Proposed Statistical Based Formulae to Calculate Valence-Arousal-Dominance Values from Text

Project Report Submitted in Partial Fulfilment of the
Requirements for the degree of
Master of Computer Application

By
Kushal Das
Master of Computer Application – III
Examination Roll Number: MCA226010
Registration Number: 149873 of 2019 – 2020

Under the guidance of
Dr. CHITRITA CHAUDHURI
Associate Professor

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Jadavpur University
Kolkata – 700032, India
June, 2022

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT FACULTY OF ENGINEERING AND TECHNOLOGY JADAVPUR UNIVERSITY

TO WHOM IT MAY CONCERN

I hereby forward the project report entitled **"*Proposed Statistical Based Formulae to Calculate Valence-Arousal-Dominance Values from Text*"** prepared by **Kushal Das, Registration Number- 149873 of 2019 – 2020 and Examination Roll Number-MCA226010 and** under my supervision to be accepted in partial fulfilment for the degree of **Master of Computer Application** in the Faculty of Engineering and Technology of Jadavpur University, Kolkata.

_____

(Dr.Chitrita Chaudhuri)
Associate
Professor **Project**
Supervisor
Dept. of Computer Science and
Engineering Jadavpur University
Kolkata – 700032

Countersigned:

_____

__ Prof. Anupam Sinha
Head, Dept. of Computer Science and
Engineering Jadavpur University
Kolkata – 700032

_____

Prof. Chandan Mazumder
Dean, Faculty of Engineering and
Technology Jadavpur University
Kolkata – 70032

Department of Computer Science and Engineering
Faculty of Engineering and Technology
Jadavpur University

CERTIFICATE OF APPROVAL *

The foregoing project report is hereby approved as a creditable study of an engineering subject carried out and presented in a manner satisfactory to the degree for which it has been submitted. It is understood that, by this approval, the undersigned do not necessary endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project report only for the purpose for which it has been submitted.

_____

Final Examination for
evaluation of the
project

_____

(Signatures of Examiners)

* Only in case the project report is approved

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

I hereby declare that this project report contains literature survey and original research work by undersigned candidate, as part of my Master of Computer Application studies.

All information in this document had been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

NAME: Kushal Das

Examination Roll Number: MCA226010

Registration Number: 149873 of 2019 - 2020

Project Title: Proposed Statistical Based Formulae to Calculate Valence-Arousal-Dominance Values from Text

Signature with Date:

ACKNOWELDGEMENT

The satisfaction and euphoria that accompanies the successful completion of this task would be incomplete without the mention of the people who made it possible. Their constant guidance and encouragement crowned my effort with success.

It is a great pleasure to express my sincerest thanks to my project supervisor Dr. Chitrita Chaudhuri, Associate Professor, Department of Computer Science and Engineering, Faculty of Engineering and Technology, Jadavpur University, for her encouragement, valuable suggestion, and constant support during the course of this project.

I would like to thank all the professors of the Department of Computer Science and Engineering, Jadavpur University, Kolkata for the guidance they provided me throughout the duration of the Master of Computer Application course.

A special note of thanks goes to Prof. Anupam Sinha, Head, Department of Computer Science and Engineering, Jadavpur University.

I am also thankful to Prof. Chandan Mazumder, Dean, Faculty of Engineering and Technology, for providing an excellent environment for completion of this project.

I am also indebted to my co-researchers Mr. Anupam Baidya for his seamless co-operation and help in completion of this project. I am thankful to my fellow classmates and my family for constant help and support.

Date: _____ _____

Kushal Das
Master of Computer Application – III
Examination Roll No. - MCA226010
Registration No: 149873 of 2019 – 2020

Chapter Content

List of Tables

List of Figures

Chapter 1

Introduction

Sentiment analysis is a process of using natural language processing (NLP) to analyse the sentiment tone behind a body of text. A sentiment analysis system for text analysis combines NLP and machine learning (ML) techniques to derive computationally whether the block of text is positive, neutral or negative [1]. Nowadays as humans can express their feelings and thoughts more openly than ever before, sentiment analysis becomes a very essential mechanism to assess a person [2]. Text data being readily available online has become a main source of human expressions. Through social media review, twitter data and other forms of correspondence, the mental condition of a user can be assessed. Using sentiment analysis on the review data of a product a company can upgrade and modify their product. From tweets one can collect the opinion of the masses regarding different sociological events. Thus the pulse of the society can be gauged through their emotions.

Valence-Arousal-Dominance (VAD), a three-dimensional model, is another metric by which the emotional involvement of a person can be evaluated. This technique was introduced by Mehrabian and Russell in 1977 [3]. The three independent dimensions are: *Valence* which represent the scale of happiness to unhappiness and expresses about pleasant or unpleasant through usage of text words, *Arousal* which gives the level of excitement in any situation, and *Dominance* which indicates the range of control from managing to being managed.

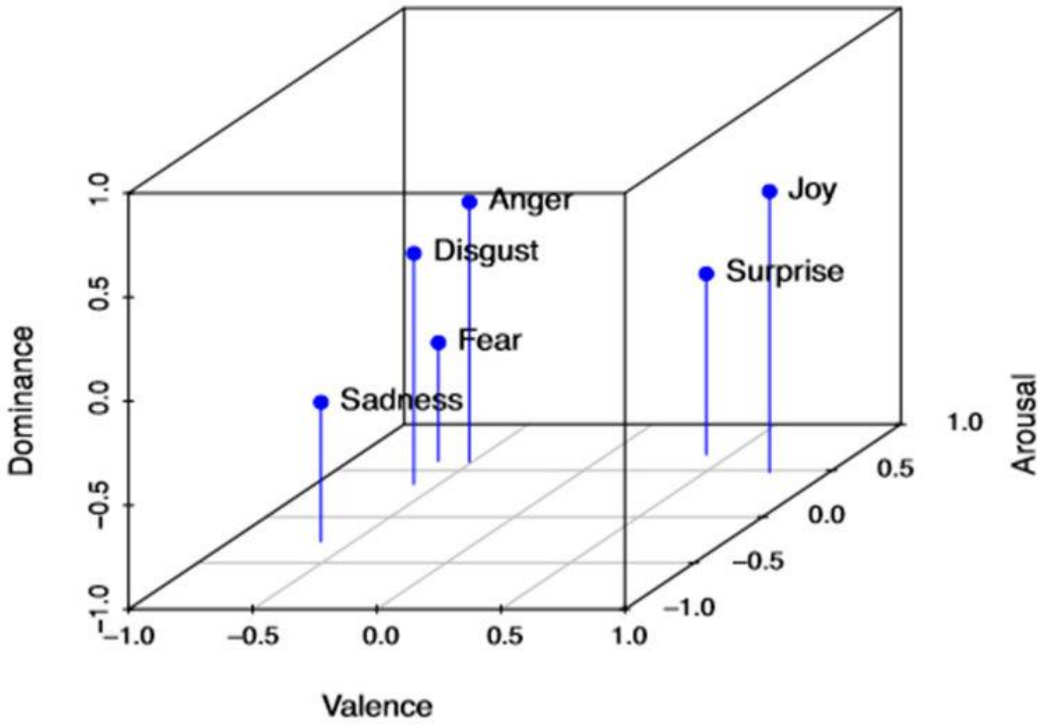

Figure 1.1: The VAD model spanned across the six basic emotions

The dataset used to test VAD attributes in this work is the EmoBank [4]. This dataset consists of 10062 sentences collected across 6 domains and 2 perspectives. Each sentence has three score representing Valence, Arousal and Dominance in the range of 1 to 5. The three independent features was derived by the annotators located in UK, US, Canada, Ireland Australia, Newzeland

and the annotation cost was $1578 [5]. The time required to annotate the total data was huge. At the time of annotating, it was not verified whether the annotators were qualified or not. Also the shortage of proper annotators was a big issue.

So for the above all reasons the three important features for monitoring a person's mental state: Valence, Arousal, Dominance are better derived computationally. In this paper the objective has been to achieve this goal. Here various machine learning models such as SVM, KNN, Naïve Bayes, Random Forest have been used to validate the annotated VAD values. The accuracies obtained by the classifiers have been further utilized to benchmark the statistically derived results achieved through the proposed calculation-based technique termed as the SBVAD Technique in this document.

The success of the proposed technique has been found to be quite high (V = more than 85%, A= more than 99%, D = around 90%), in two separate sets of experiments. In the first, only one portion of the whole data set were designated as test set data, the rest being taken to train the other classifiers. The proposed technique was tried only on this test data set in the first experiment. The second experiment involved a 10-fold cross validation technique over the whole dataset for all the other classifiers, while the proposed technique covered all the data tuples. In both cases the SBVAD technique compared favourably with the best of the classifiers. In fact, both Valence and Arousal detection was found to be best for the SBVAD techniques. So the present researcher feels that the use of the indigenous technique in evaluating VAD through statistical means is fairly justified under the circumstance.

The report is organized as follows:

In chapter 2 we give details on previous work on VAD by the human annotators. In chapter 3, we introduce some basic concepts of machine learning, natural language processing, sentiment analysis, sentiwordnet, cosine similarity and some other parameters which are important in the present work. In chapter 4 we describes the methodologies used to determine VAD using machine learning algorithm and a statistical based proposed technique. In chapter 5 we analyse the result obtained from methodology part and compare them. Finally in chapter 6 we draw a conclusion and describe about future scope.

Chapter 2

Previous Work

Following is a list of research works dealing with the Valence, Arousal and Dominance properties studied within texts for finding the emotion inclinations of the human authors or annotators.

- Osgood et al. (1957) [6] asked human participants to rate words along dimensions of opposites such as heavy– light, good–bad, strong–weak, etc. Factor analysis of these judgments revealed that the three most prominent dimensions of meaning are evaluation (good–bad), potency (strong–weak), and activity (active–passive).

- Russell (1980, 2003) [7, 8] showed through similar analyses of emotion words that the three primary independent dimensions of emotions are valence or pleasure, arousal and dominance. He argues that individual emotions such as joy, anger, and fear are points in a three-dimensional space of valence, arousal, and dominance.

- At first to detect emotion of a sentence the three independent features Valence, Arousal, Dominence was introduced by Mehrabian and Russel in 1974. They held a self-assesment study to rate the feelings on a bunch of undergraduate students through their responses to verbal reviews on various situations [9].

- After that Bardley and Lang, 1999 [10] asks the annotators to give rating on Valence, Arousal and Dominance on english word on a range 0-9. The number of English words rated is 1034. This lexicon called as Affective Norms of English Words (ANEW).

- More than a decade later Warriner et al., 2013 [11] created a similar lexicon for almost greater than 13000 words using human annotated method.

- Mohammad, 2018 [12] created another database of 20000 English words named as NRC-VAD Lexicon. It is the largest manually created VAD lexicon.

- By using these resources, recent work tried to predict VAD scores from sentences based on variational autoencoders Wu et al., 2019 [13], adversarial learning Zhu et al., 2019 [14], ensemble learning Akhtar et al., 2019 [15]

- There exist a small number of VAD lexicons in non-English languages as well, such as the ones created by Moors et al., 2013 [16] for Dutch, by Vo et al., 2009 [17] for German, and by Redondo et al., 2007 [18] for Spanish.

All the above researchers used human annotators to produce the VAD values. The limitations of the process are manifold: excessive time consumption, difference of opinion, exorbitant cost, human bias and errors, to name only a few. The urge to rectify the situation has laid the foundation of the present research.

Chapter 3

Basic Concepts

In this chapter are discussed some fundamental theories and details which helped in implementing the proposed technique.

3.1 Machine Learning

Machine learning, which is an application of artificial intelligence (AI), allows systems the ability to learn automatically and improve from experience without being explicitly programmed. It focuses on developing computer programs that can access and use data to extend its knowledge base. The process to learn begins with data in order to look for patterns and make better decisions in the future based on the data that we provide. The primary aim is to allow the computers to automatically learn and adjust actions accordingly without human intervention or assistance [19].

3.1.1 Classification

Machine learning algorithms can be mainly classified into two categories, namely Supervised learning and Unsupervised learning. Supervised learning, as the name indicates, perform task with presence of anyone's supervision. The machines are trained by providing data (called train data) which are labelled. After that the trained system are given a new set of data (called test data) to predict the outcome. In case of Unsupervised learning, the process of building a system uses data that is neither classified nor labelled and allows the algorithm to act on that data without guidance. Here the job of the machine is to group the data based on their similarities, patterns and differences without any prior knowledge of class value.

3.2 Natural Language Processing

Natural Language Processing (NLP) refers to the field of computer science more specifically Artificial Intelligence (AI), which deals with giving computers the ability to understand texts and spoken words in the same way as the human being can.

NLP combines computational linguistics (rule-based modelling of human language) with models of statistics, machine learning and deep learning. Combining this technologies, computer can process human speech in forms of text or voice data and understand with the intent and sensation of speaker and writer. This computer can also translate text from one language to another, respond in spoken commands, and also summarize large volume of texts rapidly [20].

NLP consists of two main phases: data pre-processing and algorithm development.

Data pre-processing involves preparing and cleaning text data for machines so that it can able to analyse it. Pre-processing puts the data in a usable format and highlights the features in the text for the algorithm to work. There are several ways to do this:

Tokenization: Text is broken down into smaller tokens to work with.

Stop word removal: Common words are removed from text so that unique words remains that offer the most information about the texts.

Lemmatization and stemming: Words are reduced to their root forms to process.

Part-of-speech tagging: Words are marked based on the part-of speech i.e nouns, verbs and adjectives.

After the data is pre-processed, algorithms are developed to process it. There are many different natural language processing algorithms, but two main types are commonly used.

Rules-based system: This system uses carefully designed linguistic rules. This approach was used early in the development of natural language processing and is still in use today.

Machine learning-based system: Machine learning algorithms use statistical methods. They learn to perform tasks based on the training data given to them and adjust their methods as more data is processed. Natural language processing algorithms use a combination of machine learning, deep learning, and neural networks to refine their own rules by repeating processing and learning [21].

3.3 Sentiment analysis

Sentiment is a measure, which was designed to provide helpful insights into how audiences sense approximately and understand a brand, mainly on social media systems and in editorial information media. This advertising metric is used to help the manufacturers manipulate and protect their reputations, while tracking the online environment [22].

A sentiment score is typically based on a range from -1 to 1, with -1 indicating negativity, 0 neutrality and 1 positivity.

Sentiment analysis (SA) is an intellectual process of extracting user's opinions, appraisals, and emotions toward entities, events and their attributes. It is a technique to know how people think about a product, service, company or a specific event by analysing texts, images, emoji or by reaction on the specific event.

3.3.1 Categories of Sentiment Analysis

Sentiment analysis is mainly of four types:-

A. Fine-grained Sentiment Analysis: In a customer review environment, it gives an in-depth understanding of the feedbacks received from customers. User can get accurate results related to the input polarity. This process is more laborious and costly compared to other types.

B. Emotion Detection Sentiment Analysis: This is a more sophisticated way to identify the emotions in a piece of text. Lexicons and machine learning tools are used to detect the sentiment. Lexicon is a dictionary consisting of text words which are either positive or negative. This makes it easy to separate terms based on their sentiment [23].

C. Aspect-based Sentiment Analysis: This type of sentiment analysis is used for one aspect of a service or product. Its aim is to identify an opinion regarding a specific element of the product. For example, a company selling smartphones uses this type of sentiment analysis to monitor one aspect - say camera or display performance, so that they can understand how customers feel about that specific attribute(s) of the product.

D. Intent analysis: This gives deeper understanding of the intention of the customer. Its purpose is to determine the type of intent expressed in the message. It is mainly used in customer support systems to monitor the workflow [24].

In this work mainly the category B type Sentiment Analysis is explored.

3.3.2 SentiWordNet

SentiWordNet is an opinion lexicon, which is derived from WordNet database where each word is associated with a numeric sentiment score [25]. Sample entities of SentiWordNet are shown below:

Table 3.1: Sample from SentiWordNet

POS	ID	PosScore	NegScore	SysnSet Terms	Gloss
a	02130672	0.5	0.125	tentative	unsetelled in mind or opinion; "drew a few tentative conclusions
v	02569235	0.125	0.25	oversimplify	Simplify to an excessive degree; “Don’t oversimplify the problem

Each entities in SentiWordNet has both positive and negative score lying between 0 and 1. It contains not only unigram words, but also multi word expressions (n-grams). SentiWordNet is publically available for research purpose.

3.4 Word Vectorization

It is difficult for the machine learning algorithms to analyse a raw corpus. One needs to convert such data into a numerical representation best expressed in the form of a vector. *Word Vectorization* is the process which encode the individual words into vectors, making it easier to analyse or consume that text by a machine learning algorithm. Now the model can apply mathematical operations and get insights from the data.

Hashing Vectorizer, utilized in the present work, uses a hash trick which converts token string to integer index matrix. Thus the conversion of text documents to a matrix is performed by this vectorizer, forming a spare matrix containing the number of token occurrence from the collection of documents.

3.5 Cosine Similarity

In data analysis Cosine Similarity is a measurement that quantifies the similarity between two sequences of numbers. The sequences are treated as vectors in an inner product space. It is measured by the cosine angle between the two vectors. The angle is derived with help of vector dot product and the product of magnitude of each vectors [26]. It is generally used to measure document similarity in text analysis.

3.6 Tools Used

All codes have been developed using Python toolkit version 3.10.2. Codes have been appended at the end of the report for easy reference.

Chapter 4

Methodology

In this paper Valence, Arousal and Dominance are derived using machine learning algorithms as well as a statistical based VAD technique (called as SBVAD technique). Here a description of the techniques involved are provided in detail.

4.1 System Architecture

The following figure depicts the process as a combination of three phases the outputs of which are later compared to generate the assessment of the proposed technique. The first two phases comprise of different classifier models described in more detail in the following sections. Phase 3 deals with the indigenous technique used for calculating the sentiment attributes under consideration.

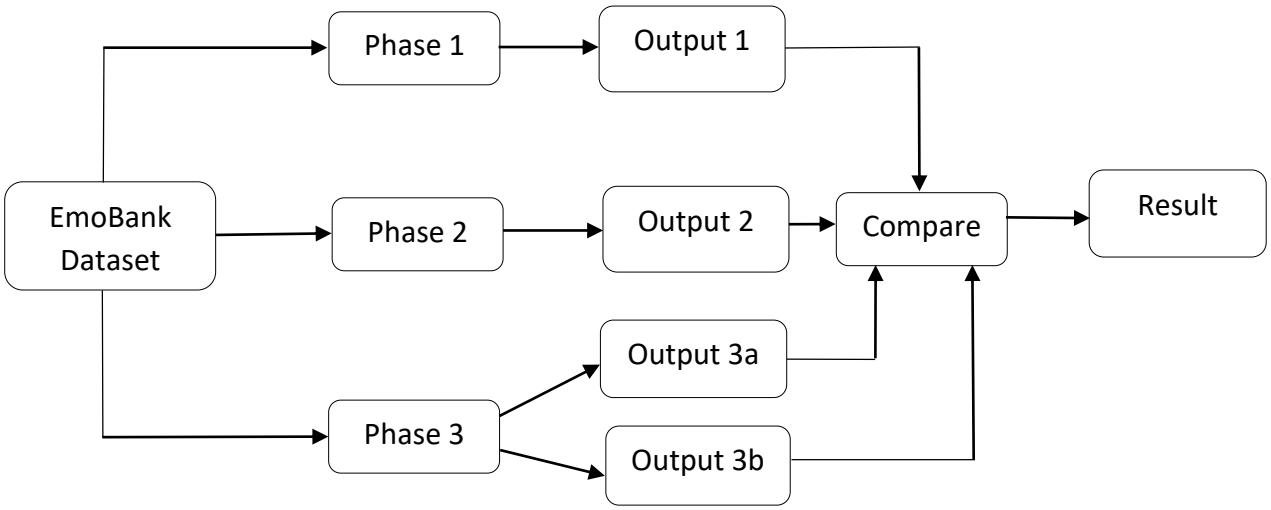

Figure 4.1: Overall System Architecture

4.2 Phase 1: Classification using Train-Test split

In this phase, is derived the Valence-Arousal-Dominance accuracy score based on the EmoBank dataset using Support Vector Machine (SVM), k-Nearest Neighbours (KNN), Naïve Bayes, and Random Forest classifiers. The train and test dataset in this phase are prefixed by the developers of the EmoBank dataset. Following is an architectural drawing of this system.

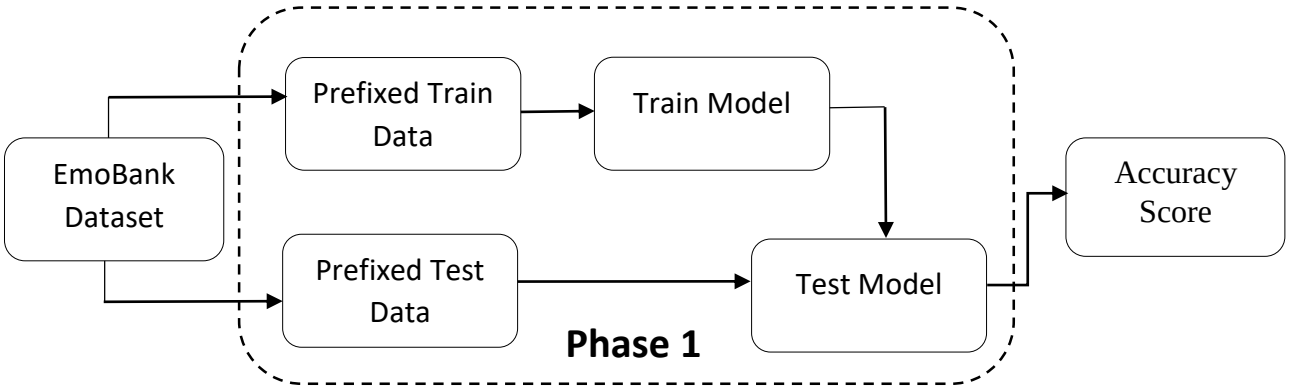

Figure 4.2: System Architecture for Phase 1

4.2.1 Classifier Details

Support Vector Machine (SVM): The goal of the SVM algorithm is to create optimal lines or decision boundaries that can divide n-dimensional space into classes so that new data points can be easily placed in the correct category in the future. This best decision boundary is called the maximum margin hyperplane. The SVM selects extreme points / vectors to help create the hyperplane. These extreme cases are called support vectors, and hence algorithm is termed as Support Vector Machine [27].

K-Nearest Neighbour (KNN): The k-nearest neighbour algorithm is a type of supervised machine learning algorithm used to solve classification and regression problems. However, it is mainly used for classification problems. At the training phase, instead of learning from the training dataset KNN just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data. For this reason KNN is called lazy learner [28].

Naïve Bayes: Naïve Bayes algorithm is a probabilistic machine learning algorithm, which is based on Bayes theorem and used for solving classification problems. Bayes theorem formula is given below:

$$P(A|B) = \frac{P(B|A)*P(A)}{P(B)}$$

Random Forest: The core unit of the Random Forest classifier is the decision tree. A decision tree is a hierarchical structure which is built using the characteristics (or independent variables) of a dataset. Each node in the decision tree is divided according to the metrics associated with the subset of features. Random forest is a classifier that contains a set of decision trees for different subsets of the given dataset and then takes the average to improve the prediction accuracy of that dataset [29].

4.2.2 Algorithm 4.1 // Phase 1

Input: EmoBank Dataset with prefixed train-test split
Output: Accuracy percentage of prefixed test data class values

1. read dataset and put it into pandas data frame named **data**

2. divide the whole data into lists: “Text” part into **train_data** and **test_data**, “V” part into **train_V**, **test_V**, “A” parts into **train_A** and **test_A**, “D” part into **train_D**, **test_D**, based on train and test split mentioned in the dataset

3. use HashingVectorizer to convert strings into vectors on **train_data** and **test_data** and keep in **train_datavector** and **test_datavector**

4. fit **train_datavector** and **train_V** to a machine learning model

5. check accuracy score of Valence by giving **test_datavector** and **test_V**

6. fit **train_datavector** and **train_A** to a machine learning model

7. check accuracy score of Arousal by giving **test_datavector** and **test_A**

8. fit **train_datavector** and **train_D** to a machine learning model

9. check accuracy score of Dominance by giving **test_datavector** and **test_D**

4.3 Phase 2: Classification using k-fold technique

Here we validate Valence-Arousal-Dominance scores for different parts of the EmoBank dataset during repeated runs, using the same set of classifiers as before, but now with the help of 10-fold cross validation technique applied on the whole dataset. Architectural drawing of Phase 2 is given below:

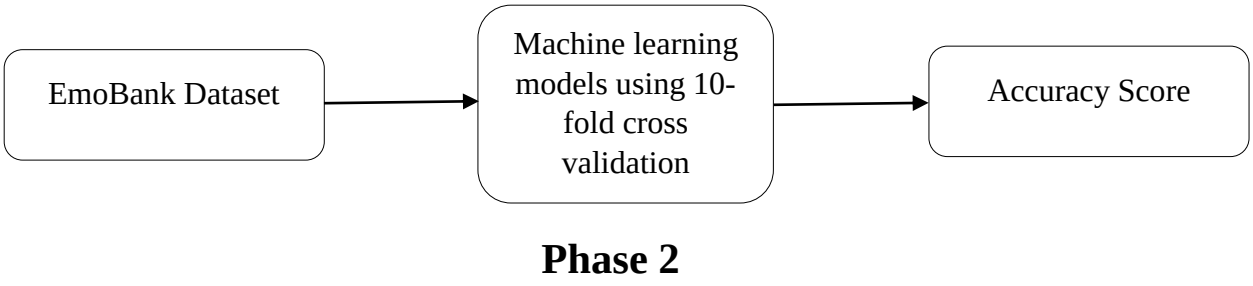

Figure 4.3: System Architecture for Phase 2

4.3.1 K-fold Cross Validation

K-fold cross validation is a very popular classification technique. It helps to solve the problem of prefixed train-test split bias. The process is depicted as follows:

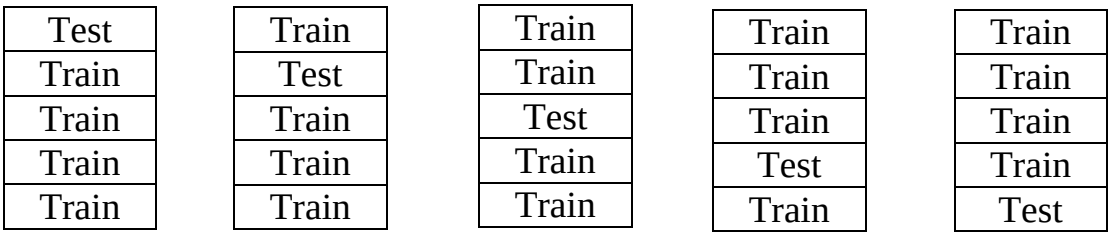

Fig 4.4: Train-test split mechanism in a 5-fold cross validation system

The above figure cites an example of a 5-fold cross validation system. First the whole data is divided into k equal parts. Then the 1st part is kept as testing data. The remaining (k-1) parts are used to train the machine learning model. The output of this first model is then saved. Next the 2nd part of the data is preserved for testing, and the rest of the data is used to train and build the 2nd model. The output of this 2nd model is also preserved after testing with the 2nd part of the data. In this way k results get accumulated. The final result is obtained by taking an average of the k results. Following is an architectural diagram of phase 2.

4.3.2 Algorithm 4.2 // Phase 2

Input: EmoBank Dataset with 10-fold train-test split

Output: Average accuracy percentage of 10-fold test data class values

1. read dataset and put it into pandas data frame named **data**
2. store "text" part in **text**, "V" part in **V**, "A" part in **A** and "D" part in **D**
3. use HashingVectorizer to convert strings into vectors on **text** and store in **vector_array**
4. call 10-fold cross validation, give a machine learning model, **vector_array** and **V**. Store in **score**
5. taking mean of **score** give accuracy score of Valence
6. call 10-fold cross validation, give a machine learning model, **vector_array** and **A**. Store in **score**
7. taking mean of **score** give accuracy score of Arousal
8. call 10-fold cross validation, give a machine learning model, **vector_array** and **D**. Store in **score**
9. taking mean of **score** give accuracy score of Dominance

4.4 Phase 3: Statistical Based Proposed Technique

In this phase NLP techniques and statistical based calculations are used to produce Valence-Arousal-Dominance scores for the sentences in the EmoBank dataset. The values are then compared with class values provided by the dataset developers. The accuracy achieved are benchmarked with the existing classifier accuracies. The process has been designated as Statistical-Based-Valence-Arousal-Dominance (SBVAD) technique. Following is the architectural diagram of Phase 3:

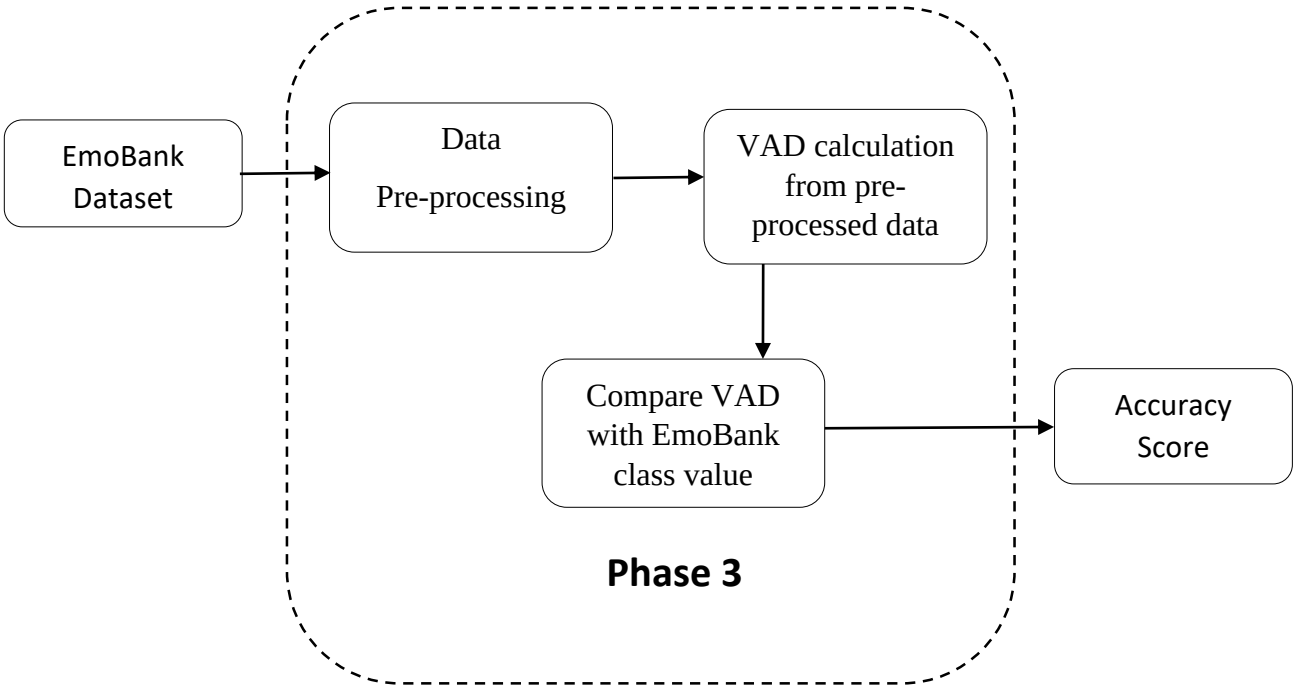

Figure 4.5: System Architecture for Phase 3

4.4.1 Data Pre-processing

The necessary data pre-processing steps are listed below. A detailed description along an architecture diagram of the model follows thereafter.

- Making lower case
- Remove punctuation
- Word Tokenization
- Stop word removal
- Lemmatization

A. Making lower case: Make the whole data into lower case.

B. Remove punctuation: Try removing several punctuations like '[', ']', '{', '}', '(', ')', '-', '@', '#', '&', '$' etc. This step helps to work properly.

C. Word Tokenization: Tokenization is the task of chopping up sentences into pieces, called tokens. Here the whole data is tokenized word by word. The process is also known as word segmentation.

D. Stop word removal: Words that are commonly excluded before processing a natural language are called stop words. This are actually the most common words in a language (articles, prepositions, pronouns etc.) and does not add much information of the text. There is a readily available dictionary (Stop Word Dictionary) in NLTK, which has been used in this work for this purpose.

E. Lemmatization: Lemmatization in linguistics is the process of grouping together the inflected forms of a word so that they can be analysed as a single item. In computational linguistics, lemmatisation is the algorithmic process of determining the lemma of a word based on its intended meaning. Lemmatisation depends on correctly identifying the intended part of speech and meaning of a word in a sentence.

am, are, is ⟶ be, car, cars, car's ⟶ car

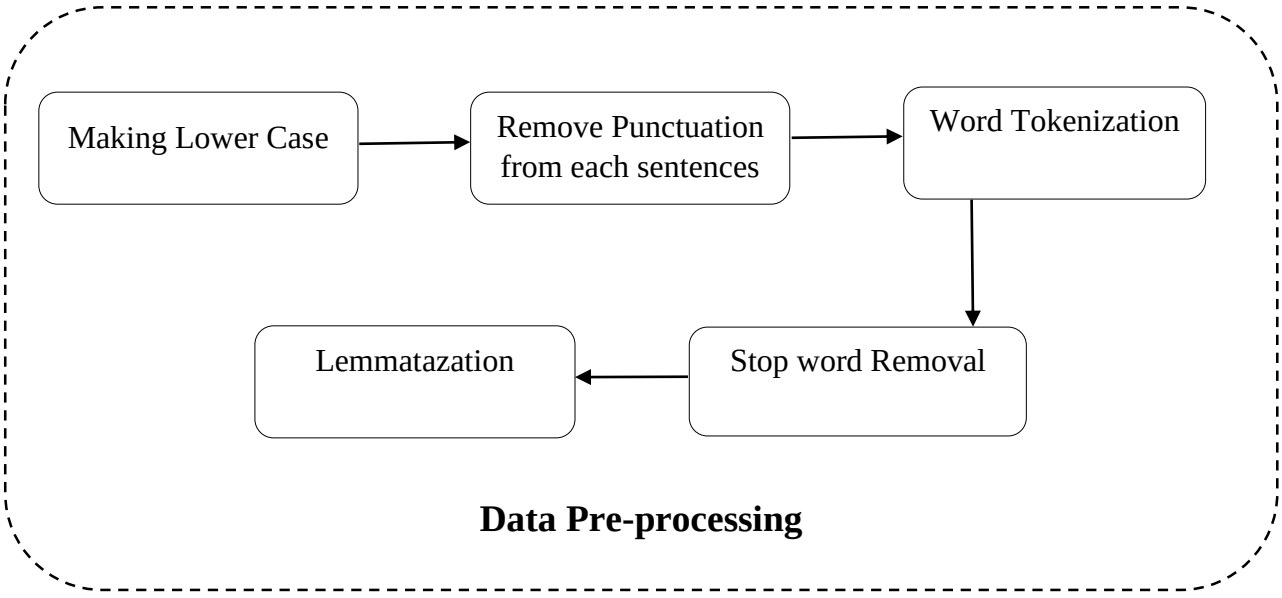

Figure 4.6: Data Pre-processing module for Phase 3

4.4.2 Valence-Arousal-Dominance

A. Calculations

Formulae: The statistical based formulae for V, A and D have been presented below:-

Valence (V) = $\frac{\sum positive\ sentiment\ score}{number\ of\ nonzero\ sentiment\ score\ words}$

Arousal (A) = $\frac{\sum |2*sentiment\ score-1|}{number\ of\ words}$

Dominance is derived basically in three steps. The first of these steps calculates a factor called Total Arousal of a Word (TAW) for each word in the whole data.

TAW = ∑|2*sentiment score of each use – 1|

The second step involves evaluating the Dominance Factor of each word in the whole data (DFw) from the normalized value of the corresponding TAW calculated in the previous step.

DFw = $\frac{TAW}{Max\ TAW\ values\ among\ all\ words\ in\ the\ text}$

The last step is to find the Dominance (D) per sentence from the Dominance Factor of all words (count = n) used in the responses made by the sentence.

Dominance (D) = $\frac{\sum DF}{n}$

B. Algorithm 4.3 //Phase 3

Input: English Text sentences obtained from EmoBank

Output : Accuracy of VAD scores of sentences vs. EmoBank class value
a. using test data set
b. using whole data set

// VAD calculation module

1. read data from sentiwordnet text file and put it in a dictionary **dict**

2. function **derive_sentiment_score ()** designed to consult **dict**

3. function **derive_valence_score ()** designed to consult **dict**

4. function **derive_pos_tag()**

    a. derive pos_tag: ***NN*** for singular noun, ***NNS*** for plural noun, ***VBP*** for verb, **VBN** for past participle verb, ***JJR*** for adjective, ***ADV*** for adverb

    b. make it lower case

    c. take its first word and replace “j” with “a” and replace “a” with “r” (since in sentiwordnet the pos-tag for adjective is “a” and for adverb is “r”)

5. save input data in a list named **data**

6. pre-process **data**

a. convert to lower case
b. remove all special characters
c. remove stopwords
d. lemmatize the input : finding the stem word

7. for each sentence in **data**

8. for each word in sentence

9. call **derive_pos_tag ()**

10. call **derive_valence_score ()** for the word and store in **score1**

11. add up **score1** to cumulative score **v_score** for each word in a senntence

12. call **derive_sentiment_score ()** for the word and store in **score2**

13. multiply each derived sentiment **score2** by 2, subtract 1 from the product, convert to its absolute value and sum up this result in **a_score** for all sentiment words in a sentence

14. dividing cumulative score **v_score** by total number of sentiment words in the sentence with non-zero sentiment score, gives the **Valence** per sentence**.** Keep this value in a list **Test_V**.

15. divide the summed up result **a_score** by total number of sentiment words in the sentence to derive **Arousal** per sentence. Keep it in a list **Test_A**

16. to derive **Dominance** per sentence, create a dictionary named **dom_dict**

a. read all input data and split by *white space* to store all words in a list named **str11**

b. for each word from list **str11**

c. call **derive_pos_tag ()**

d. call **derive_sentiment_score ()** for the word and store in **scr**

e. multiply each derived sentiment score **scr** by 2, subtract 1 from the product, convert to its absolute value and store the result in **scr1**

f. if this word does not exist in **dom_dict**
put this word and **scr1** in **dom_dict**
else add **scr1** to existing value for the word in **dom_dict**

g. find maximum valued word from dictionary **dom_dict** and store the value in **max_val**

h. divide all values in **dom_dict** by **max_val**

i. generate cumulative dominant score **d_score** for each sentence by adding up word scores per sentence from updated **dom_dict** and divide **d_score** by the total no of words in the sentence, giving **dominance** of the sentence. Store in a list **Test_D**

// VAD comparison module

18. store the V-A-D values in EmoBank dataset in three lists: **V**, **A**, **D** respectively

19. use cosine similarity on lists: **V** and **Test_V** giving accuracy score of Valence, **A** and **Test_A** giving accuracy score of Arousal, **D** and **Test_D** giving accuracy score of Dominance

Chapter 5

Result and Analysis

The following table presents the results of various techniques used to validate Valence-Arousal-Dominance in the EmoBank Dataset:

Table 5.1: Accuracy percentages of V-A-D for different techniques

Model Name	Accuracy percentage		
	Valence	Arousal	Dominance
SVM with test data	82	93.5	***95.9***
KNN - do -	81.6	92.9	95.9
Random Forest -do -	80.3	92.5	95.7
Naïve Bayes - do -	78.4	93.5	95.9
SBVAD_Technique -do -	85.99	***99.30***	91.68
SVM with 10 fold from all data	82.43	92.22	95.67
KNN - do-	81.72	92.16	95.47
Random Forest -do-	80.71	91.51	95.18
Naïve Bayes -do-	77.89	92.09	95.45
SBVAD_Technique on all data	***86.41***	99.07	87.68

The visual representation on comparison between Valence scores obtained from the different classifiers and the proposed technique are in the form of a bar chart in the following figure 5.1.

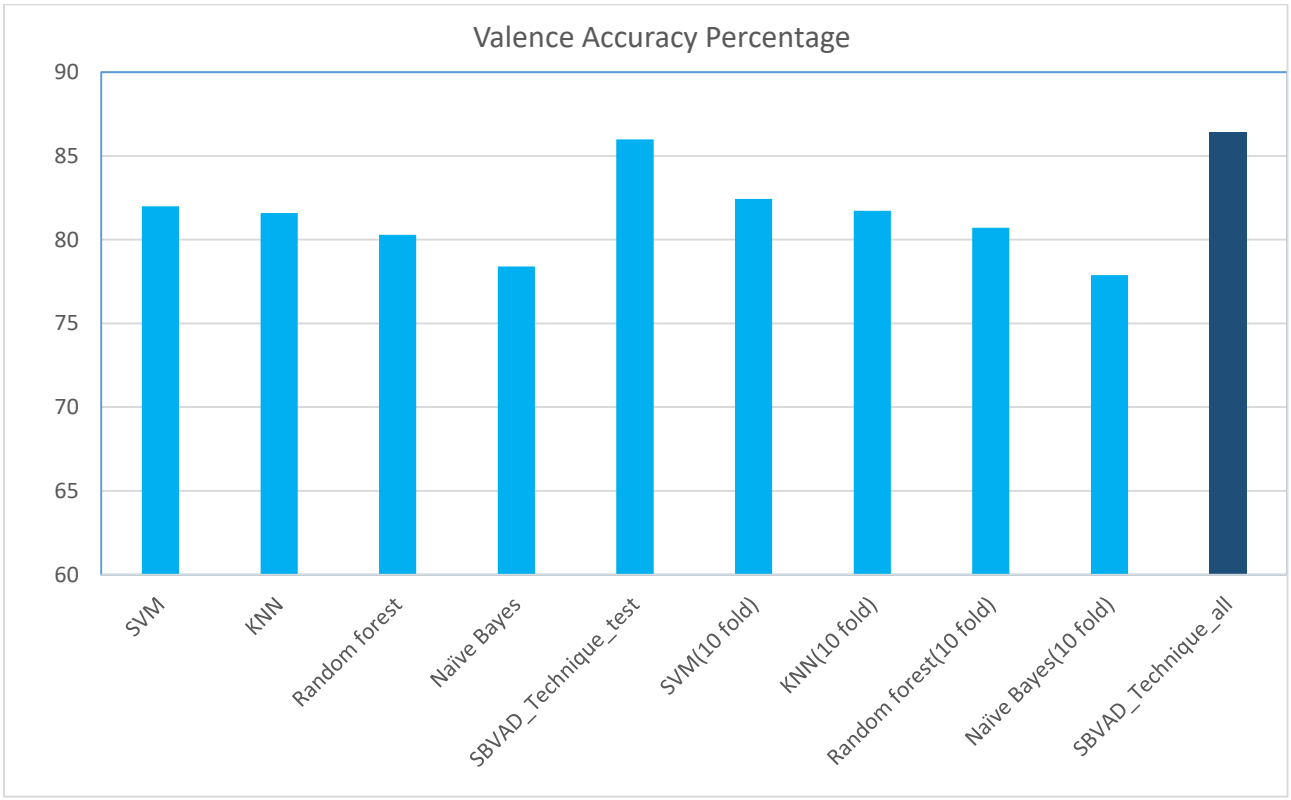

Figure 5.1: Comparison Bar Chart for Valence Accuracy

As the chart indicates the Valence accuracy is above 85% which is higher than the rest of the classifiers for both versions of the proposed technique.

Visualization of Arousal scores from the different classifiers and the proposed technique are displayed in the form of bars in the following figure 5.2. It helps to compare the accuracies obtained from the different techniques.

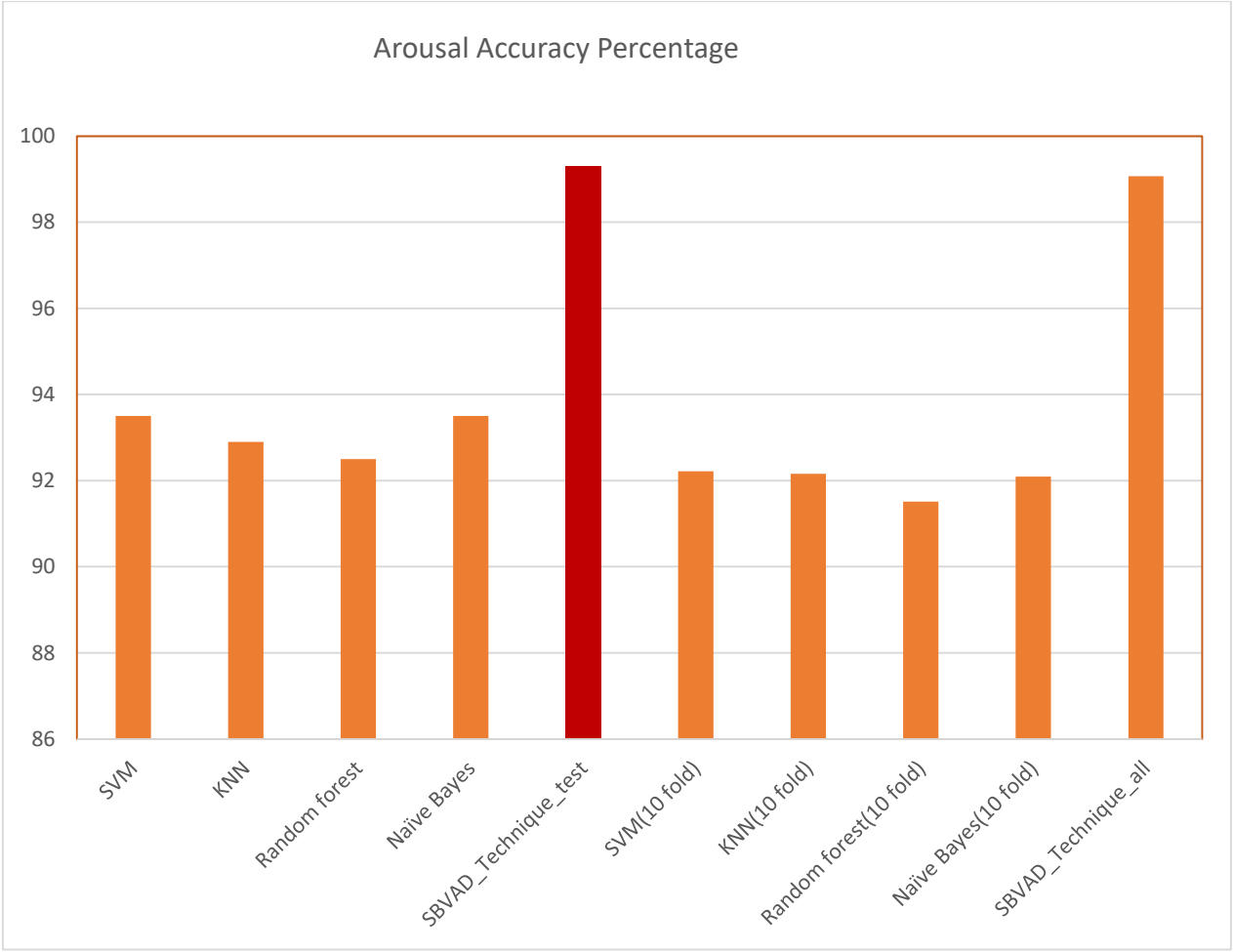

Figure 5.2: Comparison Bar Chart for Arousal Accuracy

The above chart shows that accuracy for Arousal is also higher (above 99%) for both versions of the proposed technique. In fact, they almost dwarf all other classifier values.

The Dominance scores are compared visually in the following figure 5.3.

This last figure indicates that the proposed technique fails to excel in case of Dominance determination, although the accuracy percentage is above 91% for the test data set. Even for the whole dataset the value is above 87%, which is not at all negligible. Here the winners are SVM and Naïve Bayes. In the future scope are discussed some probable methods to improve the score for the proposed technique.

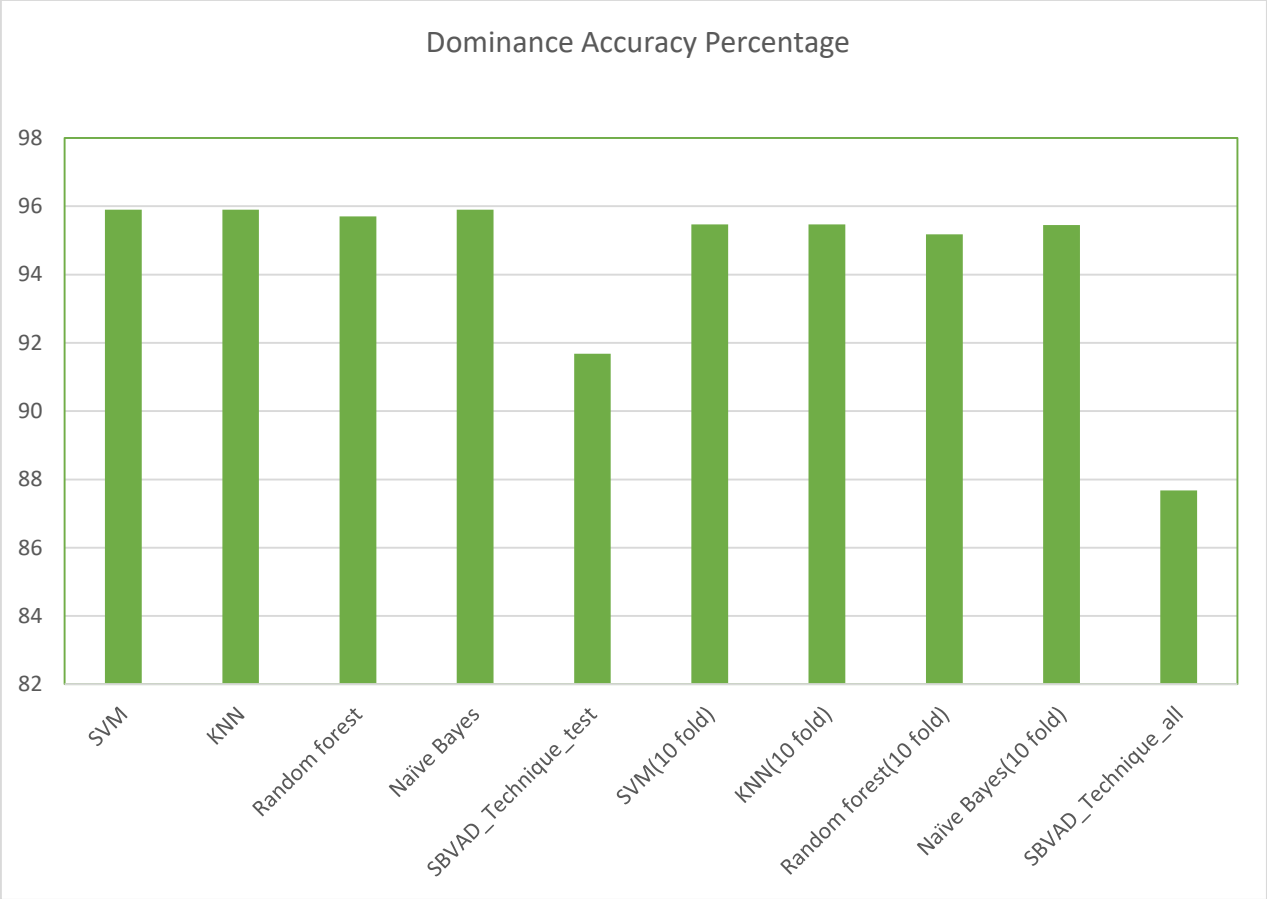

Figure 5.3: Comparison Bar Chart for Dominance Accuracy

Chapter 6

Conclusion and Future Scope

The present work set out to calculate three important sentiment features that can be obtained from natural language texts used by humans to express their emotional reactions. These are Valence, Arousal and Dominance. In the past, these have been evaluated manually by expert annotators and recorded in online dictionaries. But the process involved several disadvantages such as scarcity of experts, cost of employing them, time consumed in manual checking, and the chance of human bias and error. The calculations in the proposed technique, being mainly dependent on NLP based standard processes, helps to overcome all these adverse effects.

The experiments carried out in the work used a benchmark dataset (EmoBank) which has been utilized profitably by researchers in the past.[4, 5, 30] It contains 10k English sentences of different genre with special annotated values for the three sentiment features, and prefixed test train splits. The developer of the dataset themselves [5] expressed their scepticism over the sanctity of the annotation process. But the sizable amount of data definitely reduces the chances of errors.

The present work uses standard classification techniques over the dataset and highlights how the proposed technique out-performs those classifiers in evaluating the correct Valence and Arousal values for the majority of the dataset. In fact the Arousal value indicates a near-perfect accuracy. Even for the Dominance the value is quite high (near to 90%). The added advantage, of not needing any training phase dependent on manual class values, makes the technique more lucrative.

In future, text data from closed group conversations can also be used to assess the participants' mental condition by finding the VAD features using the proposed technique. Abnormal reactions can be traced by applying clustering techniques on the feature values obtained through the automated process proposed in this work.

So far the technique does not incorporate machine learning processes, other than those involved in the NLP techniques used at the outset. In future the researchers can think of introducing deep learning techniques to capture the underlying emotions in the sentences. The values of the VAD features may then hopefully be calculated with greater accuracy.

Reference

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

[2] https://www.datarobot.com/blog/introduction-to-sentiment-analysis-what-is-sentiment-analysis/

[3] Mehrabian, A., & Russell J., "Evidence for a Three-Factor Theory of Emotions", Journal of Research in Personality, 273-294, 1977

[4] https://github.com/JULIELab/EmoBank

[5] Buechel J. and Hahn U., "EMOBANK: Studying the Impact of Annotation Perspective and Representation Format on Dimensional Emotion Analysis", Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Vol. 2, 578-585, 2017.

[6] Osgood C., Suci G., and Tannenbaum P.,"The measurement of meaning". University of Illinois Press, 1957

[7] Russell J., "A circumplex model of affect", Journal of personality and social psychology, Vol. 39, No. 6, 1161-1178, 1980

[8] Russell J., "Core affect and the psychological construction of emotion", Psychological Review, Vol. 110, No. 1, 145-172, 2003

[9] Mehrabian A. and Russel J., "An approach to environmental psychology". Cambridge, Massachusetts: M.I.T Press, 1974

[10] Bradley M, and Lang P., "Affective norms for English words (ANEW): Instruction manual and affective ratings", University of Florida, 1999

[11] Warriner A., Kuperman V. and Brysbaert M., "Norms of valence, arousal, and dominance for 13,915 English lemmas", 2013

[12] Mohammad S., "Obtaining Reliable Human Rating of Valance, Arousal and Dominance for 20000 english word", National Research Council Canada, 2018

[13] Chuhan W., Fangzhao W., Sixing W., Zhigang Y., Junxin L., and Huang Y. "Semisupervised dimensional sentiment analysis with variational autoencoder", Knowledge-Based Systems, Vol. 165, 30-39, 2019

[14] Suyang Z., Shoushan L., and Guodong Z., "Adversarial attention modeling for multidimensional emotion regression", Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 471-480, 2019

[15] Akhtar S., Ghosal D., Ekbal A., Bhattacharyya P., and Kurohashi S., "All-in-one: Emotion, sentiment and intensity prediction using a multi-task ensemble framework", 2019

[16] Moors A., Houwer J., Hermans D., Wanmaker S., Schie K., Harmelen A., Schryver M., Winne J., and Brysbaert M., "Norms of valence, arousal, dominance, and age of acquisition for 4,300 dutch words", Behav Res 45, 169-177, 2013

[17] Melissa L., Conrad M., Kuchinke L., Urton K., Hofmann M., and Jacobs A., "The berlin affective word list reloaded", Behavior Research Methods 41, 534-538, 2009

[18] Redondo J., Fraga I., Padron I, and Comesana M., "The spanish adaptation of anew (affective norms for english words)", Behavior Research Methods 39, 600-605, 2007

[19] https://www.expert.ai/blog/machine-learning-definition/

[20] https://www.ibm.com/cloud/learn/natural-language-processing

[21] https://www.techtarget.com/searchenterpriseai/definition/natural-language-processing-NLP

[22] https://www.meltwater.com/en/blog/analyse-sentiment-with-media-intelligence

[23] https://www.upgrad.com/blog/types-of-sentiment-analysis/

[24] https://theappsolutions.com/blog/development/sentiment-analysis/

[25] Ohana B., Tierney B. "Sentiment Classification of Reviews Using SentiWordNet", Dublin Institute of Technology, 2009

[26] https://towardsdatascience.com/understanding-cosine-similarity-and-its-application-fd42f585296a

[27] https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm

[28] https://learn.g2.com/k-nearest-neighbor

[29] https://www.sciencedirect.com/topics/computer-science/random-forest-classifier

[30] Park S., Kim J., Ye S., Jeon J., Park H., Alice O., "Dimensional Emotion Detection from Categorical Emotion", arXiv:1911.02499v2 [cs.CL] 10 Sep 2021

Appendix

A.1 Python Code for Phase 1

```
import os
import re
import pandas as pd
from sklearn.feature_extraction.text import HashingVectorizer
from sklearn.neighbors import KNeighborsClassifier,KNeighborsRegressor
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB

data = pd.read_csv("emobank.csv")      # read EmoBank dataset

train_data = []
train_V = []
train_A = []
train_D = []
test_data = []
test_V = []
test_A = []
test_D = []

for i in range(len(data.index)):                    # split "text", "V", "A", "D" into train and test part
   if (data.loc[i]["split"] == "train"):
      train_data.append(data.loc[i]["text"])
      train_V.append(round(data.loc[i]["V"]))
      train_A.append(round(data.loc[i]["A"]))
      train_D.append(round(data.loc[i]["D"]))
   if (data.loc[i]["split"] == "test"):
      test_data.append(data.loc[i]["text"])
      test_V.append(round(data.loc[i]["V"]))
      test_A.append(round(data.loc[i]["A"]))
      test_D.append(round(data.loc[i]["D"]))

vectorizer = HashingVectorizer(norm = None, n_features = 20)     #convert "text" part into vector
sentence_vectors = vectorizer.fit_transform(train_data)
train_datavector = sentence_vectors.toarray()

sentence_vectors2 = vectorizer.fit_transform(test_data)
test_datavector = sentence_vectors2.toarray()


model = SVC(kernel='linear')                                    # call SVM classifier
model = KNeighborsClassifier(n_neighbors=10)                    # call KNN classifier
```

```
model = RandomForestClassifier(n_estimators=10)          # call Random Forest classifier
model = GaussianNB()                                     # call Naive Bayes classifier

model.fit(train_datavector, train_V)                     # fit classifier by train "text" and train "V"
print(model.score(test_datavector, test_V))              # test classifier by giving test "text" and test
                                                           "V" and get accuracy score

model.fit(train_datavector, train_A)                     # fit classifier by train "text" and train "A"
print(model.score(test_datavector, test_A))              # test classifier by giving test "text" and test
                                                           "A" and get accuracy score

model.fit(train_datavector, train_D)                     # fit classifier by train "text" and train "D"
print(model.score(test_datavector, test_D))              # test classifier by giving test "text" and test
                                                           "D" and get accuracy score
```

A.2 Python Code for Phase 2

```
import pandas as pd
from sklearn.feature_extraction.text import HashingVectorizer
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import cross_val_score
from numpy import mean

data = pd.read_csv("emobank.csv")                        # read EmoBank dataset

text = []
V = []
A = []
D = []

for i in range(len(data.index)):                         # appending "text", "V", "A", "D" parts into list
    text.append(data.loc[i]["text"])
    V.append(round(data.loc[i]["V"]))
    A.append(round(data.loc[i]["A"]))
    D.append(round(data.loc[i]["D"]))

vectorizer = HashingVectorizer(norm = None, n_features = 20)      # use hasing vectorizer to
                                                                    convert string into vector
text_to_vector = vectorizer.fit_transform(text)
vector_array= text_to_vector.toarray()
```

```
model = KNeighborsClassifier(n_neighbors=10)          # call KNN classifier
model = SVC(kernel='linear')                          # call SVM classifier
model = RandomForestClassifier(n_estimators=10)       # call Random forest classifier
model = GaussianNB()                                  # call Naive Bayes classifier

scores = cross_val_score(model, vector_array, V, cv = 10, scoring='accuracy')     # accuracy
                                                                                  score of Valence
print(mean(scores))

scores = cross_val_score(model, vector_array, A, cv = 10, scoring='accuracy')     # accuracy
                                                                                  score of  Arousal
print(mean(scores))


scores = cross_val_score(model, vector_array, D, cv = 10, scoring='accuracy')     # accuracy
                                                                                  score of Dominance
print(mean(scores))
```

A.3 Python Code for Phase 3

```
import os
import re
import nltk
import pandas as pd
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
pstream = PorterStemmer()

# VAD Calculation Modulo

f = open("SentiWordNet_3.0.0.txt", "r", encoding="UTF-8")       # read sentiwordnet dictionary
str = f.read()
sentence = str.split('\n')

f_list = []

for key in range(len(sentence)):
    word = sentence[key].split('\t')                                  # split by "    "
    dict = {"type": [], "pos_id": [], "pos_score": [], "neg_score": [], "word": []}
    dict["type"] = word[0]                                          # keep first five necessary value
    dict["pos_id"] = word[1]
    dict["pos_score"] = word[2]
```

```
    dict["neg_score"] = word[3]
    temp = word[4].replace("#", "")                    # remove '#'
    temp1 = re.sub(r'[0-9]', "", temp)                 # remove numbers
    dict["word"] = temp1
    f_list.append(dict)                                # append into list of dictionary

def derive_sentiment_score(search, value):             # to derive sentiment score of a word
    score = 0
    for i in range(len(f_list)):
        if (f_list[i]["word"] == search and f_list[i]["type"] == value):
            f_list[i]["pos_score"] = float(f_list[i]["pos_score"])
            f_list[i]["neg_score"] = float(f_list[i]["neg_score"])
            score = f_list[i]["pos_score"] - f_list[i]["neg_score"]
    return(score)

def derive_valence_score(search, value):               # to derive positive sentiment score of a word
    score = 0
    for i in range(len(f_list)):
        if (f_list[i]["word"] == search and f_list[i]["type"] == value):
            f_list[i]["pos_score"] = float(f_list[i]["pos_score"])
            f_list[i]["neg_score"] = float(f_list[i]["neg_score"])
            score = f_list[i]["pos_score"]
    return(score)

stop_word = stopwords.words('english')
lematizer = WordNetLemmatizer()

data = pd.read_csv("emobank.csv")                      # read EmoBank dataset

total_data = []
list_v = []
list_d = []
list_a = []

for i in range(len(data.index)):
    str = data.loc[i, "text"]
    str = str.lower()                                  # make data into lowercase
    data.loc[i, "text"] = re.sub(r'[^\w\s]', "", str)  # remove special characters
    list_data = data.loc[i, "text"].split(' ')
    str2 = ""
    for k in range(len(list_data)):
        if list_data[k] not in stop_word:              # remove stopwords
            str2 = str2 + ' ' + list_data[k]
    data.loc[i, "text"] = str2
    list_data = data.loc[i, "text"].split(' ')
    str3 = ""
```

```
    for j in range(len(list_data)):
        if (list_data[j] != ''):
            str3 = str3 + lematizer.lemmatize((list_data[j])) + ' '        # lemmatize the text
    data.loc[i, "text"] = str3

    v_score = 0
    d_score = 0
    a_score = 0
    length = 0

    str4 = data.loc[i, "text"].split(' ')
    for m in range(len(str4) - 1):
        item = str4[m]
        pos_value = nltk.pos_tag([item])                    # get pos tag
        str5 = pos_value[0][1]
        str6 = str5[0]                                      # get first letter of pos
        str6 = str6.lower()
        if(str6 =="j"):                 # since in sentiwordnet dictionary for adjective, tag is "a"
            str6 = "a"
        if(str6 == "a"):                # since in sentiwordnet dictionary for adverb, tag is "r"
            str6 = "r"
        score1 = derive_valence_score(item, str6)               # deriving Valence and Arousal
        if(score1 != 0):                                    # keep non-zero sentiment words in count
            length =length + 1
        v_score = v_score + score1

        score2 = derive_sentiment_score(item, str6)
        temp = abs(2*score2 - 1)
        a_score = a_score + temp

    if(length!=0):
        valence = v_score/length                # Valence of a sentence
    else:
        valence = 0
    if(len(str4)!=1):
        arousal = a_score/(len(str4)-1)         # Arousal of a sentence

    list_v.append(valence)
    list_a.append(arousal)
data["Test_v"] = list_v                         # store Valence of each sentence in a list
data["Test_A"] = list_a                         # store Arousal of each sentence in a list


for i in range(len(data.index)):                # deriving Dominance
    str11 = data.loc[i, "text"].split(' ')
    for r in range(len(str11)):                 # derive TAW
```

```
        if (str11[r] != ""):
            total_data.append((str11[r]))               # append all text part into list of whole data

dom_dict = {}
allwords = []

for p in range (len(total_data)):
    try:
        temp2 = total_data[p]
    except:
        l=0
    taw = 0
    pos_val = nltk.pos_tag([temp2])
    str7 = pos_val[0][1]                                # get only pos term
    str8 = str7[0]                                      # get first letter of pos
    str8 = str8.lower()
    if (str8 == "j"):                           # since in sentiwordnet dictionary for adjective, tag is "a"
        str8 = "a"
    if (str8 == "a"):                           # since in sentiwordnet dictionary for adverb, tag is "r"
        str8 = "r"
    scr = (derive_sentiment_score(temp2, str8))
    scr1 = abs(2*scr - 1)

    temp3 = pstream.stem(temp2)
    if temp3 not in allwords:
        allwords.append(temp3)
        dom_dict[temp3] = scr1                                  # TAW value of a word
    else:
        dom_dict[temp3] = dom_dict[temp3] + scr1            # TAW value of a word

val = dom_dict.values()
max_val = max(val)
count = 0
for key in dom_dict:
    dom_dict[key] = dom_dict[key]/max_val                   # deriving dominance factor of each
word

for i in range(len(data.index)):
    str10 = data.loc[i, "text"].split(' ')
    d_score = 0
    for m in range(len(str10) - 1):
        item =  pstream.stem(str10[m])
        temp10 = dom_dict[item]
        d_score = d_score + temp10
    if (len(str10) != 0):
        dominence = d_score/(len(str10))                    # Dominance score of a sentence
```

```
    list_d.append(dominence)

data["Test_D"] = list_d                                  # store Dominance of each sentence in a list

data.to_csv('result.csv')              # crating new file which contain the data of original file
                                        and VAD score of each sentence derived statiscal based formula

# VAD comparison Modulo

data1 = pd.read_csv("result.csv")                  # read the new file
l1 = []
l2 = []
l3 = []
l4 = []
l5 = []
l6 = []

for i in range (len(data1.index)):       # storing EmoBank VAD value and derived VAD value
    if(data1.loc[i]["Test_v"] != 0):
        l1.append(data1.loc[i]["Test_v"])
        l2.append(data1.loc[i]["V"])
        l3.append(data1.loc[i]["Test_A"])
        l4.append(data1.loc[i]["A"])
        l5.append(data1.loc[i]["Test_D"])
        l6.append(data1.loc[i]["D"])

from scipy import spatial
from numpy import unique
import numpy as np

result = 1- spatial.distance.cosine( l1,l2)              # accuracy score of Valence
print(result)
result = 1 - spatial.distance.cosine( l3,l4)             # accuracy score of Arousal
print(result)
result = 1 - spatial.distance.cosine( l5,l6)             # accuracy score of Dominance
print(result)
```