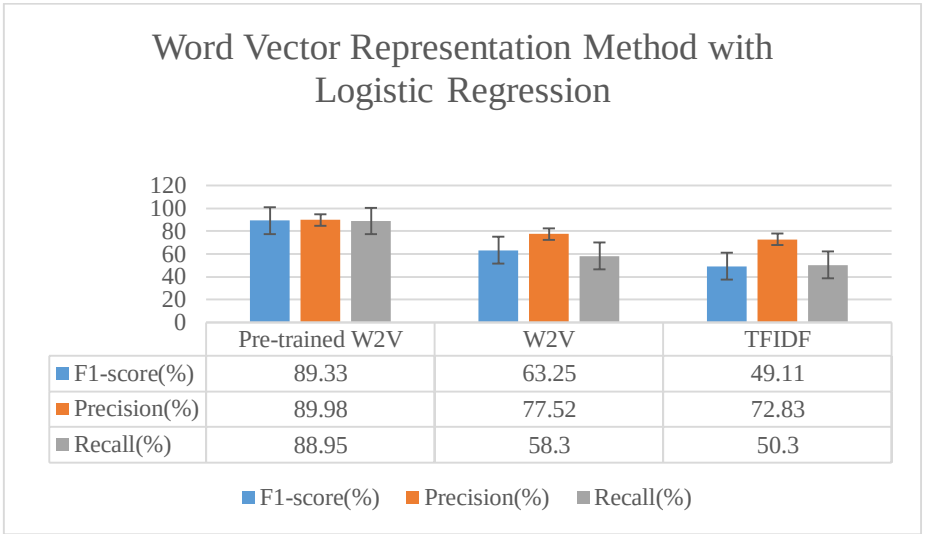

Figure 4.16 Experimental Result of Vector Representation with LG

In Figure 4.14, the pre-trained Word2Vec is shown that the maximum score of 89.33% F1-score, 89.98% precision and 88.95% recall among the three word vector representation techniques which classified the same classifier – Logistic Regression.

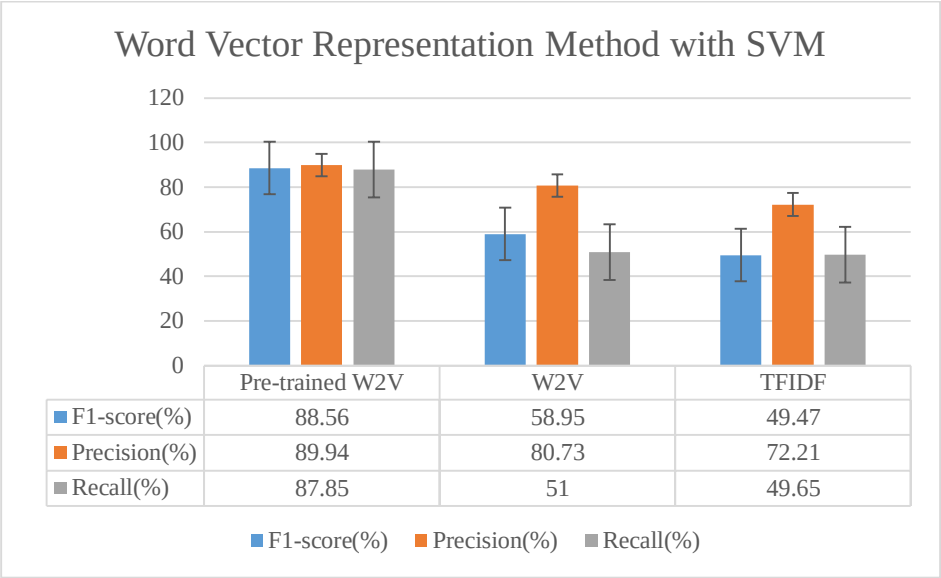

Figure 4.17 Experimental Result of Vector Representation with SVM

The test set2 classifies with the SVM classifier model. The performance of experiment is shown in figure 4.15. The pre-trained Word2Vec is also the better result than the other two vectorization models – Word2Vec and TF-IDF.

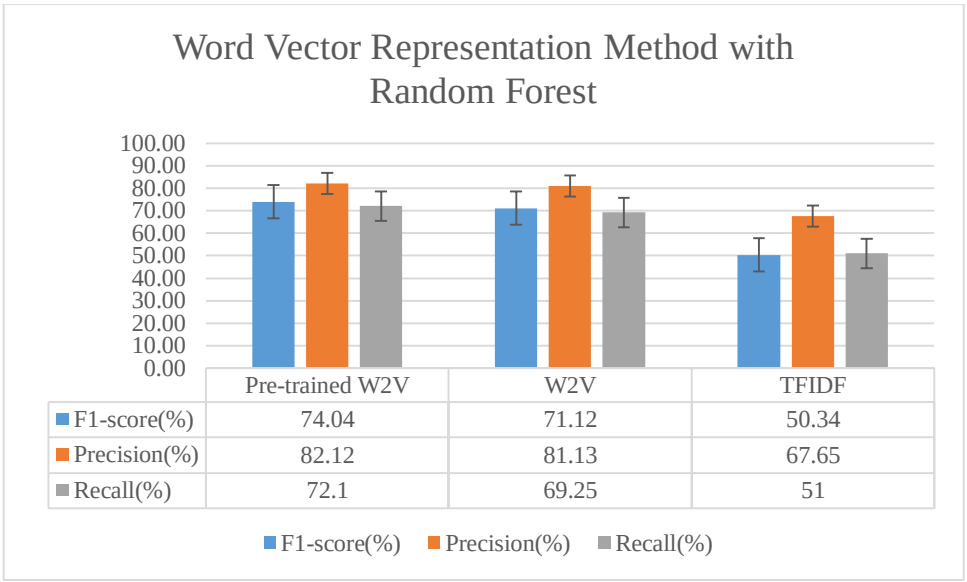

Figure 4.18 Experimental Result of Vector Representation with RF

The performance of Random Forest classifier with the three word vector representation methods are described in figure 4.16. The pre-trained Word2Vec is the

maximum result with 74.04% of F1-score, 82.12% of precision and 72.1% of recall by comparing the results of these vector representation method.

4.2.8 Results Discussion

The result shows that the maximum score work on both Pre-trained Word2Vec and Word2Vec with the Random Forest in Testset1. The Pre-trained Word2Vec with Logistic Regression classifier is the highest score in Testset2. Vectorization of pre-trained Word2Vec shows that it is better than the baseline method- TF-IDF for Testset1 and Test set 2.

Table 4.2 Summary of Experimental Results of all Score

Classifiers	Test Set	Pre-trained Word2Vec			Word2Vec			TF-IDF		
		F1-score (%)	Preci-sion (%)	recall (%)	F1-score (%)	Preci-sion (%)	recall (%)	F1-score (%)	Preci-sion (%)	recall (%)
Logistic Regress-ion	T1	82.40	82.49	82.85	55.71	73.59	51.7	66.45	67.01	70.6
	T2	89.33	89.98	88.95	63.25	77.52	58.3	49.11	72.83	50.3
SVM	T1	83.22	83.39	83.55	65.98	75.36	64.3	65.76	65.86	69.6
	T2	88.56	89.94	87.85	58.95	80.73	51	49.47	72.21	49.65
Random Forest	T1	88.25	88.29	88.4	88.13	88.3	88.35	75.16	74.93	76.4
	T2	74.04	82.12	72.1	71.12	81.13	69.25	50.34	67.65	51

Table 4.2 shows the score for each word vector representation techniques with the same three machine learning classifiers. The comparison of score of different classifiers on the social media comments dataset indicates that pre-trained Word2Vec algorithms outperform the TF-IDF vectorization techniques. Among the three machine vectorization algorithms, the pre-trained Word2Vec algorithm outperforms Word2Vec and TF-IDF.

CHAPTER 5
CONCLUSION

This study describes the experiment of the comparative study of machine learning classifiers using word vector representation for Myanmar Social Media comments. Sentiment analysis, word vector representation technique and classification techniques are discussed in detail in Chapter 2. Theoretical details of the system are described in Chapter 3 and implementation details are illustrated in Chapter 4. Since this system applies unsupervised learning for vector representation method and supervised classification method for text classification system in Myanmar text data and emotion, it has some weaknesses that need to be considered to solve in future.

Sentiment analysis' application to the massive volume of unstructured data has become a major research topic. Now, corporate groups and academia are collaborating to develop the finest sentiment analysis technology. Although some of the algorithms employed in sentiment analysis produce positive results, no technique can solve all of the issues. Comments from Myanmar online media Facebook page are collected manually and labeled. Then, these comments are preprocessed and stored as a training corpus and testing dataset. The system has been tested using real-world comment data collected from reliable Myanmar media Facebook page. In order to vectorize Myanmar social media comments, a text classifier based on pre-trained Word2Vec model with classifier is applied to the certain corpus for classification task. The performance of the system is measured by Precision, Recall and F-l measure methods. In both test sets, the pre-trained Word2Vec outperforms more than the baseline vectorizer TF-IDF. The result of pre-trained Word2Vec with Random Forest Classifier model outperforms in terms of 88.25% of F1-score, 88.29% of precision and 88.4% of recall in test set 1. In test set 2, the experiment result of pre-trained Word2Vec with Logistic Regression Classifier model is the best performance with 89.33% of F1-score, 89.98% of precision and 88.95% of recall. It has been found out that high-quality feature word vector can improve the performance of the system. According to testing results, pre-trained Word2Vec is the best of word vector representation technique. It can be concluded by saying that if the training dataset contain the more high-quality data, the performance of the system will also increase.

5.1 Advantages and Limitations of the System

The system can be applied in many real world applications in Myanmar Language. The main advantage of using Word2Vec CBOW model is very easy to understand because it is probabilistic by nature, it is generally expected to outperform deterministic approaches. It has a modest memory capacity. It does not require as much RAM as a co-occurrence matrix, which requires the storage of three large matrixes. The Word2Vec model is adjust multi-dimensional vector with the best performance in the other TF-IDF don't adjust the dimensions and depends on the occurrence of word. Although it takes time to compute for calculations, to learn unnecessary feature vectors, and to apply all feasible solutions for a better result, it is well worth the effort.

Due to learn unnecessary feature vectors, this system do not work well the text paragraph for sentiment analysis. Word2vec CBOW model, on the other hand, requires a large amount of training data to provide good results. Because the quality of the feature vector is dependent on the segmentation of words and the nature of the words, the more training data the system has, the better it performs. As a result, the improved word vector may be obtained when additional words are introduced to the word segmentation program. The amount of training data and an issue with word segmentation are to blame for some of the system's failures. Another drawback is that the system only works with well-defined labels, and for faulty data, no bounded label should be considered. To improve the performance of the classification task, the suggested system can be modified by adding more data to the training dataset.

5.2 Future Extensions

The proposed system is tested by using only the dataset with social media comments. Facebook page data always change in time by time. Therefore, the dataset can be extended with more text. The word vectors from these system will be extended in many application areas such as recommender systems, text recognition, speech to text conversion, summarization system etc.

Publications

[1] Hay Mar Su Aung, Win Pa Pa, "Analysis of Word Vector Representation Techniques with Machine-Learning Classifiers for Sentiment Analysis of Public Facebook Page's Comments with Myanmar Text", to be published in the Proceedings of the 18th International Conference on Computer Application (ICCA 2020), Yangon, Myanmar, 2020

REFERENCES

[1] Aye Myat Mon, Khin Mar Soe. "Clustering Analogous Words in Myanmar Language using Word Embedding Model", ICCA & ICFCC 2019 Conference, 27th February, 2019.

[2] C. Emelda. "A comparative Study on Sentiment Classification and Ranking on Product Reviews", ijirae, issn: 2349-2163, volume 1 issue 10, november, 2014.

[3] Chaya Liebeskind, Karine Nahon, Yaakov HaCohen-Kerner, Yotam Mano," Comparing Sentiment Analysis Models to Classify Attitudes of Political Comments on Facebook (November 2016)", issn 2395-8618

[4] Christos Troussas, Maria Virvou, Kurt Junshean Espinosa, Kevin Llaguno, Jaime Caro," Sentiment analysis of Facebook statuses using Naive Bayes classifier for language learning", conference Paper • July 2013 DOI: 10.1109/IISA.2013.6623713. https://www.researchgate.net/publication/261497806

[5] Dr.Khin Mar Soe, Dr. Khin Thandar Nwet, Aye Hnin Khine,"Automatic Myanmar News Classification System", conference paper,15th International Conference on Computer Applications (ICCA 2017), Yangon, February 2017.

[6] Kumar Saurav , Kumar Saunack, Diptesh Kanojia , and Pushpak Bhattacharyy," "A Passage to India": Pre-trained Word Embeddings for Indian Languages", arXiv:2112.13800v1 [cs.CL] 27 Dec 2021

[7] Joshua Acosta, Norissa Lamaute, Mingxiao Luo, Ezra Finkelstein, and Andreea Cotoranu. "Sentiment Analysis of Twitter Messages Using Word2Vec". Proceedings of Student-Faculty Research Day, CSIS, Pace University, May 5th, 2017.

[8] Merfat M. Altawaier, Sabrina Tiun, "Comparison of Machine Learning Approaches on Arabic Twitter Sentiment Analysis", volume.6 (2016) no. 6, ISSN: 2088-5334

[9] Md. Al- Amin, Md. Saiful Islam, Shapan Das Uzzal, "Sentiment Analysis of Bengali Comments With Word2Vec and Sentiment Information of Words", 978-1-5090-5627-9/17/$31.00 ©2017 IEEE.

[10] Mr. S. M. Vohra, 2 Prof. J. B. Teraiya," A Comparative Study Of Sentiment Analysis Techniques", Journal Of Information, Knowledge And Research In

Computer Engineering", issn: 0975 – 6760| nov 12 to oct 13 | volume – 02, issue – 02.

[11] Oscar B. Deho, William A. Agangiba,Felix L. Aryeh, Jeffery A. Ansah. "Sentiment Analysis with Word Embedding". https://www.researchgate.net/publication/332091221

[12] Radha Guha , "Exploring the Field of Text Mining", International Journal of Computer Applications (0975 – 8887) Volume 177 – No.4, November 2017.

[13] Soe Yu Maw, May Aye Khine, "Aspect based Sentiment Analysis for travel and tourism in Myanmar Language using LSTM", ICCA & ICFCC 2019 Conference Program Schedule 27th February, 2019.

[14] Yi Mon Shwe Sin and Khin Mar Soe, "Large Scale Myanmar to English Neural Machine Translation System". Proceeding of the IEEE 7th Global Conference on Consumer Electronic (GCCE 2018).

[15] Yu Mon Win Myint, War War Cho, Tin Nilar Win, "Sentiment Analysis on Customer's Comments of Myanmar Cosmetic Products", Journal of Computer Applications and Research, Volume 1, No 1, 2020

[16] https://chrisyeh96.github.io/2018/06/11/logistic-regression.html

[17] https://developers.google.com/machine-learning/crash-course/logistic-regression/calculating-a-probability

[18] https://hackernoon.com/word-embeddings-in-nlp-and-its-applications-fab15eaf7430

[19] https://machinelearningmastery.com/what-are-word-embeddings/

[20] https://medium.com/data-science-group-iitr/word-embedding-2d05d270b285

[21] https://medium.com/@shiivangii/data-representation-in-nlp

[22] https://medium.com/@social_67526/tf-idf-vs-word2vec-vectorization-techniques-for-twitter-sentiment-analysis-8c8cfc5a0820

[23] https://monkeylearn.com/sentiment-analysis/

[24] https://stats.idre.ucla.edu/r/dae/multinomial-logistic-regression/

[25] https://stats.stackexchange.com/questions/353552/support-vector-machine-calculate-w-by-hand

[26] https://www.techopedia.com/definition/29695/sentiment-analysis

[27] https://towardsdatascience.com/from-word-embeddings-to-pretrained-language-models-a-new-age-in-nlp-part-2-e9af9a0bdcd9

[28] https://towardsdatascience.com/learn-word2vec-by-implementing-it-in-tensorflow-45641adaf2ac
[29] https://towardsdatascience.com/understanding-logistic-regression-step-by-step-704a78be7e0a
[30] https://towardsdatascience.com/updated-text-preprocessing-techniques-for-sentiment-analysis-549af7fe412a
[31] https://www.analyticsvidhya.com/blog/2017/06/word-embeddings-count-word2veec/
[32] https://www.analyticsvidhya.com/blog/2021/06/how-does-backward-propagation-work-in-neural-networks/
[33] https://www.datascience.com/resources/notebooks/random-forest-intro
[34] https://www.guru99.com/word-embedding-word2vec.html
[35] https://pypi.org/project/gensim/
[36] https://www.rabbit-converter.org/Rabbit/

Appendix

1. Identify of Context word and target word for the following four sentences sample corpus.

ချစ် ခွဲ ချစ် ဆဲ ဆက် ၍ ချစ် နေ ပါ အုံး မည် ❤

Happy Birthday par

ချစ် တာ ထက်ထက် ရယ်

ဒီ ရုပ် က Miss Universe ဖြစ်ချင် သေးတာ လား ဆုတောင်း နေလိုက်

****

Target word:ချစ် .

Target vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ခွဲ', 'ချစ်'] .

Context vector: [1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ခွဲ .

Target vector: [0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ချစ်', 'ချစ်', 'ဆဲ'] .

Context vector: [1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ချစ် .

Target vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ခွဲ', 'ချစ်', 'ဆဲ', 'ဆက်'] .

Context vector: [1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ဆဲ .

Target vector: [0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ချစ်', 'ခဲ့', 'ဆက်', '၍'] .

Context vector: [1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ဆက် .

Target vector: [0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ဆဲ', 'ချစ်', '၍', 'ချစ်'] .

Context vector: [1. 0. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:၍ .

Target vector: [0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ဆက်', 'ဆဲ', 'ချစ်', 'နေ'] .

Context vector: [1. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ချစ် .

Target vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['၍', 'ဆက်', 'နေ', 'ပါ'] .

Context vector: [0. 0. 0. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:နေ .

Target vector: [0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ချစ်', '၍', 'ပါ', 'အုံး'] .

Context vector: [1. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ပါ .

Target vector: [0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Context word:['နေ', 'ချစ်', 'အုံး', 'မည်'] .

Context vector: [1. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:အုံး .

Target vector: [0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Context word:['ပါ', 'နေ', 'မည်', '❤'] .

Context vector: [0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:မည် .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Context word:['အုံး', 'ပါ', '❤', 'happy'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:❤ .
Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Context word:['မည်', 'အုံး', 'happy', 'birthday'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:happy .
Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Context word:['❤', 'မည်', 'birthday', 'par'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:birthday .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['happy', '❤', 'par', 'ချစ်'] .

Context vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:par .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['birthday', 'happy', 'ချစ်', 'တာ'] .

Context vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ချစ် .

Target vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['par', 'birthday', 'တာ', 'ထက်ထက်'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:တာ .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ချစ်', 'par', 'ထက်ထက်', 'ရယ်'] .

Context vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ထက်ထက် .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['တာ', 'ချစ်', 'ရယ်', 'ဒီ'] .

Context vector: [1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ရယ် .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ထက်ထက်', 'တာ', 'ဒီ', 'ရုပ်'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ဒီ .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ရယ်', 'ထက်ထက်', 'ရုပ်', 'က'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0. 0.]

****

Target word:ရုပ် .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ဒီ', 'ရယ်', 'က', 'miss'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0.]

****

Target word:က .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0. 0.]

Context word:['ရုပ်', 'ဒီ', 'miss', 'universe'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0. 0. 0.]

****

Target word:miss .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0. 0.]

Context word:['က', 'ရုပ်', 'universe', 'ဖြစ်ချင်'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0. 0.]

****

Target word:universe .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]

Context word:['miss', 'က', 'ဖြစ်ချင်', 'သေးတာ'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0. 0.]

****

Target word:ဖြစ်ချင် .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0. 0.]

Context word:['universe', 'miss', 'သေးတာ', 'လား'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0. 0.]

****

Target word:သေးတာ .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0. 0. 0.]

Context word:['ဖြစ်ချင်', 'universe', 'လား', 'ဆုတောင်း'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1. 0.]

****

Target word:လား .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0. 0.]

Context word:['သေးတာ', 'ဖြစ်ချင်', 'ဆုတောင်း', 'နေလိုက်'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1. 1.]

****

Target word:ဆုတောင်း .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 0.]

Context word:['လား', 'သေးတာ', 'နေလိုက်'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0. 1.]

****

Target word:နေလိုက် .

Target vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1.]

Context word:['ဆုတောင်း', 'လား'] .

Context vector: [0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 1. 1. 0.]