A Project Thesis on

Short Text Emotion Detection Using Multiclass SVM

A thesis

submitted in fulfillment of the requirement for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

In

COMPUTER SCIENCE AND ENGINEERING

by

A. VEERA VENKATA SAI RAJU

[Reg. No. 18021F0025]

Under the Supervision of

Smt. S.S.S.N USHA DEVI N

Assistant Professor

CSE, UCEK

JNTUK, Kakinada.

Submitted to

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING UNIVERSITY COLLEGE OF ENGINEERING KAKINADA (A) JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA-533003

[2018 – 2021]

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY COLLEGE OF ENGINEERING KAKINADA (A) JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY KAKINADA-53300

CERTIFICATE

This is to certify that the thesis entitled **“SHORT TEXT EMOTION DETECTION USING MULTICLASS SVM”** being submitted by **VEERA VENKATA SAI RAJU ATUKURI** bearing the Roll number **18021F0025** in fulfilment for the award of degree of **MASTER OF COMPUTER APPLICATIONS** in **COMPUTER SCIENCE AND ENGINEERING** to UCEK, JNTUK, Kakinada, Andhra Pradesh, India, is a record of bonafide work carried out by him under the guidance and supervision of the Department.

Head of the Department	**Project Supervisor**
Dr. D. HARITHA	**Smt. S.S.S.N USHA DEVI N**
Professor & HOD CSE, UCEK	Assistant Professor CSE, UCEK
JNTUK, Kakinada	JNTUK, Kakinada.

ACKNOWLEDGEMENT

I wish to express my sincere gratitude to my project guide **Smt. S.S.S.N USHA DEVI N, UCEK** for providing me with an opportunity to implement my project on “**Short Text Emotion Detection using Multiclass SVM”**. I am grateful for giving me valuable advice and constant source of encouragement throughout the project. Without her valuable inputs and continuous suggestions, this project would not have been possible. Finally, I thank my family and friends for their support.

Veera Venkata Sai Raju Atukuri

TABLE OF CONTENTS

LIST OF FIGURES

Short Text Emotion Detection Using Multiclass SVM

Abstract

Emotions capture the essence of the communication process between people and electronic communication systems. Detecting Emotions such as joy, anger, sadness, fear, and the improves the computer-generated response process for users. Recognizing this type of emotion from a human-made text plays a vital role in applications such as chat conversations, customer support forums, customer reviews, and studying a user's psychology.

Emotion detection is a multi-class classification problem. Here, the sentence may contain the co-occurrence of words related to more than one emotion; hence, classification is challenging. The existing system uses sampling algorithms to deal with multi-class data. However, it affects the time complexity of the system. We found Support Vector Machine (SVM) has a better solution to deal with multi-class data. SVM scales relatively well on multi-class data. We apply the vectorization process to convert text data into numerical feature vectors for training our SVM model. In the proposed system, our objective is to map the short text statement to a specific emotion.

Chapter 1

INTRODUCTION

Nowadays humans are interacting the machines because of advancement of technology. They communicating to machines in different formats such as text, audio and video. Predicting the emotion of the user when they communicating to machines, helps them to improve the user experience. Machines will be able to communicate accordingly based on the user emotion. It helps service providers provide tailor-made services to their customers.

Human emotions can be identified by using the text being generated by the user during the communication. Humans exhibits the emotions such as joy, sadness, anger, shame, guilt and fear. Every emotion has some special keywords associated with it. Based on the current frame of mind of a user, he uses certain keywords, these keywords help us to predict the user emotion. humans use these words to express their emotions. Emotions and their associated keyword are mapped by finding the underlying relations between them.

1.1 EMOTION DETECTION

Emotion detection (ED) is a branch of sentiment analysis that deals with the extraction and analysis of emotions. These emotion lexicons contain emotion search words or keywords such as happy, hate, angry, sad, surprise, and so on. The task is to find occurrences of these search words in a written text at the sentence level. Once the keyword is identified within the sentence, a label is assigned to the sentence.

For instance, if the constructed emotion dictionary contains joy and the written text from which emotion is to be determined reads “I was filled with so much joy on seeing my mother for the first time in five years,” the sentence is emotionally labelled with the keyword joy. This approach though simple and straightforward faces challenges, including the need for an emotion dictionary to contain reasonable number of emotion categories, since limited keywords can greatly affect the performance of the approach among ambiguity of keywords and the lack of linguistic information.

For Emotion Detection from a machine learning model different parameter should be taken into consideration. Various types of techniques are used to detect emotions from a human being like facial expressions, body movements, blood pressure, heart beat and textual information.

This project focuses on the emotion detection from textual information. Emotion Detection in text documents is essentially a content - based classification problem involving concepts from the domains of Natural Language Processing as well as Machine Learning. In this paper emotion recognition based on textual data and the techniques used in emotion detection are discussed.

In this project proposes emotion detection in short texts using multiclass SVM. It detects the seven emotions joy, Sadness, guilt, shame, fear, anger, disgust.

Chapter 2

LITERATURE REVIEW

2.1 LITERATURE

Youg-Jun Lee, Chan-Yong Park and Ho-Jin Choi in the paper "**Word-Level Emotion Embedding based on Semi-Supervised Learning for Emotional Classification in Dialogue**" [1] discussed emotion classification in dialogue based on semi-supervised word-level embedding. They used the NRC Emotion Lexicon which is list of English words and their associations with eight basic emotions. They added word-level emotion vectors to obtain utterance-level emotion vectors.

Shadi Shaheen, Wassim El-Hajj, Hazem Hajj and Shady Elbassuoni in the paper "**Emotion Recognition from Text Based on Automatically Generated Rules**" [2] proposed a framework that can recognize the emotions present in the communication or the emotions of the involved user to improve the user experience. In this work they considered syntactic and semantic structure of sentence and then generalized it by representation using wordNet and ConceptNet, which will create an emotion recognition rule (ERR).

Haji Biali, Chen Wu and Vidyasagar Potdar in the paper "**Computational Approaches for Emotion Detection in Text**" [3] discussed various emotion detection theories that provide a basics for emotion models. It shows how those models have been used in computational approaches to emotion detection. They used the SVM algorithm for validating the proposed architecture.

Dr. Sattar B. Sadkdhan and Ahmed Dheyaa Radhi in the paper "**Fuzzy Logic used in Textual Emotion Detection**" [4] discussed how the fuzzy logic can be used to detect emotion subjects from textual data. In this project, fuzzy logic was to convert the non-value member to value member. They developed a system using fuzzy logic and sentiment analysis to classify emotions represented in text.

Maruf Hassan, Mb. Sakib Bin Alam and Tanveer Ahsan in the paper "**Emotion Detection from Text using Skip-through Vectors**" [5] discussed a way of finding emotion from text by using the lexical approaches and machine learning techniques. In their work they used deep learning model named skip-thought, an approach to learning fixed length representations of sentences, to face the problem of emotion detection from text. In this project they showed skip-thought vectors were well suited for emotion detection task.

Khodijah Hulliyah, Normi Sham Awag Abu Bakar and Amelia Riathani Ismail in the project "**Emotion Recognition and Brain Mapping for Sentiment Analysis: A Review**" [6] discussed the computational linguistic areas that are interested in the attention of emotion for Sentiment Analysis. Sentiment Analysis observes the emotion conveyed by a text, and at same time, distinguishing positive and negative valence. This paper provides the overview of past and recent research on emotion detection as well as some approaches and techniques used and shows the linked between both Sentiment Analysis and Emotion Recognition.

Harpreet Kaur, Veenu Mangat and Nidhi in the project "**A Survey of Sentiment Analysis Techniques**" [7] discussed about emotion extraction using text mining. How to find the polarity of the text and classify in into positive, negative or neutral. It helps in human decision making. This paper presents the survey of main approaches used for sentiment classification.

Sophia Yat Mei Lee and Zhongqing Wang in the paper "**Multi-view Learning for Emotion Detection in Code-switching Texts**" [8] they emphasised on analyzing emotions in monolingual text, neglecting the fact that emotions are aften found in bilingual or code-switching posts in social media. They used a multi-view learning framework to learn and detect the emotions through both monolingual and bilingual views. In this project, the monolingual views are extracted from the monolingual text separately, and the bilingual view is constructed with both monolingual and translated text collectively. Empirical studies demonstrate the effectiveness of their proposed approach in detecting emotions in codeswitching texts.

Zhije Liu, Xueqiang Lv, Kum Liu and Shuicai Shi in the paper "**Study on SVM Compared with the other Text Classification Methods**" [9] discusses the applications of Support Vector Machine in text categorization. They introduced the basic principle of SVM and described the process of text classification. They showed SVM-based classification model was effective for text classification using machine learning.

A. Mathur and G. M. Foody in the paper "**Multiclass and Binary SVM Classification: Implications for Training and Classification Users**" [10] discussed how binary SVM can be extended for a one-shot multiclass classification needing a single optimization operation. In this project one-shot multiclass classification of multispectral data was more accurate than the approaches based on a series of binary class classification.

Muljono, Nurul Anisa Sri Winarshi and Catur Supriyanto in the paper "**Evaluation of Classification Methods for Indonesian Text Emotion Detection"** [11] evaluates the performance of four different classification methods: Naïve Bayes, J48, K-Nearest Neighbor and Support Vector Machine-Sequential Minimum Optimization. In this project they used Indonesian text corpus, containing 1000 sentences which consists of six emotions. They concluded that SVM-SMO classifier gives the best performance.

Tapasy Rabeya, Sanjida Ferdous, Himel Suhita Ali and Narayan Ranjan Chakraborty in the paper "**A Survey on Emotion Detection: A Lexicon Based Backtracking Approach for Detection form Bengali Text**" [12] discussed how emotions are detected from different textual data. In this project In case of lexicon-based analysis, the position of emotional lexicons really varies the state of an emotion. They presented an emotion detection model to extract emotions from Bengali text, considered two basic emotions happiness and sadness. This lexicon based backtracking approach has been introduced for recognizing the sentiments of a sentence to show how frequently people express their emotions in the last part of a sentence. Their proposed method produced a result with 77.16 accuracies.

Fika Hastarita Rachaman, Riyanatarto Sarno and Chastine Fatichah in the paper "**CBE : Corpus Based of Emotion Detection in Text Detection**" [13] discussed about forming automatic emotional corpus-Based of Emotion(CBE). CBE developed from Wordnet Affect Emotion and Affective Norms for English Words with term similarity measure and distance of node approach. Latent Dirichlet Allocation (LDA) is used too for automatically expand CBE. CBE attributes are a score of Valence (V), Arousal (A), Dominance (D) and categorical label emotion. Categorical label emotion based on six basic emotions of Ekman.

Zhaorang Zong and Changhun Hong is the paper "**On Application of Natural Language Processing in Machine Translation**" [14] discussed the natural language processing of statistical corpora with neural machine translation and concludes the natural language processing: Neural Machine translation has the advantage of deep learning, which is very suitable for dealing with the high dimension, dimension, label-free and big data of natural language, therefore, its application is more general and reflects the power of big data and big data thinking.

Feng Tian and Husian Zhang, Longzhuang Li, Qinghua Zheng and Yang Yang in the paper "**Visualizing e-Learner Emotion, Topic, and Group Structure In Chinese Interactive Texts**" discussed how e-learner's emotion combined with topics and group structure. For achieving this goal, they used a colour palette of emotions based on plutchiks's colour palette was presented, an extended cascaded PLSI algorithm using sliding window technique was proposed to detect and track topics in Chinese interactive texts, and multiple star-field variants were introduced to display the group structure.

2.4 PROBLEM STATEMENT

Emotion detection from short text is multiclassification problem. Using the labelled corpus generate co-occurrence of words and convert the dataset into word embeddings. Support Vector Machine (SVM) scales relatively well on multi-class data. Build the model using the SVM and train the model with vectorized dataset. Predict the emotion embedded in the short texts.

Chapter 3

PROPOSED SYSTEM

3.1 INTRODUCTION

Predicting the human emotion based on the text is a multiclass problem, because the machine learning model has to map every word to their corresponding emotions. Mapping the words to the emotions is difficult. The text might contain unwanted words such as stop words, different words but with the same meaning and same words with different context. It is challenging for a model form a word dictionary for every emotion.

Proposed system uses the Support Vector Machine (SVM) classifier to predict the emotion involved in a short text message. We found SVM has a better solution to deal with multi-class data. Support Vector Machine (SVM) scales relatively well on multi-class data.

Before training the model, data must be brought into a form that is predictable and analyzable for training the model. So, pre-processing the dataset is required. It helps in building the more accurate model and improves the model performance.

After preprocessing the text, we convert the text into numerical form by using word embedding techniques. Every line of text in the corpus is converted into a numerical vector. First the text is converted into a dictionary and word count format using the counter container and then counter objects are converted into numerical vectors by using the word embeddings. Dictvectorizer is used in this project to convert the counter objects into the numerical vectors. We train the SVM models with numerical data.

3.2 WORKFLOW

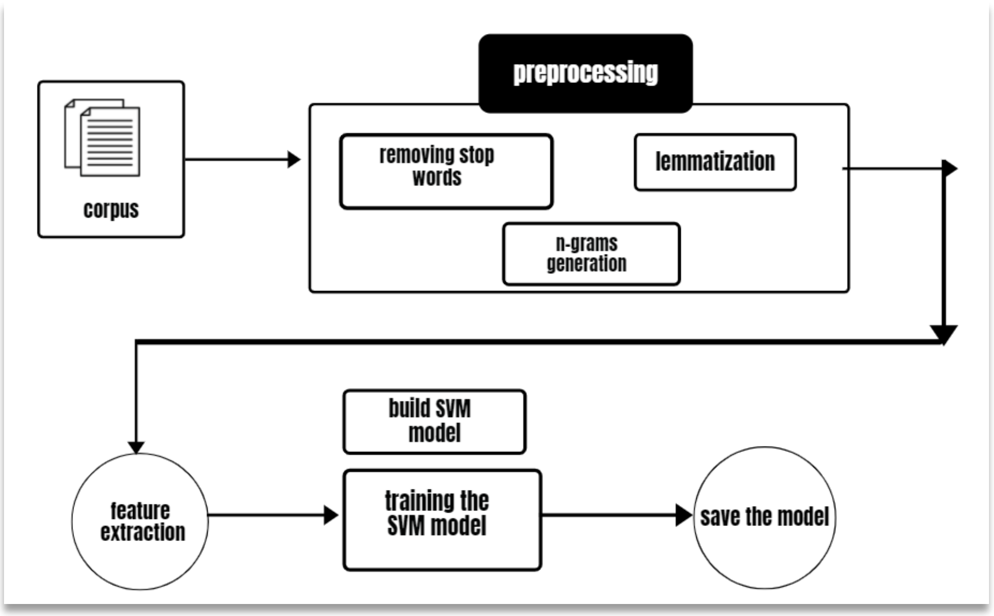


Fig 3.1 Training Phase

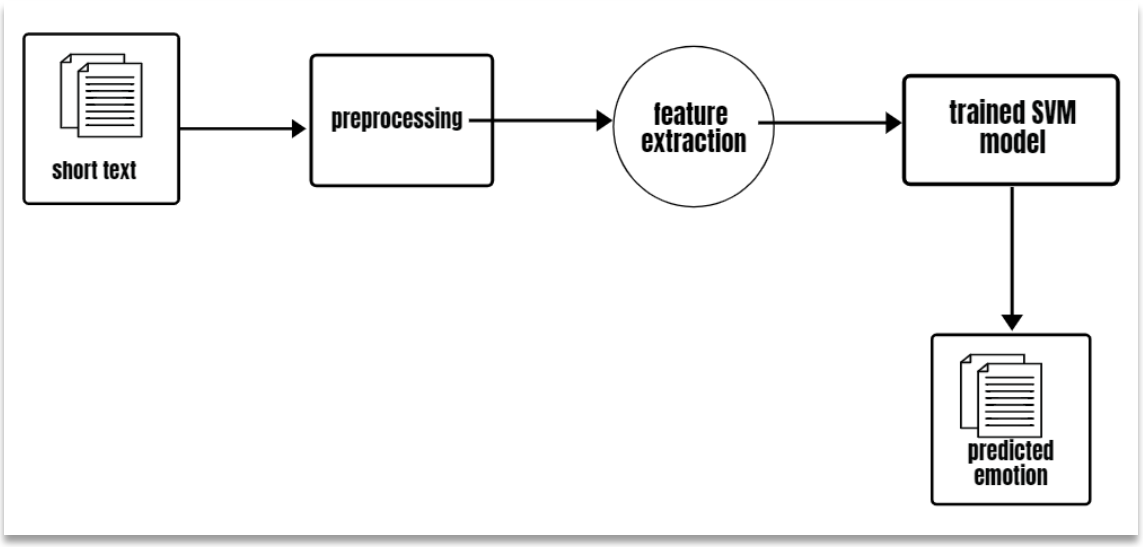


Fig 3.2 Testing Phase

3.3 MODULES

This project is divided into four modules.

1) Preprocessing
2) Feature Extraction
3) Building Model
4) Model Evaluation

➢ PREPROCESSING

Preprocessing is a stage where the data is brought into a form which used for training the model. predictable and analyzable for training the model. So, pre-processing the dataset is required. It helps in building the more accurate model and improves the model performance.

Text pre-processing is performed in the following procedure

- Converting the text into lower case letters
- Removing the stopwords.
- Removing the special character from the text [!@#$%^.......].
- Lemmatizing the text.
- N-grams generation [co-occurring words].

➢ FEATURE EXTRACTION

Machine learning algorithms cannot work with raw text directly, the text must be converted into numerical vectors. Converting the text data into numerical form is called feature extraction or feature encoding. Feature extraction is used extract and produce feature representations that are appropriate for the type of Natural Language Processing (NLP) task we are trying to accomplish and the type of model you are planning to use.

To train the model data must be in the numerical form. Since, our corpus is in the text form we need to convert the text into a numerical. This project converts the text in the corpus to numerical form using the following techniques

▪ Bag of Words

It is a way of representing text in the feature extraction phase. It is used to develop a bag-of-words model for collection of documents and helps to use different techniques to prepare a vocabulary and score words. In this project we convert each line of text in the dataset into container objects. Each container object stores the text data in dictionary format. Key is the word and value is the count of occurrence of the corresponding word in a given text.

- Vectorization

  These container objects are still not in the trainable format. So, these objects are converted to numerical vectors using the vectorization technique where each vector represent emotional text in the corpus.

  Now, the dataset is in the required format to train the model.

➢ BUILDING MODEL

Machine learning model is built by training the model with dataset. During the training process the machine learning model finds the underlying relationships in the dataset and acquires the expertise to make the prediction on the data it has never seen.

The model must be trained with the adequate amount of data, otherwise it results in the model underfitting or overfitting and it affects the accuracy of the model. We must perform the preprocessing on the data to ensure the quality in the dataset.

Before training the model, we must split the dataset for testing and training. In this project dataset is split into 80:20, 80% of dataset is used for training the model and 20% dataset is used for evaluating the model. The samples in the dataset are shuffled prior to splitting the dataset.

To build the model, we create an SVM model by using the predefined packages in the sklearn module. Read the dataset and split into two parts. And then we start training the model with training dataset.

➢ MODEL EVALUATION

Model Evaluation is an integral part of the model development process. It helps to find the best model that predicts accurately on the unseen samples. In this project 20% of the dataset is used to evaluate the model performance.

In this project after training the SVM model, the model performance is measured by using accuracy.

$$\text{Accuracy} = \text{Correct Predictions} / \text{Total Predictions}$$

Accuracy is defined as the percentage of correct predictions for the test data. It can be calculated easily by dividing the number of correct predictions by the number of total predictions.

➢ SUMMARY

In this Project dataset is preprocessed for improving the performance of the model. It generates the co-occurrence of words using n-grams technique and text is converted into vectors by using word embeddings. Dataset divided into two parts for training and testing. We build a Multiclass SVM model and we train the model using training dataset. Performance of the model is evaluated by using the testing dataset. This model predicts the seven emotions for short texts, they are joy, sadness, shame, guild, anger, disgust, fear.

Chapter 4

MODULE IMPLEMENTATION

Nowadays, we are having huge amount of text data is being generated by users over the internet. Exploring the emotions in the text which exits in large amount by using the human intervention takes lots of resources and time. Hence, we need an efficient technology to handle the huge amounts of data to explore the emotions. Machine learning is technology which trains a model with data and able to use its previous experience to predict the outcome of the future data. Every machine learning model is implemented in a programming language. In this project, we are using the python programming language.

Python combines the power of general-purpose programming languages with the ease of use of domain-specific scripting languages like MATLAB or R. Python has libraries for data loading, visualization, statistics, natural language processing, image processing, and more. This vast toolbox provides data scientists with a large array of general- and special-purpose functionality. One of the main advantages of using Python is the ability to interact directly with the code, using a terminal or other tools like the Jupyter Notebook. Machine learning and data analysis are fundamentally iterative processes, in which the data drives the analysis. It is essential for these processes to have tools that allow quick iteration and easy interaction. As a general-purpose programming language, Python also allows for the creation of complex graphical user interfaces (GUIs) and web services, and for integration into existing systems.

Python libraries used in this project

Scikit-learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

NumPy: Base n-dimensional array package

SciPy: Fundamental library for scientific computing

Matplotlib: Comprehensive 2D/3D plotting

IPython: Enhanced interactive console

Sympy: Symbolic mathematics

Pandas: Data structures and analysis

NumPy

NumPy is one of the fundamental packages for scientific computing in Python. It contains functionality for multidimensional arrays, high-level mathematical functions such as linear algebra operations and the Fourier transform, and pseudorandom number generators.

Pandas

Pandas is a popular Python package for data science, and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The DataFrame is one of these structures.

Matplotlib

Matplotlib is a Python library used for plotting beautiful and attractive Graphs. With the help of this library, we can plot 2D and 3D graphs. In data science it is very important to visualize the data. Data visualization helps to understand the data with graphs, and help to find the correct model to fit.

NLTK

The Natural Language Toolkit (NLTK) is a platform used for building Python programs that work with human language data for applying in statistical Natural Language Processing (NLP).

Programming Tool

Jupyter Notebook

The Jupyter Notebook is an interactive environment for running code in the browser. It is a great tool for exploratory data analysis and is widely used by data scientists. While the Jupyter Notebook supports many programming languages, we only need the Python support. The Jupyter Notebook makes it easy to incorporate code, text, and images.

4.1 PREPROCESSING

Before training the machine learning model, the data must be preprocessed. Preprocessing is a stage where the data is brought into a form which used for training the model. predictable and analyzable for training the model. So, pre-processing the dataset is required. It helps in building the more accurate model and improves the model performance.

Preprocessing steps

1. **Load the dataset**

   The data stored in the dataset is stored in a python variable. The labels and text of each emotional text is converted into lists.

   Code:

```
# Function for reading the dataset file
def read_data(file):
    data = []
    with open(file, 'r') as f:
        for line in f:
            line = line.strip()
            label = ' '.join(line[1:line.find("]")].strip().split())
            text = line[line.find("]")+1:].strip()
            data.append([label, text])
    return data

# File name
file = "c:\\Users\Sai raj\Desktop\My Project\dataset.txt"
data = read_data(file)
print(data[:10])
```

2. **Removing stopwords and special characters**

   Stopwords and Special characters are not so useful in training the model and also decreases the model performance. So, we need to remove the stopwords in the corpus.

Code:

```
import nltk
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
import re

nltk.download('stopwords')
stop_words = stopwords.words('english')

# Function for removing the stop word from the text
def remove_stop_words(data):
  sentences = ""
  data = data.split('\n')
  #print(data)
  for text in data:
    # Replacing each special character and numbers with a space
    text_alphanum = re.sub('[^a-z]', ' ', text)
    word_tokens = word_tokenize(text_alphanum)

    # Removing stop words
    sentence = ' '.join([w for w in word_tokens if (w not in stop_words)])
    sentences += sentence + "\n"
    #print(sentence)
  return sentence
```

3. Lemmatization

Lemmatization, unlike Stemming, reduces the inflected words properly ensuring that the root word belongs to the language. Lemmatization of text before training the model improves the model efficiency.

Code:

```
import nltk
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize

text_word_cloud = ''
# word lemmatization (Normalization)
def noun_lemmatizer(sentences):
  # Init the Wordnet Lemmatizer
```

```
    lemmatizer = WordNetLemmatizer()
    sentences = sentences.split('\n')
    #print(sentences)
    lem_text = ''
    for line in sentences:
        #print(line)
        word_tokens = word_tokenize(line)
        sentence = ' '.join([lemmatizer.lemmatize(w, 'n') for w in word_tokens])
        lem_text += sentence + '\n'
    text_word_cloud = lem_text
    return lem_text
```

4. N-Grams Generation

Using these n-grams and the probabilities of the occurrences of certain words in certain sequences could improve the predictions. For example, using a 3-gram or trigram training, model will be able to understand the difference between emotional sentences.

Code:

```
# Function for generating ngrams of words
def ngram(token, n):
    output = []
    for i in range(n-1, len(token)):
        ngram = ' '.join(token[i-n+1:i+1])
        output.append(ngram)
        #print(output)
    return output
```

4.2 FEATURE EXTRACTION

Feature extraction is used extract and produce feature representations that are appropriate for the type of Natural Language Processing (NLP) task we are trying to accomplish and the type of model you are planning to use.

To train the model data must be in the numerical form. Since, our corpus is in the text form we need to convert the text into a numerical. This project converts the text in the corpus to numerical form using the following techniques.

Code:

```
from sklearn.feature_extraction import DictVectorizer
vectorizer = DictVectorizer(sparse = True)
X_train = vectorizer.fit_transform(X_train)
X_test = vectorizer.transform(X_test)
```

4.3 BUILDING SVM MODEL

In this project, to build a model for classifying the emotional sentence we used Support Vector Machine Multiclass Classifier. We first, divide the dataset into two parts for testing and training. Build the model using training dataset.

Code:

```
from sklearn.svm import SVC
linear_svm = SVC(kernel='linear')
linear_svm.fit(x_train, y_train)
```

4.4 MODEL EVALUATION

After training the model, model performance is measured using the performance metrics. For classification problem we use the following metrics.

1. Accuracy
2. F1 Score
3. Precision
4. Recall

Code:

```
def train_test(clf, X_train, X_test, y_train, y_test):
   clf.fit(X_train, y_train)
   train_acc = accuracy_score(y_train, clf.predict(X_train))
   test_acc = accuracy_score(y_test, clf.predict(X_test))
   #precision = find_precision(y_test, clf.predict(X_test))
   precision, recall, f1_score, support = precision_recall_fscore_support(y_test, clf.predict(X_test),
            average='macro')
return train_acc, test_acc, precision, recall, f1_score
```

Chapter 5

RESULTS

After model has been trained, it has to be tested with real world data that it hasn’t seen. Testing the model help us to know whether the model is performing well or not and is it ready to deploy in the real-world applications. The model tested with the following sample, where each sample embedded with one of the seven emotions.

Text1: "I'm very good today"

Text2: "I was thinking about death"

Text3: "He does something wrong which affects me negatively"

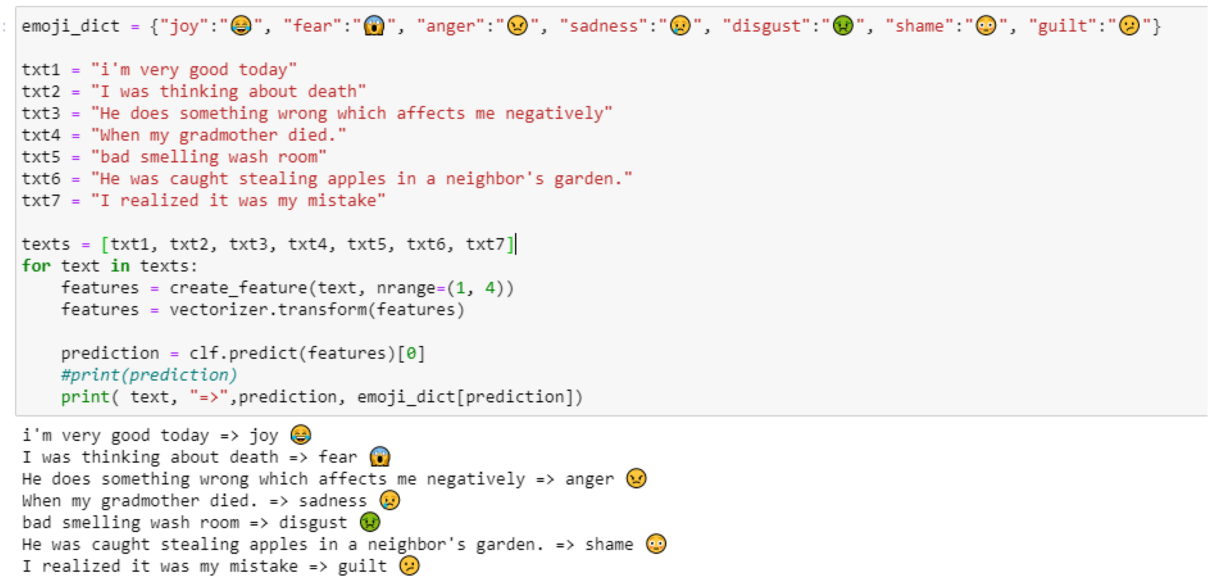
Text4: "When my grad mother died."

Text5: "bad smelling wash room"

Text6: "He was caught stealing apples in a neighbour’s garden."

Text7: "I realized it was my mistake"

Testing the model



```
emoji_dict = {"joy":"😂", "fear":"😱", "anger":"😠", "sadness":"😢", "disgust":"🤢", "shame":"😳", "guilt":"😕"}

txt1 = "i'm very good today"
txt2 = "I was thinking about death"
txt3 = "He does something wrong which affects me negatively"
txt4 = "When my gradmother died."
txt5 = "bad smelling wash room"
txt6 = "He was caught stealing apples in a neighbor's garden."
txt7 = "I realized it was my mistake"

texts = [txt1, txt2, txt3, txt4, txt5, txt6, txt7]
for text in texts:
    features = create_feature(text, nrange=(1, 4))
    features = vectorizer.transform(features)

    prediction = clf.predict(features)[0]
    #print(prediction)
    print( text, "=>",prediction, emoji_dict[prediction])
```

```
i'm very good today => joy 😂
I was thinking about death => fear 😱
He does something wrong which affects me negatively => anger 😠
When my gradmother died. => sadness 😢
bad smelling wash room => disgust 🤢
He was caught stealing apples in a neighbor's garden. => shame 😳
I realized it was my mistake => guilt 😕
```

Fig 5.1 Testing Results

Text1: "I'm very good today"

Predicted Emotion: Joy

Text2: "I was thinking about death"

Predicted Emotion: Fear

Text3: "He does something wrong which affects me negatively"

Predicted Emotion: Anger

Text4: "When my grad mother died."

Predicted Emotion: Sadness

Text5: "bad smelling wash room"

Predicted Emotion: Disgust

Text6: "He was caught stealing apples in a neighbour’s garden."

Predicted Emotion: Shame

Text7: "I realized it was my mistake"

Predicted Emotion: Guilt

The predicted outcome shows the model is performing accurately. So, it can be deployed in real-world applications.

5.1 METRICS

Detecting emotion related short texts is classification problem. In this project, we used the Support Vector Machine Classifier for building the model. After training our model we can to measure model performance. Every classification model performance us measured by using the accuracy, confusion matrix, f1 score, precision and recall. In this project, we took all these metrics for evaluating the model performance.

1. Accuracy
2. Confusion Matrix
3. Precision
4. Recall
5. F1 Score

Accuracy

It is simply the ratio of the number of correct predictions to the number of all predictions.

Accuracy = Correct Predictions / Total Predictions

In this project, the model achieved the accuracy 99% training accuracy and 81% testing accuracy.

Confusion Matrix

Confusion matrix does not return a numerical value as an evaluation. In that sense, it is hard to call it a metric. However, confusion matrix provides valuable insight into predictions. Confusion matrix goes deeper than classification accuracy by showing the correct and incorrect (i.e, true or false) predictions on each class.

This project classifies the seven classes. So, confusion matrix is 7x7. It shows the sample which are classified into correct class and the sample which are classified into wrong class.

Precision

It focuses on the positive predictions. it is the ratio of the correct positive predictions to all positive predictions. In a sense, it evaluates the model only based on the positive predictions. Precision measures how good our model is when the prediction is positive.

Precision = True Positive / (True Positive + False Positive)

Recall

Recall is also used in binary classification tasks. It focuses on the positive class. Recall is the ratio of the correct positive predictions to all observations in the positive class. Thus, it evaluates the model based on its ability to predict the positive class. Recall measures how good our model is at correctly predicting positive classes.

Recall = True Positive / (True Positive + False Negative)

F1 Score

F1 score is the weighted average of precision and recall. F1 score is a more useful measure than accuracy for problems with uneven class distribution because it takes into account both false positive and false negatives. The best value for f1 score is 1 and the worst is 0.

F1 Score = 2 (Precision * Recall)/ (Precision + Recall)

5.2 GRAPHS AND TABLES

- The below bar graph shows the frequency of sentences the below to each emotion in the corpus, which was used for training the model in this project.

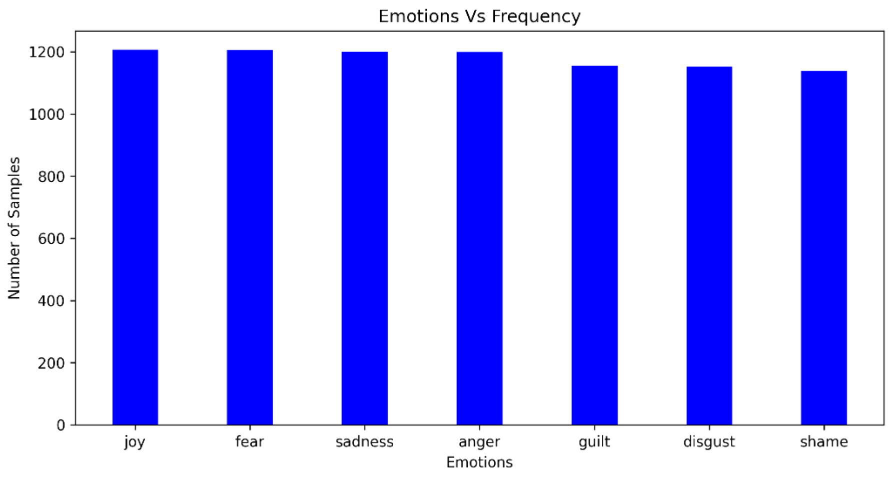
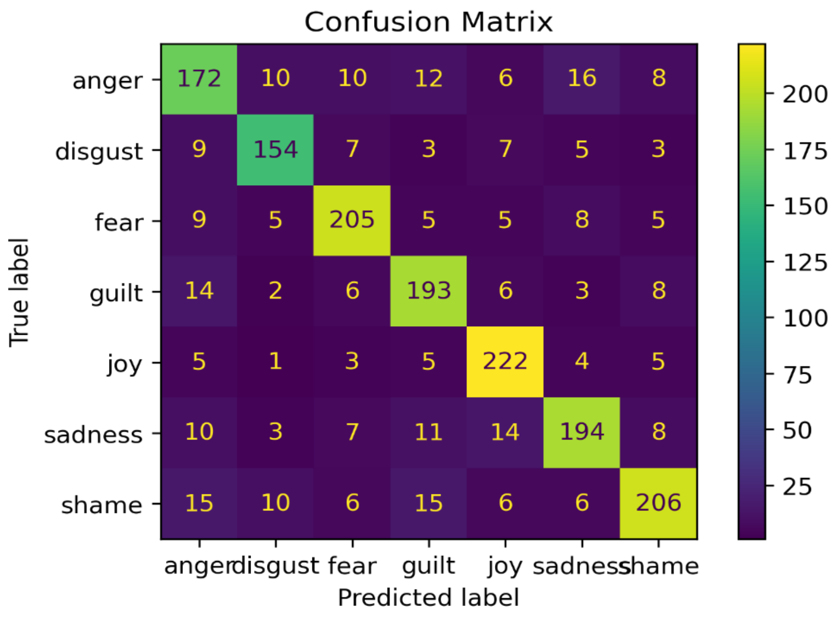


Fig 5.2 Emotions vs Number of samples

- Confusion Matrix is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 7 different combinations of predicted and actual values. It is extremely useful for measuring Recall, Precision, Specificity and Accuracy.



True label \ Predicted label	anger	disgust	fear	guilt	joy	sadness	shame
anger	172	10	10	12	6	16	8
disgust	9	154	7	3	7	5	3
fear	9	5	205	5	5	8	5
guilt	14	2	6	193	6	3	8
joy	5	1	3	5	222	4	5
sadness	10	3	7	11	14	194	8
shame	15	10	6	15	6	6	206

Fig 5.3 Confusion Matrix

- In this project, the model performance is measured using the performance metrics. The below image shows the performance metrics values obtained by the trained model.

Table 1: Accuracy of Models

Model Name	Accuracy
SVM with Linear Kernel	0.8147
SVM with Sigmoid Kernel	0.7455
Weighted Labeled Topic Model	0.6431
X-term Emotion Topic Model	0.6214

5.3 COMPARISION

The existing system used the term group co-occurring in the same context to enrich the number of features whereas the proposed system uses the co-occurrence of words generated using n-grams generation technique.

The existing system used the supervised topic modeling algorithms. It proposed two algorithms Weighted Labeled Topic Model (WLTM) and an X-Term Emotion Topic Model (XETM). The proposed system used the Support Vector Machine Multiclass Classifier (SVMMC) for predicting emotion.

The time complexity of the existing model is high because of the two supervised topic modeling algorithms. The proposed system time complexity if better when compared to the existing system.

The Proposed system is performing well when compared to the existing system as it shows the better accuracy score than the existing system. The proposed system obtained 81% accuracy.

Chapter 6

CONCLUSION AND FUTURE SCOPE

The proposed system, Short Text Emotion Detection aims to predict the emotions which are embedded in textual data. This proposed system uses one supervised machine learning algorithm for classifying the emotions. It uses the Support Vector Machine (SVM) for classifying the text into seven classes where each class represent an emotion. The emotions are joy, angry, fear, disgust, shame, sad and guilt. We performed the text lemmatization on the corpus to make the words consistent for training. It uses the co-occurrences of words for generating feature vectors. These co-occurrences of words contain the words which are used more regularly together in each emotional text. The text in the corpus is brought into numerical for by using word embedding technique. The proposed system is less time consuming and the experimental results indicated that proposed system is performing efficiently.

The proposed system will be used to improve the user experience during human and computer interaction. The computers will be able to detect the emotion of the user and they respond in a way to enhance the user satisfaction. This project classifies the text into single emotion but the text may be related to more than one emotion. In future, we extend the system to predict all the emotions embedded in an emotional text.

REFERENCES

[1] Eric Cambria, Andrew Livingstone and Amir Hussain. The Hourglass of Emotions. Cognitive Behavioural Systems 2011, LNC 7403, pp. 144-157, 2012.

[2] Jianhui Pang, Yanghui Rao. Fast Supervised Topic Model for Short Text Emotion Detection. 2168-2267, 2019

[3] S.-Y. Chen, C.-C. Hsu, C.-C. Kuo, L.-W. Ku et al., “Emotion lines: An emotion corpus of multi-party conversations,” arXiv preprint arXiv:1802.08379, 2018

[4] Y. Rao et al., “Supervised intensive topic models for emotion detection over short text,” in Proc. 22nd Int. Conf. Database Syst. Adv. Appl., 2017, pp. 408–422.

[5] S. Poria, N. Majumder, D. Hazarika, E. Cambria, A. Gelbukh, and A. Hussain, “Multimodal sentiment analysis: Addressing key issues and setting up the baselines,” IEEE Intell. Syst., vol. 33, no. 6, pp. 17–25, Nov./Dec. 2018.

[6] *Tengjun Yao, Zhengang Zhai and Bingtao Gao, “Text Classification Model Based on fastText”, 2020.*

[7] 20] J. Herzig, M. Shmueli-Scheuer and D. Konopnicki, "Emotion Detection from Text via Ensemble Classification Using Word Embeddings," in Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, 2017.

[8] R. Al-Rfou, D. Choe, N. Constant, M. Guo and L. Jones, "Character-level language modeling with deeper self-attention", *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 3159-3166, 2019.

[9] F. A. Acheampong, C. Wenyu and H. Nunoo-Mensah, "Text-based emotion detection: Advances challenges and opportunities", *Engineering Reports*, vol. 2, no. 6, pp. 1-24, 2020.

[10] F. M. Alotaibi, "Classifying text-based emotions using logistic regression", *VAWKUM Transactions on Computer Sciences*, vol. 16, no. 2, pp. 31-37, 2019.

[11] A. Kazameini, S. Fatehi, Y. Mehta, S. Eetemadi and E. Cambria, Personality trait detection using bagged SVM over BERT word embedding ensembles, 2020.

[12] Malak Abdullah, Mirsad Hadzikadicy and Samira Shaikhz, "SEDAT: sentiment and emotion detection in Arabic text using CNN-LSTM deep learning", *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE*, pp. 835-840, 2018.

[13] Ankush Chatterjee, Kedhar Nath Narahari, Meghana Joshi and Puneet Agrawal, "SemEval-2019 Task 3: EmoContext Contextual Emotion Detection in Text*", Proceedings of the 13th International Workshop on Semantic Evaluation*, pp. 39-48, 2019.

[14] Bharat Gaind, Varun Syal and Sneha Padgalwar, "Emotion Detection and Analysis on Social Media", *arXiv preprint*, 2019.

[15] Maryam Hasan, Elke Rundensteiner and Emmanuel Agu, "Automatic emotion detection in text streams by analyzing Twitter data", *International Journal of Data Science and Analytics*, vol. 7, no. 1, pp. 35-51, 2019.

[17] Moin Khan and Kamran Malik, "Sentiment Classification of Customer's Reviews About Automobiles in Roman Urdu*", Future of Information and Communication Conference*, pp. 630-640, 2018.

[18] *Lalit Mohan, Janmejay Pant, Priyanka Suyal and Arvind Kumar, "Support Vector Machine Accuracy Improvement with Classification", 2020.*