Investigations on Text-level Psychological Stress Detection Approaches

Submitted in partial fulfilment of the requirements

for the award of the degree of

DOCTOR OF PHILOSOPHY

by

K V T K N Prashanth

(Roll No. 717041)

Under the supervision of

Dr. Tene Ramakrishnudu

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
WARANGAL - 506004, TELANGANA, INDIA
APRIL 2024

Dedicated to

My Parents K Ram Mohan Rao and K E Rama Devi

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
WARANGAL - 506004, TELANGANA, INDIA

THESIS APPROVAL FOR Ph.D.

This dissertation work entitled, Investigations on Text-level Psychological Stress Detection Approaches, submitted by Mr. K V T K N Prashanth (Roll No. 717041) is approved for the degree of DOCTOR OF PHILOSOPHY at the National Institute of Technology Warangal.

Examiners

Research Supervisor

Dr. Tene Ramakrishnudu
Dept. of Computer Science and Engg.
NIT Warangal, India

Chairman

Prof RBV Subramaanyam
Dept. of Computer Science and Engg.
NIT Warangal, India

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY WARANGAL
WARANGAL, TELANGANA, INDIA - 506004

CERTIFICATE

This is to certify that the thesis entitled, Investigations on Text-level Psychological Stress Detection Approaches, submitted in partial fulfillment of requirement for the award of degree of DOCTOR OF PHILOSOPHY to National Institute of Technology Warangal, is a bonafide research work done by Mr. K V T K N Prashanth (Roll No. 717041) under my supervision. The contents of the thesis have not been submitted elsewhere for the award of any degree.

Research Supervisor

Dr. Tene Ramakrishnudu
Associate Professor
Department of CSE, NIT Warangal, India

Place: NIT Warangal
Date: April 18, 2024

DECLARATION

This is to certify that the work presented in the thesis entitled "*Investigations on Text-level Psychological Stress Detection Approaches*" is a bonafide work done by me under the supervision of Dr. Tene Ramakrishnudu and was not submitted elsewhere for the award of any degree.

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/date/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

K V T K N Prashanth

(Roll No. 717041)

ACKNOWLEDGMENTS

It is with great pleasure that I acknowledge my sincere thanks and deep sense of gratitude to my supervisor, Dr. Tene Ramakrishnudu sir, from whom I have learned so much about how to conduct oneself as a teacher. His values as a teacher and as a human being, are something I would like to take forward both in my professional and personal life. He has been very supportive to me throughout my research period and constantly encouraged me to do good work as a researcher. Because of him, I flew with my thoughts in accomplishing my ideas, especially in solving the problem of text-level stress detection through social media. He has always given me ample time for research discussions, technical help, and the revision of the work. I feel lucky to be associated with such a great human being in my life. I can strongly say that, if I would not have done my PhD under his supervision, probably, I might terminate my PhD incompletely.

I extend my gratitude to all my Doctoral Scrutiny Committee members: Prof. R.B.V. Subramaanyam, Prof. P. Radha Krishna, Prof. D. V. L. N. Somayajulu (for the first four semesters), Dr. S. Ravi Chandra and Dr. K. Venkata Reddy (Civil Engineering) for their insightful comments and suggestions during oral presentations.

I am immensely thankful to Prof. R.B.V. Subramaanyam, and Prof. P. Radha Krishna, and Dr. Tene Ramakrishnudu (my thesis supervisor), Heads of the Department of CSE during my stay in the department, for providing adequate facilities. I wish to express my thanks to the faculty members of the Computer Science and Engineering department.

I thank my colleagues and friends at NITW, Sanjib Raul, Umamaheswara Sharma, Pavan Kumar, Santosh, Shalini PV, Chandra Mohan, Manoj, Ravi Kant, Asif, Rampavan, Punam Chander, Sudarshan, Swetha, Satish Vemireddy, Varsha, Abhilash, Preethi, Hem Kumar, Vinay Raj, and Amar Kumar who have made my life at NITW easy and helped in their own ways to improve my research. I also thank colleagues under our supervisor, Poonam, Ravikanth and Dheeraj for their valuable support and

help.

I thank my close friends outside of NITW, Rashmi, Sarvaani, Bhupal, and Raghuram for being there for me always.

I show my special gratitude to the faculty at NITW, Dr. Sanjaya Panda and Dr. Venkatesh Kagita with whom I had some fruitful discussions.

I thank my parents, K Ram Mohan Rao and Rama Devi, and my siblings, Akhila and Abhishek, today for their blessings and the constant support that they have been for me. There are not enough words to express how grateful I am to them for everything they have done for me. Since my childhood, my father has consistently reiterated the importance of education in life. My mother, an inspiring teacher, had been my moral guide and had inspired me to take up research. Though I have lost my mother, her teachings will last and support will remain forever. I am always in awe of the way my mother had managed multiple tasks at home and her school. She looked after all my needs, and my father planned my education with all the financial constraints our family had. My parents' aspirations and their unwavering commitment to offering me the finest support possible have enabled me to reach this educational level. And, I am showing my gratitude towards my mother for her selfless love and care on me since I was born, and she believed in me in every tough situation that I encountered.

Above all, I would like to thank my father K Ram Mohan Rao for his love and constant support, for all the late nights and early mornings, and for keeping me sane over the past few months. Thank you for being my moral support and taking the role of my mother. But most of all, thank you for being my best friend. I owe you everything. I strongly say without the support (mental, motivational, and financial) of my father, I may never have completed this thesis. I am grateful to my colleague Sanjib Kumar Raul, for his constant moral support. I specially thank my maternal aunt, Dr D Sharada for her guidance and moral support.

K V T K N Prashanth

ABSTRACT

Psychological stress has engulfed the world and has turned out to be a major propellant for both physiological health disorders and suicides across the globe. The more concerning issue is the prevalence of psychological stress among youth and the increasing vulnerability of the young generation to stress. Hence, the detection of stress, before it becomes chronic, is of paramount importance. There exist many traditional stress detection mechanisms like interviews with psychiatrists, questionnaires, etc. However, the social stigma attached to these methods, makes people either avoid them or give incomplete or misleading information. Furthermore, the methods to detect stress using electronic devices and sensors are seen as invasive to daily life. Hence, this leads to searching for alternative approaches to detect stress early by capturing genuine emotions without any stigma. Social media data, with the growing popularity of social media usage, can be an ideal candidate in such a scenario. In micro-blogging sites like Twitter (now called X), people participate in a large scale to express their opinions, and daily activities in a free manner devoid of any social stigma. This makes Twitter a very lucrative resource for capturing human emotions and therefore, social media-based text-level stress detection has caught the attention of the researchers. The initial approaches using social media to detect text-level stress have focused on crowd-sourcing of the data for collection and labeling. Nevertheless, the crowd-sourcing approach is prone to a problem similar to that of questionnaires, while the manual labeling is laborious, time-consuming, and vulnerable to errors. To address this, automatic labeling based on sentence patterns of "I feel" was proved to be effective in labeling the social media data for stress detection. Using this strategy to collect tweets, many works in the literature had proposed tweet-level stress detection methods. These techniques have also developed more detailed hand-crafted features from the text data of tweets. Furthermore, deep-learning-based methods like stacked cross autoencoders for the classification of tweet-level stress, apart from traditional machine learning techniques, were also developed.

The literature of text-level stress detection problem has many gaps despite provid-

ing the initial solutions. First, there is an issue of data sparsity; most of the tweets are short in size, and hence existing approaches do not utilize text content of the data, content of previous or neighborhood tweets, and clues from the text data about sarcasm. Second, the lack of a large amount of labeled data for stress detection using social media makes it difficult to implement supervised algorithms. This gives motivation to explore solutions based on semi-supervised learning methods which require a lesser amount of labeled data. Third, the concept of sarcasm is found to be one of the useful attributes in detecting tweet-level stress, and with sarcasm being a classification problem on its own, there is a scope to study and develop multi-task learning approach to detect stress with the help of the auxiliary task of sarcasm detection. In this thesis, the issues mentioned are addressed using various machine learning-based solutions. Initially, a new approach called neighborhood-based tweet-level stress detection(NTSD), a modified logistic regression that includes neighborhood tweets, is developed to utilize the text content of the tweets for detecting stress. In addition, a new attribute called *Sarcasm_Level* is proposed which captures the sarcasm present in the text content of the tweet. These solutions help to address both the data sparsity problem and the utilization of the text by capturing the related concept of sarcasm. However, it has an overhead due to the additional requirement of the neighborhood tweets. To overcome this, later, a new method called sarcasm-based tweet-level stress detection (STSD) was developed, wherein sarcasm is used to develop the modified logistic regression such that the loss of sarcastic tweets, which reflect positive and friendly moments, is penalized. Simultaneously, the loss of the non-sarcastic tweets is minimized. This makes the proposed STSD perform better without the requirement of extra data like neighborhood tweets.

In the real world, most of the data is unlabeled in many scenarios, and on the contrary, most of the approaches for tweet-level stress detection are supervised models. To address this issue, a new semi-supervised approach based on logistic regression, called semi-supervised method for tweet-level stress detection (SMTSD) is developed. This utilizes semi-labeled data and the concept of sarcasm in computing the pseudo-labels, thereby improving the performance of tweet-level stress detection. Though

many approaches consider sarcasm as a useful attribute in the detection of stress at tweet-level, sarcasm detection is a classification task in itself and is related to emotion and stress detection. To this end, a multi-task approach for tweet-level stress detection (MATSD) is proposed where deep neural networks are utilized for the prediction of stress as the primary task using long short term memory (LSTM) and sarcasm as an auxiliary task using convolutional neural network (CNN), sharing a joint map layer. The simultaneous prediction of the tasks act as regularization and helps in the betterment of the efficacy of tweet-level stress prediction. To conclude, from this research, we propose different solutions for text-level stress detection by maximizing the utilization of text data.

Keywords: Psychological stress, social media, twitter, data sparsity, tweet-level stress detection, neighborhood tweets, sarcasm, logistic regression, PCA, kernel-PCA, semi-supervised learning, self-training, multi-task learning, CNN, LSTM.

Contents

ACKNOWLEDGMENTS i

ABSTRACT iii

List of Figures xii

List of Tables xv

List of Algorithms xvi

List of Abbreviations xviii

1 Introduction 1

1.1 Motivation and Objectives 6

1.2 Overview of Contributions of the Thesis 8

1.2.1 Neighborhood-based tweet-level psychological stress detection 8

1.2.1.1 Formal description 9

1.2.1.2 Research Findings: 9

1.2.2 Sarcasm-based tweet-level psychological stress detection 10

1.2.2.1 Formal Description 11

1.2.2.2 Research Findings: 11

1.2.3 Semi-supervised approach for tweet-level psychological stress detection 12

1.2.3.1 Formal Description 12

1.2.3.2 Research Findings: 13

1.2.4 A multi-task learning-based approach for tweet-level stress detection . . . 13
1.2.4.1 Formal Description . . . 14
1.2.4.2 Research Findings: . . . 14
1.3 Performance Metrics used in the thesis . . . 15
1.4 Organisation of the Thesis . . . 16

2 Literature Survey **18**
2.1 Social-media-based stress detection approaches . . . 18
2.1.1 Issues with traditional methods of stress detection . . . 18
2.1.2 Early social media-based approaches . . . 19
2.1.3 Deep learning-based text-level stress detection approaches 20
2.1.4 Need for Semi-supervised approach for tweet-level stress detection . . . 22
2.2 Role of Sarcasm and dimensionality reduction . . . 22
2.2.1 Sarcasm . . . 22
2.2.2 Importance of sarcasm in stress detection . . . 23
2.2.3 Dimensionality reduction . . . 25
2.3 Multi-task approach for tweet-level stress detection . . . 26
2.3.1 Multi-task approaches in related problems . . . 26
2.3.2 Sarcasm as an auxiliary task in related problems . . . 27
2.4 Summary . . . 28

3 Neighborhood-based Tweet-level Stress Detection **29**
3.1 Problem Formulation . . . 29
3.2 Methodology . . . 32
3.2.1 The Framework of the Model . . . 32
3.2.1.1 Tweet's features . . . 32
3.2.2 Preprocessing . . . 35
3.2.3 Proposed Model . . . 35
3.2.4 Training the Model . . . 37

3.2.5 Prediction . . . 37
3.3 Experimental Setup . . . 40
3.3.1 Dataset Collection . . . 41
3.3.1.1 Process of data collection . . . 41
3.3.1.2 Details of the datasets . . . 41
3.3.2 Comparison with Baseline Machine Learning Classifiers 42
3.4 Results . . . 43
3.4.1 Evaluation of Stress Detection Performance and Impact of Neighborhood Tweets . . . 44
3.4.2 Contribution of the *Sarcasm_Level* Attribute: An Analysis 46
3.5 Discussion . . . 50
3.5.1 Limitations of the proposed NTSD model . . . 50
3.6 Summary . . . 51

4 Sarcasm-based Tweet-level Stress Detection 52
4.1 Problem Formulation . . . 52
4.1.1 Problem Statement . . . 53
4.2 Methodology . . . 55
4.2.1 Data Collection . . . 55
4.2.1.1 Pre-processing . . . 56
4.2.2 Approach . . . 57
4.2.2.1 Feature Extraction . . . 58
4.2.2.2 Computing Sarcasm_State value . . . 58
4.2.3 Proposed STSD . . . 59
4.2.3.1 Framework . . . 59
4.2.4 Working of Sarcasm-based Tweet-level Stress Detection (STSD) . . . 61
4.2.4.1 Mathematical formulation of training of the proposed STSD model . . . 61

4.2.4.2 Steps of the proposed STSD framework and its training algorithm . . . 64
4.2.4.3 Dimensionality Reduction . . . 68
4.3 Experimental Setup . . . 69
4.3.1 Dataset Description . . . 69
4.3.2 Models utilized for comparison . . . 70
4.4 Results . . . 70
4.4.1 Visualization of PCA results . . . 71
4.4.2 Evaluation of the Proposed STSD Model . . . 71
4.4.2.1 Analyzing the effect of using dimensionality reduction with STSD . . . 75
4.5 Discussion . . . 78
4.5.1 Comparison with state-of-the-art . . . 78
4.5.2 Limitations of the proposed STSD model . . . 79
4.6 Summary . . . 79

5 Semi-supervised Approach for Tweet-level Stress Detection 81
5.1 Problem Formulation . . . 81
5.2 Methodology . . . 84
5.2.1 Features Extraction . . . 84
5.2.1.1 Computation of Sarcasm . . . 84
5.2.2 Self-training Method for Tweet-level Stress Detection (SMTSD) 85
5.2.2.1 Training Algorithm . . . 87
5.2.3 Prediction . . . 89
5.3 Experimental Setup . . . 90
5.3.1 Data collection . . . 90
5.3.2 Baseline supervised methods for comparison . . . 94
5.4 Results . . . 96
5.5 Discussion . . . 104
5.5.1 Limitations of the proposed SMTSD model . . . 106

5.6 Summary 106

6 Multi-task Learning Approach for Tweet-level Stress Detection Using Deep Learning 107

6.1 Problem Formulation 108
6.1.1 MATSD: Problem Statement 108
6.2 Methodology 110
6.2.1 MATSD: Framework 110
6.2.2 MATSD : Architecture 110
6.2.2.1 Data Collection, labeling, and preprocessing . . 110
6.2.2.2 Word Embedding 112
6.2.2.3 Sarcasm subnet - Convolution Layer 112
6.2.2.4 Sarcasm subnet - ReLU and Maxpool 112
6.2.2.5 Sarcasm subnet - Sigmoid 113
6.2.2.6 Joint Map Layer 113
6.2.2.7 Sarcasm subnet - Task specific fully-connected (FC) layer and Softmax 113
6.2.2.8 Stress subnet - LSTM Layer 114
6.2.2.9 Stress subnet - Task specific fully-connected(FC) layer and Softmax 114
6.2.3 Multi task context 115
6.2.4 Training 116
6.3 Experimental Setup 117
6.3.1 State-of-the-art methods used for comparative analysis 119
6.4 Results 122
6.4.1 Performance evaluation of the multi-task learning models over the single task learning approaches 122
6.4.2 Performance evaluation of stress detection using proposed MATSD over the state-of-the-art multi-task learning methods 123

6.4.3 The performance of the auxiliary task of sarcasm 123
6.5 Discussion 126
6.5.1 Analysis of loss function behavior for the proposed model 126
6.5.2 Limitations of the proposed MATSD model 129
6.6 Summary 129

7 Conclusion and Future Directions **131**
7.1 The Major Outcomes of the Thesis 132
7.2 Future Directions 133

Bibliography **135**

List of Publications **144**

List of Figures

1.1 An example of sarcastic and non-sarcastic tweet's text 7

3.1 Framework of the NTSD model 32

3.2 Box plots for F1-Score Results of classifiers LR, SVM, RF and NTSD. (a) & (b) present F1-Scores of the classifiers analyzing the effect of neighborhood tweets, performed on datasets DS1 and DS2, respectively. (c) & (d) depict F1-Scores of the classifiers analyzing the effect of the new attribute, *Sarcasm_Level* on datasets DS1 and DS2, respectively and neighborhood tweets on both datasets 47

3.3 Accuracy Results of Models on dataset1 48

3.4 Accuracy Results of Models on dataset2 48

3.5 F1-Score Results of Models on dataset1 48

3.6 F1-Score Results of Models on dataset2 49

4.1 Framework of the STSD model 60

4.2 The projection of Dataset D1 on two principal components after different PCA techniques 72

4.3 The projection of Dataset D2 on two principal components after different PCA techniques 72

4.4 The projection of Dataset D3 on two principal components after different PCA techniques 72

4.5 The projection of Dataset D4 on two principal components after different PCA techniques 73

4.6 The bar-plots representing the accuracies of all models on the datasets D1, D2, D3, and D4 76

4.7 The bar-plots representing the F1-measure values of all models on the datasets D1, D2, D3, and D4 77

5.1 Framework of the SMTSD model 91

5.2 The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D1. 97

5.3 The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D2. 97

5.4 The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D3. 98

5.5 The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D4. 98

5.6 The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D1. 99

5.7 The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D2. 99

5.8 The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D3. 100
5.9 The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D4. 100

6.1 Framework of the proposed MATSD model 111
6.2 Illustration of Joint Map Layer 111
6.3 Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS1 127
6.4 Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS2 128
6.5 Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS3 128
6.6 Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS4 128

List of Tables

3.1 Notations used in the NTSD model and their definitions 30
3.2 The datasets and their description 42
3.3 Model results for dataset1 - With basic attributes 44
3.4 Model results for dataset1 - By considering Saracasm_Level attribute 44
3.5 Model results for dataset2 - With basic attributes 44
3.6 Model results for dataset2 - By considering Saracasm_Level attribute 44

4.1 Description of the symbols employed in STSD 54
4.2 The Details of the Datasets 55
4.3 The number of positive and negative sentiment words in each dataset 55
4.4 Performance results for experiments on all four datasets - with original features 75
4.5 Performance results for experiments on all four datasets - By applying polynomial kernel PCA 75
4.6 Accuracy recorded by NTSD and the proposed STSD on dataset D1 . 78

5.1 Description of the symbols employed in SMTSD 83
5.2 Examples of the variables used to compute the sarcasm 85
5.3 The Details of the Datasets 93
5.4 Distribution of the class-labels in the Datasets 93
5.5 The distribution of positive and negative polarity among words, emojis and hashtags 94
5.6 The sarcasm distribution Details of the Datasets 94

5.7 Accuracy and F1-Score recorded for models implemented in this chapter. The results of baseline models correspond to the sampling of combined data considered being 1.0 96
5.8 The number of pseudo-labels included in the final combined data . . 103
5.9 Role of Sarcasm- Results 105

6.1 Notations utilized in MATSD 109
6.2 The description of the distribution of stress labels in the datasets . . 119
6.3 The description of the distribution of sarcasm labels in the datasets . 119
6.4 The results of the proposed and baseline models on dataset DS1 . . . 124
6.5 The results of the proposed and baseline models on dataset DS2 . . . 125
6.6 The results of the proposed and baseline models on dataset DS3 . . . 125
6.7 The results of the proposed and baseline models on dataset DS4 . . . 125

List of Algorithms

1 Neighborhood tweet-based Stress detection 38

2 Gradient Ascent Algorithm for training NTSD model 39

3 Stochastic Gradient Ascent Algorithm for training NTSD model 39

4 Sarcasm-based tweet-level stress detection 65

5 Gradient Descent Algorithm for training of STSD Model 67

6 Stochastic Gradient Descent Algorithm for training of STSD Model . . 67

7 Computing_Sarcasm 86

8 Proposed self-training semi-supervised based approach 92

9 Proposed Multi-task learning framework for tweet level stress detection 118

List of Abbreviations

NTSD	Neighborhood Tweet-based Stress Detection
STSD	Sarcasm-based Tweet-level Stress Detection
SMTSD	Self-training Method for Tweet-level Stress Detection
MATSD	Multi-task approach for tweet-level stress detection
LR	Logistic Regression
SVM	Support Vector Machines
RF	Random Forests
NB	Naive Bayes
CNN	Convolutional Neural Network
LSTM	Long Short Term Memory
Bi-LSTM	Bi-directional Long Short Term Memory
DNN-CAE	Deep neural Network – Cross Auto Encoder
GRU	Gated Recurrent Units
Bi-GRU	Bi-directional Gated Recurrent Unit
PCA	Principal Component Analysis
GloVe	Global Vector for word representation
FC	Fully Connected

Chapter 1

Introduction

The changing lifestyle across the globe has been a major propellant for health and psychological disorders among people, especially youngsters. This is more pronounced during the pandemic, where, sudden and harsh social isolations and lockdowns were commonplace s [1]. Psychological stress is a physiological phenomenon of being incompatible with the surrounding environment [2]. The events that trigger the stress are termed stressors [3]. Psychological stress, if it becomes chronic, can be a serious amplifier of other physical health issues or diseases [4]. Also, depression has become a general illness throughout the world with more than 264 million people suffering from it [5]. More than half of the world's population is experiencing a noticeable rise in stress, as per the worldwide survey reported by *Newbusiness* in 2010[1].

Chronic stage of stress, which arises due to the persistence of stress with no mitigation at early stages, can lead to multiple health problems and also affects sleep patterns [6, 7]. According to the China Center for Disease Control and Prevention,[2] extreme stress is concluded as a major aspect for suicide, which has become a top cause for the death of youngsters in China. The COVID-19 pandemic had further enhanced the prevalence of stress among people, especially among women [1]. In the second and third most populous countries of the world, India and the USA, the

[1]https://www.newbusiness.co.uk/articles/entrepreneurs/mark-dixon-how-keep-staff-happy-and-motivated

[2]https://theweek.com/articles/457373/rise-youth-suicide-china

stigma for COVID-19 and other related matters had acted as a major stressor causing anxiety among the people [8]. The consequences of chronic stress have been found to be taking a huge toll on health with various physiological effects [9]. Moreover, it is found that among youth, stress has both short-term and long-term effects on health [10]. Stressors are the events that act as the cause of stress [11]. The early detection of stress and identifying relevant stressors could prevent the persistence of long-term stress. Compared to the sources in the traditional ways of detecting stress, social media has evolved as a major pool of data for human emotions [12]. Moreover, the long period of the pandemic has furthered the vulnerability to negative emotions and psychological anxiety across the globe [13]. Hence, it is required to detect stress among people beforehand, especially in preventing the chronicity of stress. The early detection and diagnosis of stress would reduce the chances of stress turning chronic.

There exist many traditional methods to detect stress. The traditional methods to detect stress employed electronic monitoring of the physiological activity apart from well-known methods like interviews with psychiatrists, questionnaires, surveys, etc [14–16]. In interviews with psychiatrists and the questionnaires, the process is seen with social stigma by the patients and the affected [17–19]. Hence, in most cases, these classical procedures of interviews and questionnaires, suffer from one of the two problems- either the interviews would be avoided, or incorrect information would be given. This makes interviews and questionnaires less dependable. The traditional method of detecting stress using electronic monitoring of physiological signals are not popular as they are seen as invasive to normal life. Furthermore, these electronic device-based methods to detect stress are not cost-effective [17].

The traditional techniques, as seen earlier, are beleaguered with the challenge of social stigma, less participation, and inaccurate data. In this scenario, there is a need to detect stress through a mechanism where there exists voluntary participation of people with a relatively lesser social taboo. Social media, especially microblogging sites like **X** (formerly, Twitter)[3] is well-known for the large participation of the public [20] on their own discretion. Twitter and other related social media platforms like

[3]In this thesis, we use the old term Twitter as a substitute for the current name **X**.

Sina Weibo have changed the mode of virtual interaction among the public and paved the way for a consistent increase in the users of social media. They provide free and fair media where users express their feelings and opinions with open minds and hardly hide or fear social ramifications. Moreover, Twitter also captures the users' opinions on daily routines like lunch, meetings, interaction with friends, etc, in a timeline. Hence, with the huge rise in their popularity, social media sites like Twitter are an ideal choice for capturing the opinions of users with reliability [19]. In addition, they provide an easy and effective way to collect the data through many tools.

Though social media is vulnerable to misinformation, it is an important part of networking life where instant sharing of information, messages, and bringing up issues is possible [20]. Therefore, using social media for stress detection is both feasible and accurate [19]. However, this has many challenges throughout the data mining pipeline - from the collection and cleaning of the data to the building of efficient models and interpretation of the results. Various works in the literature of stress detection using social media had addressed many of these problems [18, 19, 21–26].

The usage of social media for expressing opinions about COVID-19 pandemic depicts the potential of social media as an important source of public opinion data. COVID-19 had devastated the daily lives of people all over the world, with more than 409 million people infected and over 5.7 million deaths reported globally until January 31, 2022[4]. The pandemic has also changed the lifestyle due to isolation and quarantines, which, along with an unhealthy diet, intensify stress, resulting in an increase in cardiovascular illnesses, especially among women [1]. Also, the world's second and third most populous countries, India and the United States of America, have been severely affected by the second and third waves of COVID-19, respectively [8, 27] [5]. And the impact was observed in social media usage as well as in the population [28, 29]. Unsurprisingly, people during the period of isolation and quarantine have increased their usage of social media communication with peers for mobilizing emotional support—a process called the buffer effect [30]. Moreover, it

[4] https://covid19.who.int/

[5] https://www.sciencedaily.com/releases/2022/02/220210154230.htm

is observed that due to the ripple effect of COVID-19, the pandemic is acting as a stressor [6], impacting psychological stress on people of various walks of life, ranging from students to migrant workers [31]. The greater reach of social media allowed larger sections of society to express themselves. However, with the change in lifestyle owing to the measures taken as part of COVID-19 control, there is an increased trend of expressing negative emotions on social media [32]. The surge in negative emotions reflected in tweets related to COVID-19 gives an opportunity to automatically detect text-level stress. It is also understood that text features have the greatest individual contribution to stress detection [17, 19]. All this further reinforce the importance of social media as the source for detecting psychological stress.

With the gaining popularity of social media like Twitter and the need for early diagnosis of stress, social media-based stress detection is an area of increasing research interest [17–19]. The problem of detecting stress from a given social media posting (tweet, in the case of Twitter) is modeled as a classification problem and termed tweet-level or text-level stress detection [18]. Early methods of text-level stress detection relied on crowd-sourcing for data collection. In addition, the labeling for the data was either manual or through questionnaires [18, 19, 21–26]. Also, the initial methods of stress detection were of binary classification by analyzing a given text content of a tweet [21, 22]. Later approaches used multi-class models to classify a text into different categories of stress [18, 24, 25]. In addition, the transmission of misinformation and fake news about the epidemic via social media has long-term consequences, particularly in terms of psychological health [33]. To this end, research in social media-based stress detection has recently gained a lot of traction over traditional stress detection techniques, as the latter suffer from many inconsistencies in data collection apart from social stigma [14–16]. Twitter data, especially tweets that contain textual information, was widely used to detect stress [18, 19, 24, 34]. Hence, in this thesis, tweet-level stress and text-level stress are used interchangeably.

To address the issue of text-level stress detection, social media-based stress detection methods gained traction in the research community [22, 23]. In early methods,

[6] Any event that acts as a trigger for stress is a stressor [3]

n-grams were used as features to represent text data and the problem is to detect the stress from a given snippet of text's features [21]. Hand-crafted features based on text were used to represent text data in other social media-based text-level stress detection methods [18, 25, 34]. Off late, deep-learning multi-modal approaches were also proposed for detecting text-level stress using social media posts [18]. The text content of the tweets had more role in determining the stress and therefore giving scope for utilizing more information from text content. Sarcasm, a way of expressing the content that contradicts the intention, or a kind of ironical statement, is said to have an influence on the human emotional usage [35]. It is understood that sarcasm is mostly used with known people by exchanging funny talk or actions [36]. Hence, employing the sarcasm recorded in a given text-content as an additional information or feature for determining stress will also be a case of increase in utilization of the text-content for tweet-level stress detection.

Furthermore, all of these methods address the problem of text-level stress detection using supervised learning methods [3, 18, 34, 37]. Moreover, in many real-world applications, it takes a great deal of effort to label the data given the fact that there is a scarcity of labeled data and an abundance of unlabeled data. This requires methods to utilize unlabeled data to improve the performance of supervised learning, which leads to semi-supervised learning. Hence, it is required to devise semi-supervised strategies for detecting text-level stress detection. Also, predicting sarcasm is a classification task on its own and is related to many human emotions. Hence, there is a scope to develop sarcasm as an auxiliary task that helps in detection of stress. Though deep learning-based solutions for the problem of text-level stress detection, the solutions are mostly based on single-task-based approaches and there is ample opportunity to utilize information from the related text-level classification problems for better utilization of text data [3, 18, 19, 37, 38]. All these issues are addressed as part of the objectives.

1.1 Motivation and Objectives

As mentioned earlier, the detection of stress through the text-contents of the user's social media postings is very important for the early diagnosis and prevention of stress. Social media microblogging sites like Twitter are suitable for capturing human emotions. Hence, text-level or tweet-level stress detection approaches are widely studied in the literature [17–19, 21–26]. However, the constraint on the size of the tweet, of at most 280 characters since 2017[7] acts as a major challenge. The size limit of tweets was 140 characters, earlier. According to Twitter, the additional space allotted for tweets did not increase the average size of tweets significantly. Nevertheless, this increase in the size of tweets improved the number of people who engaged online apart from an increase in the number of mentions [8]. However, the research in the domain of tweet-level stress detection concludes that the linguistic or text attributes are major contributors to the performance of detecting stress from tweets [17, 19]. This motivates us to explore the text content of the tweets for the betterment of the tweet-level classification of stress.

In addition, the concept of sarcasm is widely used for expressing human emotions. It is understood that sarcasm can be expressed in many ways in human and virtual communication [39–41]. The usage of sarcasm in expressing opinions and other cases shows that it is used mostly in a fun interaction with close pals or known persons [36]. Though there exist different forms of sarcasm in textual content, illocutionary sarcasm captures the contradictory sentiment between the facial expression and the sentence spoken [39, 40]. This illocutionary sarcasm can be adapted to the text contents of the tweets and its effect in improving the performance of tweet-level stress detection models could be studied. This gives ample motivation for exploring the effect of sarcasm on tweet-level stress detection approaches. It is understood that for the sarcastic tweets, which have inconsistency in emojis and text sentiment, the inherent and the explicit meanings conveyed contradict each other. This gives scope to explore the utilization of illocutionary sarcasm in detecting text-level stress detection. Figure

[7]https://blog.twitter.com/en_us/topics/product/2017/tweetingmadee asier.html
[8]https://www.theverge.com/2018/2/8/16990308/twitter-280-character-tweet-length

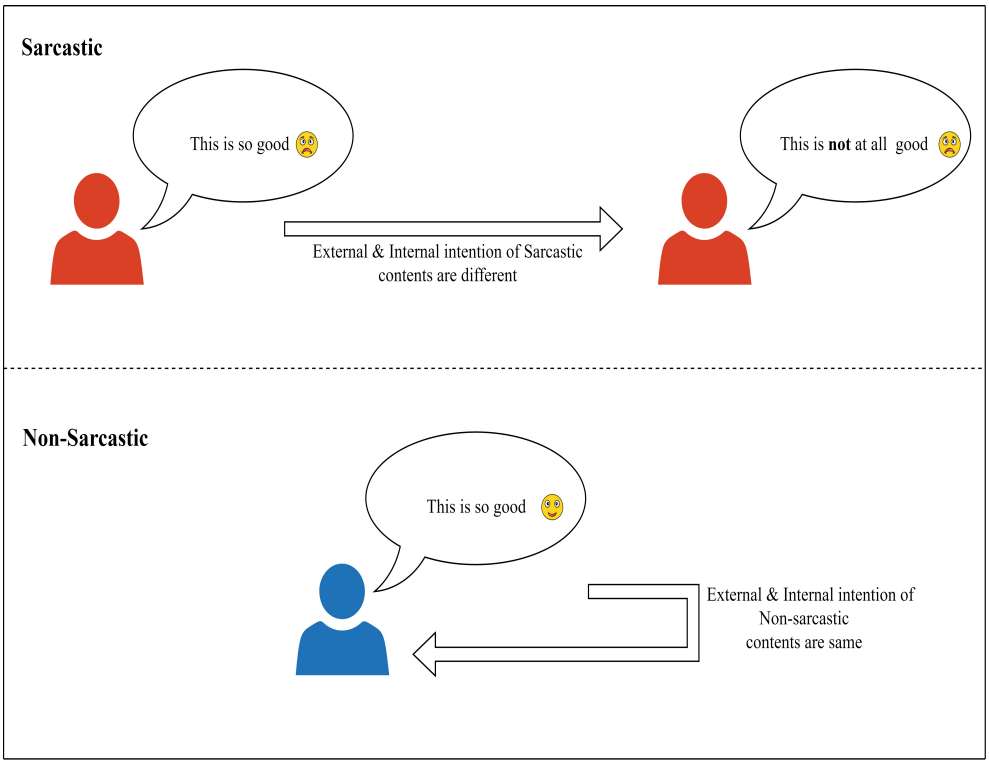

Figure 1.1: An example of sarcastic and non-sarcastic tweet's text

1.1 explains the concept of illocutionary sarcasm expressed through textual content and emojis.

Though many traditional and social media-based tweet-level stress detection solutions based on machine learning models are proposed in the literature, most of them are supervised learning approaches, requiring huge amounts of labeled training data [17–19, 42]. In many real-world problems, it is hard to collect large labeled data. As there is a scarcity of labeled data in many of the real-world problem domains related to text-level stress detection, it leaves room to explore the semi-supervised paradigms in building classifiers.

Though there exist few multi-task learning approaches for the detection of stress [43], the possibility of utilizing multi-task learning to detect tweet-level stress in conjunction with learning sarcasm, which is related to stress, is not explored in the literature [18, 19, 38, 42, 44, 45]. To better utilize information from text content, there is a scope to develop multi-task learning with stress detection as the primary task and a related task of sarcasm detection as the auxiliary task. This helps in better

sharing of parameters and extraction of content from the text.

All the gaps mentioned above act as important motivations for formulating the following objectives in this thesis:

1. Developing a model to detect tweet-level stress using neighborhood tweets. The neighborhood tweet-based stress detection (NTSD) model also incorporates a new feature called *Sarcasm_Level*.

2. Utilizing the concept of sarcasm to improve the performance of tweet-level stress detection by penalizing the tweets with sarcastic content. In addition, the dimensionality reduction techniques are utilized for better performance.

3. Developing a semi-supervised approach based on self-training for the tweet-level stress detection problem by utilizing the information of sarcasm in the tweets.

4. Developing a multi-task learning-based approach for detecting tweet-level stress with sarcasm detection as an auxiliary task.

1.2 Overview of Contributions of the Thesis

This section presents the chapter-wise contributions to the thesis. The following subsections summarize the contributions corresponding to each chapter.

1.2.1 Neighborhood-based tweet-level psychological stress detection

Tweet-level stress detection gained the interest of the research community. Nevertheless, the aforementioned reasons for size constraints on tweets pose a challenge of data sparsity. The objective of this contribution is to develop a neighborhood-based tweet-level stress detection (NTSD) mechanism that addresses the issue of data sparsity and also to include the concept of sarcasm as a feature for better utilization of a tweet's text information. For this purpose, initially, exploratory data analysis

is performed to develop evidence and intuition for the similarity in the distribution of tweets and their neighborhood tweets concerning a few statistics on class labels. Consequently, based on this intuition, a novel neighborhood-based tweet-level stress detection is developed and explained.

1.2.1.1 Formal description

Let D be the set of historical tweets collected from Twitter over a period of time, given by $\bigcup_i \mathbf{x_i}$, where $\mathbf{x_i}$ is i-th tweet of the dataset belonging to some user $u \in U$. Let $|D| = N$, where N represents the total number of tweets in the primary dataset and $y_i \in \{0, 1\}$ gives the stress label of tweet $\mathbf{x_i}$. Let, D_δ be the set of neighborhood tweets within the neighborhood window δ for each tweet in primary dataset, D. Formally, $D_\delta = \bigcup_{i=1}^{N} \bigcup_{j=1}^{N_\delta} \mathbf{x_j^i}$, where, $\mathbf{x_j^i}$ is the j-th previous tweet corresponding to the primary tweet $\mathbf{x_i}$ of user u under the neighborhood window δ and N_δ is maximum possible neighborhood tweets for any primary tweet allowed under the neighborhood window δ. And, the size of this dataset is $N < |D_\delta| \leq NN_\delta$. Therefore, based on the principle of NTSD, the proposed NTSD approach aims to optimize the following loss function

Therefore, based on the principle of NTSD, neighborhood tweets should also be considered in training the datasets for text-level stress detection , to compute weights $\mathbf{w}$ and predictions of logistic-regression-based prediction, the likelihood function for the proposed model is defined as follows:

$$f(\mathbf{w}) = \sum_{i=1}^{N} \Big\{ y_i \log(p(\mathbf{x_i})) + (1 - y_i) \log(1 - p(\mathbf{x_i})) \Big\} \quad + \sum_{i=1}^{N} \sum_{j=1}^{N_\delta} \Big\{ y_i \log(p(\mathbf{x_j^i})) + (1 - y_i) \log(1 - p(\mathbf{x_j^i})) \Big\} \quad (1.1)$$

1.2.1.2 Research Findings:

Based on the experiments conducted as part of the work on the datasets collected using tweepy API, it is observed that the neighborhood tweets of current tweets have

similar distribution on few statistics with respect to class labels. In addition, from the experiments executed on the proposed NTSD model and other baseline machine learning classifiers like LR, SVM, and RF, it can be noted that the proposed NTSD outperforms the other models significantly when previous tweets are included. Moreover, when the new attribute *Sarcasm_Level* was considered, there was a significant improvement in the performance of all the models.

1.2.2 Sarcasm-based tweet-level psychological stress detection

It is understood from the previous contribution that sarcasm as an attribute has an important role in improving the performance of the tweet-level stress detection model. Also, from the previous contribution, it can be seen that the problem of constraint in tweet's length is addressed through the model called NTSD. The usage of neighborhood tweets of the user for the current in detecting the stress of the current tweet forms the crux of NTSD. In addition, a new attribute called *Sarcasm_Level* was proposed to increase the utilization of the textual information. The results show that the usage of sarcasm and neighborhood tweets improved the performance, Nevertheless, this approach has its own shortcomings. One, the usage of neighborhood tweets makes the process data-intensive and computationally slow. In addition, the large number of hand-picked attributes affect the performance of the model. The sarcasm present in the tweet's text content can be utilized to better exploit the textual information present in the tweets. The objective of this contribution is to develop a sarcasm-based tweet-level stress detection method called sarcasm-based tweet-level stress detection, STSD, which uses information from sarcasm in logistic loss for better utilization of text content. The aim is to maximize the loss for sarcastic tweets and to minimize the loss for non-sarcastic tweets. This is due to the observed fact that sarcastic expression is mostly used to express happiness and exchange with friends [36]. In addition, it is also verified if the usage of the dimensionality reduction

techniques improves the performance of the model.

1.2.2.1 Formal Description

The principle of the proposed STSD is to utilize the information from sarcasm by penalizing the loss of sarcastic tweets. For this purpose, a new loss function based on logistic regression loss was proposed to include the sarcasm attribute. If $\mathbf{x_i}$ is a tweet feature vector and let the set of tweets, $\bigcup_i \mathbf{x_i}$ forms the training dataset D, whose corresponding class label is y_i, and if the *Sarcasm_value* of the tweet $\mathbf{x_i}$ is s_i, then the proposed STSD model learns weights by optimizing the following loss function:

$$f(\mathbf{w}) = -\sum_{i=1}^{N}(1-s_i)\Big\{y_i \log(p(\mathbf{x_i})) + (1-y_i)\log(1-p(\mathbf{x_i}))\Big\} + \sum_{i=1}^{N} s_i\Big\{y_i \log(1-p(\mathbf{x_i})) + (1-y_i)\log(p(\mathbf{x_i}))\Big\} \quad (1.2)$$

1.2.2.2 Research Findings:

Based on the experiments conducted as part of the work on the datasets collected using Tweepy API, it is observed that the proposed STSD model outperforms other basic ML models like LR, SVM, RF, etc. Apart from the above, the proposed model also records better performance than the NSTD approach with no previous tweets. In addition, it is observed that all the models record an improvement in performance after applying the dimensionality reduction techniques like kernel-PCA. It is noted that the proposed STSD records the highest improvement after application of polynomial kernel PCA. The case study conducted also validates that the proposed STSD outperforms all other basic ML models.

1.2.3 Semi-supervised approach for tweet-level psychological stress detection

Many studies in the literature on social of tweet-level stress detection utilize supervised learning mechanisms for building classifiers [17, 46]. However, real-world data is often largely unlabeled. Hence, the tweet-level stress detection mechanisms need to develop models in semi-supervised mechanisms for better handling real-world scenarios. The objective of this contribution is to develop a semi-supervised approach for the detection of stress at tweet-level by utilizing text content. To this end, in this chapter, a new model based on the self-training approach for the semi-supervised paradigm called the self-training method for tweet-level stress detection (SMTSD), is developed. This model utilizes the text information in the tweets' textual content to compute sarcasm value, which is later used to finalize the pseudo-labels computed as part of the self-training process. This approach, therefore, proposes semi-supervised solutions for tweet-level stress detection, which can be useful in real-world cases, where there is a paucity of adequate labeled data.

1.2.3.1 Formal Description

The principle of the proposed SMSTD is to utilize the information of sarcasm in tweet-level stress detection in a semi-supervised method of self-training approach. The concept is to invert the pseudo-labels of sarcastic tweets and append the modified pseudo-labeled data to the existing labeled dataset. Let $\mathbf{x_i^L}$ and $\mathbf{x_i^\nu}$ denote the Let D be the dataset containing a small amount of labeled tweet data, $D^L = \{\mathbf{x_i}^L, y_i^L\}$ for the tweet vector $\mathbf{x_i}^L \in D^L$ users $u \in U$, and a large amount of unlabeled tweet data $D^\nu = \{\mathbf{x_i}^\nu\}$, for the users $w \in U$. Also, let the set of tweet features in labeled training data is $D_X^L = \{(\mathbf{x}|(\mathbf{x}, y) \in D^L)\}$. If D_{TR}^L is the training part of $\tilde{D}$ is the pseudo-labeled data updated based on sarcasm values, then the combined loss function which the proposed SMSTD needs to be optimized is given as follows:

$$Combined\_Loss = \sum_{j=1}^{|D_{TR}^{L} \cup \tilde{D}|} l(y_j^*, \hat{g}(\mathbf{x_j}, \gamma)) \tag{1.3}$$

where, $l()$ is the cross-entropy function as defined in equation 5.2. Also, y_j^* is the labels of combined data, $D_{TR}^{L} \bigcup \tilde{D}$. While $\hat{g}$ is the sigmoid function for the given tweet vector, $\mathbf{x_j} \in D_{TR}^{L} \bigcup \tilde{D}, \quad \forall j \in \{1, 2, ...|D_{TR}^{L} \bigcup \tilde{D}|\}$

1.2.3.2 Research Findings:

The datasets used for this contribution are collected using Twitter tweepy API and a small part of them are labeled manually by experts. This partially labeled data is used to build the models. From the results obtained from the experiments conducted, it is noted that the proposed SMTSD model outperforms all the basic supervised models like LR, SVM, RF and NB. In addition, when the basic ML models are also used as base classifiers implemented in the semi-supervised self-training paradigm, the proposed SMTSD outperforms all other self-training approaches. Furthermore, the ablation study was conducted to view how the proposed SMTSD outperforms the basic traditional self-training approach without the inclusion of sarcasm.

1.2.4 A multi-task learning-based approach for tweet-level stress detection

Earlier contributions have shown the importance of illocutionary sarcasm as an attribute in both supervised and semi-supervised models. As sarcasm detection for a given piece of text turns out to be a classification problem, there is ample scope to consider sarcasm detection as a related task contributing to the task of early stress detection [17,42,46]. The objective of this chapter is to develop a deep-learning-based approach to tweet-level stress detection. In addition, the objective is to utilize the information from sarcasm in the text to help improve the performance of stress. To this end, a multi-task learning paradigm-based deep learning solution called Multi-

task approach for tweet-level stress detection (MATSD) is developed. In addition, the contribution of individual components of the architecture is also analyzed. For this purpose, an ablation study is performed. Also, the performance of the model needed to evaluated against the state-of-the-art techniques.

1.2.4.1 Formal Description

The proposed MATSD involves joint training of all the models (tasks) that are part of the multi-task framework. In order to achieve this, the loss of the multi-task model needs to be minimized. However, the loss of the model is characterized as the sum of losses of constituent tasks. Hence, in the MATSD, the total loss of the model is given as the weighted sum of the losses of tasks of sarcasm and stress. If $D = (\mathbf{x}_i, y_i^s, y_i^a)_{i=1}^N$ is the dataset of size N consisting of tweet ($\mathbf{x}_i$) along with the labels for both stress (y^s) and sarcasm (y^a), the total loss to be optimized by the MATSD is given by the following equation:

$$L_T(\alpha) \leftarrow \lambda_1 L_s + \lambda_2 L_a \tag{1.4}$$

Where λ_1 and λ_2 are the weights for the losses of tasks. While L_s and L_a are losses corresponding to stress and sarcasm tasks, respectively. They are defined based on the cross entropy loss [47] as follows:

$$L_s = \sum_{i=1}^{N} \left\{ y_i^s \log(h_{\beta,\theta}(\mathbf{x}_i)) + (1 - y_i^s)(\log(1 - h_{\beta,\theta}(\mathbf{x}_i))) \right\} \tag{1.5}$$

$$L_a = \sum_{i=1}^{N} \left\{ y_i^a \log(h_{\gamma,\theta}(\mathbf{x}_i)) + (1 - y_i^a)(\log(1 - h_{\gamma,\theta}(\mathbf{x}_i))) \right\} \tag{1.6}$$

1.2.4.2 Research Findings:

From the implementation and evaluation of the proposed model and other baseline models, it is observed that the proposed MATSD outperforms the basic ML models in terms of accuracy and F1-score. In addition, it can be noted that the proposed MATSD model records a statistically better performance than the state-of-the-art

methods. The usage of sarcasm as an auxiliary task helped in improving the performance, as seen from the ablation study. Furthermore, the sigmoid link between the two tasks also helped in increasing the performance of stress detection, as observed in the ablation study. This makes the proposed MATSD approach paves way for multi-task learning approaches for detecting tweet-level stress.

1.3 Performance Metrics used in the thesis

To analyze the performance and effectiveness of the developed model against the other standard models, the popular metrics of Accuracy and F1-score are utilized. These two are the main measures are used in the thesis for performance evaluation of the proposed models with the existing models.

Accuracy: The number of correctly classified data points among all data points used for testing. [48]. The predicted data points are categorized into the following categories:

- **True positives (TP):** The number of stress positive test samples correctly classified by the model.
- **True negatives(TN):** The number of stress negative test samples correctly classified by the model.
- **False positives (FP):** The number of stress negative test samples incorrectly classified by the model.
- **False negatives (FN):** The number of stress positive test samples incorrectly classified by the model.

Then, the accuracy is computed as:

$$Accuracy = \frac{(TP + TN)}{(TP + FP + TN + FN)} \tag{1.7}$$

F1-Score: It is the harmonic mean of precision and recall. If precision, a, is the ratio of correct predictions among all predictions and if recall, b, is the ratio of correct predictions out of the total actual observations [49], then F1-score is defined as;

$$F1 - score = \frac{2 \times a \times b}{(a + b)} \tag{1.8}$$

1.4 Organisation of the Thesis

The thesis focuses on investigating various machine-learning approaches for psychological stress detection at the text level. In addition, the thesis focuses on incorporating the concept of illocutionary sarcasm in the prediction of stress at text level. To this end, supervised models like NTSD and STSD are proposed along with an empirical analysis of the performance of these models on various datasets. In addition, the semi-supervised paradigm, called SMTSD, was developed for text-level stress detection and studied with extensive empirical experiments. Moreover, the thesis also discusses the deep-learning-based multi-task learning model called MATSD for the classification of text-level stress. The thesis is organized into seven chapters.

Chapter 1: This chapter presents the introduction and broad topics related to the thesis. In addition, a crisp and short description of the research objectives is also presented in this chapter.

Chapter 2: This chapter covers a thorough and detailed review of the problem of text-level stress detection.

Chapter 3: This chapter presents a detailed and formal approach for tweet-level or text-level stress detection. In addition to the collection of the data, extensive pre-processing is performed to get an intuition of the usage of neighborhood data. The concept of sarcasm is used as an attribute, to improve the performance

Chapter 4: In this chapter, the new supervised model of STSD is employed to

predict tweet-level stress. It also gives tries to investigate the role of the PCA in text-level stress detection.

Chapter 5: This chapter presents a model with semi-supervised learning. Self-training mechanism of semi-supervised paradigm, called SMTSD, is employed to implement tweet-level stress detection. In addition, extensive experiments were conducted to understand the effectiveness of a semi-supervised approach

Chapter 6: This chapter proposed a deep-learning model for detecting tweet-level (text-level) stress. The multi-task learning approach is presented in this chapter to detect the primary task of tweet-level stress with the help of auxiliary task of sarcasm.

Chapter 7: This chapter presents the conclusion and future work for the thesis.

Chapter 2

Literature Survey

This chapter presents a detailed background literature for the text-level [1] stress detection problem. In this chapter, initially, the traditional stress detection approaches are presented. Later, the details of various methods in social media-based tweet-level stress detection approaches are presented. In addition, the role of sarcasm and dimensionality reduction are discussed. Also, the multi-task learning solutions in the problems related to tweet-level stress detection are discussed. The chapter concludes with the summary and major gaps found in the literature.

2.1 Social-media-based stress detection approaches

2.1.1 Issues with traditional methods of stress detection

Traditional methods used to detect stress through signals from wearable sensors or from personal interviews with psychiatrists, questionnaires and feedback [14–16]. But most of them suffer from the fact that they are not proactive in detecting the stress. In addition, they suffer from the quality of data as the procedure of data collection is error-prone, preoccupied with labour and as most users don't prefer to undergo such a process due to social stigma. Moreover, some of the traditional methods

[1] As mentioned in chapter 1, most of the literature for detecting social media-based text-level stress detection is based on twitter datasets, hence, the tweet-level and text-level are used interchangeably throughout the thesis.

use wearable sensors to monitor physiological activity for detecting stress. But this method interferes with normal life making people to avoid it. Hence, social media-based approach for detecting stress has gained prominence over traditional methods.

2.1.2 Early social media-based approaches

Numerous studies have been conducted on the methods used to detect health-related issues using social media [50]. But detecting mental health-related issues is difficult because psychological health issues or stress in social media data is reflected by subtle changes in the behavior and language used [21]. The possibility of utilizing social media to identify and diagnose the commonly occurring mental illness like *major depressive disorder* is explored in [22]. An advance prediction of significant variations in the behavior of mothers during the postpartum period is studied in [23]. In [51], a simple keyword-based classifier is developed to determine the sentiment in short messages like tweets. Beyond depression, detecting *post traumatic stress disorder* (PTSD) using social media postings was first studied in [21]. The broader set of mental health illnesses are analyzed for intuitive linguistic signals from social media postings in [52], indicating that Twitter and other social media are a good source for getting unique quantifiable signals of human behavior [21]. An improved microblog-based system to detect stress in teenagers in advance and help them to overcome the stress was proposed in [53]. Wherein, Gaussian process-based classifier is used to detect the tweet-level stress, which is later aggregated to compute user-level stress in that category [53]. Despite throwing new light on the approaches to detect stress using social media, the above techniques suffer from either of the following two drawbacks. First, they depend upon manual filtering for labeling the data [21,52], which is time-consuming and laborious. Second, they are based on the data collected through crowdsourcing and surveys [22,23], which takes time and is erroneous because many individuals avoid taking surveys.

Adolescent stress is detected using tweets of teenagers in [25] by extracting various linguistic features and classifying the detected stress in each category into six levels in the integer range [0,5]. But there is no justification provided in [25] for using

the dependency tree to label training data. In [34], the effect of the sensitivity of time-based comment and response interactions on detecting the stressful state of a user is analyzed. It does not consider the importance of neighborhood tweets of the given tweet of the user for identifying stress. Moreover, the labeling scheme employed in [34] is a manual procedure, which is a major drawback. Different patterns of adolescents' stress based on microblog postings of teens are considered in [26]. Chronic stress is computed with the aggregation over individual tweet-level stress states [26]. Similar to earlier approaches, this scheme suffers from the significant drawback of using questionnaires as the ground truth for labeling data [26], making it less applicable. In [54], an intelligent system, *TensiStrength* is proposed for automatic detection of stress state from a wealth of short text messages, so as to use this information in other applications like intelligent transport systems. But, it does not consider the impact of neighborhood messages in detecting stress levels in a given message. In [55], a web-based application is proposed for determining the depression state of Twitter users based on the sentiment of their tweets. However, it did not explore the impact of neighborhood tweets.

2.1.3 Deep learning-based text-level stress detection approaches

The problem of detecting stress using techniques of deep learning is gaining research interest. [18] is an initial work that uses deep sparse neural network to detect whether a given tweet reflects the stressed state of its author. This approach labels training data based on the ground-truth that the words in hashtags of the tweets represent the opinion of the user, which turns out to be the major shortcoming because it is intuitive that hashtagged phrases may not correctly reflect the perspective of the user. In [24], several hand-selected features such as linguistic and statistical attributes are computed at the tweet and user levels, respectively, and then fed into a convolutional neural network to generate modality invariant attributes at the user-level. After that, a model based on a deep neural network is designed to solve the problem of detecting stress, with modality invariant attributes as input [24]. In [3], a rich set of features that are oriented towards stress are extracted to identify stressor subjects

and stressor events from a given social media posting, where a multi-task learning model is employed. Also, [3] uses convolutional neural networks to learn tweet-level features from word-level vectors. But the procedure relies on manual labeling of data, making it infeasible to scale for larger datasets. According to [19], the authors have proposed a hybrid model of convolutional neural network and partially labeled factor graph to solve the problem of detecting the user-level stress. In addition, [19] had performed a systematic investigation on the correlation of users' stress states and social interaction networks; wherein it was concluded that stressed users have 14% higher sparse connections than non-stressed users. In the works, [3, 19, 24], the training data is labeled based on the presence of tweets with sentence patterns like "I feel stressed" or "I feel stressed this week", as it is proved that "I feel" based sentence patterns are effective in the emotional analysis [56]. Therefore, the preceding works [3, 24], suffer from the major drawback of lacking time-sensitive analysis of comment, response interactions and also there is no study on the effect of neighborhood postings of the user in the tweet-level stress detection.

In [57], big data framework like Apache Spark and several word embeddings are exploited to develop a method in order to operate a multi-class multi-label classification of a discussion within a range of six classes of toxicity. But it does not consider the impact of sarcasm and previous comments. Recently, [38] proposed a deep learning-based fusion net model to detect depressed users based on multi-task learning of text based vectors and statistical features. Though this work aims to maximize the utilization of the text content of the tweets, it does not address the impact of immediate previous tweets. In [58], a deep learning approach based on convolutional neural networks(CNN) and long short term memory(LSTM) is presented to detect the sentiment in tweets with the extraction of various features with word2vec and stopwords to capture the context. [59] explores a deep learning-based solution for detecting clickbaits in social media posts, which may mislead the users and affect their stress levels. But both of these works fail to capture the importance of immediate previous tweets. In [60], a machine learning-based approach is developed to detect clues of depression in the text. It presents an method for identifying depressive

profiles on Twitter using machine learning, sentiment analysis, and natural language processing techniques. However, the impact of neighborhood tweets in detecting the stress level of the present tweet has not been investigated.

2.1.4 Need for Semi-supervised approach for tweet-level stress detection

Semi-supervised techniques fall somewhere between supervised and unsupervised techniques and are useful when there is a large amount of unlabeled data but only a small amount of labeled data available [61]. This case is true for the problem of social media-based text-level stress detection [19, 45]. There were few deep learning-based solutions utilizing semi-supervised mechanisms for detecting user-level stress [19, 62, 63]. However, the tweet-level stress detection approaches using classical machine learning approaches are all based on supervised learning models [25, 34]. Furthermore, even the deep learning-based solutions for the tweet-level stress detection are largely based on supervised learning approaches [18, 37]. Hence, it is concluded that there is a gap in applying semi-supervised learning techniques to tweet-level stress detection. There also exists scope to utilize information from sarcasm in the paradigm of semi-supervised solutions for tweet-level stress detection, as sarcasm in conjunction with semi-supervised models was not explored in existing literature [17, 19, 37, 62, 63].

2.2 Role of Sarcasm and dimensionality reduction

2.2.1 Sarcasm

Sarcasm is a type of communication wherein the meaning conveyed is the opposite to the meaning stated explicitly [64]. In [65], a behavioural modelling framework is employed to detect sarcasm in users' tweets. It is proposed in [66] that for the effective automatic recognition of sarcasm in Twitter data, both text content features and authorial style criteria are required. In [67], a fuzzy-logic-based approach is presented for detecting sarcasm in tweets and classifying sarcasm into four categories based on

the intensity of hurt in the statement. The impact of sarcasm, embedded in the text content of the tweet, on detecting stress is not studied in the literature [14, 17–19, 24]. Essentially, sarcasm is a way of expressing verbal irony intended to convey contempt or ridicule [2]. There is a mention of four different types of sarcasm in [39, 40] :

i) ***Propositional sarcasm***: A situation in which the meaning of the proposition made is opposite to the explicit meaning. For example, *"You sound fantastic!"* may be interpreted as non-sarcastic without the context.

ii) ***Embedded sarcasm***: In this type of sarcasm, there is an embedded incompatibility in the expressed words or phrases. For example, in the sentence *"John has turned out to be such a diplomat that no one takes him seriously."*, the incompatibility is embedded in the word of "diplomat" and rest of the sentence.

iii) ***Like-Prefixed sarcasm***: A Like-phrase is used for the denial of immediate meaning conveyed. For example, the sentence *"Like you care!"* is a common sarcastic response.

iv) ***Illocutionary sarcasm***: In this case, sarcasm is expressed with non textual clues, like facial expressions to convey an inverted sentiment and attitude to a sincere utterance. For example, rolling one's eyes when pronouncing the phrase *"Yeah right"*.

In this work, a new feature called *Sarcasm_Level* is computed based on the concept of illocutionary sarcasm, where facial expression is represented in the form of emojis, by the users, to exhibit the sarcasm.

2.2.2 Importance of sarcasm in stress detection

The aim of this thesis is to develop a binary classifier to detect stress at tweet-level using information of sarcasm. And the problem addresses to develop a binary classifier that learns from the features of textual data. Though most of the literature on tweet-level stress uses multi-class classification, there are a few binary classification

[2] https://www.thefreedictionary.com/

approaches as well [68,69]. The detection of psychological stress is of high importance during and after the COVID-19 pandemic era, where there is a wide presence of mental health problems [70] . Moreover, the utility of sarcasm, which exists in the tweet's text data, in detecting stress is yet to be explored to its full potential. Sarcasm, a word play that hides the original intent of the speaker, has an important role in conveying the opinion of users and affects their mood [41]. Sarcasm is subjective and is relatively tough to detect as compared to sentiment. However, few studies have concluded that sarcastic conversations can have a different emotional impact compared to non-sarcastic responses [35]. In much of the existing work related to stress detection, the role of sarcasm is not investigated thoroughly [18, 19, 21–26]. The main intention of the usage of sarcasm is to express a contradictory emotion than what is explicitly specified. Furthermore, it is noted that sarcasm, as an expression of humour, can be seen as an increase in creativity [71]. Due to the implicit presence of sarcasm within the tweet's text content, the influence of sarcasm in the stress detection helps in maximizing the utilization of information from the text. In this regard, the work in [45] had employed a new attribute whose computation is derived from the principle of illocutionary sarcasm in order to capture sarcasm to improve the performance of stress detection.

In [45], the authors have proposed two solutions for overcoming the problem of sparsity in the detection of stress at the granularity of tweets. First, an attribute called *Sarcasm_Level* is extracted to reflect the sarcasm that exists in the content of a tweet for the detection of tweet-level stress. Second, a novel method called Neighborhood-based Tweet-level Stress Detection (NTSD) is proposed to utilize the information from neighboring tweets to increase the availability of related data. And it was concluded that the use of sarcasm as a binary-valued feature helps in increasing the performance of stress detection. Also, the NTSD in combination with the new attribute outperforms all the basic models considered. But the model has an overhead as it includes the content of neighborhood tweets as well.

Sarcasm is a form of expressing irony where the literal meaning contradicts the intended meaning. Though sarcasm is complex and subjective, few studies have con-

cluded that sarcastic conversations can cause a different and contradictory emotional impact than non-sarcastic responses [35]. Furthermore, usage of sarcasm is observed to be more prevalent during "funny" movements and targeted towards known persons [36]. All this helps to frame a new model that utilizes the information from sarcasm to detect stress.

2.2.3 Dimensionality reduction

Dimensionality reduction analysis for improving the performance of tweet-level stress detection is not addressed in the literature [3, 18, 19, 26, 34, 45, 72]. It is observed that Kernel Principal Component Analysis (PCA) helps in obtaining the dimensionality reduction when the classes are linearly non-separable [73]. The reasons for choosing PCA over other dimensionality reduction methods are listed as follows:

- There are many dimensionality reduction techniques that rely on the use of classifiers, such as forward selection, chi square, backward selection, etc. Whereas, PCA is an unsupervised dimensionality reduction technique and hence does not use any classifiers for the reduction of dimensions [74].
- The data taken is replete with many sparse values in many of the features, and hence there is a need for a dense representation of the feature sets. This is possible with a linear or nonlinear combination of the original features, unlike simple feature selection algorithms, which just remove the features, making little change in the sparsity of the feature values.
- PCA captures the distribution of datasets and selects the directions with the highest variations for projections, making a new projection to capture most of the information within a smaller number of dimensions with only a small loss of information [47, 75].

2.3 Multi-task approach for tweet-level stress detection

The concept of simultaneous learning of multiple related classification problems using a single model from a given data can be termed multi-task learning [38, 44, 76]. The simultaneous learning of multiple tasks is similar to regularization with each task acting as regularizer to the other task. This could lead to better models with the sharing of information of parameters [47].

2.3.1 Multi-task approaches in related problems

Though multi-task learning (MTL) approaches were not explored in the solutions for text-level stress detection, they are explored in other closely related text classification problems of depression and complaint detection [38, 44].

In [76], one of the earliest approaches using MTL mechanisms for mental health and suicide risk on social media data, apart from mental health issues, gender is predicted as an auxiliary task, and the conditions of health are considered as different tasks in MTL. Nevertheless, this work did not consider the influence of sarcasm in detecting stress.

MTL-based CNN architectures were developed by considering the multi-input strategy of emotional categories in [43]. This model was developed for detection of the related mental illness tasks like depression and post-traumatic stress disorder (PTSD) in a multi-task framework using multi-channel CNN architecture. It is concluded in the work that despite the lack of availability of large training data, the performance achieved is high due to the shared information across the tasks. However, this work did not consider the influence of sarcasm, in detecting stress.

In 2021, [44], there is a novel approach to develop an MTL mechanism to predict emotion and personality traits by providing a variety of information flow gates between the networks of two tasks at various levels. Nevertheless, the utilization of text information like sarcasm is not considered in the work. In [77], a multi-task learning

approach is developed to detect sentiment from social media postings using complaint detection as an auxiliary task. The pipe-lined architecture in the model is based on a Bi-directional gated recurrent unit (Bi-GRU) and the complaint predicted is used for sentiment detection [77].

In all of these works the concept of sarcasm was not utilized in detecting stress as part of multi-task learning. As stress and sarcasm are related tasks, there is a need for developing a multi-task classifier that depicts the utility of sarcasm in the detection of stress. In this thesis, a few of the state-of-the-art multi-task learning approaches are adopted to stress detection, for better validation and comparison of the proposed system.

2.3.2 Sarcasm as an auxiliary task in related problems

Sarcasm can be defined as an expression where the explicit intention conveyed is the opposite of the actual intention and hence, sarcasm can affect stress and human emotion [40]. Earlier literature proved that sarcasm in the textual part of the tweet affects the stress state of the tweet [45]. Also, multi-task learning-based techniques utilize sarcasm as a task related to emotions. In [78], a multi-task learning approach was employed in developing generalized embeddings called Emo2Vec. The concept is based on a multi-task learning mechanism with the utilization of information from six related emotions, including stress and sarcasm. This gives a clue that sarcasm and stress are related tasks, giving better performance than existing embeddings, especially on smaller datasets. In [42], the authors develop a semi-supervised scheme to detect tweet-level stress by utilizing information of illocutionary sarcasm in filtering of the pseudo-labels. Hence, from the above literature, it is intuitive that sarcasm helps in better capturing of the stress that exists in the user's tweet. Therefore, there is a scope to develop a multi-task-based model, where stress is the primary task and sarcasm is considered as the auxiliary task that influences the detection of stress.

2.4 Summary

In essence, most of the social media-based text-level stress detection literature relied on only current tweets, making way for the problem of data sparsity. In addition, the related concept of sarcasm is not examined for its role in improving the performance of tweet-level stress detection models. Hence, as a part of this thesis, the problems of data sparsity and usage of sarcasm in tweet-level stress detection are addressed.

It is known that in the real world, finding or building large labeled datasets for classification is a tough task and the same applies to the problem of tweet-level stress detection. This motivates in exploring the semi-supervised approach in this thesis to develop classifiers for the tweet-level stress detection as it is observed that most of the models in existing literature related to tweet-level stress detection employ supervised learning approaches. Sarcasm identification is found to be a related task for tweet-level stress detection. From the literature, it is found that there is an acute lack of multi-task learning approaches to detect tweet-level stress. Furthermore, the scope to explore the understudied concept of using sarcasm as an auxiliary task in the paradigm of multi-task learning to detect tweet-level stress.

Chapter 3

Neighborhood-based Tweet-level Stress Detection

As discussed in chapter 1, this chapter proposes two solutions for solving tweet-level stress detection problem. It considers the information from neighborhood tweets and the sarcasm present in the tweet.

Organization of the chapter:

The chapter is organized as follows. Section 3.1 discusses the problem formulation, including the formal notations used in the chapter. The proposed NTSD model and its training are discussed in sections 3.2. The attributes used in the models including the proposed new attribute of *Sarcasm_level* is discussed in section 3.2.1.1. Whereas, the experimental setup and the data collection procedure are briefed in section 3.3. The results and the findings of the proposed model are presented in section 3.4. Finally, the summary of the chapter was presented in section 3.6.

3.1 Problem Formulation

In this chapter, we propose a novel neighborhood-based tweet-level stress detection approach. All the notations used in this chapter are summarized in the Table 3.1.

Let U be the set of users and let $\mathbf{x}$ be the tweet posted by a user $u \in U$. Let $y \in \{0, 1\}$ is the stress label for the tweet $\mathbf{x}$. The datasets of tweets considered for the

model is composed of two sets, primary and auxiliary. Before defining the problem, few definitions are presented.

Table 3.1: Notations used in the NTSD model and their definitions

Notation	Definition
U	Set of users
u	Any user $u \in U$,where U is set of users
$\mathbf{x_i}$	i-th tweet (tweet feature vector) in primary dataset D and belongs to some user u
N	Total number of tweets in the primary dataset D
M	Total number of features or dimensions (Length of feature vector)
δ	The neighborhood window
N_δ	Maximum number of previous tweets considered for a given tweet under neighborhood window δ
$\mathbf{x_j^i}$	The j-th previous tweet corresponding to the primary tweet $\mathbf{x_i}$ of user u under the window δ
x_{im}	The m-th feature value of primary tweet $\mathbf{x_i}$, where $m \in \{0, 1, ..., M\}$
x^i_{jm}	The m-th feature value of auxiliary tweet $\mathbf{x_j^i}$, where $m \in \{0, 1, ..., M\}$
D	The set of labeled primary tweets collected for analysis.
D_δ	Set of auxiliary tweets collected from primary tweets dataset under window δ
$y_i \in \{0, 1\}$	Stress state label corresponding to tweet $\mathbf{x_i} \in D$
$y^i_j \in \{0, 1\}$	Stress state label corresponding to neighborhood tweet $\mathbf{x_j^i} \in D_\delta$
Y	Set of class labels for primary dataset, D
$\mathbf{w}$	The feature weight vector of size M, $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M]^T$
$f(\mathbf{w})$	The log-likelihood function of the proposed model
$(D^{T_q}, D_\delta^{T_q})$	Training data sample corresponding primary tweets dataset, D, and auxiliary tweets dataset, D_δ
$(D^{\mathcal{V}_q}, D_\delta^{\mathcal{V}_q})$	Validation data sample corresponding primary tweets dataset, D, and auxiliary tweets dataset, D_δ
$\hat{Y}_q$	Predicted class labels for validation sample of primary tweets, D^{V_q}
∇f	Gradient of the objective function, $f(\mathbf{w})$ with respect to parameter vector $\mathbf{w}^T$
∇f_i	Gradient of the objective function at ith training example with respect to parameter vector $\mathbf{w}^T$
$\Delta\mathbf{w}$	The change in weight vector $\mathbf{w}^T$
η	Learning rate
$P(y_i\|\mathbf{x_i})$	Prediction probability of class y_i for primary tweet $\mathbf{x_i}$
$P(y_i\|\mathbf{x_j^i})$	Prediction probability of class y_i for auxiliary tweet $\mathbf{x_j^i}$ corresponding to primary tweet $\mathbf{x_i}$
$P_c\left(y_i \middle\| \mathbf{x_i}, \bigcup_j \mathbf{x_j^i}\right)$	Combined prediction probability of class y for tweet $\mathbf{x_i}$ when the set of auxiliary or previous tweets are also considered
α	The weight for prediction probability of primary tweet
γ_k	The weight for prediction probability for k^{th} auxiliary tweet, $1 \leq k \leq N_\delta$
v	The number of tress in the Random forest
h	The average depth of the trees in Random forest classifier

Definition 1 (Primary tweet dataset): Let D be the set of historical tweets collected from the Twitter over a period of time, given by $\bigcup_{i=1}^{N} \mathbf{x_i}$, where $\mathbf{x_i}$ is i-th tweet of the dataset and the tweet $\mathbf{x_i}$ belongs to some user $u \in U$. Let $|D| = N$, where N represents the total number of tweets in the primary dataset and each of

the tweets in this dataset are labeled, where $y_i \in \{0, 1\}$ gives the stress label of tweet $\mathbf{x_i}$. In this chapter, the terms primary tweets, present tweets and current tweets are used interchangeably.

Definition 2 (Neighborhood window): The neighborhood window, denoted by δ, provides the limit on the set of most recent tweets for each tweet in the primary dataset. That is, it is the set of the most recent tweets considered from the current tweet of primary dataset. For any given window δ, let N_δ be the maximum possible neighborhood tweets allowed in the window for any tweet in the primary dataset.

Definition 3 (Auxiliary tweet dataset): Let, D_δ be the set of neighborhood tweets within the neighborhood window δ for each tweet in primary dataset, D. Formally, $D_\delta = \bigcup_{i=1}^{N} \bigcup_{j=1}^{N_\delta} \mathbf{x_j^i}$, where, $\mathbf{x_j^i}$ is the j-th previous tweet, of user u, corresponding to the primary tweet $\mathbf{x_i}$ of same user u under the neighborhood window δ and N_δ is maximum possible neighborhood tweets for any primary tweet allowed under the neighborhood window δ. And, the size of this dataset is $N < |D_\delta| \leq NN_\delta$. It is to be noted that we do not consider time-frame in this approach. We only consider count of previous tweets

Problem Definition: Given a set of labeled tweets (D) and corresponding auxiliary tweets (D_δ), the problem is to learn a function g that predicts a label for any unlabeled tweet $\mathbf{x_r}$ by incorporating the content from its auxiliary tweets $\bigcup_j \mathbf{x_j^r}$ considered under neighborhood window δ. The function g is the tweet classifier which is described as $g\colon D \otimes D_\delta \rightarrow C$, where, $D \otimes D_\delta$ is the collection of tweets along with the corresponding previous tweets, and C is the set of unique labels used in the classification (here, $C = \{0, 1\}$). Then, the classifier g is used to learn the unknown label y_r of the unlabeled tweet $\mathbf{x_r}$ based on the content of the neighborhood tweets $\bigcup_j \mathbf{x_j^r}$ as following:

$$y_r = g\big(\mathbf{x_r}, \bigcup_{j=1}^{N_\delta} \mathbf{x_j^r}\big) \tag{3.1}$$

where $y_r \in \{0, 1\}$

3.2 Methodology

3.2.1 The Framework of the Model

Basic framework of the proposed model is depicted in figure 3.1. The primary tweets are collected by crawling over Twitter using Twitter's API *tweepy* and for each primary tweet, the corresponding neighborhood tweets of the user of the tweet are collected. Following this, all the required features are extracted for both sets of tweets. Later, the proposed NTSD model is trained using the feature vectors of primary tweets and the corresponding auxiliary tweets. The trained NTSD model is then used to predict the stress-state label of any unlabeled tweet by utilizing the information from the content of the given tweet and its immediate previous tweets.

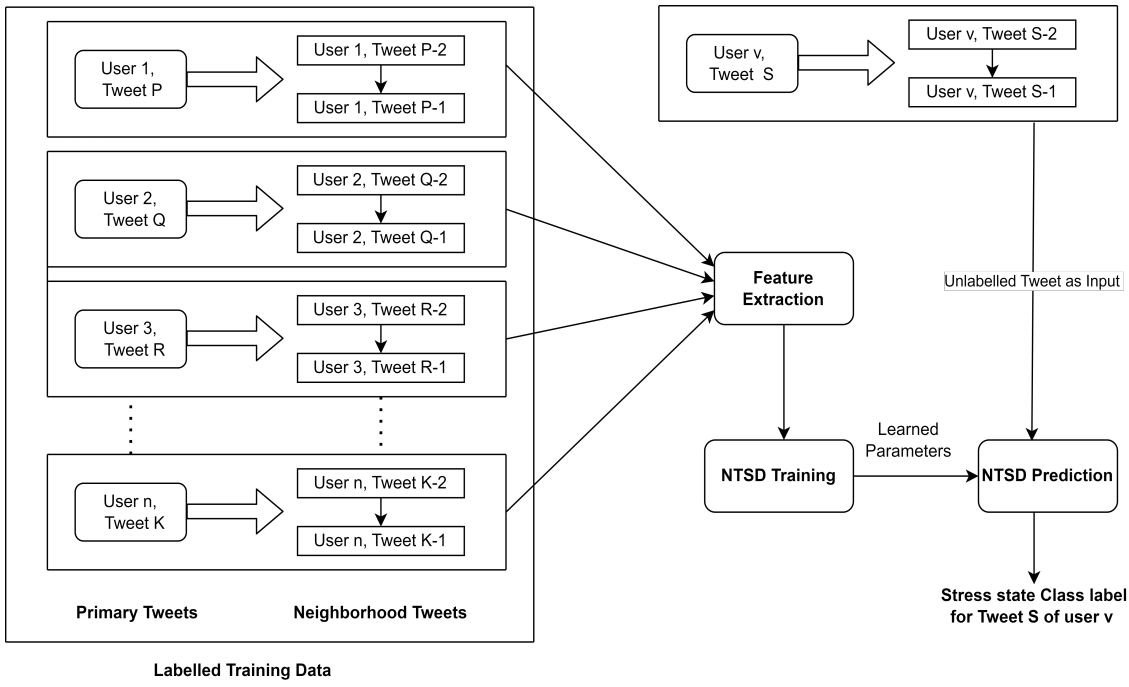

Figure 3.1: Framework of the NTSD model

3.2.1.1 Tweet's features

Before presenting the model, the tweet's feature space, which is employed in this problem, is described. As the problem addresses the tweet level stress detection, the textual and social attributes specific to each tweet are extracted. Most of the

features are based on tweet's textual content so as to leverage the text content of the tweets and their neighborhood tweets, while the social attributes at the tweet level are extracted to understand the social engagement of the tweets. In addition, a new text-based feature called *Sarcasm_Level* is defined to understand the contribution of the sarcasm, present in the text content of the tweet, in detecting the stress at the tweet level. The features used in the model are summarized as follows:

- Linguistic/Textual Features
  - *The word-counts for ten categories of EMPATH based linguistic-psychology library*: EMPATH is a free-of-cost library for psychological -linguistic word count [79]. This library is used to extract the word count related to 10 categories of linguistic psychological clues for the text content of the tweet. The categories include - Family, health, school, academics, exam, disease, business, interpersonal, office and informal language. This is a vector of length 10.
  - *Number of positive and negative verbs*: It is a vector of size 2 containing the count of positive and negative verbs found in the tweet.
  - *Number of positive and negative adjectives*: It is a vector of size 2 specifying the count of positive and negative adjectives found in the tweet.
  - *Degree adverbs*: The attribute computes the intensity or level of adverbs used in the tweet as computed in [18, 19, 24]. The scale of the degree is in the range of $[-3, 3]$. For example, "*I feel stressed*" has degree -1, while "*I feel more stressed*" has a degree of -2. Similarly, "*I am happy*" has degree $+1$, while "*I am extremely happy*" will be given a degree of $+3$.
  - *Emoji Count*: The number of positive and negative emojis present in the tweet. It is a vector of size 2. NLTK and Emoji libraries of Python are employed to extract the sentiment of emojis.
  - *Punctuation*: It is the count of punctuation marks that reflect emotions found in the text content of the tweet like question marks (?), exclamation marks (!) and dots (...). This is a vector of size 3.

– *Sarcasm_Level*: It is one of the two proposed solutions for utilizing text content of the tweet to mitigate the problem of data sparsity. This attribute captures the sarcasm embedded in the content of the tweet. It indicates the inconsistency in the tweet if the polarity of text has one sentiment but the emojis and words in the hashtags of the tweet are of opposite sentiment. This is necessary as the emotion conveyed in a sarcastic tweet generally contradicts with the explicit emotion found in the tweet. This is common in illocutionary sarcasm [40].

To understand the computation of this attribute, few notations are defined. Let the total positive and negative polarity words in the tweet are pos and neg, respectively. Also, let the total number of positive and negative emojis in the tweet are e_{pos} and e_{neg}, respectively, while h_{pos} and h_{neg} be the count of hashtags with positive polarity and negative polarity in the tweet content, respectively. Then, the *Sarcasm_Level* of tweet t is defined as follows:

$$Sarcasm\_Level(t) = \begin{cases} 1, \text{ if } \Big\{ (e_{pos} \geq e_{neg} \ \wedge \ pos \leq neg) \ \vee \\ \qquad (e_{pos} \leq e_{neg} \ \wedge \ pos \geq neg) \ \vee \\ \qquad (h_{pos} \geq h_{neg} \ \wedge \ pos \leq neg) \ \vee \\ \qquad (h_{pos} \leq h_{neg} \ \wedge \ pos \geq neg) \Big\} \\ 0, \text{ otherwise} \end{cases} \tag{3.2}$$

Where, the symbols $\wedge$ and $\vee$ respectively denote the operations of logical AND and logical OR. The hashtags should not be filtered out in the preprocessing as they are used to compute this attribute.

- Social Features of Tweet: The social attributes extracted in this chapter describe the social engagement of the tweet. They are computed as the number of favorites or likes, retweets and comments for a given tweet.

3.2.2 Preprocessing

The tweets are preprocessed to filter out noisy data and then the required features are extracted. For data cleaning, the tweets that contain only URL information or the tweets which contain only image content are filtered out. Also excluded are the tweets of users who block timeline access, preventing the extraction of previous tweets in the user's timeline. Finally, the dataset of tweets is normalized using z-score normalization [48]. In this manner, the collected tweets are preprocessed.

3.2.3 Proposed Model

In this section, the working of the proposed model, NTSD, is presented. Stress is a psychological phenomenon with contiguity, persisting over a longer period of time. The central idea is that if the immediate previous tweets are stressful, then there is a high chance that the given tweet is also stressed. For this, a simple and efficient model is developed to address the problem of data sparsity at the tweet level stress detection. The concept is that for computing the class label of a given tweet, the information of the immediate neighborhood tweets of the user of the given tweet is utilized. The neighborhood window for neighborhood tweets is constructed based on count of previous tweets considered from corresponding primary tweet. In other words, the window contains the recent previous tweets of the user of a given primary tweet. Then, both primary and neighborhood tweet datasets are used to train the model. The proposed model is developed over logistic regression, which employs a sigmoid or logistic function to predict the probability of the input tweet vector $\mathbf{x}$ belonging to class stressed $(y = 1)$:

$$P(y = 1|\mathbf{x}) = \frac{1}{1 + \exp\left(-\mathbf{w}^T.\mathbf{x}\right)} = p(\mathbf{x}) \tag{3.3}$$

where $\mathbf{w}$ is the feature weight vector that must be solved [80, 81]. The problem is to find the appropriate $\mathbf{w}$ so that the log-likelihood, $L(\mathbf{w})$, of the training data is maximized, where, $L(\mathbf{w}) = \sum_{k=1}^{N} y_k * \log p(\mathbf{x_k}) + (1 - y_k) * (1 - \log p(\mathbf{x_k}))$.

In this chapter, the combined training of primary tweets and corresponding neigh-

borhood tweets is performed. To train the model, the likelihood that for every tweet $\mathbf{x_i}$ in primary dataset D and its associated neighborhood tweets $\bigcup_j \mathbf{x_j^i}$ in auxiliary dataset D_δ belonging to the same class is maximized [80]. Therefore, based on the notations specified in the table 3.1, the likelihood function for the proposed model is defined as follows:

$$l(\mathbf{w}) = \prod_{i=1}^{N} \left\{ p(\mathbf{x_i})^{y_i} (1 - p(\mathbf{x_i}))^{1-y_i} \prod_{j=1}^{N_\delta} \left[p(\mathbf{x_j^i})^{y_j^i} (1 - p(\mathbf{x_j^i}))^{1-y_j^i} \right] \right\} \tag{3.4}$$

For simpler computation, log-likelihood is considered [82]. Hence, the objective function of this model is to maximize the log-likelihood of the current tweet along with the loss of its related neighborhood tweets such that they have a common class label. The parameters at which the likelihood is maximum gives the weights of the model, which are then used to predict the class label of an unlabeled tweet. The log-likelihood of the proposed model is computed as:

$$\begin{aligned} f(\mathbf{w}) &= \log(l(\mathbf{w})) \\ &= \sum_{i=1}^{N} \left\{ y_i \log(p(\mathbf{x_i})) + (1 - y_i) \log(1 - p(\mathbf{x_i})) \right\} \\ &\quad + \sum_{i=1}^{N} \sum_{j=1}^{N_\delta} \left\{ y_j^i \log(p(\mathbf{x_j^i})) + (1 - y_j^i) \log(1 - p(\mathbf{x_j^i})) \right\} \end{aligned} \tag{3.5}$$

According to the proposed model, if the neighborhood tweets $\mathbf{x_j^i}$ of i^{th} primary tweet are stressed, then the i^{th} primary tweet is also stressed. Therefore, $y_j^i = y_i, 1 \leq i \leq N$ and $1 \leq j \leq N_\delta$. Hence, the objective function becomes,

$$\begin{aligned} f(\mathbf{w}) &= \sum_{i=1}^{N} \left\{ y_i \log(p(\mathbf{x_i})) + (1 - y_i) \log(1 - p(\mathbf{x_i})) \right\} \\ &\quad + \sum_{i=1}^{N} \sum_{j=1}^{N_\delta} \left\{ y_i \log(p(\mathbf{x_j^i})) + (1 - y_i) \log(1 - p(\mathbf{x_j^i})) \right\} \end{aligned} \tag{3.6}$$

3.2.4 Training the Model

Training the model is an optimization problem because it must find the parameter vector that maximizes the objective function. In order to optimize the objective function given in equation (6), a greedy solution of gradient ascent is used. Gradient ascent is a maximization version of gradient descent in which the solution is searched by taking large steps in the direction of the gradient [80, 83]. The parameter vector to be learned is $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M]^T$, where M is the total number of features and $\mathbf{w}_0$ is the bias. The process starts with the initialization of the weight vector and then updates its weight until no more change in weights is observed. This is known as the point of convergence [81]. The learning rate η is a small real-valued constant ($0 < \eta < 0.1$). It is a value which can decay with iterations. Algorithm 19 gives the complete working process of the proposed model, NTSD. The variable k in the algorithm represents the value of k in the implemented k-fold cross-validation. Typically, k=10. The gradient ascent solution to learn the parameter vector of the the proposed model is given in algorithm 7 and it is invoked from algorithm 19.

As normal gradient ascent is a batch processing technique that requires scanning of the entire set of training examples before updating the weights, it slows down with increase in the size of the data [80, 83]. In contrast to this, algorithm 3 presents the solution to find optimal values of the weight vector $\mathbf{w}$ using the stochastic gradient ascent method. This method is also called incremental gradient ascent, where the parameter vector is updated in each iteration instead of a batch update [80, 83]. Stochastic gradient ascent with small batch size is implemented in the solution, considering its efficiency and fast convergence [82, 83]. And the model is trained for various window-sizes chosen for the neighborhood window to understand the impact of neighborhood tweets in the detection of stress for the given primary tweet.

3.2.5 Prediction

After learning the feature weights of the model, the prediction is performed to find the class label of an unlabeled tweet. Based on the notations specified in the table

Algorithm 1: Neighborhood tweet-based Stress detection

Input: Primary Dataset D and auxiliary dataset D_δ

Output: Model object with learned parameters and performance metrics

Function `NTSD`(D, D_δ)**:**

- $Y \leftarrow \Big\{ y_i \Big| (\mathbf{x_i}, y_i) \in D, \quad 1 \leq i \leq N \Big\}$;
- Invoke *NTSD_preprocessing*(D, D_δ);
- $\mathbf{w} \leftarrow$ *NTSD_Gradient_Ascent*(D, D_δ) ;
- **for** $q \leftarrow 1$ *to* k **do**
  - count $\leftarrow 0$;
  - $(D^{T_q}, D_\delta^{T_q}) \leftarrow q^{th}$ iteration's Training Sample from Dataset (D, D_δ);
  - $(D^{\mathcal{V}_q}, D_\delta^{\mathcal{V}_q}) \leftarrow q^{th}$ iteration's Validation Sample from Dataset (D, D_δ);
  - $Y_q \leftarrow \Big\{ y_i \Big| (\mathbf{x_i}, y_i) \in D^{\mathcal{V}_q}, \ 1 \leq i \leq |D^{\mathcal{V}_q}| \Big\}$;
  - $\mathbf{w_q} \leftarrow$ *NTSD_Gradient_Ascent*$(D^{T_q}, D_\delta^{T_q})$;
  - $\hat{Y}_q \leftarrow \Big\{ \hat{y}_i \Big| \hat{y}_i = \underset{y_i \in dom(Y_q)}{\text{argmax}} P_c\Big(y_i \Big| \mathbf{x_i}, \bigcup_j \mathbf{x_j^i} \Big), \ 1 \leq i \leq |Y_q|, \quad 1 \leq j \leq |N_\delta|, \quad (\mathbf{x_i}, y_i) \in D^{\mathcal{V}_q}, \ \mathbf{x_j^i} \in D_\delta^{\mathcal{V}_q} \Big\}$ ▷ Set of predicted stress labels for qth validation sample $D^{\mathcal{V}_q}$;
  - $Matched\_Labels \leftarrow \Big\{ y_i \Big| y_i = \hat{y}_i; y_i \in Y_q \ , \quad \hat{y}_i \in \hat{Y}_q, 1 \leq i \leq |Y_q| \Big\}$ ▷ Set of correctly predicted labels among the predicted labels for q^{th} validation sample $D^{\mathcal{V}_q}$;
  - Accuracy$[q] \leftarrow \frac{|Matched_Labels|}{|Y_q|}$;
  - F1-Score$[q] \leftarrow$ Harmonic_Mean$(Precision[q], Recall[q])$;
- **end**
- Final_Accuracy $\leftarrow$ Average(Accuracy[1], Accuracy[2], $\cdots$, Accuracy[k]);
- Final_F1-Score $\leftarrow$ Average(F1-Score[1], F1-Score[2], $\cdots$, F1-Score[k]);
- *result_object* $\leftarrow$ ($\mathbf{w}$, Final_Accuracy, Final_F1-Score);

return *result_object*

Algorithm 2: Gradient Ascent Algorithm for training NTSD model

Input: Training Data consisting of D, features of primary tweets dataset, and D_δ, features of auxiliary tweets dataset under neighborhood window δ and condition for convergence, η

Output: Parameter Vector $\mathbf{w}$ of the objective function

Function `NTSD_Gradient_Ascent(`D, D_δ`):`

Initialize parameter vector $\mathbf{w}$ to small random values ;

while *convergence is not reached* **do**

$$\nabla f = \sum_{i=1}^{N} (y_i - p(\mathbf{x_i}))x_{im} + \sum_{i=1}^{N} \left\{ y_i \sum_{j=1}^{N_\delta} \mathbf{x_j^i} - \sum_{j=1}^{N_\delta} (p(\mathbf{x_j^i})x_{jm}^i) \right\}$$

Update the weight vector, $\mathbf{w} \leftarrow \mathbf{w} + \eta * \nabla f$

end

return w

Algorithm 3: Stochastic Gradient Ascent Algorithm for training NTSD model

Input: Training Data consisting of D, features of primary tweets dataset, and D_δ, features of auxiliary tweets dataset under neighborhood window δ and condition for convergence, η

Output: Parameter Vector $\mathbf{w}$ of the objective function

Function `NTSD_Stochastic_Learning(`D, D_δ`):`

Initialize parameter vector $\mathbf{w}$ to small random values ;

while *Convergence is not reached* **do**

for *each training example* $(\boldsymbol{x_i}, y_i)$ **do**

$$\nabla f_i = (y_i - p(\mathbf{x_i}))x_{im} + \left\{ y_i \sum_{j=1}^{N_\delta} \mathbf{x_j^i} - \sum_{j=1}^{N_\delta} (p(\mathbf{x_j^i})x_{jm}^i) \right\}$$

Update the weight vector as, $\mathbf{w} \leftarrow \mathbf{w} + \eta * \nabla f_i$

end

end

return w

3.1, the combined prediction probability $P_c\left(y_i \middle| \mathbf{x_i}, \bigcup_j \mathbf{x_j^i}\right)$ of a primary tweet $\mathbf{x_i}$ using the content of its own tweet and set of its neighborhood tweets is computed as the weighted harmonic mean of the prediction probabilities of the given tweet and its neighborhood tweets within the neighborhood window δ. If there are no previous tweets available for the user, the prediction probability becomes the original prediction probability of the given tweet itself. The combined prediction probability of the model is specified in the following equation:

$$P_c\left(y_i \middle| \mathbf{x_i}, \bigcup_j \mathbf{x_j^i}\right) = \begin{cases} \frac{\alpha + \sum_{k=1}^{N_\delta} \gamma_k}{\alpha * \frac{1}{P(y_i|\mathbf{x_i})} + \sum_{k=1}^{N_\delta} \gamma_k * \frac{1}{P(y_i|\mathbf{x_k^i})}} & \text{, if } N_\delta > 0 \\ P(y_i|\mathbf{x_i}) & \text{, otherwise} \end{cases} \tag{3.7}$$

Where, $\alpha, \gamma_1, \gamma_2, ..., \gamma_{N_\delta}$ are the weights for the prediction probabilities. Also, $P(y_i|\mathbf{x_i})$ and $P(y_i|\mathbf{x_k^i})$ are computed according to equation (3.3). For simplicity, in this chapter, all weights are treated equal to one. Hence, it reduces to a simple harmonic mean of probabilities of each of the neighborhood tweets and the given primary tweet. For any unlabeled tweet that is given as input, its neighborhood tweets are extracted and using the learned parameter vector of the model, the prediction probabilities of the unlabeled tweet and its corresponding neighborhood tweets are computed.

3.3 Experimental Setup

The description of the datasets, the baseline models used for comparison and the performance measures are discussed in this section. All the experiments for the collection of tweets, data preprocessing and building of classifier models are implemented in Python using the Jupyter Notebook 4.1.1 as IDE.

3.3.1 Dataset Collection

3.3.1.1 Process of data collection

For experimental evaluation, two datasets of tweets are collected. The Twitter's API *tweepy* was used to collect tweets from the Twitter. Initially, primary tweets are collected and labeled automatically by extracting them using matching sentence patterns as specified in earlier works [19]. Later, these are used to collect auxiliary tweets. For example, in the automatic tweet labeling, tweets that match the searched patterns like "*I feel stressed*", "*I feel stressed so much*" are collected and labeled with positive stress, ($y_i = 1$). And the tweets that match the search queries like "*I feel relaxed*" and "*I feel non-stressed*" are collected and labeled with negative stress, ($y_i = 0$). The tweets collected in this way form the part of primary dataset. The most recent previous tweets of the users of each tweet in primary dataset are extracted using Twitter's API, *tweepy*. The auxiliary tweets, for a given tweet's username, are extracted using the method called *API.user_timeline()* belonging to twitter API class. This method is invoked with the current primary tweet and the parameter *count*. The function returns the number of tweets specified in the parameter *count* of the function.

These tweets form the part of auxiliary dataset. The auxiliary dataset is grouped into subsets based on the size of neighborhood window. The window sizes considered in this chapter are $N_\delta = 1$ and $N_\delta = 2$.

3.3.1.2 Details of the datasets

Datasets used in the experiments are summarized in the Table 2. The dataset DS1(dataset1) consists of 2040 primary tweets of 1400 users spanning over one month, crawled using Twitter's API, *tweepy*. The dataset contains 1020 tweets that have been positively labeled and 1020 tweets that have been negatively labeled. In addition, the auxiliary data of DS1 consists of 4,000 tweets. Similarly, dataset DS2(dataset2) consists of 16,000 primary tweets of 4,000 users spanning over two months, which consists

of 8,000 are positively labeled tweets and other 8,000 are negatively labeled tweets. In addition, the auxiliary dataset consists of 31,200 tweets. Auxiliary datasets are grouped into two subsets based on the size of the neighborhood window. The window sizes considered in this chapter are $N_\delta = 1$ and $N_\delta = 2$.

Table 3.2: The datasets and their description

Dataset	**#tweets (Primary)**	**#neighborhood tweets (Auxiliary)**	**#users**
DS1	2040 tweets (1020 positive class and 1020 negative class)	4000 tweets	1400
DS2	16,000 tweets (8000 positive tweets and 8000 negative tweets)	31200 tweets	4000

3.3.2 Comparison with Baseline Machine Learning Classifiers

For analyzing the performance of the proposed NTSD model, it is compared with the popular classification techniques such as Support Vector Machines (SVM), Random Forest (RF), and Logistic Regression (LR). These classifiers are chosen for two reasons. First, they are known for better performance among the machine learning classifiers in the literature on stress detection based on social media, especially, when text and related information is exploited [18, 24–26, 34]. Second, all these 3 different baseline models build the model based on different mathematical properties [48, 80, 82, 84–86]. Hence comparison with these models supports the generalizability of the results. The metrics used for comparison are Accuracy and F1-Score, which are defined in section 1.3.

1. **SVM:** Support Vector Machines is a discriminatory classifier that finds a decision boundary to separate data points of different classes. SVM is known for high performance and hence used for wide range of classification problems [48,82,84]. The principle of SVM is to map the data into higher dimension space and to find the maximum margin hyper-plane that separates data points of different classes [48, 84].

2. **RF:** Random Forests is a kind of ensemble method over decision tree classifiers that takes decision based on majority vote [82, 85]. Bagging is used to improve

the consistency and stability of the results of individual decision trees in the forest. The aggregate value of all decisions of individual trees forms the resultant decision of the ensemble [82, 85].

3. **LR:** Logistic regression is a popular method of binary classification which uses logistic function to predict posterior probability of the class [86]. Logistic regression is a generalized linear model that uses sigmoid or logistic function as the link function [80]. LR is a baseline model from which the proposed model in this chapter is developed.

All these classifiers are implemented along with the proposed NTSD model on both the datasets DS1 and DS2.

The following list of experiments are conducted on two datasets DS1 and DS2:

- Building all the classifiers specified in this section along with the proposed NTSD model, to detect the stress when;
  1. no neighborhood tweets are included.
  2. at most one neighborhood tweet is considered along with primary tweets.
  3. at most two neighborhood tweet is considered along with primary tweets.
- In each of the above experiments, two separate models are built for each classifier to analyze the effect of the new attribute, *Sarcasm_Level* - one with only basic features and the other with the new attribute, *Sarcasm_Level*. So, on each of the datasets - DS1 and DS2 - a total of six experiments are run for each classifier.

3.4 Results

This section presents the results of experiments conducted.

Table 3.3: Model results for dataset1 - With basic attributes

Input_dataset / Classifier Model	LR		SVM		RF		NTSD	
	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score
Without neighborhood tweets	69.4	0.40	47.87	0.22	59.7	0.24	**69.4**	**0.51**
With 1 neighborhood tweet	72.29	0.45	52.67	0.28	63.4	0.33	**73.4**	**0.62**
With 2 neighborhood tweets	72.7	0.49	51.06	0.35	68.9	0.39	**75.2**	**0.69**

Table 3.4: Model results for dataset1 - By considering Saracasm_Level attribute

Input_dataset / Classifier Model	LR		SVM		RF		NTSD	
	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score
Without neighborhood tweets	69.94	0.56	50.05	0.25	62.5	0.32	**69.95**	**0.55**
With 1 neighborhood tweet	73.29	0.50	52.67	0.36	63.5	0.42	**74.01**	**0.65**
With 2 neighborhood tweets	73.81	0.66	53.61	0.45	69.6	0.54	**79.51**	**0.73**

Table 3.5: Model results for dataset2 - With basic attributes

Input_dataset / Classifier Model	LR		SVM		RF		NTSD	
	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score
Without neighborhood tweets	70.4	0.42	51.97	0.38	61.72	0.35	**70.5**	**0.55**
With 1 neighborhood tweet	72.59	0.53	54.57	0.42	65.4	0.42	**73.3**	**0.63**
With 2 neighborhood tweets	72.9	0.58	52.23	0.47	69.7	0.48	**75.5**	**0.73**

3.4.1 Evaluation of Stress Detection Performance and Impact of Neighborhood Tweets

To evaluate the performance of the proposed NTSD model, the experiments are conducted on the datasets DS1 and DS2. The performance measures in each experiment are recorded after 10-fold cross-validation. Furthermore, to understand the significance of the results, *t*-tests are conducted for each experiment as part of cross-validation and the null hypothesis is rejected when the results of the proposed model are compared with other models, with a p-value of $\leq .01$, making the findings statistically significant.

Table 3.6: Model results for dataset2 - By considering Saracasm_Level attribute

Input_dataset / Classifier Model	LR		SVM		RF		NTSD	
	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score	Acc(%)	F1-score
Without neighborhood tweets	70.94	0.45	53.05	0.42	64.5	0.38	**70.95**	**0.59**
With 1 neighborhood tweet	73.18	0.55	54.57	0.47	65.5	0.46	**74.1**	**0.66**
With 2 neighborhood tweets	74.01	0.60	55.71	0.51	69.76	0.55	**80.26**	**0.80**

Table 3.3 records the results of different experiments conducted on dataset DS1 when only existing basic features are considered. Here, the highest performance for all the classifiers is observed when two neighborhood tweets are included. When two neighborhood tweets are considered, the NTSD has an improvement in its stress detection accuracy by 5.8% compared to the case when no neighborhood tweet is included. Whereas, the F1-score also increases by 0.18 points. Similarly, LR has an improvement of 2.8% in accuracy and an increase of 0.09 points in F1-score when two neighborhood tweets are included. The baseline model employed in developing in the proposed NTSD model is the LR without inclusion of the neighborhood tweets. When compared to the baseline model, the NTSD with the inclusion of two neighborhood tweets improves accuracy by 5.8 percent and F1-score by 0.28 points. Moreover, NTSD built by considering two neighborhood tweets, also has better performance compared LR model implemented with two neighborhood tweets. RF and SVM also show improvement in performance with the inclusion of neighborhood tweets. RF records highest accuracy of 68.9% and F1 measure of 0.39. On the other hand, SVM records the lowest performance. When implemented on the dataset DS1, the improvement in accuracy of the classifiers with the inclusion of the neighborhood tweets can be observed in the blue bars present in the figure 3.3. Figure 3.5 also illustrates that when neighborhood tweets are taken into account, the F1-score of the classifiers increases significantly. Hence, it can be inferred that including the neighborhood tweets enhances the performance of the model.

The results of the experiments on the dataset DS2 with the basic attributes are recorded in table 3.5. Here, the highest performance for all the classifiers is observed when two neighborhood tweets are included. The NTSD model, built by considering two neighborhood tweets, has a better performance compared to the LR model implemented with the same case. Also, there is a significant increase in the F1-score of the classifiers when the neighborhood tweets are considered as seen in the blue bars present in figure 3.5.

In the figures 3.2a and 3.2b, F1-Scores of all the models are plotted by varying the number of neighborhood tweets. As the median of F1-Scores of all models for the

inclusion of two neighborhood tweets is higher than the inter-quartile region for cases where neighborhood tweets are not considered, it is intuitive that taking previous tweets into account significantly improves the performance of stress detection. Also, the in the case of the proposed model, NTSD – when considered with inclusion of two neighborhood tweets – the median F1-Score is significantly higher than the other cases in all other models. This is true for experiments conducted on both datasets DS1 and DS2. Hence, it can be concluded that considering the immediate neighborhood tweets improves the performance of detecting stress significantly, and the proposed model, NTSD, outperforms the other base-line models.

3.4.2 Contribution of the *Sarcasm_Level* Attribute: An Analysis

Table 3.4 demonstrates the result of applying various experiments on dataset DS1 by including the new attribute of *Sarcasm_Level* in addition to the existing basic features. While table 3.3 presents the results of the experiments on same dataset without including this new attribute. Figure 3.3 shows the bar plot for the accuracy measure of these models. The inclusion of the new attribute has improved the performance of all the models in detecting stress. But, the improvement is greater when the neighborhood tweets are also considered. The NTSD model with two neighborhood tweets has highest improvement in accuracy of 4.31% over NTSD implemented without the new attribute, but has only a small positive increment of F1-score by 0.04 points. When compared to LR model implemented with the new attribute in the case of including two neighborhood tweets, the proposed NTSD model has an improvement in accuracy by 5.7% and F1-score by 0.07 points. There is also a significant improvement in the F1-score when the models are implemented by considering the new attribute *Sarcasm_Level* when compared to the baseline LR. The next highest improvement in F1-score with the inclusion of the new attribute is noted in RF, while SVM models show least change in the F1-score.

When *Sarcasm_Level* is considered, there is nearly 9.56% improvement in the

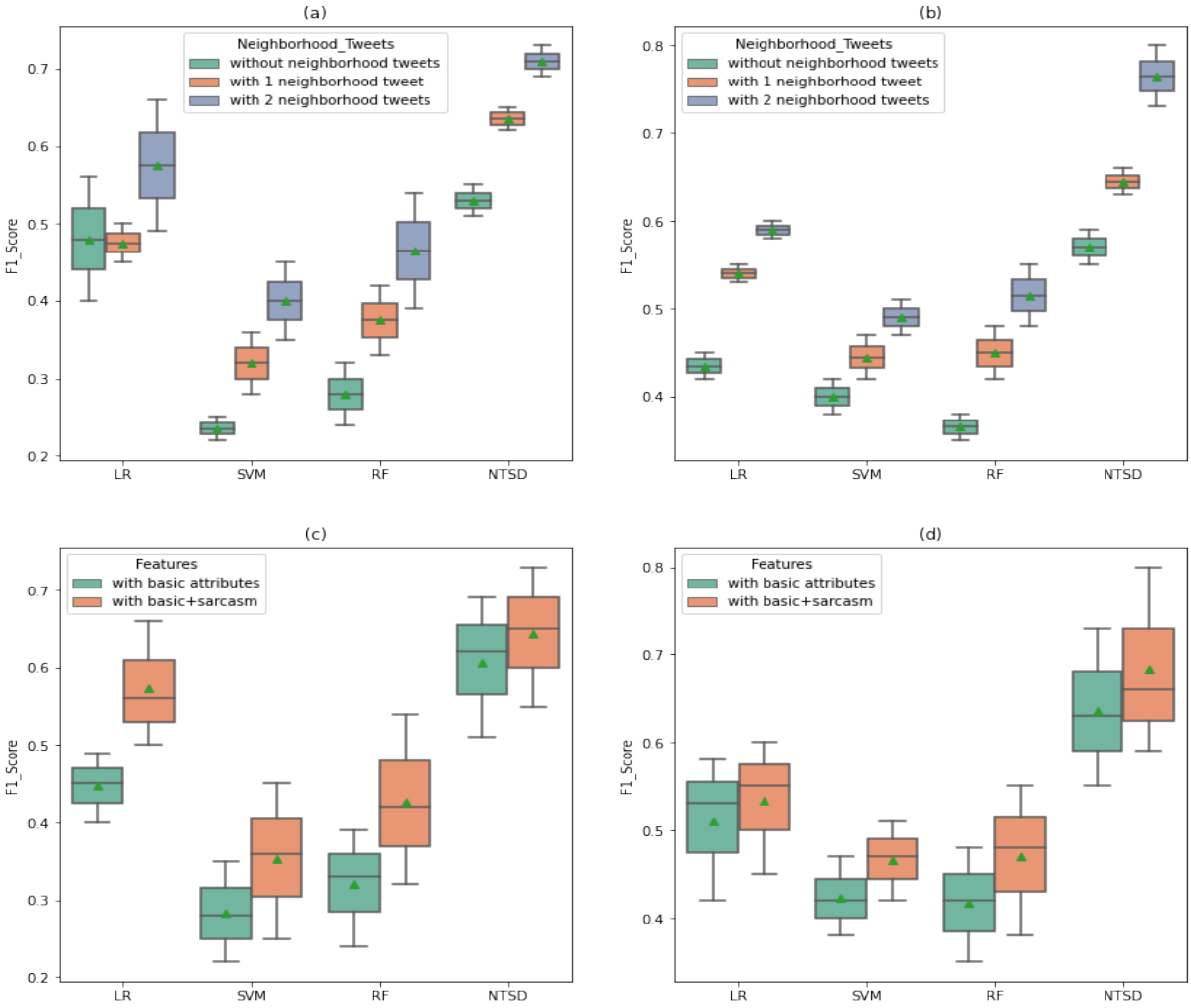

Figure 3.2: Box plots for F1-Score Results of classifiers LR, SVM, RF and NTSD. (a) & (b) present F1-Scores of the classifiers analyzing the effect of neighborhood tweets, performed on datasets DS1 and DS2, respectively. (c) & (d) depict F1-Scores of the classifiers analyzing the effect of the new attribute, *Sarcasm_Level* on datasets DS1 and DS2, respectively and neighborhood tweets on both datasets

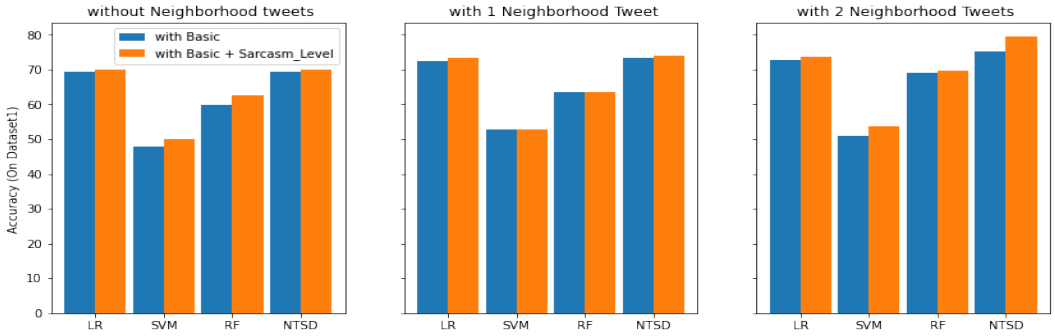

Figure 3.3: Accuracy Results of Models on dataset1

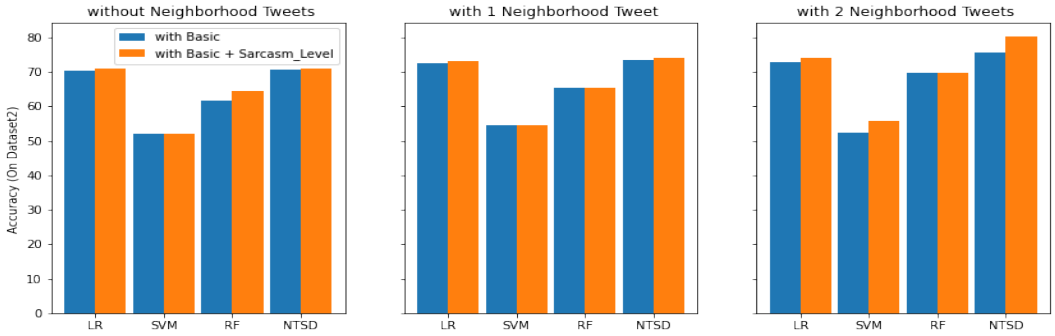

Figure 3.4: Accuracy Results of Models on dataset2

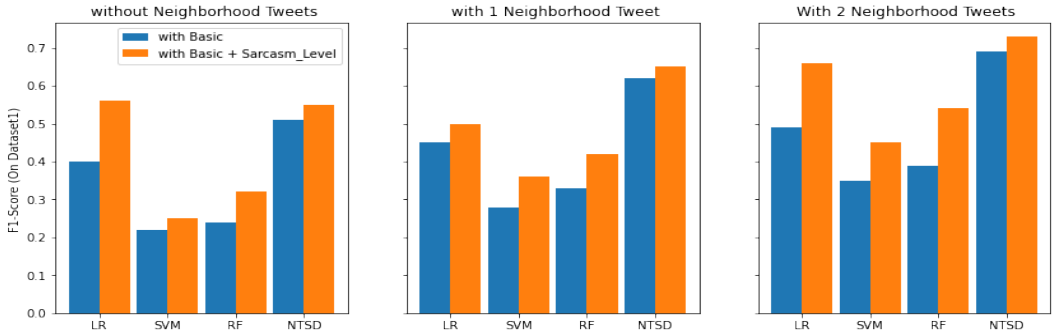

Figure 3.5: F1-Score Results of Models on dataset1

accuracy of detecting stress by NTSD model implemented without the inclusion of the neighborhood tweets, when compared to the baseline LR model. Also, the NTSD model that is implemented with the inclusion of the *Sarcasm_Level* and with two previous tweets, there is a considerable increase of 10.11% in accuracy and 0.33 points in F1-score, when compared to the baseline model.

The results of the experiments on the dataset DS2, by including the *Sarcasm_-Level* attribute, are recorded in table 3.6. According to the results, when two neigh-

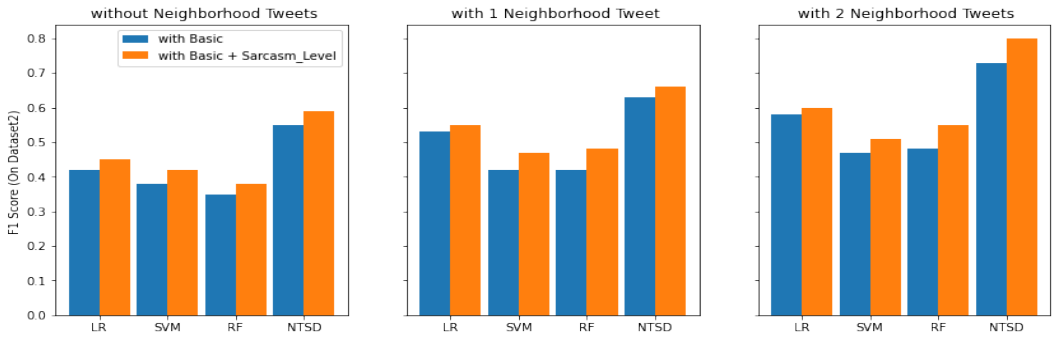

Figure 3.6: F1-Score Results of Models on dataset2

borhood tweets are included, NTSD has a 4.74 percent improvement in accuracy when *Sarcasm_Level* is included over the same model when implemented without the new attribute. When compared to the baseline LR model, the NTSD model - by including the new attribute and considering two neighborhood tweets - has a greater enhancement in the accuracy, and F1-score by 9.86%, and 0.38 points respectively. Also, NTSD model implemented with a new attribute by considering two neighborhood tweets, has an improvement of 6.25% in accuracy, and 0.20 points in F1-score over LR model under the same case. Figure 3.6 presents the bar plot for the F1-scores of the classifiers experimented on the dataset DS2. There is only a small improvement in F1-score of the models when only the attribute *Sarcasm_Level* is included, and it should be noted that both RF and the proposed NTSD model improve F1-score by 0.07 points when compared to other models. It can be seen that when the classifier is built by considering two neighborhood tweets, the performance improvement is greatest in both cases of including and excluding the new attribute, *Sarcasm_Level*.

In the figure 3.2c and 3.2d, F1-Scores of all the considered models are plotted for the cases of including the new attribute *Sarcasm_Level* with basic and the case of only basic attributes. It can be seen that the median of F1-Scores for the case of including the *Sarcasm_Level* attribute in addition to the basic attributes is consistently higher than the medians of other cases. Hence, it is intuitive that taking into account the new attribute, Sarcasm Level, improves the performance of the stress detection. Moreover, the median F1-Score for the proposed model, NTSD, is significantly higher than the other cases for all models when the new attribute is incorporated. This holds true

for all experiments performed on both datasets, DS1 and DS2. Hence, it is concluded that the inclusion of the new attribute improves the performance of stress detection.

3.5 Discussion

From the experiments conducted in this chapter, it was observed that the addition of neighborhood tweets and a new attribute, Sarcasm_Level, improves the performance of tweet-level stress detection. The results show that the textual information of tweets helps in detecting tweet-level stress. Sarcasm embedded in the tweet contributes to the improved performance of stress detection. Also, it is concluded from the experiments that the proposed NTSD model - when implemented with neighborhood tweets and the new attribute *Sarcasm_Level* - outperforms the baseline machine learning models employed in this chapter. Moreover, in the same case, the proposed NTSD model has an improvement of 10.11% in accuracy and 0.33 points in F1-score over baseline logistic regression model on the dataset DS1. While in the same scenario on the dataset DS2, NTSD model has an enhancement in the accuracy by 9.6% and F1-score by 0.38 points. The experiments are conducted on different combinations of features and neighborhood tweets. In all the cases, NTSD model exhibits better than the baseline models. The discussion on the time complexity analysis is given below. Furthermore, the section 3.5.1 discusses the limitations of the proposed model.

3.5.1 Limitations of the proposed NTSD model

There are few limitations in the proposed model, such as:

- The presence of tweets with only URLs does not provide useful content to the model in understanding the stress state of current tweet.
- If some of the neighborhood tweets are not relevant to the current tweet, it provides very little information for understanding the stress state of current

tweet. Consequently, the proposed method will not yield good performance in such situations.

3.6 Summary

Stress has become a major contributor to people's health problems all over the world, necessitating early diagnosis before it advances to chronic disease. Social media-based stress detection has gained popularity due to the large number of people who use it to freely express their opinions. In this chapter, two solutions were proposed to maximize the utilization of text content of tweets for stress detection. First, a new textual content-based attribute of *Sarcasm_Level* is computed to capture the sarcasm embedded in the tweet content. Second, neighborhood tweet-based stress detection, a model that incorporates the contents of a given tweet and those of its neighboring tweets is developed for tweet-level stress detection. This addresses the problem of data sparsity in tweet-level stress detection. The data is collected using Twitter's API, Tweepy. And for every primary tweet, neighborhood tweets are also collected and labeled in preprocessing stage. Also, the experiments are conducted by varying the number of neighborhood tweets in order to understand the impact of the neighborhood tweets on stress detection. The effectiveness of the proposed model is validated against the standard and widely used classifiers like SVM, RF and LR on the two different datasets collected. It is demonstrated that the proposed NTSD model outperforms the other models significantly when implemented with the new attribute, *Sarcasm_Level*.

Chapter 4

Sarcasm-based Tweet-level Stress Detection

From the chapter 3, it is observed that the tweet-level stress detection problem can be addressed by considering neighborhood tweets and a new attribute to compute illocutionary sarcasm in the given tweet. However, the usage of neighborhood tweets is an extra overhead. In this chapter, we develop a solution to tweet-level stress detection using sarcasm attribute without the requirement of neighborhood tweets. In addition, dimensionality reduction is considered to improve the performance.

Organization of the chapter:

The chapter is organized as follows. Section 4.1 discusses the problem formulation, including the formal notations used in this chapter. The proposed STSD model and its training are discussed in section 4.2. The experimental setup and the data collected are presented in section 4.3. The results and the findings of the chapter are presented in section 4.4. The discussion of this chapter is presented in section 4.5. Finally, section 4.6 gives the summary of the chapter.

4.1 Problem Formulation

In this chapter, we develop sarcasm-based tweet-level stress detection using logistic regression-based approach. All the notations used in this chapter are presented in the

table 4.1.

4.1.1 Problem Statement

The aim of the chapter is to build a classifier model for tweet-level stress detection by utilizing information about sarcasm present in the tweet. The problem is to find a function G that takes an unlabeled tweet (z) and considers its *Sarcasm_state* (s), to produce a label (y) for the tweet, where, $y \in \{0, 1\}$ is class label denoting stress of the tweet. The parameters of the function G are learned from training data such that the log-likelihood loss for sarcastic tweets ($s_i = 1$) is penalized while the log-likelihood loss for non-sarcastic tweets ($s_i = 0$) is minimized. The function g is described as $g\colon D \rightarrow C$, where C is the set of unique labels used in the classification (here, $C = \{0, 1\}$). Subsequently, the classifier G is employed to predict the label y of the unlabeled tweet $\mathbf{z}$ by utilizing the information of its *Sarcasm_state*.

Table 4.1: Description of the symbols employed in STSD

Symbol	**Description**
U	Collection of users.
u	Any particular user u, where $u \in U$.
$\mathbf{x_i}$	It is feature vector of i-th tweet in dataset D and there exists some owner $u \in U$, for this tweet.
N	The cardinality or the number of tweets present in the dataset D.
M	Count of attributes in the model or length of the tweet's feature vector $\mathbf{x_i}$.
s_i	The value of *Sarcasm_state* for the tweet $\mathbf{x_i}$.
D_s	The set of sarcastic tweets in the training data.
D_s'	The set of non-sarcastic tweets in the training data, $D = D_s \bigcup D_s'$.
$y_i \in \{0, 1\}$	Class label denoting Stress state of the tweet $\mathbf{x_i} \in D$.
Y	Collection of class labels of all tweets, $\mathbf{x_i}$, $\forall i \in 1, 2, ..., N$, where $N = \|D\|$.
$\mathbf{w}$	The weight vector of features. It is of size M, $\mathbf{w} = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_M]^T$.
$p(\mathbf{x})$	The sigmoid function or logistic function, defined as $p(\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}.\mathbf{x})}$.
$f(\mathbf{w})$	The loss function for the proposed model of STSD.
∇f	Gradient for the loss function, $f(\mathbf{w})$ with respect to parameter vector $\mathbf{w}^T$.
∇f_i	Gradient for loss function at ith sample of training data,D, with respect to weight vector $\mathbf{w}^T$.
η	Learning rate.
$P_s(y_i\|\mathbf{x_i})$	Prediction probability for STSD model, specifying the probability that the class y_i corresponds to tweet $\mathbf{x_i}$.

Table 4.2: The Details of the Datasets

Dataset	#tweets	#positive labeled tweets	#negative labeled tweets	#users	COVID-19 Period
D1	7,289	6,134	1,155	1732	February, 2021
D2	16,532	12,628	3904	5119	March, 2021
D3	4062	3176	886	1888	December, 2021
D4	7209	5495	1714	2558	January, 2022

4.2 Methodology

In this section, the description of datasets along with the operations performed on them is presented. In addition, the methods that form the basis of the proposed model of Sarcasm-based Tweet-level Stress Detection (STSD) are discussed.

4.2.1 Data Collection

To evaluate the model, four datasets of tweets are extracted using Tweepy, Twitter's API. The datasets D1 and D2 were collected during the starting stage of the second wave of COVID-19 in India during the months of February and March 2021. The rest of the datasets are collected in two stages. Dataset D3 is extracted during the starting stage of the third wave in India, from December 2021 to January 2022. Whereas, dataset D4 is collected during the peak stage to tail-end of the third wave in India, from January 2022 to February 2022. The datasets from different time periods are used for implementation to evaluate the model's generalizability. The details of the datasets collected are presented in Table 4.2. The distribution of positive and negative polarity words in each dataset is presented in Table 4.3.

Table 4.3: The number of positive and negative sentiment words in each dataset

Dataset	#positive words	#negative words
D1	4,466	2,689
D2	10,773	5,334
D3	2,687	1,246
D4	4,625	2,031

4.2.1.1 Pre-processing

The datasets collected are first preprocessed, and later features are extracted. Subsequently, the datasets are normalized. The preprocessing and normalization are described in this section.

Tweets are extracted using the Tweepy API. After the collection of tweets, preprocessing should be performed to remove the noise and missing information in the tweets. The procedure is comprised of the following steps:

- *Purging of tweets that are not useful*: The data is cleaned by removing tweets that have no textual content in them. Also, tweets that contain only URLs are removed. In addition, all tweets that contain text in a language other than English are removed
- The collected tweets are processed such that the stopwords and punctuation marks in the text content of the tweets are removed. In the process of cleaning the tweets, the symbols giving no valuable information, like @, $, etc., are removed from the tweets.
- The URLs present in the text content of the tweets are removed.
- The user mentions and retweets labels (RT) are eliminated from the tweets.
- Later, the hashtags from each tweet are collected and later utilized for computing the sarcasm, as shown in equation 4.3 of section 4.2.2.2. Following this, the hashtags are removed from the text content of the tweets.

This filtered set of tweets is used for extracting features relevant for the classification of tweet-level stress. Later, the filtered set of tweets is used in exploratory data analysis. Furthermore, for the extraction of the attributes, the natural language toolkit is utilized for the process of stopword removal and stemming of the words in the tweet.

(b) Normalization: In this chapter, normalization is applied to the datasets after the extraction of the features. Normalization forms an essential pre-processing step for applying Principal Component Analysis [87]. Normalization maps all values to a magnitude of the smaller range, making way for faster computation.

The normalization technique of Z-score is applied on the datasets. Z-score normalization transforms the dataset values such that the resultant data has features with zero mean and unit variance [48]. If x represents the data value and if μ and σ respectively denote the mean and standard deviation of the attribute of the dataset considered, the following is the formula employed in the computation of the Z-score normalization [48].

$$Z = \frac{x - \mu}{\sigma}$$

4.2.2 Approach

The base method employed to develop the proposed model is logistic regression. It is a classical supervised machine learning (ML) algorithm known for its good performance in binary classification [48]. The classification using logistic regression is based on a special function called sigmoid, which is also termed a logistic function and predicts the probability of class for a given feature vector [88]. The parameters used in the sigmoid function are learned by maximizing the likelihood of the training data. From the work [88], it is noticed that, in a logistic regression model, for any feature vector $\mathbf{x}$, whose corresponding class label is y, the likelihood of the tweet $\mathbf{x}$ belonging to the class y is given as:

$$p(\mathbf{x})^{y}(1 - p(\mathbf{x}))^{(1-y)} \tag{4.1}$$

Where, $p(\mathbf{x})$ is sigmoid or logistic function, defined as follows:

$$p(\mathbf{x}) = \frac{1}{1 + \exp\left(-\mathbf{w}.\mathbf{x}\right)}$$

Therefore, the log-likelihood in this scenario is given as:

$$y \log(p(\mathbf{x})) + (1 - y) \log(1 - p(\mathbf{x})) \tag{4.2}$$

In the proposed method, the log-likelihood is developed based on the log-likelihood of logistic regression, shown in equation 4.2.

4.2.2.1 Feature Extraction

In order to predict tweet-level stress more accurately, the information from a tweet's text content and clues of sarcasm in it are leveraged. For this purpose, linguistic-content features are extracted. In this chapter, except the value of *Sarcasm_state*, all the remaining linguistic and social attributes related to tweet's content and social engagement are same as in chapter 3 and are computed as per the procedure discussed in section 3.2.1.1 of chapter 3. The procedure is according to the existing literature [19, 45].

4.2.2.2 Computing Sarcasm_State value

The computation of this value is similar to earlier literature, but with some modifications. Based on the work [45], a new value called *Sarcasm_state* is derived to represent the sarcasm that exists in the tweet's text, built on the notion of illocutionary sarcasm. In this chapter, the computation of sarcasm is further modified to capture the contradictory emotions within words and hash tags, apart from reflecting the inconsistency between the polarity of text content (like words and hashtags) and the polarity of non-verbal expressions like emojis and emoticons. Broadly, the *Sarcasm_state* is allocated a value of 1, when there is a disagreement between the polarity of the majority of words and the polarity of the majority of the hashtags within the text part of the tweet. In addition, the *Sarcasm_state* is given a value of 1 when there is a disagreement between the polarity of the majority of words in a tweet's text content and the polarity of the majority of non-verbal expressions like emojis and emoticons. Similarly, the *Sarcasm_state* is assigned a value of 1 in cases

where there is a disagreement between the polarity of the majority of hashtags and the polarity of the majority of non-verbal expressions like emojis and emoticons. In all the remaining cases, where there is similar polarity for the majority of text content and the majority of non-verbal expressions like emojis and emoticons, the *Sarcasm_-state* is assigned a value of 0. Hence, the process of computing *Sarcasm_State* is an extension of computation as shown in equation 3.2 of chapter 3. the computation of *Sarcasm_state* is presented in equation 4.3.

$$Sarcasm\_State(t) = \begin{cases} 1, \text{ if } \Big\{ (e_{pos} \geq e_{neg} \ \wedge \ pos \leq neg) \ \vee \\ \qquad (e_{pos} \leq e_{neg} \ \wedge \ pos \geq neg) \ \vee \\ \qquad (h_{pos} \geq h_{neg} \ \wedge \ pos \leq neg) \ \vee \\ \qquad (h_{pos} \leq h_{neg} \ \wedge \ pos \geq neg) \ \vee \\ \qquad (h_{pos} \geq h_{neg} \ \wedge \ e_{pos} \leq e_{neg}) \ \vee \\ \qquad (h_{pos} \leq h_{neg} \ \wedge \ e_{pos} \geq e_{neg}) \Big\} \\ 0, \text{ otherwise} \end{cases} \tag{4.3}$$

4.2.3 Proposed STSD

In this section, the concept, formulation, and procedure of the proposed model of Sarcasm-based Tweet-level Stress Detection (STSD) are discussed. The aim of this approach is to develop a classification model to detect stress at tweet-level by availing sarcasm information, thereby increasing the utilization of the text information for better performance.

4.2.3.1 Framework

The framework representing the concept of the proposed STSD model is depicted in figure 4.1. The figure shows the proposed Sarcasm-based Tweet-level Stress Detection (STSD) method consists of two phases - STSD training and STSD prediction. During training, the tweets are preprocessed and the feature extraction is performed. The resulting processed training data is categorized into two sets based on the value of

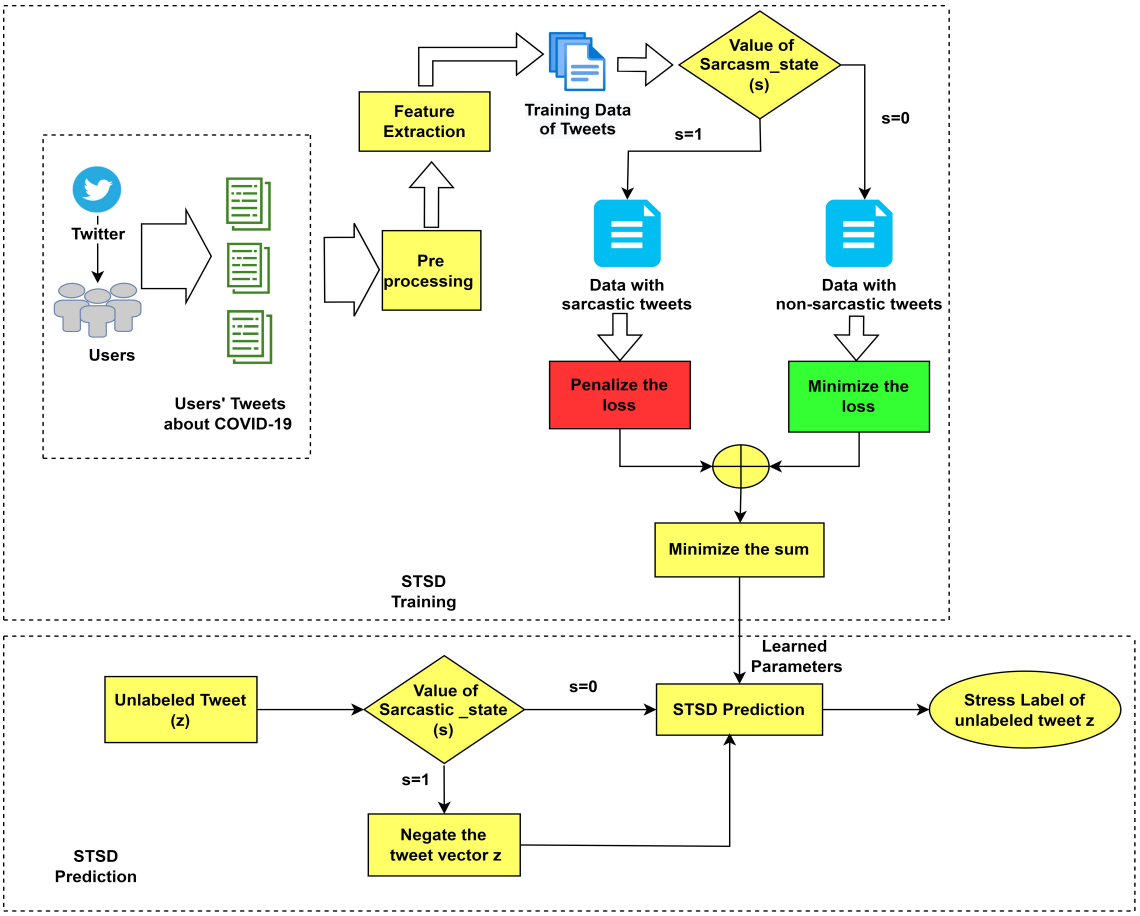

Figure 4.1: Framework of the STSD model

Sarcasm_state. In this approach, the label for stress is learned by minimizing the loss for non-sarcastic tweets and penalizing the loss for sarcastic tweets, as sarcastic tweets generally have subtle sentiment that is contrary to the explicit emotion specified. For this purpose, the value of sarcasm is utilized in the loss function of the proposed approach. During the prediction stage, the unlabelled input vectors are processed based on the value of the *Sarcasm_state*. The symbols employed to formulate and implement this problem are described in Table 4.1. The detailed procedure of all the steps involved in the process is presented in section 4.2.3.

4.2.4 Working of Sarcasm-based Tweet-level Stress Detection (STSD)

In this section, the detailed process of the proposed STSD method is described, from the initial preprocessing of tweets to prediction. Also, the idea and working mechanism of the proposed Sarcasm-based Tweet-level Stress Detection (STSD) are presented. Finally, dimensionality reduction techniques for improving the performance of the proposed STSD model are also discussed. As noted in the works related to sarcasm in human emotions across many diverse geographies, the reason for the usage of sarcasm is more for cases like "to be funny", and having fun or entertainment with a group of friends [36]. The principle of STSD is formulated based on this idea - the sarcastic tweets are likely to be on a lighter or funny note than the non-sarcastic tweets.

4.2.4.1 Mathematical formulation of training of the proposed STSD model

In this section, the training procedure and related algorithms of the proposed STSD model are discussed. To understand the loss function and training procedure of the proposed model, the idea and working mechanism of the proposed model of STSD are presented.

The principle of the proposed STSD model is to maximize the likelihood of a tweet belonging to the class of stress if it is a non-sarcastic tweet and to minimize the likelihood of a tweet belonging to the class of stress if it is a sarcastic tweet. This can be interpreted as-to minimize the loss for non-sarcastic tweets and penalize the loss for sarcastic tweets.

From the work [88], it is noticed that, in a logistic regression model, for any feature vector $\mathbf{x}$, whose corresponding class label is y, the likelihood of the tweet $\mathbf{x}$ belonging to the class y is given as:

$$p(\mathbf{x})^{y}(1-p(\mathbf{x}))^{(1-y)}$$

Where, $p(\mathbf{x})$ is Sigmoid or logistic function, defined as follows:

$$p(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}.\mathbf{x})}$$

Therefore, the log-likelihood in this scenario is given as:

$$y \log(p(\mathbf{x})) + (1 - y) \log(1 - p(\mathbf{x}))$$

According to the principle of the proposed model, the likelihood of the model is constructed based on the likelihoods of sarcastic and non-sarcastic tweets as follows:

1. For non-sarcastic tweets ($s_i = 0$), the likelihood of belonging to the class stress is maximized. Hence, the likelihood of the tweets $\mathbf{x_i} \in D_s{}'$ is presented as follows [88]:

   $$p(\mathbf{x_i})^{y_i}(1 - p(\mathbf{x_i}))^{1-y_i}$$

   Then the log-likelihood in this scenario is given by

   $$y_i \log(p(\mathbf{x_i})) + (1 - y_i) \log(1 - p(\mathbf{x_i})) \tag{4.4}$$

2. The equations used in this step and later on are developed as part of the proposed STSD model. Following the principle of STSD, the likelihood of belonging to the class stress is to be minimized for sarcastic tweets ($s_i = 1$). This is achieved by maximizing the likelihood of belonging to the class non-stressed and minimizing the likelihood of belonging to the class stress. Hence, the likelihood for the tweets $\mathbf{x_i} \in D_s$ is

   $$p(\mathbf{x_i})^{1-y_i}(1 - p(\mathbf{x_i}))^{y_i}$$

   Then the log-likelihood in this case is given as

   $$(1 - y_i) \log(p(\mathbf{x_i})) + y_i \log(1 - p(\mathbf{x_i})) \tag{4.5}$$

Hence, using the equations (4.4) and (4.5), the total log-likelihood, $l(\mathbf{w})$, of the proposed model, computed for all the tweets in the training dataset, is given as:

$$l(\mathbf{w}) = \sum_{i=1}^{N}(1-s_i)\Big\{y_i \log(p(\mathbf{x_i})) + (1-y_i)\log(1-p(\mathbf{x_i}))\Big\} + \sum_{i=1}^{N} s_i\Big\{y_i \log(1-p(\mathbf{x_i})) + (1-y_i)\log(p(\mathbf{x_i}))\Big\}$$

Then the loss of the proposed model STSD, $f(\mathbf{w})$, is computed as negative value of log-likelihood $l(\mathbf{w})$:

$$f(\mathbf{w}) = -l(\mathbf{w})$$

$$f(\mathbf{w}) = -\Big(\sum_{i=1}^{N}(1-s_i)\Big\{y_i \log(p(\mathbf{x_i})) + (1-y_i)\log(1-p(\mathbf{x_i}))\Big\} + \sum_{i=1}^{N} s_i\Big\{y_i \log(1-p(\mathbf{x_i})) + (1-y_i)\log(p(\mathbf{x_i}))\Big\}\Big) \quad (4.6)$$

In other words, the concept of the proposed STSD model is defined as minimizing the loss for non-sarcastic tweets while penalizing the loss for sarcastic tweets. For solving the parameters of the proposed model, gradient descent-based algorithms are implemented. The optimization algorithms of gradient descent require the computation of the gradient of the loss function of the proposed model, which is derived in Equation 4.8. As the prediction function, p, is a composite function of both weight vector, $\mathbf{w}$ and feature vector $\mathbf{x_i}$, the computation of the gradient of loss function of the proposed STSD model is as follows:

$\frac{\partial f}{\partial \mathbf{w}} = \nabla f$

$$= -\Bigg[\sum_{i=1}^{N} \frac{\partial\Big\{(1-s_i)\Big\{y_i \log(p(\mathbf{x_i})) + (1-y_i)\log(1-p(\mathbf{x_i}))\Big\}\Big\}}{\partial \mathbf{w}} + \sum_{i=1}^{N} \frac{\partial\Big\{(s_i)\Big\{y_i \log(1-p(\mathbf{x_i})) + (1-y_i)\log(p(\mathbf{x_i}))\Big\}\Big\}}{\partial \mathbf{w}}\Bigg] \quad (4.7)$$

$$= -\Bigg[\sum_{i=1}^{N}(1-s_i)\Big\{y_i \frac{1}{p(\mathbf{x_i})}\frac{\partial p(\mathbf{x_i})}{\partial \mathbf{w}} + (1-y_i)\frac{1}{(1-p(\mathbf{x_i}))}\frac{\partial(1-p(\mathbf{x_i}))}{\partial \mathbf{w}}\Big\} + \sum_{i=1}^{N} s_i\Big\{y_i \frac{1}{(1-p(\mathbf{x_i}))}\frac{\partial(1-p(\mathbf{x_i}))}{\partial \mathbf{w}} + (1-y_i)\frac{1}{p(\mathbf{x_i})}\frac{\partial p(\mathbf{x_i})}{\partial \mathbf{w}}\Big\}\Bigg]$$

$$= -\Bigg[\sum_{i=1}^{N}\Big\{(1-s_i)(y_i - p(\mathbf{x_i}))x_{im}\Big\} + \sum_{i=1}^{N}\Big\{s_i(1-y_i-p(\mathbf{x_i}))x_{im}\Big\}\Bigg]$$

$$\implies \nabla f = -\Big[\sum_{i=1}^{N}\Big\{y_i - 2y_i s_i + s_i - p(\mathbf{x_i})\Big\}x_{im}\Big] \quad (4.8)$$

4.2.4.2 Steps of the proposed STSD framework and its training algorithm

The whole process describing the training and prediction of the proposed STSD model is presented in the Algorithm 4. The Algorithm describes how the information of sarcasm present in the tweet is extracted from each tweet of the training data and is utilized for learning the parameters using gradient descent. This can be seen from steps 4-9 in Algorithm 4. The procedure call at step 6 invokes the method to compute sarcasm using equation 4.3 and returns the value of sarcasm. The procedure call of *Gradient_Descent_STSD* at step 9 invokes Algorithm 5, which returns learned parameters, $\mathbf{w}$. To predict labels for unlabeled tweets, first the value of *Sarcasm_state*

of the tweets is determined and then the label is predicted with the help of sigmoid function.

Algorithm 4: Sarcasm-based tweet-level stress detection

Input: Labeled Dataset of Tweets, D
Output: Object with learned parameters and labels for unlabeled tweets

Function `STSD(`D`)`**:**
- $Y \leftarrow \left\{ y_i \middle| (\mathbf{x_i}, y_i) \in D, \quad 1 \leq i \leq N \right\}$
- Perform initial preprocessing of the data
- $\mathbf{s} \leftarrow \phi$
- **for** $\boldsymbol{x_i} \in D$ **do**
  - $s\_i \leftarrow Sarcasm\_state(\mathbf{x_i})$;
  - $\mathbf{s} \leftarrow \mathbf{s} \bigcup s\_i$
- **end**
- $\mathbf{w} \leftarrow Gradient\_Descent\_STSD(D, \mathbf{s})$
- $p(\mathbf{x}) \leftarrow \frac{1}{1+e^{-\mathbf{w}.\mathbf{x}}}, \quad \forall$ tweet $\mathbf{x}$
- $\mathbf{Z} \leftarrow Unlabeled\_tweets$
- $Y_Z \leftarrow \phi$
- **for** $\boldsymbol{z} \in \boldsymbol{Z}$ **do**
  - $s_z \leftarrow Sarcasm\_state(\mathbf{z})$;
  - **if** $s_z = 1$ **then**
    - $prediction\_probability \leftarrow p(-\mathbf{z})$
  - **else**
    - $prediction\_probability \leftarrow p(\mathbf{z})$
  - **end**
  - **if** $prediction\_probability > 0.5$ **then**
    - $y_z \leftarrow 1$
  - **else**
    - $y_z \leftarrow 0$
  - **end**
  - $Y_Z \leftarrow Y_Z \bigcup y_z$
- **end**
- $result\_object \leftarrow \{\mathbf{w}, Y_Z\}$

return $result\_object$

Theorem 4.2.1. *The loss function of the proposed STSD model is a special form of logistic regression loss.*

Proof. From the loss function of the proposed STSD model, presented in Equation (4.6), it is observed that at any given iteration, only one component of the loss

function is computed due to the binary variable s_i. For the tweets with $s_i = 1$, only the second component is computed, while only the first component is computed for the tweets with s_i=0.

From the definition of the sigmoid function [48]:

$$p(-\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}.(-\mathbf{x}))} = \frac{\exp(-\mathbf{w}.\mathbf{x})}{1+\exp(-\mathbf{w}.\mathbf{x})} = 1 - \frac{1}{1+\exp(-\mathbf{w}.\mathbf{x})} = 1 - p(\mathbf{x})$$

Hence, using the above relation, the loss function is rewritten as:

$$\begin{aligned} & f(\mathbf{w}) = -l(\mathbf{w}) \\ \implies & f(\mathbf{w}) = -\Big(\sum_{i=1}^{N} (1-s_i) \Big\{ y_i \log(p(\mathbf{x_i})) + (1-y_i)\log(1-p(\mathbf{x_i})) \Big\} \\ & \qquad + \sum_{i=1}^{N} s_i \Big\{ y_i \log(p(-\mathbf{x_i})) + (1-y_i)\log(1-p(-\mathbf{x_i})) \Big\} \Big) \end{aligned} \tag{4.9}$$

From Equation (4.9), it is noticed that the two components in the summation are characterized by the value of *Sarcasm_state*, s_i. The first component of the loss exists only if the value of *Sarcasm_state* vanishes if $s_i = 0$ and the second component exists if $s_i = 1$. Hence, equation (4.9) can be rewritten into a single component as follows:

$$f(\mathbf{w}) = -\Big(\sum_{i=1}^{N} \Big\{ y_i \log(p(\mathbf{x_{ti}})) + (1-y_i)\log(1-p(\mathbf{x_{ti}})) \Big\} \Big) \tag{4.10}$$

where,

$\mathbf{x_{ti}} = \mathbf{x_i}$,if $s_i = 0$

$\mathbf{x_{ti}} = -\mathbf{x_i}$, if $s_i = 1$

The equation (4.10) is similar to a logistic loss function [88]. Hence, it is demonstrated that the proposed STSD model's loss function reduces to a special case of the logistic regression loss. □

The gradient descent and stochastic gradient descent algorithms for learning the parameters of the proposed STSD model are described in Algorithms 5 and 6, respectively. Both the Algorithms use the gradient of the loss function of the proposed

model, ∇f, as specified in Equation (4.8).

Algorithm 5: Gradient Descent Algorithm for training of STSD Model

Input: Training dataset consisting of D containing the features of tweets in dataset, with $|D| = N$ and $\mathbf{s} = \{s_i | i = 1, 2, ..., N\}$, a vector representing the *Sarcasm_state* of each tweet and η, the rate of convergence

Output: Parameter Vector $\mathbf{w}$ of the objective function

Function `STSD_Gradient_Descent(`$D, \mathbf{s}$`):`
- Initialize parameter vector $\mathbf{w}$ to small random values ;
- **while** *convergence is not reached* **do**
  - $\nabla f = \sum_{i=1}^{N}(y_i - 2y_i s_i + s_i - p(\mathbf{x_i}))x_{im}$;
  - Update the weight vector, $\mathbf{w} \leftarrow \mathbf{w} + \eta * \nabla f$;
- **end**

return w

Algorithm 6: Stochastic Gradient Descent Algorithm for training of STSD Model

Input: Training dataset consisting of D containing the features of tweets in dataset, with $|D| = N$ and $\mathbf{s} = \{s_i | i = 1, 2, ..., N\}$, a vector representing the *Sarcasm_state* of each tweet and η, the rate of convergence

Output: Parameter Vector $\mathbf{w}$ of the objective function

Function `STSD_Stochastic_Learning(`$D, \mathbf{s}$`):`
- Initialize parameter vector $\mathbf{w}$ to small random values ;
- **while** *Convergence is not reached* **do**
  - **for** *each training example* $(\boldsymbol{x_i}, y_i)$ **do**
    - $\nabla f_i = (y_i - 2y_i s_i + s_i - p(\mathbf{x_i}))x_{im}$;
    - Update the weight vector as, $\mathbf{w} \leftarrow \mathbf{w} + \eta * \nabla f_i$
  - **end**
- **end**

return w

The model's parameters, which have been learned during training, are used in the prediction function to predict the label for the unlabeled tweets. Based on the principle of STSD, the prediction function for the proposed STSD model is defined

with respect to the value of *Sarcasm_state*, s_i, of the tweet $\mathbf{x_i}$ as shown below:

$$P_s(y_i|\mathbf{x_i}) = \begin{cases} P(y_i|\mathbf{x_i}) & \text{, if } s_i = 0 \\ P(y_i|-\mathbf{x_i}) & \text{, otherwise} \end{cases}$$

where, $P(y_i|\mathbf{x_i})$ is sigmoid function and hence, $P(y_i|-\mathbf{x_i}) = p(-\mathbf{x_i}) = \frac{1}{1+\exp(-\mathbf{w}^T.\mathbf{x})}$.

4.2.4.3 Dimensionality Reduction

In this chapter, various techniques of dimensionality reduction are applied so as to improve the classification performance of the model. Given the large number of features and sparse nature of the training data of tweet-level stress detection, applying an effective dimensionality reduction technique is essential to reduce the features and to improve the performance of the model [48]. An exploratory analysis of the datasets is conducted by applying popular dimensionality reduction techniques such as Linear Principal component Analysis (PCA) and Nonlinear or Kernel Principal component Analysis [89].

In linear PCA, the data is projected onto a linear subspace of lower dimensions [48]. But linear PCA does not perform well when the data is linearly non-separable [73].

In non-linear or kernel-PCA, the features are transformed into higher dimensions using a non-linear mapping and, later, linear PCA is applied in the transformed higher space [89]. Given the high computational cost in computing the transformation to higher dimensions, kernel trick is used [48]. The kernel trick is helpful as it avoids the computation of mapping to higher dimensions but allows the computation of linear sub-spaces of higher dimensions using kernel functions within the original input space.

The three popular kernels used in kernel PCA are the Linear kernel, Radial Basis Function (RBF) kernel, and Polynomial kernel. In this chapter, Linear kernel PCA, RBF kernel PCA, and Polynomial kernel PCA are applied on 4 different datasets, and it is observed that polynomial kernel PCA (with degree 3) has good separation of classes when projected onto the two principal components. Hence, the proposed

model employs kernel PCA with a polynomial kernel for dimensionality reduction. The detailed discussion of the obtained results using three PCA techniques on all the datasets is presented in section 4.4.

4.3 Experimental Setup

This section describes the experimental setup, utilized datasets, and the employed machine learning models, which were used to assess the proposed model's performance. All the experiments were conducted using Python with the IDE Jupyter Notebook 4.1.1.

4.3.1 Dataset Description

To evaluate the model, four datasets of tweets were extracted using Tweepy, Twitter's API. The datasets D1 and D2 were collected during the starting stage of the second wave of COVID-19 in India during the months of February and March, 2021. The rest of the datasets were collected in two stages. Dataset D3 is extracted during the starting stage of the third wave in India, from December 2021 to January 2022. Whereas, dataset D4 was collected during the peak stage to tail-end of third wave in India -January 2022 to February 2022. The datasets belonging to different time-periods were used for implementation to validate the generalizing ability of the model.

Tweets were extracted using Tweepy with two types of queries. The tweets were collected using the query "*I feel Stressed*" along with the search words "*COVID-19*",'*covid*". All the tweets collected using this query and keywords were labelled as stressed ($y_i = 1$). While the tweets collected using the query "*I feel relaxed*", "*I don't feel stressed*" along with the search words "*COVID-19, covid*" are labelled non-stressed ($y_i = 0$). This is because "*I feel*" pattern-based extraction has proved to be effective in labelling tweets [19, 45]. The collected datasets were cleaned to remove noise. The details of the datasets collected are presented in Table 4.2.

4.3.2 Models utilized for comparison

Popular ML classifiers such as Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), and Naive Bayes (NB) are used to evaluate the performance of the proposed STSD model. All of these models, except Naïve Bayes (NB) are discussed in section 3.3.2 of chapter 3. The metrics used for comparison are Accuracy and F1-Score, which are defined in section 1.3. The abstract details of the Naïve Bayes model are given below:

Naïve Bayes: Naïve Bayes (NB) is a popular binary classification algorithm built upon the concept of Bayes theorem [48]. It predicts posterior probabilities for each class label for a given unlabeled tuple using Bayes theorem [48]. The tuple is assigned to a class with the highest posterior probability. Naïve
Bayes assumes class-conditional independence, where each of the input features is independent of each other [48, 66, 90].

On the four datasets-D1, D2, D3 and D4-the following experiments were conducted:

1. All the models considered in this chapter (baseline models and the proposed STSD model) are built with original features without applying any dimensionality reduction technique.

2. Transforming the data by applying the polynomial kernel PCA.

3. Building all the models considered in this chapter on data of reduced dimensions.

4.4 Results

This section contains the results for all the experiments that are performed as part of this chapter.

4.4.1 Visualization of PCA results

This section discusses the results of exploratory data analysis after applying PCA techniques as part of the dimensionality reduction task that is described in section 4.2.4.3. In this chapter, to reduce the dimensionality of the data and to mitigate the low performance of the models, a linear PCA technique and a kernel PCA technique with two different kernels—RBF Kernel and Polynomial Kernel—were employed. In the figures 4.2, 4.3, 4.4 and 4.5, the plots respectively show the projection of the datasets D1, D2, D3, and D4 on the two principal components after applying various PCA techniques used in this chapter.

For the dataset D1, the separation of points into stressed and non-stressed classes is more clear after applying a polynomial kernel, as noted from the figure 4.2. But after applying Linear PCA and RBF kernel PCA, the classes were not linearly separable. Similarly, for dataset D2, the points of the stressed class are hidden behind the non-stressed points after applying polynomial kernel PCA, as seen from the figure 4.3. For dataset D3, it is observed from figure 4.4 that the data-points were linearly separable after applying polynomial kernel PCA. Also, in the case of dataset D4, there was a clustering of two classes in both directions of the axes of principal components, as observed in the figure 4.5. There was no clear linear separation of the points in the projection after polynomial kernel PCA in the case of dataset D4. Nonetheless, in all the four datasets, the data points were grouped into nearly linearly separable classes after the application of polynomial kernel PCA. Consequently, this helps in better classification. Hence, the polynomial kernel PCA technique was used for feature reduction in this chapter before building the classifiers.

4.4.2 Evaluation of the Proposed STSD Model

For evaluating the proposed STSD model's performance, many experiments were implemented on four datasets such as D1, D2, D3, and D4. All the results were recorded following the 10-fold cross validation of the model. Furthermore, the statistical significance of the results is analysed by conducting *one-sample Wilcoxon signed rank*

Figure 4.2: The projection of Dataset D1 on two principal components after different PCA techniques

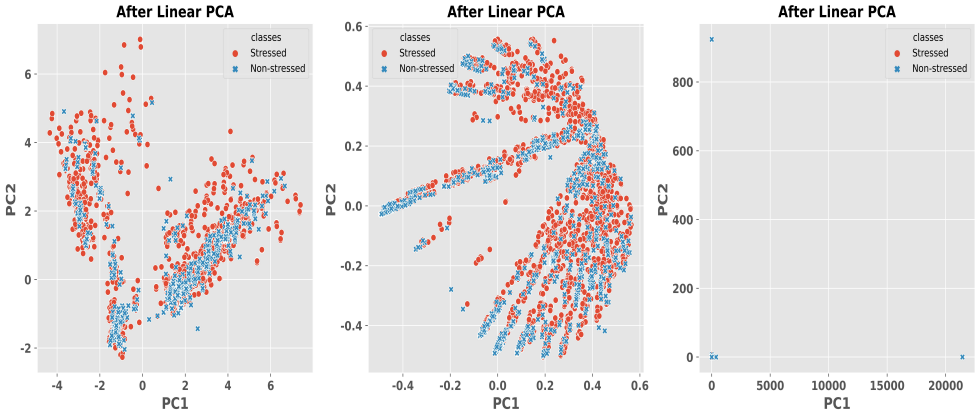

Figure 4.3: The projection of Dataset D2 on two principal components after different PCA techniques

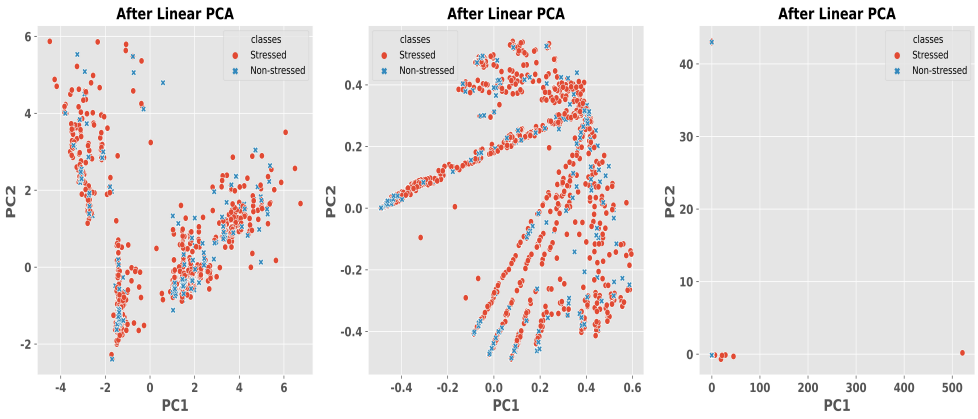

Figure 4.4: The projection of Dataset D3 on two principal components after different PCA techniques

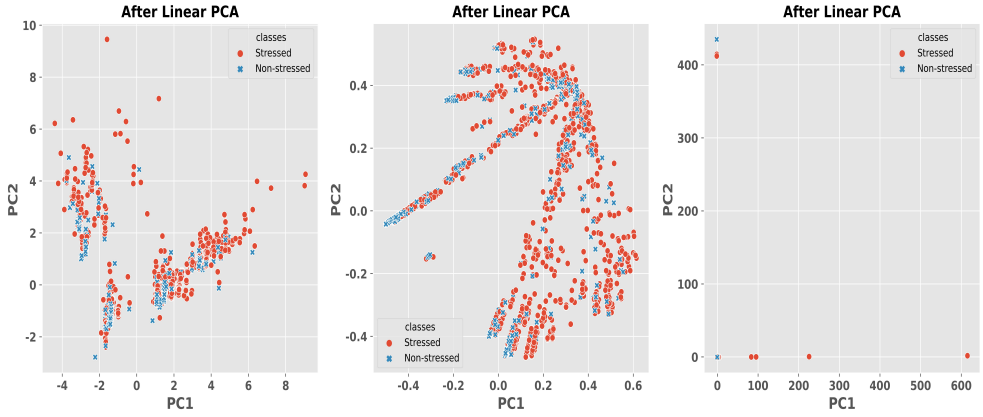

Figure 4.5: The projection of Dataset D4 on two principal components after different PCA techniques

test [91]. The null and alternate hypothesis for the *one-sample Wilcoxon signed rank test* were given as:

H_0 : The mean performance of other models is equal to the mean performance of the proposed STSD model.

H_1 : The mean performance of the other models differs from the mean performance of the proposed STSD model.

The decision of rejecting the null hypothesis is determined when observations of a proposed model are analysed with other models. In this chapter, the rejection of the null hypothesis was done with a $p-$value ≤ 0.05, inferring that the recorded results are statistically significant [91].

Table 4.4 stores the results of the experiments executed on the baseline models using the four datasets of D1, D2, D3, and D4 with original features. Initially, experiments were conducted on baseline models of Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), and Naïve Bayes (NB) along with the proposed STSD model. In Table 4.4, it can be observed that the proposed STSD performed better than all other models on all the four datasets. The STSD model delivered an accuracy of 77.63% on dataset D1 and recorded an F1-score of 0.762. The SVM performed second best with respect to accuracy but records a low F1-score when compared with the proposed model. In the experiments on dataset D2, the

proposed STSD model had leading performance with 76.79% accuracy and F1-score of 0.770. Here, SVM had the second best performance with 73.94% accuracy and an F1-score of 0.761. In the experiments on datasets pertaining to the third wave, D3 and D4, the STSD outperformed other baseline models. The STSD achieved an accuracy of 77.24% and an F1-score of 0.792 in experiments on dataset D3. While SVM and LR were ranked second and third in terms of both accuracy and F1-score. Also, on dataset D4, STSD records an accuracy of 76.75% and an F1-Score of 0.780, outperforming all other baseline models. Hence, when implemented with original features, the proposed STSD model delivered better performance when compared with all other baseline ML models.

Table 4.5 recorded the performances of all the models in this chapter after the application of polynomial kernel PCA. From Table 4.5, it is noted that the performance of all the models on all datasets improves significantly by applying polynomial kernel PCA. The proposed STSD model delivers a leading performance with accuracy of 86.82% and F1-Score of 0.926 on dataset D1, while SVM has second highest accuracy of 83.74% and RF had second highest F1-Score of 0.880. For dataset D2, the proposed STSD records a high accuracy 83.01% with a high F1-Score of 0.855, while SVM recorded the second highest accuracy of 77.17% and RF recorded the second best F1-score of 0.799. After implementation on the datasets concerned with the third wave, D3 and D4, the proposed STSD model exhibited better performance than any other baseline models. The STSD exhibited a high accuracy of 86.3% and an F1-Score of 0.907 on dataset D3, while NB recorded the second best accuracy of 79.2% and SVM gave the second best F1-Score of 0.816. In the case of dataset D4, the STSD lead with an accuracy of 82% and an F1-Score of 0.889, while SVM had the second best accuracy of 76.86% and RF recording the second best F1-Score of 0.785. Accordingly, it was concluded that the proposed STSD leads in accuracy and F1-Score when compared to all other baseline ML models considered.

The bar-plots representing accuracy for each of the classifiers when implemented with original features and with polynomial kernel PCA, are presented in figure 4.6. It is observed that the STSD has the highest accuracy in both cases-first, with original

features, shown with red coloured bars; and second, with polynomial kernel PCA, shown with cyan coloured bars. And the STSD model exhibits better accuracy than any other baseline model in both cases for all four datasets. Similarly, the bar-plots for F1-Score for each of the models considered with original features and with polynomial kernel PCA are depicted in figure 4.7. And the STSD model exhibits a better F1-Score than all other baseline models in both cases-first, with original features, shown with red coloured bars; and second, with polynomial kernel PCA, shown with cyan coloured bars. This is true for all the four datasets.

Table 4.4: Performance results for experiments on all four datasets - with original features

Datasets	**D1**		**D2**		**D3**		**D4**	
Models/Measures	**Accuracy** (%)	**F1-Score**	**Accuracy** (%)	F1-Score	**Accuracy** (%)	**F1-Score**	**Accuracy** (%)	**F1-Score**
LR	74.62	0.734	73.92	0.750	74.91	0.772	74.4	0.761
SVM	77.30	0.744	73.94	0.761	75.96	0.772	75.06	0.762
RF	74.40	0.734	72.01	0.755	73.80	0.780	74.30	0.761
NB	74.30	0.734	72.70	0.750	74.01	0.772	74.70	0.761
STSD	**77.63**	**0.762**	**76.79**	**0.770**	**77.24**	**0.792**	**76.75**	**0.780**

Table 4.5: Performance results for experiments on all four datasets - By applying polynomial kernel PCA

Datasets	**D1**		**D2**		**D3**		**D4**	
Models/Measures	**Accuracy** (%)	**F1-Score**	**Accuracy** (%)	F1-Score	**Accuracy** (%)	**F1-Score**	**Accuracy** (%)	**F1-Score**
LR	82.71	0.805	77.16	0.781	78.08	0.798	76.34	0.770
SVM	83.74	0.821	77.17	0.788	77.64	0.816	76.86	0.770
RF	82.2	0.880	76.8	0.799	78.1	0.807	76.4	0.785
NB	82.6	0.860	76.3	0.765	79.2	0.790	76.2	0.766
STSD	**86.82**	**0.926**	**83.01**	**0.855**	**86.3**	**0.907**	**82**	**0.889**

4.4.2.1 Analyzing the effect of using dimensionality reduction with STSD

The low performance of the models in their original features was noted in Table 4.4. The dimensionality reduction technique of polynomial kernel PCA was applied because it grouped the classes of tweets, as seen from the results of preprocessing in

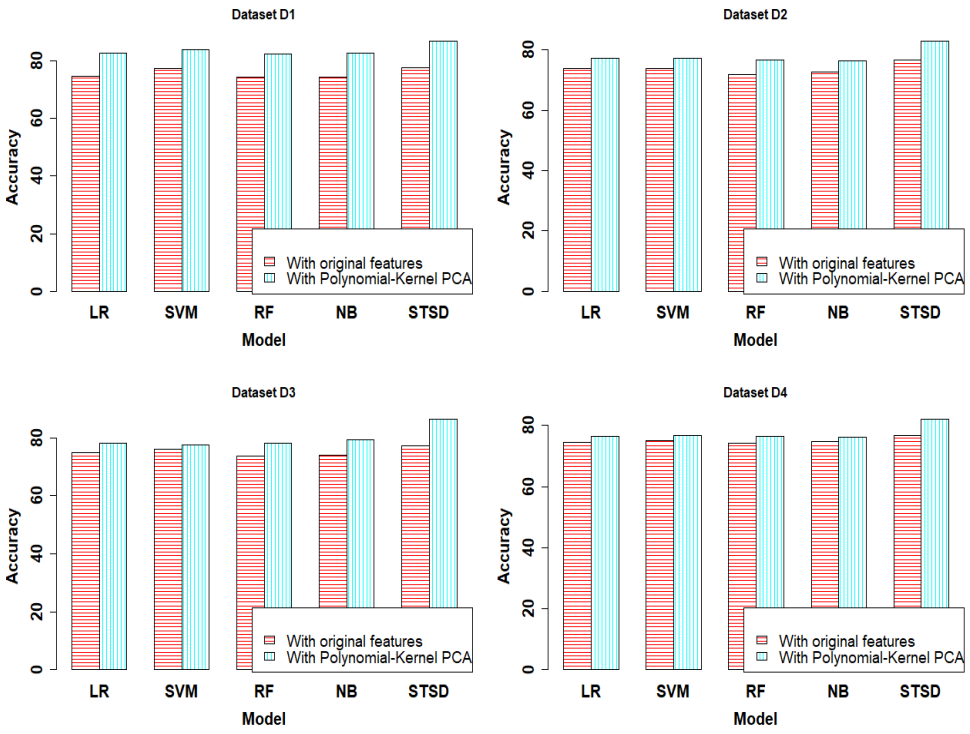

Figure 4.6: The bar-plots representing the accuracies of all models on the datasets D1, D2, D3, and D4

section 4.4.1. This helps in better classification. The accuracy and F1-score results on experiments conducted in this chapter on all four datasets with original features were presented in Table 4.4. While the results of experiments after applying polynomial kernel PCA were presented in Table 4.5. The bar-plot in figure 4.6 depicted the comparison of accuracies exhibited by all the models when implemented with original features and polynomial kernel PCA, for all the four datasets. The bar-plot in figure 4.7 compares the F1-Scores obtained by all models when implemented with original features and polynomial kernel PCA. The cyan bars in the figure 4.6 denote the accuracy of the models after the application of kernel PCA. Hence, it was observed that for all the models considered, the accuracy improves with the dimensionality reduction technique of polynomial kernel PCA. And the largest improvement was seen in the proposed STSD on all the four datasets. STSD with kernel PCA records at least 9.19% improvement in accuracy when compared to all the models implemented without PCA on dataset D1. While the same STSD recorded an improvement in

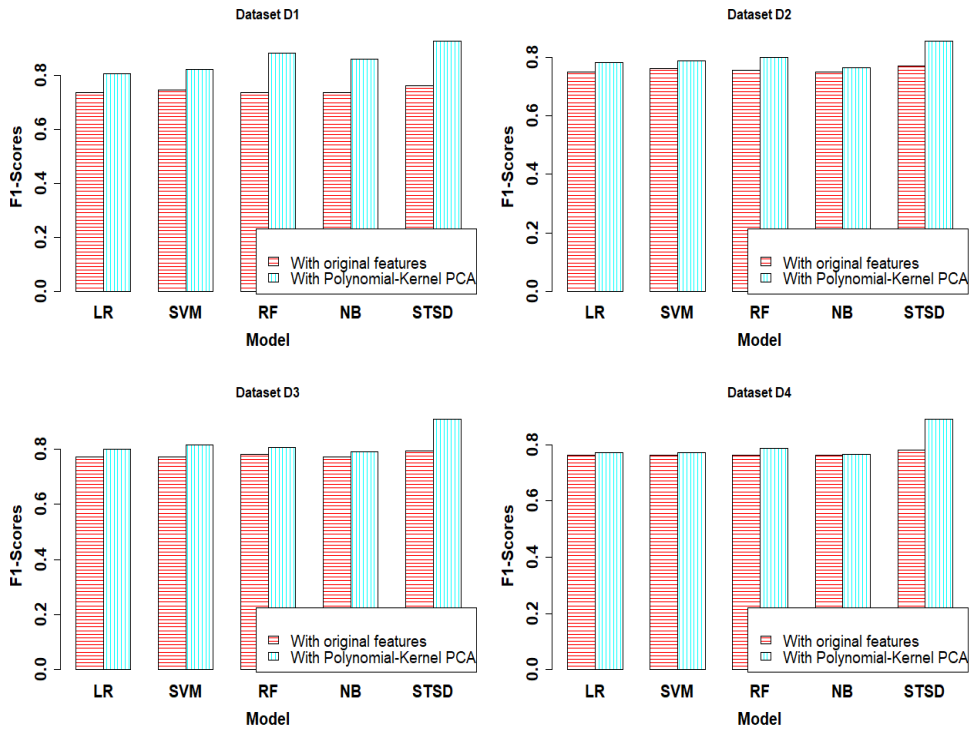

Figure 4.7: The bar-plots representing the F1-measure values of all models on the datasets D1, D2, D3, and D4

accuracy by 6.22%, 9.06% and 5.25% when compared to other models implemented on datasets D2, D3, and D4, respectively. The least improvement in accuracy was observed when models were implemented on dataset D4. This was due to a lack of clear separation of classes after the application of polynomial kernel PCA, as observed from figure 4.5.

Similarly, the cyan bars in the figure 4.7 denote the F1-scores of the models after the application of the polynomial kernel PCA. It was observed that the F1-Score of all the models has seen an improvement after using the dimensionality reduction technique of polynomial kernel PCA. Moreover, the largest increment in F1-Score over all the datasets was noted for the proposed STSD model. When compared to all models implemented without PCA on dataset D1, STSD with kernel PCA improved F1-Score by at least 0.164 points. Whereas, the same STSD recorded an improvement of 0.085 points, 0.115 points, and 0.109 points when compared to other models implemented on datasets D2, D3, and D4, respectively. The least improvement

in F1-Score was observed when models were implemented on dataset D4. This was due to the lack of clear separation of classes after the application of polynomial kernel PCA, as observed from figure 4.5.

4.5 Discussion

This section presents the discussion of the results of the proposed STSD model when compared to the state-of-the-art models. It is known from the previous section that the proposed STSD had outperformed the baseline models on the datasets collected from two different time periods of COVID-19, as evidenced by the results presented in section 4.4.2.

4.5.1 Comparison with state-of-the-art

The proposed STSD is compared with two state-of-the-art models, like the NTSD proposed in [45]. However, in [45], the methodology used neighborhood tweets, which would be computationally costly. And the proposed STSD approach doesn't address the concept of stress detection with the inclusion of neighborhood tweets. Nevertheless, NTSD, in the case of no-neighborhood tweets, can be compared with the proposed STSD. It includes sarcasm as an attribute. Table 4.6 shows the best results obtained in terms of accuracy and F1-score by the models of NSTD and proposed STSD. These performances are recorded when implemented on dataset D1. After application of the PCA, the proposed STD outperformed the NTSD in accuracy and F1-score.

Table 4.6: Accuracy recorded by NTSD and the proposed STSD on dataset D1

Model/Performance	**Accuracy**	**F1-Score**
NTSD without PCA	75.92	0.758
STSD without PCA	77.63	0.762
NTSD with PCA (polynomial kernel)	84.16	0.846
STSD with PCA (polynomial kernel)	**86.82**	**0.926**

It is concluded that with the use of sarcasm information, STSD outperforms other baseline models. Furthermore, with the effective use of the PCA techniques, a better separation of classes is achieved. This resulted in good improvements in the performances by all the models considered. And, the proposed STSD model recorded the highest magnitude of enhancement in performance. Hence, STSD with polynomial kernel PCA is the better model for tweet-level stress detection, by enhancing the utilization of tweet-content.

4.5.2 Limitations of the proposed STSD model

The limitations of the proposed STSD model are presented below:

- The proposed model will work only if the tweets contain textual content information.
- As text is the major source for extracting the features, the proposed model would perform well only for tweets that contain sarcasm or its forms present in textual content rather than embedded in media other than text.

4.6 Summary

Psychological stress has emerged as a major global health problem, and social media-based stress detection has caught the attention of numerous researchers. In this chapter, for detecting tweet-level stress, a novel sarcasm-based tweet-level stress detection (STSD) classifier has been developed for availing the information related to sarcasm present in the tweet-content.

The principle of the STSD model is to minimize the loss for non-sarcastic tweets while penalizing the loss for sarcastic tweets. Subsequently, a theorem is framed to prove the loss function of the proposed STSD model as the special form of the standard logistic loss. The experiments are conducted on four different datasets, with datasets D1 & D2 collected during the second wave of the COVID-19 pandemic and the datasets D3 & D4 collected during the third wave of the pandemic in India.

A thorough preprocessing is performed and an appropriate dimensionality reduction technique of polynomial kernel PCA is applied to all the datasets so as to enhance the performance of the models. From the experimental results, it is noted that the proposed STSD outperforms all the other baseline models in both the cases of the implementation with original features and the implementation after polynomial kernel PCA. The STSD model with polynomial kernel PCA records accuracies of 86.82%, 83.01%, 86.3%, and 82%, with datasets D1, D2, D3, and D4 respectively. Also, STSD with polynomial kernel PCA achieves an improvement in accuracy of at least 9.19%, 6.22%, 9.06%, and 5.25% when compared to the baseline models implemented without PCA on datasets D1, D2, D3, and D4 respectively. Moreover, STSD with polynomial kernel PCA records F1-Scores of 0.926, 0.855, 0.907, and 0.889 with datasets D1, D2, D3, and D4 respectively. Also, STSD with polynomial kernel PCA achieves an improvement in F1-score by at least 0.164 points, 0.085 points, 0.115 points, and 0.109 points when compared to all other baseline models implemented without PCA on datasets D1, D2, D3, and D4 respectively. In addition, all the models considered in this chapter record improved performance when implemented with polynomial kernel PCA as compared to their implementation without PCA. Wherein, the proposed STSD shows the highest improvement in accuracy and F1-score when compared to the increment recorded by all the other baseline models.

Chapter 5

Semi-supervised Approach for Tweet-level Stress Detection

Supervised learning-based solutions of NTSD and STSD are proposed in chapters 3 and 4, respectively, for tweet-level stress detection with better utilization of text data. Since there is a severe lack of availability of large amounts of labeled data for the problem, in this chapter, we propose a semi-supervised approach to solve the problem of text-level stress detection.

Organization of the chapter:

The chapter is organized as follows. Section 5.1 discusses the problem formulation, including the formal notations used in this chapter. The proposed SMTSD model and its training are discussed in section 5.2. The experimental setup and the data collected is presented in section 5.3. The results and the findings of the chapter are presented in section 5.4. The discussion of the results of this chapter is presented in section 5.5. Finally, section 5.6 gives the summary of the chapter.

5.1 Problem Formulation

In this chapter, a semi-supervised approach called self-training method for tweet-level stress detection (SMTSD) is proposed. All the notations used in this work are presented in Table 5.1.

Let D be the dataset containing a small amount of labeled tweet data, $D^L = \{\mathbf{x_i}^L, y_i^L\}$ for the users $u \in U$, and a large amount of unlabeled tweet data $D^\nu = \{\mathbf{x_i}^\nu\}$, for the users $w \in U$. Also, let the set of tweet features in labeled training data is $D_X^L = \{(\mathbf{x}|(\mathbf{x}, y) \in D^L)\}$ and let C is the set of unique labels used in the classification (here, $C = \{0, 1\}$). The problem is to find a function g, described as $g : D_X^L \bigcup D^\nu \longrightarrow C$, that learns the parameters so that it predicts labels for unlabeled tweets, D^ν. The function g is later used to predict the label for any unlabeled tweet $\mathbf{z}$.

Table 5.1: Description of the symbols employed in SMTSD

Symbol	**Description**
U	Collection of users
u	Any particular user u, where $u \in U$ for labeled data
w	Any particular user u, where $w \in U$ for unlabeled data
D^{ν}	Unlabeled dataset of tweets
D^{L}	Labeled dataset of tweets
D_X^L	Set of tweets for Labeled dataset of tweets
D	The total training data, $D = D^L \bigcup D^{\nu}$
$\mathbf{x_i}^L$	It is feature vector of i-th tweet in labeled dataset D^L and there exists some owner $u \in U$, for this tweet.
$\mathbf{x_i}^{\nu}$	It is feature vector of i-th tweet in unlabeled dataset D^{ν} and there exists some owner $w \in U$, for this tweet.
N_L	The cardinality or the number of tweets present in the labeled dataset D^L
N_{ν}	The cardinality or the number of tweets present in the unlabeled dataset D^{ν}
N	The cardinality or the number of tweets present in the dataset D , $N = N_L + N_{\nu}$
ψ	The length of the tweet's feature vector $\mathbf{x}$
s_i	The value of *Sarcasm* for the i^{th} tweet $\mathbf{x_i}$
$s[j]$	The value of *sarcasm* in j^{th} element of *Sarcasm* vector, $s_j = s[j]$
D_s	The set of sarcastic tweets in Training data.
D_s'	The set of non-sarcastic tweets in Training data, $D = D_s \bigcup D_s'$
$y_i \in \{0, 1\}$	Class label denoting Stress state of the tweet $\mathbf{x_i} \in D$
Y^L	Collection of class labels of all tweets, $\mathbf{x_i}^L$, $\forall i \in 1, 2, ..., N_L$, where $N_L = \lvert D^L \rvert$
y_j^*	The label in combined data, $D_{TR}^L \cup \tilde{D}$, where $j \in \left\{1, 2, ... \lvert D_{TR}^L \bigcup \tilde{D} \rvert\right\}$
$\tilde{Y}$	Collection of pseudo-labels that needs to be predicted for unlabeled tweets, $\mathbf{x_i}^{\nu}$, $\forall i \in 1, 2, ..., N_{\nu}$, where $N_{\nu} = \lvert D^{\nu} \rvert$
β, γ	The weight vectors of features. They are of size ψ,$\beta = [\beta_1, \beta_2, ..., \beta_{\psi}]^T$, $\gamma = [\gamma_1, \gamma_2, ..., \gamma_{\psi}]^T$,
pos	number of positive polarity words in the tweet $\mathbf{x}$
neg	number of negative polarity words in the tweet $\mathbf{x}$
e_+	number of positive polarity emojis and emoticons in the tweet $\mathbf{x}$
e_-	number of negative polarity emojis and emoticons in the tweet $\mathbf{x}$
h_+	number of hashtags with positive polarity in the tweet $\mathbf{x}$
h_-	number of hashtags with negative polarity in the tweet $\mathbf{x}$
$p(\mathbf{x})$	The sigmoid function or logistic function, defined as $p(\mathbf{x}) = \frac{1}{1+\exp(-\mathbf{w}.\mathbf{x})}$
$f()$	The sigmoid function learned by the model, when trained on labeled data
$\hat{f}(\mathbf{x_i}, \beta)$	predicted label after training with labeled data, D_{TR}^L
$g()$	The sigmoid function learned by the model, when trained on combined data of labeled and pseudo-labeled data
$l(\beta, \hat{f}())$	The loss function for training the labeled training data, D_{TR}^L
$\hat{g}(\mathbf{x_i}, \gamma)$	predicted label after training with combined data, $D_{TR}^L \bigcup \tilde{D}$
$l(\gamma, \hat{g}())$	The loss function for the combined proposed model of SMTSD after training the combined data, $D_{TR}^L \bigcup \tilde{D}$
$P_s(y_i \lvert \mathbf{x_i})$	Prediction probability for SMTSD model, specifying the probability that the class y_i corresponds to tweet $\mathbf{x_i}$
β^*	weight vector learned by the model after learning from labeled data, D_{TR}^L. It is of size ψ.
γ^*	weight vector learned by the model after learning from combined data, $D_{TR}^L \bigcup \tilde{D}$, It is of size ψ.

5.2 Methodology

This section presents the proposed approach and methodology developed for a semi-supervised solution for tweet-level stress detection. Here, the problem statement is defined first, and then the features required for the model are discussed. Later, the proposed model is discussed in detail.

5.2.1 Features Extraction

this chapter focuses on utilizing semi-supervised learning methods for improving the performance of tweet-level stress detection. The aim is to exploit the information inherently present in the text content of the tweets for better stress detection. The features employed in the proposed method are based on state-of-the-art works [19,45]. Except for the value of sarcasm, all the set of features to be computed and the procedure to compute them is according to the section 3.2.1.1 of chapter 3. However, the sarcasm is computed as per the procedure discussed in 4.2.2.2 of chapter 4.

The proposed solution incorporates a sarcasm vector in its approach, the computation of which is similar to the works [45, 46]. Here, the concept of "illocutionary sarcasm" is implemented, which captures the contrast between the explicit meaning conveyed in the text and the emotions specified in emojis and expressions [40, 45]. In this method, the concept of sarcasm is used as an attribute in the basic self-training and other baseline approaches as per the previous literature in this domain [45]. However, in case of the proposed SMTSD method, sarcasm is not considered as one of the attributes.

5.2.1.1 Computation of Sarcasm

The value of sarcasm is computed based on the existing literature [45, 46]. It is developed on the basis of illocutionary sarcasm, where the explicit meaning of the sentence is in contradiction to the facial expressions [40]. In this chapter, the sarcasm is computed for all the tweets in the dataset, and a vector storing the sarcasm value for each tweet is computed and returned. The process for calculating sarcasm for a

set of tweets is described in algorithm 7. Here, the sarcasm of a tweet is assigned a value of 1, if there is a contrast in the polarity of the majority of text content and non-textual content. Also, sarcasm of tweet is assigned 1, if there is a contradiction between the polarity of the majority of hashtags and the majority of words in the text content. For all the remaining cases, it is assigned a value of 0. In Table 5.2, examples of the variables used to compute the sarcasm function are presented.

It is to be noted that the aim of the proposed method is to utilize the information of sarcasm in the self-training process to decide the pseudo-labels. To this end, the computation of sarcasm is performed during the process of self-training in the proposed model of SMTSD. The procedure for computing is invoked after predicting the confidence of the pseudo-labels.

Table 5.2: Examples of the variables used to compute the sarcasm

Variable	**Examples of the variable**
Positive Words	'successful', 'good', 'HAPPY', 'easy', 'healthy', 'free', 'incredible', 'extraordinary', etc.
Negative Words	'grief', 'mental', 'poor', 'alcoholic', 'tough', 'forced', 'difficult', 'worst', 'disappointment', etc.
Positive Hashtags	#love, #joy, #HappyHolidays, #LoveYourHeart, #PrimaryCare, #'CredibleMind, #CleanAir, #HotTea.., etc.
Negative Hashtags	#anxious, #anger, #LongCovid #StaySafe, #MentalHealth, #Grief, #LongTermCare, #FakeNews, etc.
Positive Emojis	 etc.
Negative Emojis	etc.

5.2.2 Self-training Method for Tweet-level Stress Detection (SMTSD)

Using the self-training approach in semi-supervised learning, this chapter proposes a solution for the tweet-level stress detection problem. In semi-supervised learning, self-training is a popular approach in which unlabeled data is assigned pseudo-labels based on predictions made by a model trained on a small set of labeled data [61, 92]. The pseudo-labels predicted with high confidence are added back to the labeled data, thereby expanding it. This process is repeated iteratively until one of two conditions is met: either there are no more unlabeled tweets or there are no pseudo-labels predicted with high confidence [93]. This approach is popular and has been found to

Algorithm 7: Computing_Sarcasm

Input: Dataset of Tweets,D consisting of tweet vectors $D^L = \{\mathbf{x}_1, \mathbf{x}_2, ..\}$
Output: Sarcasm_vector, **s**

Function `Sarcasm(`D`)`**:**
  sarcasm_level $\leftarrow \phi$
  $i \leftarrow 0$
  for $\mathbf{x} \in D$ **do**
    $pos \leftarrow$ number of positive polarity words in the tweet **x**
    $neg \leftarrow$ number of negative polarity words in the tweet **x**
    $e_+ \leftarrow$ number of positive polarity emojis and emoticons in the tweet **x**
    $e_- \leftarrow$ number of negative polarity emojis and emoticons in the tweet **x**
    $h_+ \leftarrow$ number of hashtags with positive polarity in the tweet **x**
    $h_- \leftarrow$ number of hashtags with negative polarity in the tweet **x**
    if $e_+ \geq e_- \,\&\, pos < neg$ **then**
      $s_i \leftarrow 1$
    end
    if $e_+ < e_- \,\&\, pos \geq neg$ **then**
      $s_i \leftarrow 1$
    end
    if $h_+ \geq h_- \,\&\, pos \leq neg$ **then**
      $s_i \leftarrow 1$
    end
    if $h_+ \leq h_- \,\&\, pos \geq neg$ **then**
      $s_i \leftarrow 1$
    end
    if $h_+ \geq h_- \,\&\, e_+ \leq e_-$ **then**
      $s_i \leftarrow 1$
    end
    if $h_+ \leq h_- \,\&\, e_+ \geq e_-$ **then**
      $s_i \leftarrow 1$
    else
      $s_i \leftarrow 0$
    end
    $i \leftarrow i + 1$
  end
  $Sarcasm\_Vector \leftarrow \bigcup_{i \in |D|} s_i$
return $Sarcasm\_Vector$

be successful in improving model performance in cases of abundant unlabeled data as it gradually uses the information from unlabeled data [92, 93].

In the proposed approach of SMTSD, self-training approach with logistic regression as the base-learner is extended by utilizing the information of sarcasm of the tweet. The framework of the proposed model is depicted in Figure 5.1. The process is similar to the generic self-training approach until the prediction of pseudo-labels. Later, in the proposed model, the information from the value of sarcasm (s) is computed for each tweet in the dataset of confident pseudo-labels. And it is used to categorize the tweets into sarcastic ($s = 1$) and non-sarcastic ($s = 0$). In pseudo-labeled data with high confidence, for the tweets falling under the "sarcastic" category, the predicted pseudo-labels are negated, and the updated data is added back to the labeled training data. For pseudo-labeled tweets falling under the "non-sarcastic" category, the pseudo-label remains unchanged. The extended training data after the addition of updated pseudo-labeled data is retrained. This process is repeated until either no more unlabeled data is present or no more confident pseudo-labels are predicted. Subsequently, the test data taken from the original labeled data is used to assess the performance of the model. The parameters learned are later utilized for predicting the stress labels for any new unlabeled tweets in the future. In the proposed solution of self-training, the base-learner employed to build the models on labeled data is logistic regression, as it is known for better performance on the binary classification of tweet-level stress [45].

5.2.2.1 Training Algorithm

The training algorithm for the proposed SMTSD is based on the principle of self-training of the semi-supervised learning mechanism, extended by employing information related to sarcasm.

The training process is presented in algorithm 8. The algorithm's initial input training dataset is divided into two sets, labeled (D^L) and unlabeled (D^ν), with sizes N_L and N_ν, respectively, where $N_\nu >> N_L$. As unlabeled data is updated and changed for each iteration, the unlabeled data available for any iteration is maintained

in the dataset, $D^{\nu}_{current}$, by removing tweets of high-confidence pseudo-labels. And the labeled dataset, D^L, is further subdivided into training and testing sets, D^L_{TR} and D^L_{TS}, respectively. The traditional supervised model of logistic regression is trained over D^L_{TR}. The algorithm is trained in this chapter to minimize the loss, $l()$, between the actual label y^L_i and the predicted label, $\hat{f}(\mathbf{x_i}^L, \beta)$, where the $model^{(L)}$ learns the sigmoid function $f()$ with the parameters β.

For the labeled training dataset, D^L_{TR}, the total loss is defined as :

$$Labeled\_Loss = \sum_{i=1}^{|D^L_{TR}|} l(y^L_i, \hat{f}(\mathbf{x_i}^L, \beta)) \tag{5.1}$$

In the case of logistic regression, the loss function l is given by the cross-entropy coefficient [88], which is given as follows:

$$l(y^L_i, \hat{f}(\mathbf{x_i}^L, \beta)) = y^L_i \log(p(\mathbf{x_i}^L)) + (1 - y^L_i) \log(1 - p(\mathbf{x_i}^L)) \tag{5.2}$$

where, p is the sigmoid function and is defined as follows [88] :

$$p(\mathbf{x_i}^L) = \hat{f}(\mathbf{x_i}^L, \beta) = \frac{1}{1 + \exp(-\beta.\mathbf{x_i}^L)} \tag{5.3}$$

The sigmoid function represents the posterior probability of class, y^L_i for the given tweet vector, $\mathbf{x_i}^L \in D^L_{TR}$.

The resultant model is then utilized to predict pseudo-labels for an unlabeled dataset, where, the pseudo-labels predicted with high-confidence are of main interest. The criterion for determining a pseudo-label's confidence is computed using the posterior probability of the label, $p(y|\mathbf{x})$. And labels y that exceed the threshold, τ, in their posterior probability predictions are referred to as "confident labels". The sarcasm value s_i of confident pseudo-labels is then computed for each data point $\mathbf{x_i}$ in the dataset of confident pseudo-labels, $\tilde{D}$. The sarcasm vector, hence, is defined as $\mathbf{s} = \bigcup_{i \in \tilde{D}} s_i$. Subsequently, based on the value of the sarcasm computed for the pseudo-labeled tweets, the pseudo-labels are modified. The labels are negated for

the data vectors whose value of sarcasm(s) is 1. The pseudo-labels for the vectors corresponding to non-sarcastic ($s = 0$) tweets, on the other hand, remain unchanged. Finally, the updated pseudo-labeled dataset is added back to the training data, extending it. The updated data is then trained again to minimize the combined loss, which is given as follows:

$$Combined\_Loss = \sum_{j=1}^{|D_{TR}^{L} \cup \tilde{D}|} l(y_j^*, \hat{g}(\mathbf{x_j}, \gamma)) \tag{5.4}$$

where, $l()$ is the cross-entropy function as defined in equation 5.2. Also, y_j^* is the labels of combined data, $D_{TR}^{L} \bigcup \tilde{D}$. While $\hat{g}$ is the sigmoid function for the given tweet vector, $\mathbf{x_j} \in D_{TR}^{L} \bigcup \tilde{D}, \quad \forall j \in \{1, 2, ...|D_{TR}^{L} \bigcup \tilde{D}|\}$ and is defined as in the equation 5.3:

$$\hat{g}(\mathbf{x_j}, \gamma) = p(\mathbf{x_j}) = \frac{1}{1 + \exp(-\gamma.\mathbf{x_j})} \tag{5.5}$$

The current unlabeled dataset, $D_{current}^{\nu}$, is updated after each iteration by removing tweets with high-confidence pseudo-labels, $\tilde{D_X}$. The final model of the proposed SMTSD is formed by the resultant parameters, γ^*, after all iterations. The idea of availing the information related to sarcasm in the model is to maximize the utilization of text content in predicting tweet-level stress and improve the performance of the general semi-supervised approach.

The loss functions in equations (5.1) and (5.4) are functions of the parameters β, γ. The goal of the training is to learn the optimal values of these parameters, β^*, γ^* at which the loss functions are at their minimum. These optimal values are solved using optimization of the loss functions (5.1) and (5.4) [88].

5.2.3 Prediction

After training the model with confident pseudo-labels for unlabeled data, the model is built, and the logistic function is employed to predict the labels of unlabeled and new tweets. The prediction is given as follows [88]:

$$P(y|\mathbf{x_i}) = \frac{1}{1 + \exp\left(-\gamma^*.\mathbf{x_i}\right)} \tag{5.6}$$

The labels are assigned based on the prediction probability as follows:

$$y = \begin{cases} 1 & \text{, If } P(y|\mathbf{x_i}) \geq 0.5 \\ 0 & \text{, If } P(y|\mathbf{x_i}) < 0.5 \end{cases} \tag{5.7}$$

5.3 Experimental Setup

This section presents the datasets collected and the experiments conducted as part of the proposed SMTSD model. All the experiments are conducted on Python 3.7.15 using the IDE Jupyter Notebook 4.1.1.

5.3.1 Data collection

Twitter's tweepy API is used to collect five different datasets of tweets. The datasets D1 and D2 were respectively collected during February and March, 2021, when India was reeling under the rise of the second wave of COVID-19, which made people experience more stress. The dataset D3 was collected in December 2021, whereas the dataset D4 was collected in January 2022, when the third wave of COVID-19 affected India. Of the collected data, only about thirty percent of the tweets are labeled. Using the process mentioned in literature [19, 45], the data is collected with the Tweepy search API by using the patterns "I feel stressed" and "covid-19" for positive stress. And the patterns used to collect negative stress are "I am not feeling stressed" and "covid-19". The pattern "covid-19" is employed in data collection so as to collect user opinions related to the pandemic. The datasets are manually labeled by three experts, and the majority label is considered. In each of the collected datasets, only about thirty percent are labeled, leaving seventy percent of the data unlabeled. Among the thirty percent, ten percent is kept aside as test data for measuring the performance. In this chapter, we evaluate the performance of basic supervised machine learning

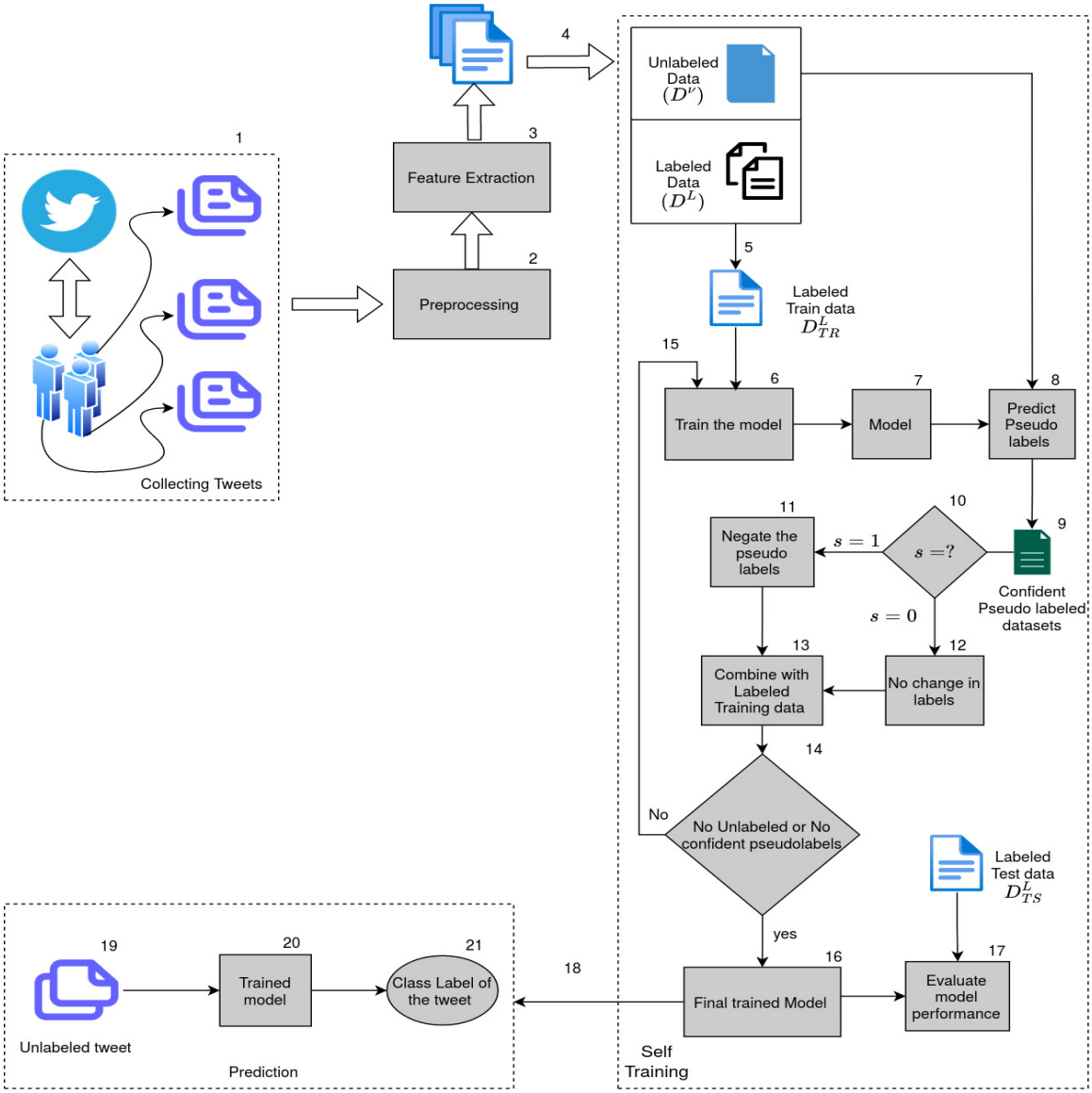

Figure 5.1: Framework of the SMTSD model

Algorithm 8: Proposed self-training semi-supervised based approach

Input: Dataset of Tweets,D consisting of labeled part $D^L = \{(\mathbf{x}_1^L, y_1^L), (\mathbf{x}_2^L, y_2^L), ..., (\mathbf{x}_M^L, y_M^L))\}$ and unlabeled part $D^\nu = \{\mathbf{x}_1^\nu, \mathbf{x}_2^\nu, ..., \mathbf{x}_{N-M}^\nu\}$ and confidence threshold τ

Output: Model object with learned parameters and labels for unlabeled tweets

Function `SMTSD(`D`)`**:**

Perform initial preprocessing of the data;

Organise labeled data into training and testing parts, $D^L \leftarrow D_{TR}^L \bigcup D_{TS}^L$

$c_p = 1$

$D_{current}^\nu \leftarrow D^\nu$

for $D_{current}^\nu \neq \phi$ *and* $c_p \neq 0$ **do**

$model^{(L)} \leftarrow \arg\min_{\beta} \sum_{i=1}^{M} l(y_i^L, \hat{f}(\mathbf{x_i}^L, \beta))$;

$(f, \beta^*) \leftarrow model^{(L)}$;

$\tilde{y_i} \leftarrow f(\mathbf{x_i}^\nu, \beta^*) \quad \forall i \in \{1, 2, ..., |D_{current}^\nu|\}$;

$\tilde{D} \leftarrow \{(\mathbf{x_i}^\nu, \tilde{y_i}) \big| P(y|\mathbf{x}) >= \tau \text{ and } \mathbf{x_i}^\nu \in D_{current}^\nu\}$;

$c_p \leftarrow |\tilde{D}|$

$\mathbf{s} \leftarrow$ Computing_Sarcasm($\tilde{D}$);

$D_s \leftarrow \phi$;

$count \leftarrow 0$

for $(\mathbf{x}, y) \in \tilde{D}$ **do**

if $\mathbf{s}[count] = 1$ **then**

$\tilde{D} \leftarrow \tilde{D} \backslash \{(\mathbf{x}, y)\}$;

$y \leftarrow \neg y$;

$D_s \leftarrow D_s \bigcup \{(\mathbf{x}, y)\}$;

end

$count \leftarrow count + 1$;

end

$\tilde{D} \leftarrow \tilde{D} \bigcup D_s$;

$D^{new} \leftarrow D_{TR}^L \bigcup \tilde{D}$;

$model \leftarrow \arg\min_{\gamma} \{ \sum_{i=1}^{|D^{new}|} l(y_i^*, \hat{g}(\mathbf{x_i}^{new}, \gamma)) \}$;

$(g, \gamma^*) \leftarrow model$

$D_{TR}^L \leftarrow D^{new}$;

$M \leftarrow |D_{TR}^L|$;

$\tilde{D_X} \leftarrow \{\mathbf{x} | (\mathbf{x}, y) \in \tilde{D}\}$;

$D_{current}^\nu \leftarrow D^\nu \backslash \tilde{D_X}$;

end

$result\_object \leftarrow \{g, \gamma^*\}$;

return *result_object*

models and existing state-of-the art works of [18] and [37], in the literature by adding the predicted pseudo-labels of the proposed semi-supervised model of SMTSD to the original labeled data. The details of the datasets, mentioning the number of tweets, the number of unlabeled tweets, and the exact time periods during which the datasets are collected, are presented in Table 5.3.

The distributions of aggregates of various features among the collected features are presented in various tables. In Table 5.4, the number of labeled tweets and the class label distribution of the two classes are presented. The dataset is balanced, as the number of data points in both positive and negative classes of stress is nearly equal. In table 5.5, the polarity-based distributions of words, emojis, and hashtags are presented. The distribution of the value of sarcasm in the tweets, predicted using algorithm 7, is noted in table 5.6.

Table 5.3: The Details of the Datasets

Dataset	**#tweets**	**#users**	**#Unlabeled tweets**
D1	7,289	1,732	5,193
D2	16,532	5,119	11,779
D3	4,062	1,888	2,894
D4	7,209	2,558	5,136

Table 5.4: Distribution of the class-labels in the Datasets

Dataset	**#tweets**	**#labeled tweets**	**#positive labeled tweets**	**#negative labeled tweets**
D1	7,289	2,096	1,048	1,048
D2	16,532	4,753	2,376	2,377
D3	4,062	1,168	584	584
D4	7,209	2,073	1,036	1,037

Table 5.5: The distribution of positive and negative polarity among words, emojis and hashtags

Dataset	**(#positive words, #negative words)**	**(#positive emojis, #negative emojis)**	**(#positive hastags, #negative hashtags)**
D1	(4,466 , 2,689)	(366 , 251)	(12 , 30)
D2	(10,773 , 5,334)	(1,017 , 440)	(55 , 127)
D3	(2,687 , 1,246)	(545 , 272)	(14 , 7)
D4	(4,625 , 2,031)	(416 , 167)	(22 , 74)

Table 5.6: The sarcasm distribution Details of the Datasets

Dataset	**#tweets**	**#tweets with sarcasm value=1**	**#tweets with sarcasm value=0**
D1	7,289	397	6,892
D2	16,532	1,198	15,334
D3	4,062	504	3,558
D4	7,209	955	6,254

5.3.2 Baseline supervised methods for comparison

Tweet-level stress detection techniques are applied on a variety of popular machine learning models like logistic regression, support vector machines, naïve bayes and random forests. Of them, logistic regression and support vector machines are considered to be high performers for tweet-level stress detection [19, 94]. All the basic machine learning models used here are discussed in section 4.3.2 of chapter 4. However, there are other deep learning-based models like Deep Neural Network with Cross-Auto Encoder (DNN-CAE) and Bidirectional long short term memory (Bi-LSTM) which are considered additionally in this chapter. In the baseline models, all the classical machine learning models except the state-of-the-art Bi-LSTM use features similar to the previous work [45] and have sarcasm as one of the features. The Bi-LSTM model, on the other hand, relies on word embedding techniques rather than hand-crafted

features [37].

1. **Deep Neural Network with Cross-Auto Encoder (DNN-CAE)**: A Deep Neural Network with Cross Auto Encoder (DNN-CAE) is used in [18] to predict tweet-level stress. It takes cross-media data from tweets, consisting of tweets' text, social, and visual contents, and then finds a middle-level representation using CAE. Later, DNN is used for classification. It performed better than the classic ML algorithms, when data from all three different media was considered.

2. **Bidirectional long short term memory (Bi-LSTM) [37] :** This is a state-of-the-art work in this domain. It develops a sequential model of bidirectional long short term memory (Bi-LSTM) to identify the psychological stress disclosed by minority group. The input text data, after being preprocessed into tokens, is fed into the embedding layer, which outputs the word vectors for each word. The dimension of the output vectors from the embedding layer is 300. The embedding layer output is then sent to the Bi-LSTM layer. The Bi-LSTM architecture expands the capabilities of sequential models like LSTM. It achieves this by considering the context of or influence on a word from neighboring words in both directions. The context in Bi-LSTM captures the influence on a word from previous text sequences and future text sequences as well. The output of the Bi-LSTM layer is fed into a dropout or regularization layer. The final feature vector is then fed into a fully connected neural network for classification. Bi-LSTM performed better at capturing true positives and true negatives.

In this chapter, all the baseline models considered above are implemented on all the four datasets. In the experiments, the supervised machine learning models are implemented on the combined data of labeled and predicted pseudo-labeled data. The pseudo-labels are generated using two ways- first, from both basic self-training approach and second, from the proposed SMTSD approach. All the baseline models presented in this section are implemented using the combined data formed using the two methods mentioned above. The comparative analysis includes the performance

comparison among baseline models, including the state-of-the-art Bi-LSTM model and the proposed SMTSD method. The metrics used for comparison are Accuracy and F1-Score, which are defined in section 1.3.

Table 5.7: Accuracy and F1-Score recorded for models implemented in this chapter. The results of baseline models correspond to the sampling of combined data considered being 1.0

Model/Dataset		D1		D2		D3		D4	
		Accuracy (%)	**F1-Score**	**Accuracy (%)**	**F1-Score**	**Accuracy (%)**	**F1-Score**	**Accuracy (%)**	**F1-Score**
Proposed Semi-supervised methods	Basic Self-training	83.01	0.905	82.01	0.911	83.2	0.880	81.6	0.860
	Proposed SMTSD	**87.16**	**0.929**	**86.77**	**0.918**	**85.8**	**0.901**	**85.2**	**0.919**
Baseline models trained with the combined data of labeled and pseudo-labeled data from the proposed SMTSD model	LR	84.4	0.915	78.95	0.87	78.59	0.87	78.92	0.87
	SVM	82.3	0.902	75.74	0.86	78.9	0.88	76.30	0.865
	RF	84.4	0.914	77.0	0.87	78.8	0.88	76.3	0.86
	NB	84.4	0.916	75.7	0.86	75.8	0.86	77.5	0.86
	DNN-CAE [18]	72.43	0.735	71.60	0.710	71.8	0.731	70.8	0.733
	Bi-LSTM [37]	84.41	0.915	82.01	0.901	82.4	0.891	80.6	0.901
Baseline models trained with the combined data of labeled and pseudo labeled data from the basic Self-training model	SVM	80.3	0.792	75.2	0.79	76.9	0.8	74.9	0.799
	RF	81.4	0.884	76.4	0.78	76.8	0.791	75.3	0.771
	NB	81.1	0.798	73.7	0.78	73.9	0.75	73.2	0.79
	DNN-CAE [18]	71.27	0.72	70.6	0.70	70.3	0.709	68.8	0.703
	Bi-LSTM [37]	82.6	0.809	76.55	0.773	79.9	0.801	77.07	0.789

5.4 Results

This section presents the results of the experiments conducted as part of this chapter and the analysis of their performance.

The performances recorded for every experiment are presented in Table 5.7. All the results obtained are tested for statistical significance using the *one-sample Wilcoxon signed rank test* [91]. It is found that the null hypothesis is rejected with $p \leq 0.05$ and hence the results are considered statistically significant.

The experiments are initially implemented for the proposed SMTSD and basic self-training. Here, a simple implementation of generic self-training without the us-

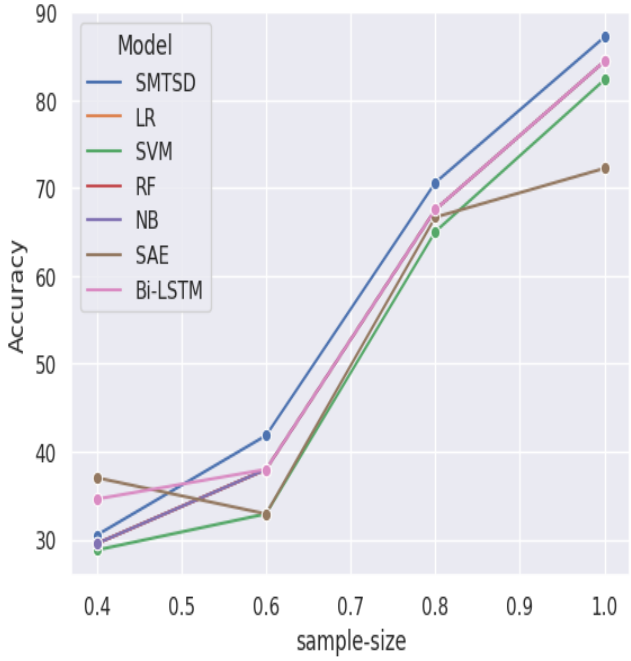

Figure 5.2: The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D1.

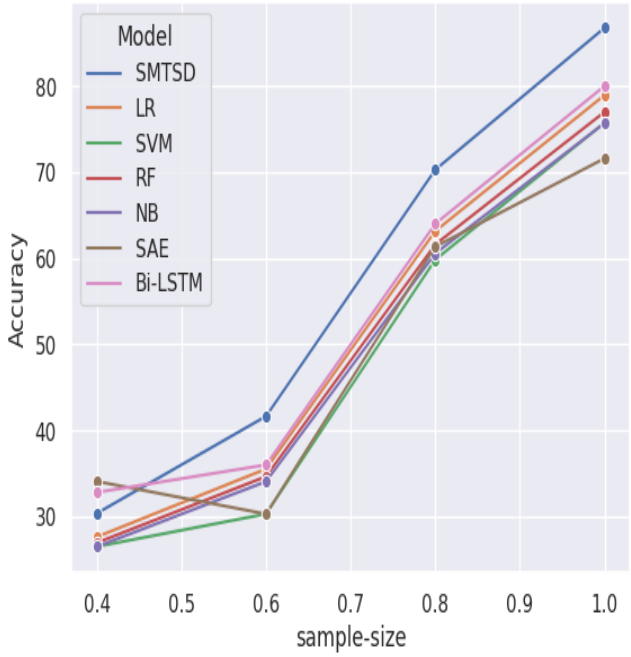

Figure 5.3: The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D2.

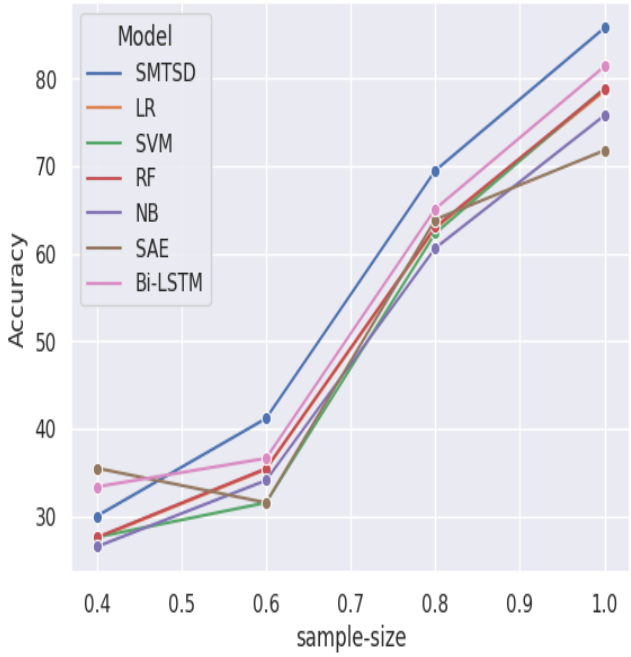

Figure 5.4: The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D3.

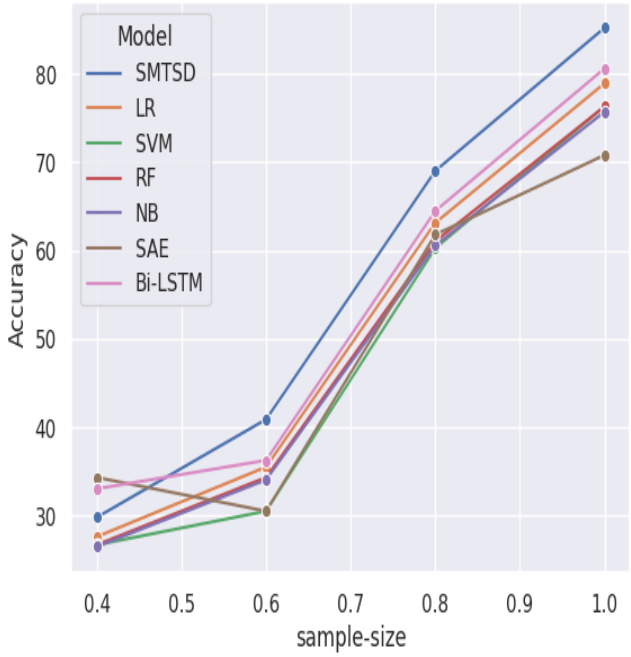

Figure 5.5: The variation in accuracy of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D4.

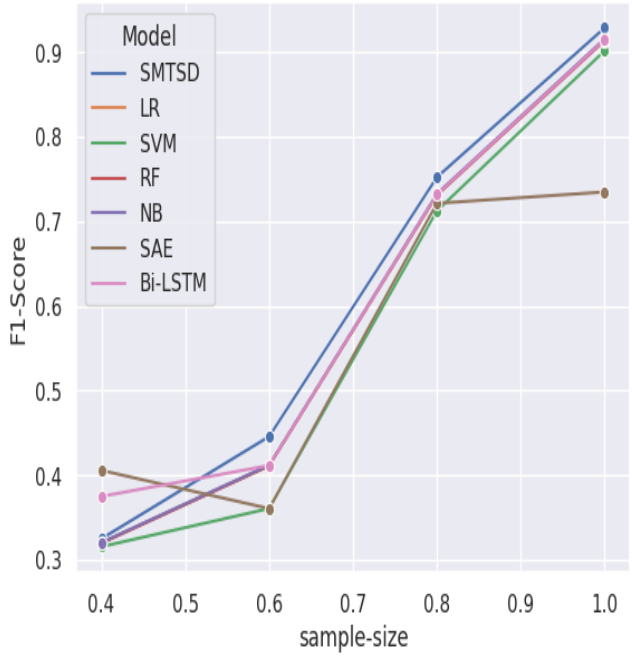

Figure 5.6: The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D1.

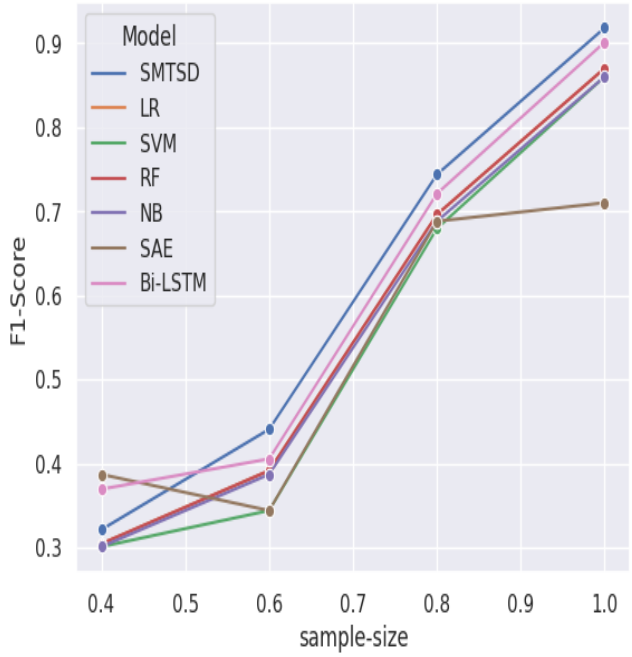

Figure 5.7: The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D2.

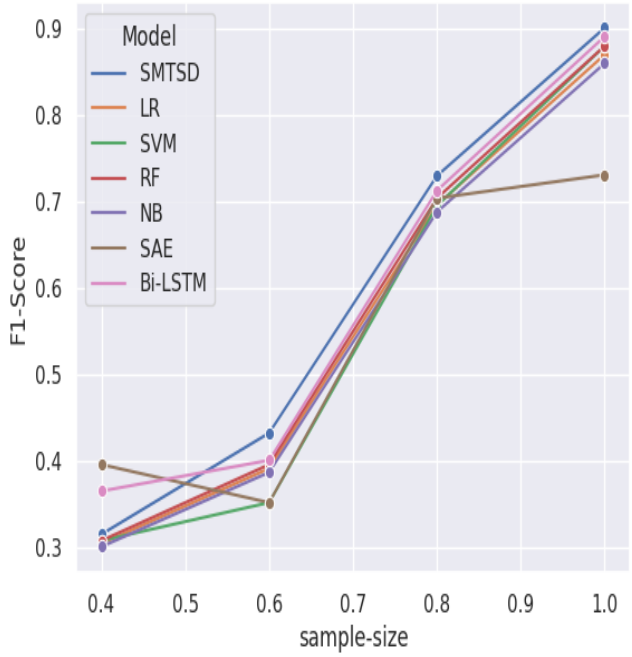

Figure 5.8: The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D3.

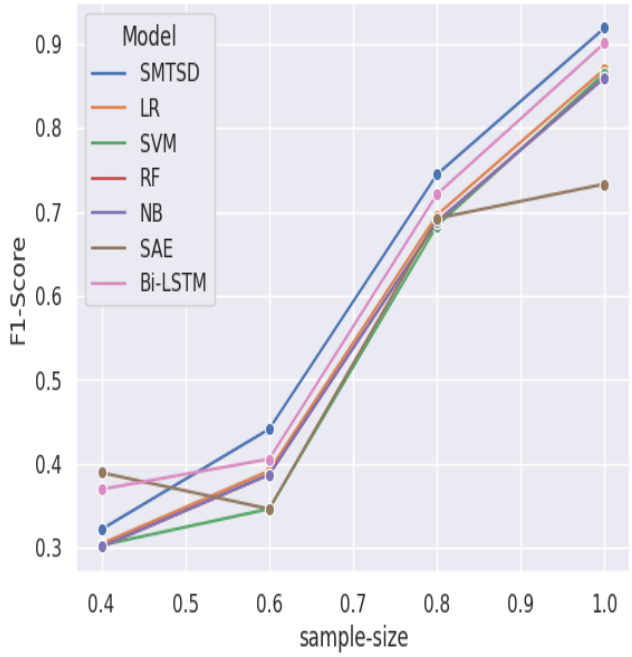

Figure 5.9: The variation in F1-Scores of the proposed SMTSD model and other supervised baseline models for different sample sizes of combined data containing labeled and pseudo-labeled data predicted by SMTSD on Dataset D4.

age of information related to sarcasm for predicting pseudo-labels, is termed "basic self-training". The predicted pseudo-labeled data is then combined with the labeled data. The combined sets of labeled and unlabeled data are formed correspondingly for all four datasets. The baseline supervised models are then implemented on the combined datasets. The combined datasets are formed in two ways. First, by adding the predicted pseudo-labeled data from the proposed SMTSD method. Second, by adding the pseudo-labels from the basic self-training approach. The combined data, formed by pseudo-labels from the proposed SMTSD approach, is trained on the supervised algorithms, including logistic regression. On the other hand, the combined data formed by the pseudo-labels from the basic self-training approach is trained on supervised algorithms other than logistic regression. Because the base classifier is the same in the basic self-training approach and supervised logistic regression, the sarcasm is used as a feature in the supervised models. Table 5.7 presents the performances in terms of accuracy and F1-Score for the proposed SMTSD and baseline self-training models on all four datasets. Also, the table records the performance of the baseline supervised models when trained with combined data prepared from two semi-supervised models on all four datasets.

From Table 5.7, it is observed that the proposed SMTSD model outperforms the basic self-training model and all other baseline supervised ML models, including the existing works on DNN-CAE and Bi-LSTM models in the literature. On the other hand, the basic self-training-based approach outperforms other supervised models, except the Bi-LSTM, when implemented on datasets D2, D3, and D4. But on dataset D1, the basic self-training approach records low performance when compared to Bi-LSTM and basic ML models, except for SVM. The state-of-the-art model, Bi-LSTM, gives the second-best performance in both accuracy and F1-Score on all the datasets after the proposed SMTSD approach.

SMTSD records the highest performance on dataset D1, where it records the maximum improvement in accuracy and F1-Score by 4.86% and 0.027 points compared to the performance of SVM. While, in the case of datasets D2, D3, and D4, the proposed SMTSD records the highest improvement in accuracy over the baseline ML algorithm

of naïve bayes. In the case of F1-Score improvement recorded by SMTSD, the highest is observed in dataset D2 when compared to both naïve bayes and SVM. While, in the case of datasets D3 and D4, the highest improvement in F1-Scores is recorded over naïve bayes. Moreover, as shown in Table 5.7, proposed model of SMTSD has the largest improvement in performance scores compared to the performance achieved by the supervised model of DNN-CAE. Also, when trained on the combined data formed from the pseudo-labeled data predicted by the proposed SMTSD, the proposed model of SMTSD records the highest improvement in accuracy by 4.76% over the state-of-the-art Bi-LSTM model when implemented on dataset D2. In the same case, it is observed that the proposed model achieves the highest improvement in the F1-Score by 0.1 points when implemented on dataset D3.

Furthermore, it is noted that the proposed SMTSD model outperforms the basic semi-supervised self-training model on all datasets. This provides empirical validation for the use of information related to sarcasm in the proposed model. The maximum improvement in accuracy of 4.4% is recorded by the proposed SMTSD when implemented with dataset D2. Also, the maximum increase in F1-Score observed is 0.07 points when implemented using the dataset D2. This confirms that the use of sarcasm was effective in maximizing the performance of stress detection. Also, it is observed in table 5.7 that the performances of the baseline algorithms decrease when the baseline models are trained with combined labeled and pseudo-labeled data predicted from the basic self-training, when compared to the performances recorded with combined data formed by the pseudo-labels predicted from the proposed SMTSD model. It is also noted that the basic self-training model, implemented with logistic regression, records better performance than the other baseline and state-of-the-art techniques when trained with the combined labeled and pseudo-labeled data obtained from the basic self-training method.

Table 5.8: The number of pseudo-labels included in the final combined data

Dataset	**#pseudo-labeled data points included in final dataset**
D1	636
D2	2,709
D3	390
D4	1,517

In addition to the above, the combined data, comprising labeled and pseudo-labeled data from the proposed SMTSD model, is sampled for four different sizes: 0.4, 0.6, 0.8, and 1.0. Here, each sample size reflects the proportion of combined data. And the baseline supervised models are implemented on each of these sampled datasets. Figure 5.2 depicts the accuracy recorded by all the supervised models considered as part of this experiment, along with the proposed SMTSD model, at varying levels of sampling size of the combined data for dataset D1. Similarly, 5.3, 5.4 and 5.5 depict the accuracy recorded by all the supervised models considered as part of this experiment, along with the proposed SMTSD model, at varying levels of sampling size of the combined data for the datasets D2, D3, and D4, respectively. And the F1-Scores of all the supervised models and the proposed SMTSD model with varying sample sizes of the combined pseudo-labeled data, for dataset D1, are shown in Figure 5.6. While figures 5.7, 5.8, and 5.9 present the F1-Scores of all the supervised models and the proposed SMTSD model with varying sample sizes of the combined pseudo-labeled data, for the datasets D2, D3, and D4, respectively. From figures 5.2, 5.3, 5.4, 5.5 and figures 5.6, 5.7, 5.8,5.9, it is observed that, with the increase in sample size taken from the predicted labels of the proposed SMTSD, the accuracy and F1-Score of all the models also grow steadily. This confirms that the pseudo-labels predicted by the proposed semi-supervised approach are valid. Also, even in the scenario of considering the entire predicted labeled dataset ($sample\_size = 1.0$), all of these models record lower performance when compared to the proposed SMTSD model. Moreover, from figures 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, and 5.9, it is observed

that the proposed SMTSD model outperforms the state-of-the-art Bi-LSTM model. Also, it depicts the gradual increase in the performances of the Bi-LSTM classifier with the increase in sample size from 0.6. Table 5.8 gives the number of pseudo-labeled data points included in the final combined data when the proposed SMTSD was implemented with respect to all four datasets.

5.5 Discussion

In this section, the discussion of the contribution of the proposed SMTSD model and results are presented. In addition, the limitations of the proposed method are also presented.

The tweet-level stress detection problem is handled only when there is availability of labeled data [3,18,19,21–26]. Since it is not possible to get huge amounts of labeled data in a real world scenario, a semi-supervised approach is devised to develop a tweet-level stress detection model. In existing works on the tweet-level, either using traditional machine learning classifiers or using deep learning models such as DNN-CAE [18] and Bi-LSTM [37], only supervised mechanisms were proposed. In this chapter, the semi-supervised models developed using self-training mechanisms handle the problem of the scarcity of labeled data and also improve the performance.

Furthermore, the proposed SMTSD method in this chapter employs the concept of sarcasm in self-training for better prediction of pseudo-labels. This enabled better utilization of the text content of the tweet for improving tweet-level stress detection performance. The concept of utilizing information from text-content through sarcasm is not addressed in the existing literature. The proposed SMTSD model is developed to utilize the information of sarcasm value in predicting the pseudo-labels during the process of self-training. Hence, with regards to the concept of utilizing the information of sarcasm in the proposed model, there are three cases –

- Utilizing sarcasm within the semi-supervised self-training framework to compute pseudo-labels (SMTSD with logistic regression as the base learner).

Table 5.9: Role of Sarcasm- Results

Model	Accuracy				F1-Score			
	D1	D2	D3	D4	D1	D2	D3	D4
Proposed SMTSD (Utilizing sarcasm to predict pseudo labels in self-training)	87.16	86.77	85.8	85.2	0.929	0.918	0.901	0.919
Basic self-training with sarcasm as an attribute	83.01	82.01	83.2	81.6	0.905	0.911	0.880	0.860
Basic self-training without sarcasm as an attribute	80.32	79.10	79.9	78.7	0.83	0.79	0.765	0.78

- Utilizing sarcasm only as an attribute in data without using it in computing the pseudo-labels during the process of self-training.
- Not using sarcasm even as an attribute.

From the table 5.9, it can be gauged that with the usage of sarcasm, the performance of the tweet-level stress detection increases. There are two ways in which sarcasm could be used in the models. First, as one of the features, and second, as the variable in deciding the pseudo-labels during self-training. In the first case, when basic self-training is implemented with sarcasm as an attribute, there is a significant increase in the performance of tweet-level stress detection when compared to the basic self-training model implemented without sarcasm as an attribute. It is observed that there is an improvement of at least 2.69% in accuracy and 0.075 points in the F1-Score. In the second case, when sarcasm is utilized to decide pseudo-labels as part of self-training, as in the proposed SMTSD model, there is a steady improvement in performance in both accuracy and F1-Score compared to the basic self-training implemented with sarcasm as one of the features. It is observed that the proposed SMTSD method has an improvement over basic self-training with sarcasm as an attribute, by at least 2.6% in accuracy and 0.007 points in F1-Score. Moreover, when the second case is implemented without using sarcasm in deciding pseudo-labels during self-training, it reduces to the case of basic self-training without sarcasm as an attribute. The proposed SMTSD, when compared to basic self-training without sarcasm as an attribute, records an improvement of at least 5.9% in accuracy and 0.099 points in F1-Score. Hence, from the study of the impact of sarcasm in the proposed semi-supervised approach for tweet-level stress detection, it is intuitive that the pro-

posed SMTSD model, which is a combination of a semi-supervised approach with the usage of sarcasm in self-training, outperforms other baseline models.

5.5.1 Limitations of the proposed SMTSD model

The limitations of the proposed SMTSD model are given as follows:

1. The proposed model doesn't perform well when the tweets lack text-content in them.
2. The proposed model may have different performance for the tweets which have sarcasm embedded without the usage of emojis.

5.6 Summary

In this chapter, a semi-supervised solution is proposed for tweet-level stress detection, as there is a scarcity of labeled data in the real-world. As a solution to this problem, a self-training approach based on logistic regression, Self-training Method for Tweet-level Stress Detection (SMTSD), is proposed. Unlike a general self-training approach, this method relies on utilizing the information of sarcasm in predicting the pseudo-labeled data. For empirical verification, four different datasets are collected using the Twitter API, tweepy, with only a small portion of them being labeled manually. It is observed that, on all four datasets, the proposed SMTSD model outperforms the baseline supervised ML models as well as the state-of-the-art methods like Bi-LSTM and other techniques like DNN-CAE method in the existing literature. The expanded data with predicted pseudo-labels is then sampled with four different sample sizes, and the supervised models considered in this chapter are trained on all of these samples. It is noted that the performance of the supervised models increased as the sample size of the combined data increased. All results obtained are statistically significant.

Chapter 6

Multi-task Learning Approach for Tweet-level Stress Detection Using Deep Learning

As discussed in previous chapters, sarcasm, as an attribute helps in improving the performance of the stress detection at text-level. From chapters 3 and 4, it is found that sarcasm helped in detecting stress. And in chapter 5, sarcasm helped in deciding the pseudo-label. It is intuitive that sarcasm is closely related to stress and could help in detecting the stress. Hence, there is a scope to explore the multi-task learning approach for detecting stress, by using sarcasm as an auxiliary task. In this chapter, we propose a multi-task deep learning-based framework as a solution to tweet-level stress detection.

Organization of the chapter:

The chapter is organized as follows. Section 6.1 discusses the problem formulation, including the formal notations used in this chapter. The proposed MATSD model and its training are discussed in section 6.2. The experimental setup and the data collected are presented in section 6.3. The results and the findings of the chapter are presented in section 6.4. The discussion of this chapter is presented in section 6.5. Finally, section 6.6 gives the summary of the chapter.

6.1 Problem Formulation

In this chapter, a multi-task deep learning-based approach of MATSD was proposed to detect the psychological stress in a given tweet, along with the auxiliary task of sarcasm. The notations used in this chapter are presented in table 6.1.

6.1.1 MATSD: Problem Statement

If $D = (\mathbf{x}_i, y_i^s, y_i^a)_{i=1}^N$ is the dataset of tweets containing the labels for both stress (y^s) and sarcasm (y^a), then the problem is to learn a function that approximates the mapping from tweet ($\mathbf{x}$) to labels of stress (y^s) and sarcasm (y^a). For greater clarity, the problem of the proposed MATSD is - If $D = (\mathbf{x}_i, y_i^s, y_i^a)_{i=1}^N$ is the dataset of tweets containing the labels for both stress and sarcasm, then the problem is to learn functions $h_s : D \longrightarrow C^s$ and $h_a : D \longrightarrow C^a$ jointly for the tasks of stress and sarcasm respectively. Here, C^s and C^a represent unique class labels in the classifcation tasks of stress and sarcasm, respectively (here, $C^s = \{0, 1\}$ and $C^a = \{0, 1\}$ as both are binary classifiers). For any unlabeled tweet $\mathbf{z}$, the learned functions h_s, h_a are used to predict stress and sarcasm labels respectively as $p^s = h_s(\mathbf{z})$ and $p^a = h_a(\mathbf{z})$.

Table 6.1: Notations utilized in MATSD

Notation	**Definition**
U	Set of users
$\mathbf{x_i}$	i^{th} tweet in dataset D, of user $w, \exists w \in U$,
N	The total size of the dataset D, $N = \|D\|$.
D^{TR}	The training-set part of the dataset D
D^{TS}	The validation-set part of the dataset D
$y_i^s \in \{0,1\}$	dependent variable or class label denoting the state of stress for some the tweet $\mathbf{x}_i \in D^{TR}, \forall i \in \{1,2,..\|D^{TR}\|\}$
$y_i^a \in \{0,1\}$	dependent variable or class label denoting the state of sarcasm for some the tweet $\mathbf{x}_i \in D^{TR}, \forall i \in \{1,2,..\|D^{TR}\|\}$
Y_s^{TR}	set of stress labels for training dataset, D^{TR}
Y_a^{TR}	set of sarcasm labels for training dataset, D^{TR}
$p^s \in \{0,1\}$	predicted class label denoting the state of stress for some the unknown tweet $\mathbf{z} \notin D^{TR}$
$p^a \in \{0,1\}$	predicted class label denoting the state of sarcasm for some the unknown tweet $\mathbf{z} \notin D^{TR}$
w	any word in in a tweet $\mathbf{x}$ in training dataset D^{TR}
E_w	embedding vector for the word w in tweet $\mathbf{x} \in D^{TR}$
E_m	embedding matrix for the tweet $\mathbf{x} \in D^{TR}$. It is vector of embedding vectors for every w in tweet $\mathbf{x} \in D^{TR}$
E_D	set of embedding matrices, where each matrix corresponds to a tweet $\mathbf{x} \in D^{TR}$
L_s	Training Loss function for the task of stress for all tweets $\mathbf{x} \in D^{TR}$
L_a	Training Loss function for the task of sarcasm for all tweets $\mathbf{x} \in D^{TR}$
λ_1	weight for Training Loss function for the task of stress
λ_2	weight for Training Loss function for the task of sarcasm
L_T	Combined or total Training Loss function of the multi-task network for $\mathbf{x} \in D^{TR}$, $L_T = \lambda_1 L_s + \lambda_2 L_a$
∇L_T	Gradient of the Combined Training Loss, L_T
α	parameters learned for the total multi-task neural network of the proposed MATSD model. Total loss L_T
β	parameters learned that are present only in the subnet for prediction of stress
γ	parameters learned that are present only in the subnet for prediction of sarcasm
θ	parameters learned that are common for both the subnets predicting stress and sarcasm
$h_{\beta,\theta}(\mathbf{z})$	The predicted label of stress task classifier for some tweet $\mathbf{z}$
$h_{\gamma,\theta}(\mathbf{z})$	The predicted label of sarcasm task classifier for some tweet $\mathbf{z}$

6.2 Methodology

This section presents a detailed description of the proposed MATSD (Multi-task Approach for Tweet-level stress detection) model and its training procedure.

6.2.1 MATSD: Framework

The framework of the proposed model is depicted in the figure 6.1. The tweets collected from the users are preprocessed during the initial stages. The preprocessed tweets are then vectorized using standard embedding techniques like Glove. After the embedding phase, the embedded matrices of tweets are then passed as input into networks corresponding to two tasks of sarcasm and stress. After convolution and max-pooling stages in the subnet of sarcasm, the resultant feature vector is passed through the sigmoid gate, which is represented in figure 6.1 as σ. Later it is mapped with the LSTM final layer on the stress subnet. Finally, both tasks have their own task-specific fully connected, and Softmax layers. The output of the Softmax layers forms the predictions for the tasks. The maxpooled feature vectors are shared by subnets of both tasks.

6.2.2 MATSD : Architecture

The proposed MATSD is a multi-task learning solution based on stress detection using information from sarcasm. The architecture consists of two subnets. One of them is for the primary task of learning the stress and the other is for the auxiliary task of learning the sarcasm. As in general multi-task approaches, few layers are shared between both tasks. There are many components in the proposed MATSD architecture. The details of all the components are described in this section.

6.2.2.1 Data Collection, labeling, and preprocessing

Twitter's Tweepy API is used to collect tweets during the periods of 2021 March and 2021 December to 2022 January and February. The collected tweets are preprocessed to remove noise and other unreadable characters, etc. The details are described in

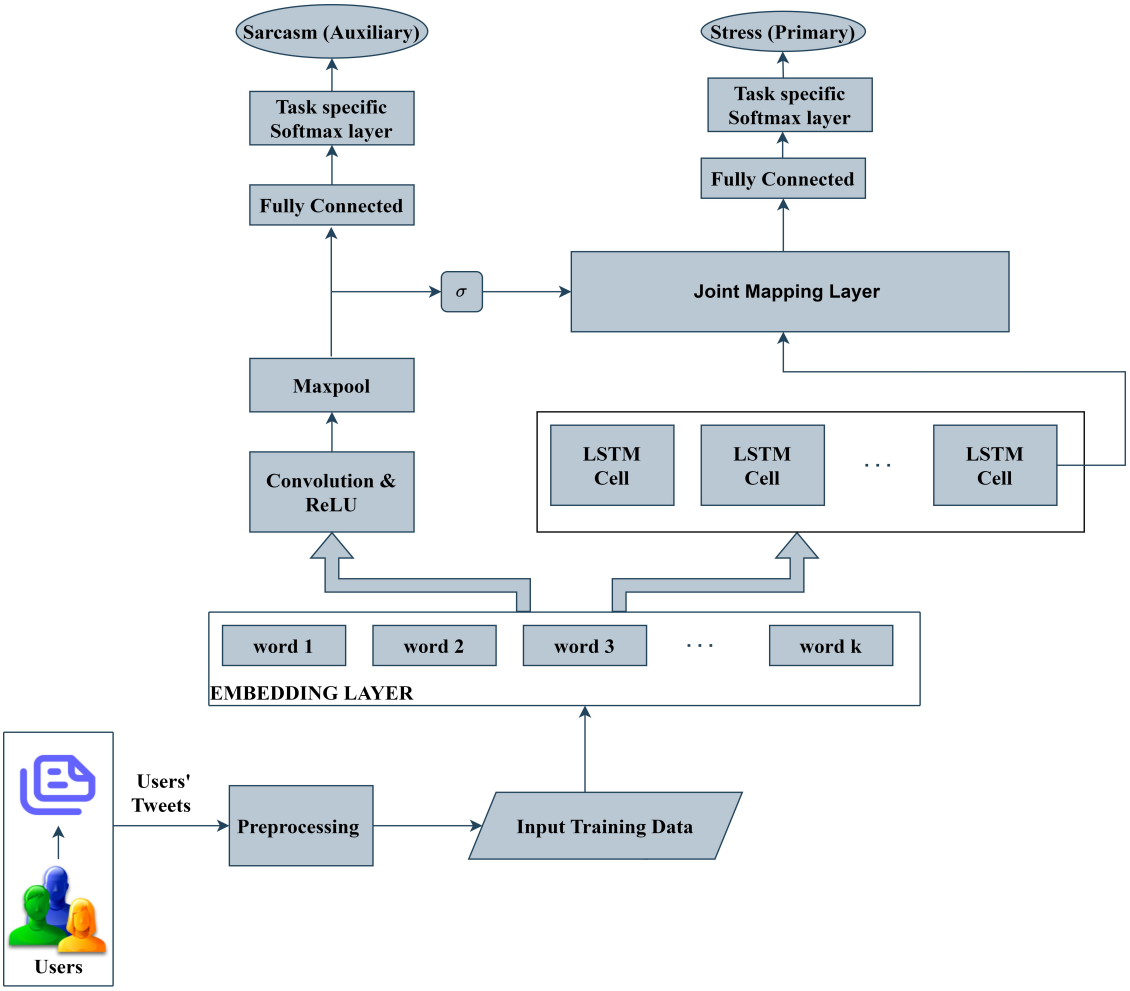

Figure 6.1: Framework of the proposed MATSD model

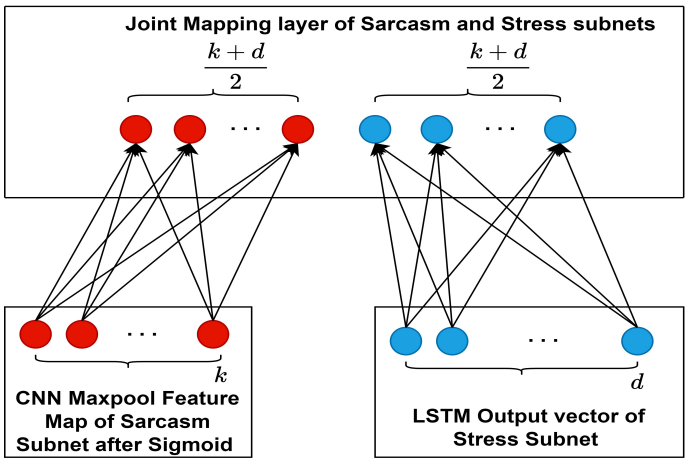

Figure 6.2: Illustration of Joint Map Layer

section 6.3. The datasets are same as those collected in 4. For the extracted data using the queries "I feel stressed", "I feel relaxed", the labels for the primary task of stress are assigned as 1 and 0 respectively. Later, the sarcasm value computed in chapter 4, for all the datasets collected. In this chapter, we use this attribute as the class-label for the sarcasm task.

Preprocessing of the datasets in this chapter is similar to the preprocessing procedure that was performed in chapter 5. In the case of data preprocessing, the tweets with only images or GIFs are also removed as the focus of the model was more on textual information. The preprocessed tweets with the class-labels corresponding to the two tasks are then passed as input to the word embedding layer.

6.2.2.2 Word Embedding

In this layer, the GloVe Embedding technique is employed to build word vectors. Global vector for word representation (GloVe) builds the word vectors using an unsupervised learning algorithm trained on large global corpus [95]. This method is proven to produce vectors that better capture the syntax and semantics of the vocabulary space.

6.2.2.3 Sarcasm subnet - Convolution Layer

In the sarcasm subnet, which is the network pertaining to the task of detecting sarcasm, the convolutional neural network architecture is employed. The traditional CNN layers for text classification are known to use 1D convolution filters of sizes 2 and 3 [47]. In this chapter, the filters of sizes 2 and 3 are employed. A total of 128 filters are used. Figure 6.1 illustrates the convolution. The resultant feature maps are given as input to the activation function of the rectified linear unit (ReLU) function.

6.2.2.4 Sarcasm subnet - ReLU and Maxpool

The ReLU function stands for the rectified linear unit, which provides non-linearity and helps in mitigating the problem of vanishing gradient [47]. The ReLU function is 0 for all negative inputs. However, for the positive valued inputs, the value of the

ReLU function is a maximum of zero and the input value. The output of the ReLU function is sent for aggregation in the max-pooling layer. In the max pooling stage, the aggregate values of the feature maps are considered. The resultant is a hidden or latent representation of input.

6.2.2.5 Sarcasm subnet - Sigmoid

The resultant of the maxpool layer is sent to the sigmoid function to get a resultant vector which is then sent to the stress subnet through a joint map layer. Here the sigmoid gate takes the vector generated by the maxpool layer to produce a resultant vector, where each element is the output of the sigmoid function applied to the corresponding element in the vector from the maxpool layer. The size of this resultant vector is considered to be k.

6.2.2.6 Joint Map Layer

In this the feature map of the maxpool layer on the sarcasm subnet and the LSTM output signal on the stress subnet are mapped to a new layer such that the LSTM layer is mapped as fully connected to one half chunk of the resultant vector and the maxpool layer from the sarcasm CNN subnet is mapped to another half chunk of the resultant vector. Each of these chunks of size $\frac{k+d}{2}$. This is illustrated in figure 6.2.

6.2.2.7 Sarcasm subnet - Task specific fully-connected (FC) layer and Softmax

The flattened maxpool feature vector is sent to a fully connected layer that is specific to each task. This performs hidden layer mapping (i.e., mapping to other dimension space), typical to general shallow artificial neural networks.

The output of the FC layer is sent to the sigmoid or Softmax layer to produce probabilities of prediction ($h_{\gamma,\theta}(\mathbf{x})$) for each class of sarcasm for the given input tweet. Also, the error or loss in the prediction is computed at this layer. The total loss(L_a) for all the training data is computed using cross entropy and it is specific to the

subnet of sarcasm [47]. It is given as follows:

$$L_a = \sum_{i=1}^{N} \left\{ y_i^a \log(h_{\gamma,\theta}(\mathbf{x}_i) + (1 - y_i^a)(\log(1 - h_{\gamma,\theta}(\mathbf{x}_i))) \right\}$$

6.2.2.8 Stress subnet - LSTM Layer

The primary task of the proposed MATSD method is to detect stress. The subnet for detecting stress is composed of gated recurrent neural networks called long short term memory (LSTM) [47]. The input embeddings are passed to the first cell of the LSTM layer, along with the initial cell state. At every block, the input to the block consists of three vectors- the previous cell state, the previous hidden state, and the current input word vector. Every block or cell of the LSTM layer produces an output vector. The output of the final block is considered as the feature representation of the input tweet. The architecture of the LSTM network used in the proposed MATSD approach learns a feature vector of size d. This feature vector of the final LSTM cell in the layer is sent to the joint map layer. The number of LSTM cells used in this chapter is 64.

6.2.2.9 Stress subnet - Task specific fully-connected(FC) layer and Softmax

The task-specific fully connected layer of the stress subnet takes the input as the resultant vector of the joint map layer or connecting layer between two subnets. The fully connected layer maps the feature vector to other spaces.

The output of the FC layer is sent to the sigmoid or Softmax layer to produce probabilities of prediction ($h_{\beta,\theta}(\mathbf{x})$) for each class of stress for the given input tweet. Also, the error or loss in the prediction is computed at this layer. The total loss(L_s) for all the training data is computed using cross entropy and it is specific to the subnet of sarcasm [47]. It is given as follows:

$$L_s = \sum_{i=1}^{N} \left\{ y_i^s \log(h_{\beta,\theta}(\mathbf{x}_i) + (1 - y_i^s)(\log(1 - h_{\beta,\theta}(\mathbf{x}_i))) \right\}$$

$$L_s = \sum_{i=1}^{N} \left\{ y_i^s \log(h_{\beta,\theta}(\mathbf{x}_i) + (1 - y_i^s)(\log(1 - h_{\beta,\theta}(\mathbf{x}_i))) \right\}$$

This Softmax layer of the stress subnet is the final layer of the stress subnet takes the output of the previous fully connected layer and inputs it to the Softmax or sigmoid layer to produce a probability distribution of predictions of stress class for the given input tweet.

The predicted output at this layer is used to compute the error or loss which is then propagated back to the previous layers and used to compute the gradients. These errors will be propagated to both subnets of both layers from the point-wise multiplication layer.

6.2.3 Multi task context

The subnet for the auxiliary task is based on CNN while the subnet for the primary task is based on LSTM. Both tasks have common embeddings as input. Also, the vector output of the max pool layer in the sarcasm subnet is used as a shared layer in both tasks.

A multi-task framework is a generalization of regularization, where each task acts as a regularizer to other tasks making all the tasks learn from shared parameters. This helps in improving the performance of all tasks through the sharing of knowledge of tasks. This is useful, especially in the case of data scarcity. Moreover, the multi-task learning should be done carefully among the related tasks, otherwise, the shared parameters of both the tasks can mislead each other decreasing the performance of all the tasks.

Hence, in this chapter, we propose a gate-based sharing where only the important information of a task is shared with the other task via sigmoid gate (σ) and mapping with the joint map layer, instead of direct concatenation of the vectors.

6.2.4 Training

The proposed MATSD is a multi-task learning approach wherein multiple tasks using a set of shared parameters. This saves a lot of time and space resources when compared to training multiple tasks individually. In essence, by sharing of parameters, for each task, all the other tasks act as regularizers. Thus apart from sharing of information, the tasks also constrain each other, paving the way for better prediction models.

To this end, the multiple tasks represented by each subnet in the MATSD are trained together. The proposed MATSD architecture is based on soft-parameter sharing as only a few parameters are shared between the two tasks and the sharing is achieved through the transfer of information through the sigmoid gate (σ) and joint map layer.

Training a multi-task learning architecture like MATSD involves joint training of all the models (tasks) that are part of the multi-task framework. In order to achieve this, the loss of the multi-task model needs to be minimized. However, the loss of the model is characterized as the sum of losses of constituent tasks. Hence, in the MATSD, the total loss of the model is given as the weighted sum of the losses of tasks of sarcasm and stress. This is given by the following equation:

$$L_T(\alpha) \leftarrow \lambda_1 L_s + \lambda_2 L_a \tag{6.1}$$

Where λ_1 and λ_2 are the weights for the losses of tasks. While L_s and L_a are losses corresponding to stress and sarcasm tasks, respectively. They are defined based on the cross entropy loss [47] as follows:

$$L_s = \sum_{i=1}^{N} \left\{ y_i^s \log(h_{\beta,\theta}(\mathbf{x}_i)) + (1 - y_i^s)(\log(1 - h_{\beta,\theta}(\mathbf{x}_i))) \right\}$$

$$L_a = \sum_{i=1}^{N} \left\{ y_i^a \log(h_{\gamma,\theta}(\mathbf{x}_i)) + (1 - y_i^a)(\log(1 - h_{\gamma,\theta}(\mathbf{x}_i))) \right\}$$

To train the network, the weights or parameters α at which combined loss, $T(\alpha)$,

presented in the equation 6.1 is minimized. Formally the following equation represents the goal of the learning in the proposed MATSD model.

$$\alpha \leftarrow \arg\min_{\alpha}\{L_T(\alpha)\} \tag{6.2}$$

The entire process of training the proposed MATSD model is presented in Algorithm 9. The algorithm takes dataset of Tweets, D consisting of set of tweets and labels, $D = \{(\mathbf{x}_i, y_i^s, y_i^a\}_{i=1}^N$, as input. The learned parameters are the output of the algorithm. The initial steps involve preprocessing the data and partitioning the data into training and testing parts. Steps 4 and 5 denote the initialization of the class labels from the training dataset (D^{TR}), corresponding to the tasks of stress and sarcasm, respectively. The steps 6 to 9 denote the initialization of various parameters. The tweets are then sent to the embedding layer, where the embedding matrix, $\mathbf{E}_m$ are computed for every tweet. The set of embedding matrices corresponding to all the tweets is denoted by $\mathbf{E}_D$. Later, in each epoch, the forward pass of the network is executed to compute the final error at the output layers. The gradient of total loss, L_T, denoted by ∇L_T, is computed using the backpropagation strategy. The parameters, α, are updated with the gradient value as shown in line 32 of the algorithm 9. The cycle of epochs runs as long as the number of iterations crosses the threshold value. Finally, obtained weights after convergence, are partitioned as per the network architecture and each task gets its own sets of parameters and also the set of shared parameters as well. All three sets of parameters form the result object and are returned as output of the algorithm. In this chapter, the input batch size is taken as 32 and the number of epochs considered is 100.

6.3 Experimental Setup

This section describes dataset details and the environment setup of the experiments conducted. The experiments are implemented in Python 3.8 version using the development environment of Jupyter-notebook, of version 6.5.2.

Datasets are collected as per the procedure followed in section 4.2.1. The collected

Algorithm 9: Proposed Multi-task learning framework for tweet level stress detection

Input: Dataset of Tweets (D), consisting of tweets and labels of two tasks, $D = \{(\mathbf{x}_i, y_i^s, y_i^a\}_{i=1}^N$; The maximum number of epochs (*epoch_threshold*); learning rate(η)

Output: Model object with learned parameters.

Function `MATSD1(`D`):`

- Perform the Preprocessing of the data;
- Reorganize the data into training and testing sets, such that, $D \leftarrow D^{TR} \bigcup D^{TS}$;
- $\mathbf{Y}_s^{TR} \leftarrow \{y_i^s | (\mathbf{x}_i, y_i^s, y_i^a) \in D^{TR}, \forall i \in \{1, 2, ...|D^{TR}|\}\}$;
- $\mathbf{Y}_a^{TR} \leftarrow \{y_j^a | (\mathbf{x}_j, y_j^s, y_j^a) \in D^{TR}, \forall j \in \{1, 2, ...|D^{TR}|\}\}$;
- $\mathbf{E}^D \leftarrow \phi$;
- Initialize all weights (α) in the network to default values ;
- $epoch\_count = 0$;
- $\tau_1 \leftarrow epoch\_threshold$;
- $\mathbf{E}_D \leftarrow \phi$;
- **for** $\mathbf{x} \in D^{TR}$ **do**
  - $\mathbf{E}_m \leftarrow \phi$;
  - **for** $w \in \mathbf{x}$ **do**
    - $\mathbf{E}_w \leftarrow$ Embedding_Vector(w) ;
    - $\mathbf{E}_m \leftarrow \mathbf{E}_m \oplus \mathbf{E}_w$;
  - **end**
  - $\mathbf{E}_D \leftarrow \mathbf{E}_D \bigcup \mathbf{E}_m$;
- **end**
- **while** $epoch\_count < \tau_1$ **do**
  - **for** $\mathbf{E}_m \in \mathbf{E}_D$ **do**
    - $\mathbf{f_1} \leftarrow$Conv($\mathbf{E}_m$);
    - $\mathbf{f_2} \leftarrow$ReLU($\mathbf{f_1}$);
    - $\mathbf{f_3} \leftarrow$Max_Pool($\mathbf{f_2}$);
    - $\mathbf{f_4} \leftarrow FC(\mathbf{f_3})$;
    - $\mathbf{f_5} \leftarrow Soft\_Max(\mathbf{f_4})$;
    - $\mathbf{g_1} \leftarrow LSTM(\mathbf{E}_m)$;
    - $\sigma_\mathbf{f} \leftarrow \sigma(\mathbf{f_4})$;
    - $\mathbf{J}_{\mathbf{L_1}} \leftarrow FC(\sigma_\mathbf{f})$;
    - $\mathbf{J}_{\mathbf{L_2}} \leftarrow FC(\mathbf{g_1})$;
    - $\mathbf{g_2} \leftarrow FC(\mathbf{J}_{\mathbf{L_1}} \oplus \mathbf{J}_{\mathbf{L_2}})$;
    - $\mathbf{g_3} \leftarrow Soft\_max(\mathbf{g_2})$;
  - **end**
  - Compute the Gradient ∇L_T of the combined (total) loss function ;
  - Update weights α as $\alpha \leftarrow \alpha - \eta.\nabla L_T$;
  - $epoch\_count \leftarrow epoch\_count + 1$;
- **end**
- Partition $\alpha \leftarrow \beta \bigcup \gamma \bigcup \theta$;
- $\theta \leftarrow$ Parameters shared by both the tasks;
- $\beta \bigcup \theta \leftarrow$ Parameters specific to the task of detecting Stress;
- $\gamma \bigcup \theta \leftarrow$ Parameters specific to the task of detecting Sarcasm;
- $result\_object \leftarrow \{< \theta, \beta, \gamma >\}$;
- **return** *result_object*

return *result_object*

and labeled tweets are then preprocessed in order to remove unnecessary symbols and links, as specified in the seciton 4.2.1.1. This makes the data ready for classification The description of the information related to datasets is presented in tables 6.2 and 6.3. Table 6.2 records the label distribution of the primary task of stress. Table 6.3 presents the distribution of sarcasm labels in the datasets.

Table 6.2: The description of the distribution of stress labels in the datasets

Dataset	**Number of tweets**	**Number of stressed tweets**	**Number of non-stressed tweets**
DS1	7,289	6,134	1,155
DS2	16,532	12,628	3,904
DS3	4,062	3,176	886
DS4	7,209	5,495	1,714

Table 6.3: The description of the distribution of sarcasm labels in the datasets

Dataset	**Number of tweets**	**Number of sarcastic tweets**	**Number of non-sarcastic tweets**
DS1	7,289	397	6,892
DS2	16,532	1,198	15,334
DS3	4,062	504	3,558
DS4	7,209	955	6,254

6.3.1 State-of-the-art methods used for comparative analysis

This section presents the list of baseline machine learning models and state-of-the-art models in tweet-level stress detection. The baseline classic machine learning models are known for their better performances in text classification.

The evaluation of the results of the proposed MATSD model is compared with the popular single-task learning algorithms as well as the state-of-the-art multi-task learning algorithms. The metrics used for comparison are Accuracy and F1-Score, which are defined in section 1.3.

- *Convolutional Neural Network (CNN)* [47, 96]: It is one of the popular deep neural networks used for classification. The network uses convolution and max pooling operations for the automatic extraction of the features specific to the classification problem. The input text sequences are embedded into a matrix and fed as input to the convolution network. It consists of three stages- convolution, maxpoling, and fully connected layer. A set of filters is used to perform convolution over the input embedding matrix, producing multiple feature maps. The activation function of rectified linear unit (ReLU) is applied on the resultant features. Later the max pooling stage applies the aggregate operation, to extract a compact feature map. The final set of feature maps are flattened and sent as input to the connected layer and Softmax layer for classification.

- *Long-short-term-memory (LSTM)* [97]: It is a kind of recurrent neural network to learns long-term dependencies from the sequence data. It consists of a set of three gates (input gate, forget gate, and output gate) in each cell and a set of LSTM cells form an LSTM layer. At every cell, the input consists of cell state and hidden state. Every cell generates hidden states at the next time step. Also, the hidden state. Moreover, the hidden state produced at each cell is also considered as the output of the cell. For sequence-to-sequence and vector-to-sequence prediction problems, the output of each cell is activated. On the other hand for the classification problems or sequence-to-vector prediction problems, the output of every cell except the final cell is ignored. The output of the final cell is passed into the fully connected layer for the purpose of classification.

- *Bi-LSTM with dropout* [37] : This is a single-task learning mechanism for learning stress and state-of-the-art work in the detection of stress at tweet-level. The model is developed for the identification of the stress experienced by minority groups in any given demography. To this end, it uses a bi-directional sequential model of Bi-directional long short-term memory (Bi-LSTM). The input is initially preprocessed and sent to the layer of embedding, which produces an output vector of 300 dimensions for each word. Later, the embedded vectors are

given as input to the Bi-LSTM layer. The output of this layer is then sent to the dropout layer. The resultant vector is flattened and passed to a fully-connected layer. This Bi-LSTM is implemented separately for each of the single tasks.

- *Multi task Multi-channel CNN* [43]: This is a state-of-the-art multi-task learning-based model for the detection of psychological stress. This model uses multi-channel convolutional neural network architecture in the multi-task learning scenario for learning three tasks simultaneously. This model is adapted for two tasks sarcasm and stress in implementation.

- *Multi task Hybrid CNN Bi-GRU* [98]: this chapter utilizes a multi-task architecture consisting of two components. The first component of the convolutional neural network (CNN) is to learn the global feature representation and the gated recurrent units(GRU) in combination with the attention layer to learn the local representation of the features. This chapter is adapted to the current problem of stress detection using sarcasm as auxiliary task. The concept of stress and sarcasm are learned by learning the global and local features.

- *Multi task Bi-GRU with Attention* [77]: this chapter proposed three different multi-task learning architectures for complaint detection as the primary task with sentiment detection as an auxiliary task. The embedding layer generates the word embeddings which are later fed into the Bi-directional gated recurrent units, which store the context of the words in a sentence, and the output of this layer is fed into the attention layer. Following this, there exist task-specific layers. In the pipelined multitask architecture of complaint detection, the output of the sentiment task is concatenated with the task-specific layer of complaint detection and then fed as input to the output layer of the complaint detection task. This multi-task pipelined system of these architectures is selected for comparison with the proposed MATSD in this chapter.

6.4 Results

This section gives a description of the experimental results of this chapter. It presents the evaluation of the working performance of the proposed model and other state-of-art models on various datasets. The performance results of the experiments of various models are measured in terms of accuracy and F1-score and are recorded in tables 6.4, 6.5, 6.6, and 6.7. The obtained results are evaluated using the statistical hypothesis test of "Students' t-test" and the null hypothesis is rejected at the critical value of $\alpha < 0.05$, making the obtained results statistically significant.

6.4.1 Performance evaluation of the multi-task learning models over the single task learning approaches

The multi-task learning can act as regularization and help in better performance. From table 6.4, it is noted that the multi-task learning approaches record better accuracy and F1 scores compared to the single-task learning models. This is true for all the datasets, as can be observed in the tables 6.5, 6.6, and 6.7. It is noted that in the multi-task approaches, only the multi-channel CNN approach of [43] records lesser performance than the single-task learning models for the prediction of both tasks. However, it is observed that when the task of stress is predicted using the proposed MATSD approach, it outperforms the accuracy in stress detection models based on single task learning by at least 8.2%, 16.4%, 11.4%, 8.4% when implemented with datasets DS1, DS2, DS3, and DS4 respectively. And it is concluded that the proposed MATSD model records an improvement in F1-score by at least 0.043 points, 0.0925 points, 0.0661 points, and 0.0294 points respectively, on implementation with datasets DS1, DS2, DS3, and DS4. Therefore, it is concluded that the proposed multi-task approach of MATSD records a better performance than the popular traditional single-task learning approaches.

6.4.2 Performance evaluation of stress detection using proposed MATSD over the state-of-the-art multi-task learning methods

The multi-task approach using the proposed MATSD framework develops a deep neural network model, where the information learned from sarcasm detection is shared with the stress detection subnet after applying a sigmoid function. From the table 6.4, it is observed that the proposed MATSD model records at least a 2.75 % increment in accuracy over the other state-of-the-art models. Similarly, from tables 6.5, 6.6, and 6.7, it is noted that when the models are implemented on datasets DS2, DS3, and DS4, the MATSD algorithm shows an increment in accuracy by 6.55%, 1.84%, 10.98% respectively.

The proposed MATSD outperforms the state-of-the-art models in the F1 score as well. From the tables 6.4, 6.5, 6.6, and 6.7, it is observed that the performance of the proposed MATSD model, when implemented on the datasets DS1, DS2, DS3 and DS4, records an improvement in F1-Score by a minimum of 0.0128 points, 0.0365 points, 0.0147 points, 0.0685 points, respectively. Hence, it can be concluded that the proposed MATSD outperforms the state-of-the-art multi-task learning strategies in both accuracy and F1-score.

6.4.3 The performance of the auxiliary task of sarcasm

In multi-task learning mechanisms, the auxiliary tasks act as regularizers in constraining the weights of the primary task. In the proposed multitask learning model of MATSD, sarcasm is utilized as the auxiliary task and the information learned as part of sarcasm detection is shared with the stress task through a joint map layer. This joint map is preceded with an application of a sigmoid function to the output of maxpooling layer of the sarcasm subnet. Hence, the joint map layer contains information from the both the tasks and helps in stress prediction.

It is also noted that the sarcasm detection performance improves when implemented using multi-task approaches when compared to the sarcasm detection perfor-

mance when implemented as a single-task learning approach. It is observed from the tables, 6.4, 6.5, 6.6 and 6.7,that except for the model of [98], the sarcasm detection shows an improved performance when implemented as a multi-task learning model, when compared to the sarcasm prediction as a single task. The tables 6.4, 6.5, 6.6 and 6.7 depict that the performance of sarcasm predicted by the proposed MATSD model records an improvement of accuracy by a minimum of 8.35% 8.19% 5.9%, and 11.59% on implementation with datasets DS1, DS2, DS3, and DS4, respectively. Furthermore, from the tables6.4, 6.5, 6.6 and 6.7, it is noted that the minimum increment recorded in F1-Scores of sarcasm detection over the single-task learning models as 0.107 points, 0.113 points, 0.091 points, and 0.107 points on datasets DS1, DS2, DS3 and DS4, respectively. Hence, it can be concluded that multi-task learning helped in the sharing of parameters and helped improve the performance of all the tasks when compared to the single-task learning models.

Table 6.4: The results of the proposed and baseline models on dataset DS1

Approach	**Classifier**	**Stress (Primary task)**		**Sarcasm (Auxiliary Task)**	
		Accuracy	**F1-Score**	**Accuracy**	**F1-Score**
Single-task Learning	CNN	0.845	0.9159	0.837	0.81
	LSTM	0.8553	0.9162	0.846	0.801
	Bi-LSTM [37]	0.9053	0.946	0.8472	0.827
Multi-task Learning	Multi-Channel CNN [43]	0.845	0.9159	0.882	0.803
	GRU-CNN-GRU [98]	0.93	0.9589	0.8744	0.729
	Pipelined Bi-GRU [77]	0.9595	0.9762	0.9082	0.887
	Proposed MATSD	**0.987**	**0.989**	**0.9307**	**0.934**

Table 6.5: The results of the proposed and baseline models on dataset DS2

Approach	**Classifier**	**Stress (Primary task)**		**Sarcasm (Auxiliary Task)**	
		Accuracy	**F1-Score**	**Accuracy**	**F1-Score**
Single-task Learning	CNN	0.7693	0.8695	0.8335	0.8234
	LSTM	0.7693	0.8595	0.845	0.8151
	Bi-LSTM [37]	0.8231	0.8949	0.853	0.8165
Multi-task Learning	Multi-Channel CNN [43]	0.7693	0.8695	0.8935	0.8449
	GRU-CNN-GRU [98]	0.9217	0.9509	0.9026	0.7008
	Pipelined Bi-GRU [77]	0.7693	0.8695	0.9135	0.8442
	Proposed MATSD	**0.9872**	**0.9874**	**0.9349**	**0.9362**

Table 6.6: The results of the proposed and baseline models on dataset DS3

Approach	**Classifier**	**Stress (Primary task)**		**Sarcasm (Auxiliary Task)**	
		Accuracy	**F1-Score**	**Accuracy**	**F1-Score**
Single-task Learning	CNN	0.8155	0.8972	0.8547	0.7983
	LSTM	0.8204	0.9058	0.8659	0.8383
	Bi-LSTM [37]	0.8659	0.9141	0.8684	0.80
Multi-task Learning	Multi-Channel CNN [43]	0.8069	0.8931	0.8747	0.8105
	GRU-CNN-GRU [98]	0.9606	0.9655	0.8973	0.8067
	Pipelined Bi-GRU [77]	0.8069	0.8931	0.9047	0.8217
	Proposed MATSD	**0.979**	**0.9802**	**0.9274**	**0.9298**

Table 6.7: The results of the proposed and baseline models on dataset DS4

Approach	**Classifier**	**Stress (Primary task)**		**Sarcasm (Auxiliary Task)**	
		Accuracy	**F1-Score**	**Accuracy**	**F1-Score**
Single-task Learning	CNN	0.7693	0.8695	0.8029	0.7921
	LSTM	0.7693	0.8695	0.8135	0.795
	Bi-LSTM [37]	0.8942	0.9486	0.8535	0.8609
Multi-task Learning	Multi-Channel CNN [43]	0.7693	0.8695	0.8935	0.8423
	GRU-CNN-GRU [98]	0.8682	0.9095	0.9008	0.733
	Pipelined Bi-GRU [77]	0.7693	0.8695	0.9035	0.8621
	Proposed MATSD	**0.978**	**0.978**	**0.9694**	**0.968**

6.5 Discussion

This section presents a discussion on the results of the experiments conducted.

6.5.1 Analysis of loss function behavior for the proposed model

The value of the loss function computed during the training of the model is termed as training loss. The aggregated value of the loss function computed during the validation phase on the validation dataset is called validation loss. The relative variation of training loss and validation loss of the proposed MATSD model, when executed over the large number of epochs shows whether the proposed model is an underfit, or overfit or good fit [47]. If both training and validation loss are high and validation loss doesn't reduce, it is termed as an underfit. If the training loss decreases gradually, and the validation loss is high and increases after some epochs, it is a scenario of overfitting. The good fit scenario tells that the training and validation loss both decrease with the number of epochs and after some epochs, the validation loss also reduces and becomes either equal or less than that of the training loss variation. In this chapter, the cross entropy loss is considered for computing loss function values.

Figure 6.3 shows the training loss and validation loss when the proposed MATSD model is implemented on dataset D1. This is recorded over 100 epochs. It can be observed that the training loss has a smooth decrease and few peaks of smaller magnitude less number of peaks. Whereas the validation loss is varying and not smooth. However, the magnitude of the difference between validation and training losses is not very high. At the end of epochs, it decreases and nearly converges with training loss. Figure 6.4 shows the training loss and validation loss when the proposed MATSD model, implemented on dataset D2, which is recorded over 100 epochs. It can be observed that the training loss has smooth decrease after one epoch. The validation loss has some peak at initial epochs, but it is smooth and not varying much. The training and validation loss maintain nearly constant small difference after 20 epochs.

Figure 6.5 shows the training loss and validation loss when the proposed MATSD

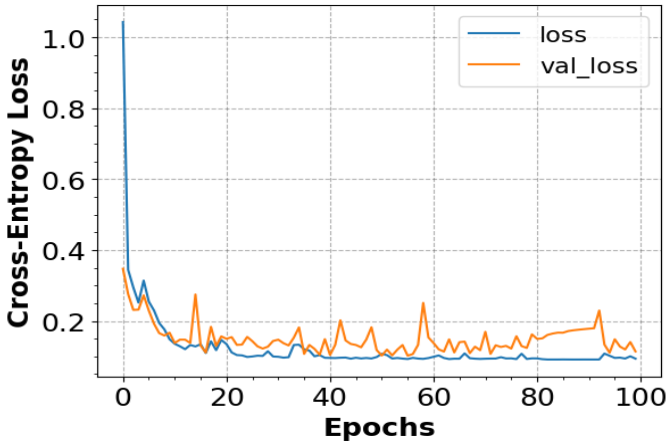

Figure 6.3: Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS1

model is implemented on dataset D3. This is recorded over 100 epochs. It can be observed that the training loss has a smooth decrease after the first epoch and few peaks of smaller magnitude less number of peaks. The validation loss too varies smoothly after the first epoch. However, the validation loss has few peaks and a valley though the the magnitude of the difference between validation and training losses is not very high. At the end of epochs, it decreases and nearly converges with training loss. Figure 6.6 shows the training loss and validation loss when the proposed MATSD model, during implementation on dataset D4, which is recorded over 100 epochs. It can be observed that the training loss has two peaks before 40 epochs and it converges to nearly zero later on. The validation loss decreases after two epochs, and later a plateau kind of peak is observed from epochs 15 to 36. During the later epochs, the variation in validation loss is nearly unchanging except for a dip between the epochs of 65 to 70. The training and validation loss maintains a nearly constant small difference after 35 epochs and the validation loss is nearing to training loss after 80 epochs. It can be concluded that on all four datasets the validation loss and training loss curves are having less difference in magnitude and in a few cases converge.

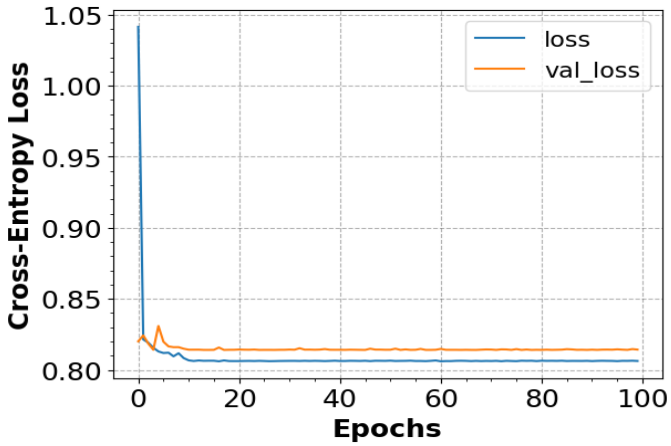

Figure 6.4: Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS2

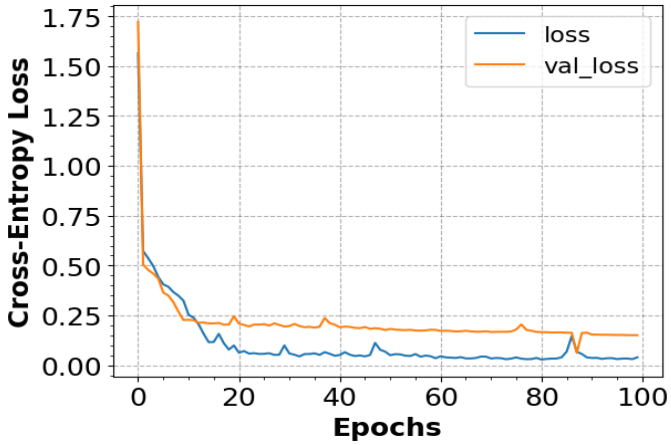

Figure 6.5: Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS3

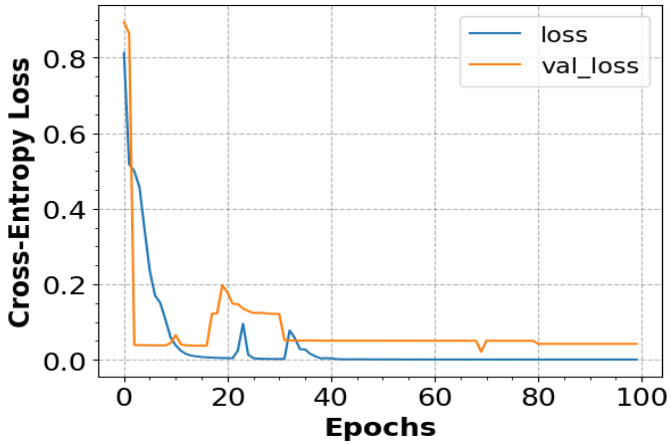

Figure 6.6: Graph showing loss functions of training and validation of the proposed MATSD model implemented on dataset DS4

6.5.2 Limitations of the proposed MATSD model

The Limitations of the proposed MATSD model are given as follows:

- The proposed model of MATSD doesn't perform well in case the tweets are devoid of textual information.
- The proposed model of MATSD may not give the desired performance if the sarcasm expressed in the content is not in a textual form such as any other means of media.

6.6 Summary

In this chapter, a deep multi-task learning-based framework called MATSD is proposed for detecting tweet-level stress detection using sarcasm present in the tweets' content. The datasets required for the training are crawled from Twitter's API Tweepy and the labels of the sarcasm task are taken from the sarcasm value computed in chapter 4. In the proposed MATSD, the detection of sarcasm is an auxiliary task learned using CNN and the detection of of stress is the primary task implemented using the LSTM network. Both of these tasks are jointly learned so that performance for the task of stress detection is improved when compared to the case where stress detection is developed using a standalone single-task model. In addition, the proposed model uses the sigmoid function before sharing the information from the sarcasm task subnet with the stress task subnet. An extensive set of experiments is conducted on various baselines and state-of-the-art techniques to validate the results of the proposed method. Also, the loss function values are analyzed for the proposed model over all four datasets. It is observed that the proposed MATSD records an improvement in F1 scores of a minimum of 0.0128 points, 0.0365 points, 0.0147 points, and 0.0685 points, respectively, on the four datasets DS1, DS2, DS3, DS4 when compared to the state-of-the-art multi-task learning models. Similarly, the proposed model records an increment in the accuracy by a minimum of 2.75%, 6.55%,

1.84%, and 10.98% respectively on the four datasets DS1, DS2, DS3, and DS4 compared to the state-of-the-art multi-task learning models. This is significantly greater than the baseline models and state-of-the-art techniques implemented with the same datasets.

Chapter 7

Conclusion and Future Directions

In this thesis, we had investigated text-based stress detection approaches like NTSD, STSD, SMTSD and MATSD. As chronic stress can be detrimental to both physical and mental health, early detection and mitigation of stress has attained importance. Traditional techniques to detect stress have shown the initial way. The traditional methods to detect stress used questionnaires, interviews with psychiatrists, measuring physiological activity with electronic devices, etc. However, these techniques suffer from social stigma and electronic device-based detection is viewed as invasive to normal activities. The growing popularity of social media- as a medium where people can express themselves freely - made researchers to explore the potential of social media postings as the data source for detecting stress. In this thesis, the popular social media microblogging site called Twitter was used to collect data.

The problem of stress detection at tweet-level is investigated in the literature, but the influence of the neighboring or previous tweets in detecting stress is studied as part of neighborhood-based tweet-level stress detection (NTSD) in chapter 3. Also, the concept of sarcasm, especially illocutionary sarcasm, is utilized as an attribute in NTSD. Furthermore, the concept of sarcasm is used to train the loss function in STSD, as discussed in chapter 4.

In most of the works on stress detection, as observed in the literature and also the above two objectives, the supervised classification models are employed. However, in real-world, the paucity of labeled datasets makes us to investigate a semi-supervised

solution for stress detection, called SMTSD, which is presented in chapter 5. With sarcasm being used as an important attribute in all the initial three objectives, it was intuitive to explore the multi-task approach to detect tweet-level stress with sarcasm detection as an auxiliary task. In chapter 6, a multi-task learning-based for detecting stress, MATSD is presented. It is also known from the literature that the text-level stress detection approaches have not explored the concept of multi-task learning.

7.1 The Major Outcomes of the Thesis

The chapters from 3 to 6 form the major contributions of this Thesis. The detailed contributions are listed as follows:

1. **Chapter 3 :** The role of neighborhood tweets in improving the performance of the tweet-level stress detection is presented through the model of NTSD. In addition, the concept of illocutionary sarcasm is formulated as a new attribute called *Sarcasm_Level*. It is also observed from extensive experiments that the inclusion neighborhood tweets and the new sarcasm attribute has shown better performance than the traditional classifiers without including them. The proposed NTSD had outperformed the traditional baseline ML classifiers like LR, SVM and RF.

2. **Chapter 4:** The concept of illocutionary sarcasm is further extended to include hashtags. In addition, a new model called STSD is proposed which utilizes the concept of illocutionary sarcasm in the loss function, so as to maximize the likelihood of non-sarcastic tweets and penalize the likelihood of sarcastic tweets. In addition, dimensionality reduction techniques like non-linear PCA was applied and its usage was found to improve the performance of the proposed model. For evaluating the performance of the model, an extensive set of experiments were conducted and the proposed model is compared with the standard baseline ML

models and the zero-neighborhood case of the NTSD model. It is observed that the proposed STSD records a better performance in all these cases.

3. **Chapter 5:** The semi-supervised approach to detect tweet-level stress called SMTSD is proposed. In this solution, self-training method of semi-supervised approach is employed. The concept of illocutionary sarcasm is utilized to invert the pseud-labels. For evaluation of the performance, large set of experiments were conducted using the proposed SMTSD as well as baseline self-training approach against the basic supervised models of LR, SVM, RF and NB. In addition, the state-of-the art methods like Bi-LSTM and CNN-DAE are also utilized. It is observed that the proposed SMTSD outperforms all other methods. Moreover, the supervised models also record better performance with the pseudo-labeled data set formed from the proposed SMTSD against the pseudo-labeled data from the baseline self-training method.

4. **Chapter 6:** Develops a multi-task approach called MATSD to detect tweet-level stress with sarcasm as an auxiliary task. The proposed MATSD outperforms state-of-the-art multi-task learning architectures like Multi-channel CNN, Bi-GRU, etc. Hence, it can be intuitive that tasks of detecting sarcasm and stress can share information thereby improving the performance.

7.2 Future Directions

There is an ample potential to extend future research from this thesis. Importance of stress detection has only grown with time and the recent pandemic has further re-affirmed the need to detect stress before it turns chronic. The NTSD proposed in chapter 3 can be extended to by considering the similarity co-efficient of the previous tweets with the current tweet. This is used to include relevant neighborhood tweets, among the neighborhood tweets, to the given primary tweet. In both chapters 3 and 4, the text-level data can be extended to include all modalities of input tweet data -text, image, and multimedia. Furthermore, modality-invariant prediction models

need to be built for the purpose. Finally, the NTSD model can be extended to use neighborhood tweets based on time-units rather than count.

From chapter 5, in which we proposed a semi-supervised approach of SMTSD as a solution for the problem of tweet-level stress detection, there is a future scope to develop unsupervised models to solve the problem. It is intuitive from the chapter 6, the proposed multi-task deep learning-based solution called MATSD for detecting tweet-level stress using sarcasm as auxiliary attribute can be extended to multi-modal datasets. In addition, the semi-supervised modeling can also be utilized with multi-task learning approach, given the lack of large labeled datasets in realworld.

In all the four chapters from 3 to 6, illocutionary sarcasm is computed as an attribute and utilized in the models either as an attribute or as an auxiliary task. There is a scope to extend the computation of illocutionary sarcasm in future by considering the multi-modal datasets. In addition, multi-stage learning approaches could also be employed for filtering the tweets with sarcasm and then detecting stress in both sarcastic and non-sarcastic categories.

Bibliography

[1] Anna Vittoria Mattioli, Susanna Sciomer, Silvia Maffei, and Sabina Gallina. Lifestyle and stress management in women during covid-19 pandemic: impact on cardiovascular risk burden. *American journal of lifestyle medicine*, 15(3):356–359, 2021.

[2] Hilary Dobson and RF Smith. What is stress, and how does it affect reproduction? *Animal reproduction science*, 60:743–752, 2000.

[3] Huijie Lin, Jia Jia, Liqiang Nie, Guangyao Shen, and Tat-Seng Chua. What does social media say about your stress?. In *IJCAI*, pages 3775–3781, 2016.

[4] Ronald Glaser and Janice K Kiecolt-Glaser. Stress-induced immune dysfunction: implications for health. *Nature Reviews Immunology*, 5(3):243–251, 2005.

[5] Spencer L James, Degu Abate, Kalkidan Hassen Abate, Solomon M Abay, Cristiana Abbafati, Nooshin Abbasi, Hedayat Abbastabar, Foad Abd-Allah, Jemal Abdela, Ahmed Abdelalim, et al. Global, regional, and national incidence, prevalence, and years lived with disability for 354 diseases and injuries for 195 countries and territories, 1990–2017: a systematic analysis for the global burden of disease study 2017. *The Lancet*, 392(10159):1789–1858, 2018.

[6] Hwan-Cheol Park and Jihyun Oh. The relationship between stress and sleep quality: the mediating effect of fatigue and dizziness among patients with cardiovascular disease. *Medicine*, 102(20):e33837, 2023.

[7] Rumana Ferdousi Siddique, Oli Ahmed, and Kazi Nur Hossain. Relationship between the fear of covid-19 disease and sleep quality: The mediating role of stress. *Heliyon*, 7(5), 2021.

[8] David N Sattler, Boldsuren Bishkhorloo, Kendall A Lawley, Ruth Hackler, Chuluunbileg Byambajav, Michidmaa Munkhbat, and Brooklyn Smith-Galeno. Stigma, post-traumatic stress, and covid-19 vaccination intent in mongolia, india, and the united states. *International Journal of Environmental Research and Public Health*, 20(3):2084, 2023.

[9] Inah Kim, Min Ji Koo, Hye-Eun Lee, Yong Lim Won, and Jaechul Song. Overwork-related disorders and recent improvement of national policy in south korea. *Journal of occupational health*, 61(4):288–296, 2019.

[10] Maja Lindell and Anna Grimby-Ekman. Stress, non-restorative sleep, and physical inactivity as risk factors for chronic pain in young adults: A cohort study. *PloS one*, 17(1):e0262601, 2022.

[11] Huijie Lin, Jia Jia, Jie Huang, Enze Zhou, Jingtian Fu, Yejun Liu, and Huanbo Luan. Moodee: an intelligent mobile companion for sensing your stress from your social media postings. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[12] Yipeng Zhang, Hanjia Lyu, Yubao Liu, Xiyang Zhang, Yu Wang, Jiebo Luo, et al. Monitoring depression trends on twitter during the covid-19 pandemic: Observational study. *JMIR infodemiology*, 1(1):e26769, 2021.

[13] Nilanjan Dey, Rishabh Mishra, Simon James Fong, KC Santosh, Stanna Tan, and Rubén González Crespo. Covid-19: Psychological and psychosocial impact, fear, and passion. *Digital Government: Research and Practice*, 2(1):1–4, 2020.

[14] Cristian Paul Bara, Michalis Papakostas, and Rada Mihalcea. A deep learning approach towards multimodal stress detection. In *Proceedings of the AAAI-20 Workshop on Affective Content Analysis, New York, USA, AAAI*, pages 67–81, 2020.

[15] Jennifer A Healey and Rosalind W Picard. Detecting stress during real-world driving tasks using physiological sensors. *IEEE Transactions on intelligent transportation systems*, 6(2):156–166, 2005.

[16] Andrew Raij, Patrick Blitz, Amin Ahsan Ali, Scott Fisk, Brian French, Somnath Mitra, Motohiro Nakajima, Minh Hoai Nguyen, Kurt Plarre, Mahbubur Rahman, et al. mstress: Supporting continuous collection of objective and subjective measures of psychosocial stress on mobile devices. *ACM Wireless Health 2010 San Diego, California USA*, 2010.

[17] Prashanth KVTKN and Tene Ramakrishnudu. A novel method for detecting psychological stress at tweet level using neighborhood tweets. *Journal of King Saud University-Computer and Information Sciences*, 34(9):6663–6680, 2022.

[18] Huijie Lin, Jia Jia, Quan Guo, Yuanyuan Xue, Jie Huang, Lianhong Cai, and Ling Feng. Psychological stress detection from cross-media microblog data using deep sparse neural network. In *2014 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2014.

[19] Huijie Lin, Jia Jia, Jiezhong Qiu, Yongfeng Zhang, Guangyao Shen, Lexing Xie, Jie Tang, Ling Feng, and Tat-Seng Chua. Detecting stress based on social interactions in social networks. *IEEE Transactions on Knowledge and Data Engineering*, 29(9):1820–1833, 2017.

[20] Nilanjan Dey, Rosalina Babo, Amira S Ashour, Vishal Bhatnagar, and Med Salim Bouhlel. *Social networks science: Design, implementation, security, and challenges: From social networks analysis to social networks intelligence*. Springer, 2018.

[21] Glen Coppersmith, Craig Harman, and Mark Dredze. Measuring post traumatic stress disorder in twitter. In *Eighth international AAAI conference on weblogs and social media*, 2014.

[22] Munmun De Choudhury, Michael Gamon, Scott Counts, and Eric Horvitz. Predicting depression via social media. In *Seventh international AAAI conference on weblogs and social media*, 2013.

[23] Munmun De Choudhury, Scott Counts, and Eric Horvitz. Predicting postpartum changes in emotion and behavior via social media. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 3267–3276, 2013.

[24] Huijie Lin, Jia Jia, Quan Guo, Yuanyuan Xue, Qi Li, Jie Huang, Lianhong Cai, and Ling Feng. User-level psychological stress detection from social media using deep neural network. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 507–516, 2014.

[25] Yuanyuan Xue, Qi Li, Li Jin, Ling Feng, David A Clifton, and Gari D Clifford. Detecting adolescent psychological pressures from micro-blog. In *International Conference on Health Information Science*, pages 83–94. Springer, 2014.

[26] Yuanyuan Xue, Qi Li, Liang Zhao, Jia Jia, Ling Feng, Feng Yu, and David A Clifton. Analysis of teens' chronic stress on micro-blog. In *International Conference on Web Information Systems Engineering*, pages 121–136. Springer, 2016.

[27] Purva Asrani, Mathew Suji Eapen, Md Imtaiyaz Hassan, and Sukhwinder Singh Sohal. Implications of the second wave of covid-19 in india. *The Lancet Respiratory Medicine*, 9(9):e93–e94, 2021.

[28] Malvika Chhatwani, Sushanta Kumar Mishra, and Himanshu Rai. Active and passive social media usage and depression among the elderly during covid-19: does race matter? *Behaviour & Information Technology*, pages 1–12, 2022.

[29] Praveen Sv, Remya Lathabhavan, and Rajesh Ittamalla. What concerns indian general public on second wave of covid-19? a report on social media opinions. *Diabetes & metabolic syndrome*, 15(3):829, 2021.

[30] Yousri Marzouki, Fatimah Salem Aldossari, and Giuseppe A Veltri. Understanding the buffering effect of social media use on anxiety during the covid-19 pandemic lockdown. *Humanities and Social Sciences Communications*, 8(1):1–10, 2021.

[31] Ajay Agarwal. Ripple effect of a pandemic: Analysis of the psychological stress landscape during covid19. *PsyArXiv*, 2020.

[32] Sharath Chandra Guntuku, Garrick Sherman, Daniel C Stokes, Anish K Agarwal, Emily Seltzer, Raina M Merchant, and Lyle H Ungar. Tracking mental health and symptom mentions on twitter during covid-19. *Journal of general internal medicine*, 35(9):2798–2800, 2020.

[33] Surekha Borra and Nilanjan Dey. Misinformation about covid-19 and confidential information leakage: Impacts on the psychological well-being of indians. *Current Psychiatry Research and Reviews Formerly: Current Psychiatry Reviews*, 16(4):283–287, 2020.

[34] Liang Zhao, Jia Jia, and Ling Feng. Teenagers' stress detection based on time-sensitive micro-blog comment/response actions. In *IFIP International Conference on Artificial Intelligence in Theory and Practice*, pages 26–36. Springer, 2015.

[35] Bethany Pickering, Dominic Thompson, and Ruth Filik. Examining the emotional impact of sarcasm using a virtual environment. *Metaphor and Symbol*, 33(3):185–197, 2018.

[36] Dawn G Blasko, Victoria A Kazmerski, and Shariffah Sheik Dawood. Saying what you don't mean: A cross-cultural study of perceptions of sarcasm. *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale*, 75(2):114, 2021.

[37] Cory J Cascalheira, Shah Muhammad Hamdi, Jillian R Scheer, Koustuv Saha, Soukaina Filali Boubrahimi, and Munmun De Choudhury. Classifying minority stress disclosure on social media with bidirectional long short-term memory. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 16, pages 1373–1377, 2022.

[38] Yiding Wang, Zhenyi Wang, Chenghao Li, Yilin Zhang, and Haizhou Wang. A multitask deep learning approach for user depression detection on sina weibo. *arXiv preprint arXiv:2008.11708*, 2020.

[39] Elisabeth Camp. Sarcasm, pretense, and the semantics/pragmatics distinction. *Noûs*, 46(4):587–634, 2012.

[40] Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)*, 50(5):1–22, 2017.

[41] Simona Frenda, Alessandra Teresa Cignarella, Valerio Basile, Cristina Bosco, Viviana Patti, and Paolo Rosso. The unbearable hurtfulness of sarcasm. *Expert Systems with Applications*, page 116398, 2022.

[42] Prashanth KVTKN and Tene Ramakrishnudu. Semi-supervised approach for tweet-level stress detection. *Natural Language Processing Journal*, page 100019, 2023.

[43] Prasadith Kirinde Gamaarachchige and Diana Inkpen. Multi-task, multi-channel, multi-input learning for mental illness detection using social media text. In *Proceedings of the tenth international workshop on health text mining and information analysis (LOUHI 2019)*, pages 54–64, 2019.

[44] Yang Li, Amirmohammad Kazameini, Yash Mehta, and Erik Cambria. Multitask learning for emotion and personality detection. *arXiv preprint arXiv:2101.02346*, 2021.

[45] Prashanth KVTKN and Tene Ramakrishnudu. A novel method for detecting psychological stress at tweet level using neighborhood tweets. *Journal of King Saud University-Computer and Information Sciences*, 2021.

[46] Prashanth KVTKN and Tene Ramakrishnudu. Sarcasm-based tweet-level stress detection. *Authorea*, 2022.

[47] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[48] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[49] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21(1):6, 2020.

[50] John S Brownstein, Clark C Freifeld, and Lawrence C Madoff. Digital disease detection—harnessing the web for public health surveillance. *The New England journal of medicine*, 360(21):2153, 2009.

[51] Matthias Baumgarten, Maurice D Mulvenna, Niall Rooney, and John Reid. Keyword-based sentiment mining using twitter. *International Journal of Ambient Computing and Intelligence (IJACI)*, 5(2):56–69, 2013.

[52] Glen Coppersmith, Mark Dredze, and Craig Harman. Quantifying mental health signals in twitter. In *Proceedings of the workshop on computational linguistics and clinical psychology: From linguistic signal to clinical reality*, pages 51–60, 2014.

[53] Qi Li, Yuanyuan Xue, Jia Jia, and Ling Feng. Helping teenagers relieve psychological pressures: A micro-blog based system. In *EDBT*, pages 660–663. Citeseer, 2014.

[54] Mike Thelwall. Tensistrength: Stress and relaxation magnitude detection for social media texts. *Information Processing & Management*, 53(1):106–121, 2017.

[55] Bernice Yeow Ziwei and Hui Na Chua. An application for classifying depression in tweets. In *Proceedings of the 2nd International Conference on Computing and Big Data*, pages 37–41, 2019.

[56] Sepandar D Kamvar and Jonathan Harris. We feel fine and searching the emotional web. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 117–126, 2011.

[57] Salvatore Carta, Andrea Corriga, Riccardo Mulas, Diego Reforgiato Recupero, and Roberto Saia. A supervised multi-class multi-label word embeddings approach for toxic comment classification. In *KDIR*, pages 105–112, 2019.

[58] Usha Devi Gandhi, Priyan Malarvizhi Kumar, Gokulnath Chandra Babu, and Gayathri Karthick. Sentiment analysis on twitter data by using convolutional neural network (cnn) and long short term memory (lstm). *Wireless Personal Communications*, pages 1–10, 2021.

[59] Rajesh Kumar Mundotiya and Naina Yadav. Forward context-aware clickbait tweet identification system. *International Journal of Ambient Computing and Intelligence (IJACI)*, 12(2):21–32, 2021.

[60] Ricardo Martins, José João Almeida, Pedro Rangel Henriques, and Paulo Novais. Identifying depression clues using emotions and ai. In *ICAART (2)*, pages 1137–1143, 2021.

[61] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. Semi-supervised learning. adaptive computation and machine learning. *Methods*, 1(1):4–8, 2010.

[62] Han Yu and Akane Sano. Semi-supervised learning and data augmentation in wearable-based momentary stress detection in the wild. *arXiv preprint arXiv:2202.12935*, 2022.

[63] Nibraas Khan. Semi-supervised generative adversarial network for stress detection using partially labeled physiological data. *arXiv preprint arXiv:2206.14976*, 2022.

[64] Peng Liu, Wei Chen, Gaoyan Ou, Tengjiao Wang, Dongqing Yang, and Kai Lei. Sarcasm detection in social media based on imbalanced classification. In *International Conference on Web-Age Information Management*, pages 459–471. Springer, 2014.

[65] Ashwin Rajadesingan, Reza Zafarani, and Huan Liu. Sarcasm detection on twitter: A behavioral modeling approach. In *Proceedings of the eighth ACM international conference on web search and data mining*, pages 97–106, 2015.

[66] Shubhadeep Mukherjee and Pradip Kumar Bala. Sarcasm detection in microblogs using naïve bayes and fuzzy clustering. *Technology in Society*, 48:19–27, 2017.

[67] Karthik Sundararajan and Anandhakumar Palanisamy. Multi-rule based ensemble feature selection model for sarcasm type detection in twitter. *Computational intelligence and neuroscience*, 2020, 2020.

[68] Bishal Shaw, Sriparna Saha, Santosh Kumar Mishra, and Angshuman Ghosh. Investigations in psychological stress detection from social media text using deep architectures. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pages 1614–1620. IEEE, 2022.

[69] Aryan Rastogi, Qian Liu, and Erik Cambria. Stress detection from social media articles: New dataset benchmark and analytical study. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.

[70] Junling Gao, Pinpin Zheng, Yingnan Jia, Hao Chen, Yimeng Mao, Suhong Chen, Yi Wang, Hua Fu, and Junming Dai. Mental health problems and social media exposure during covid-19 outbreak. *Plos one*, 15(4):e0231924, 2020.

[71] Li Huang, Francesca Gino, and Adam D Galinsky. The highest form of intelligence: Sarcasm increases creativity for both expressers and recipients. *Organizational Behavior and Human Decision Processes*, 131:162–177, 2015.

[72] Bayu Yudha Pratama and Riyanarto Sarno. Personality classification based on twitter text using naive bayes, knn and svm. In *2015 International Conference on Data and Software Engineering (ICoDSE)*, pages 170–174. IEEE, 2015.

[73] Narjiss Satour, Badreddine Benyacoub, Badr El Mahrad, and Ilias Kacimi. Kpca over pca to assess urban resilience to floods. In *E3S Web of Conferences*, volume 314, page 03005. EDP Sciences, 2021.

[74] Manoranjan Dash, Hua Liu, and Jun Yao. Dimensionality reduction of unsupervised data. In *Proceedings ninth ieee international conference on tools with artificial intelligence*, pages 532–539. IEEE, 1997.

[75] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to data mining*. Pearson Education India, 2016.

[76] Adrian Benton, Margaret Mitchell, and Dirk Hovy. Multi-task learning for mental health using social media text. *arXiv preprint arXiv:1712.03538*, 2017.

[77] Apoorva Singh, Sriparna Saha, Md Hasanuzzaman, and Kuntal Dey. Multitask learning for complaint identification and sentiment analysis. *Cognitive Computation*, pages 1–16, 2022.

[78] Peng Xu, Andrea Madotto, Chien-Sheng Wu, Ji Ho Park, and Pascale Fung. Emo2vec: Learning generalized emotion representation by multi-task training. *arXiv preprint arXiv:1809.04505*, 2018.

[79] Ethan Fast, Binbin Chen, and Michael S Bernstein. Empath: Understanding topic signals in large-scale text. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4647–4657, 2016.

[80] Andrey Kim, Yongsoo Song, Miran Kim, Keewoo Lee, and Jung Hee Cheon. Logistic regression model training based on the approximate homomorphic encryption. *BMC medical genomics*, 11(4):83, 2018.

[81] Scott Menard. *Applied logistic regression analysis*, volume 106. Sage, 2002.

[82] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[83] Tom M Mitchell et al. Machine learning, 1997.

[84] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152, 1992.

[85] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[86] Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, David Heckerman, Chris Meek, et al. *Introduction to statistical relational learning*. MIT press, 2007.

[87] Khalid Raza. Machine learning in single-cell rna-seq data analysis.

[88] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[89] John A Lee and Michel Verleysen. *Nonlinear dimensionality reduction*, volume 1. Springer, 2007.

[90] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, 29:103–130, 1997.

[91] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine learning research*, 7:1–30, 2006.

[92] Samet Oymak and Talha Cihad Gulcu. Statistical and algorithmic insights for semi-supervised learning with self-training. *arXiv preprint arXiv:2006.11006*, 2020.

[93] Vivian Lay Shan Lee, Keng Hoon Gan, Tien Ping Tan, and Rosni Abdullah. Semi-supervised learning for sentiment classification using small number of labeled data. *Procedia Computer Science*, 161:577–584, 2019.

[94] KVTKN Prashanth and Tene Ramakrishnudu. Sarcasm-based tweet-level stress detection. *Expert Systems*, page e13534, 2024.

[95] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[96] Laith Alzubaidi, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamaría, Mohammed A Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: Concepts, cnn architectures, challenges, applications, future directions. *Journal of big Data*, 8:1–74, 2021.

[97] Kamilya Smagulova and Alex Pappachen James. A survey on lstm memristive neural network architectures and applications. *The European Physical Journal Special Topics*, 228(10):2313–2324, 2019.

[98] Guangquan Lu, Jiangzhang Gan, Jian Yin, Zhiping Luo, Bo Li, and Xishun Zhao. Multi-task learning using a hybrid representation for text classification. *Neural Computing and Applications*, 32:6467–6480, 2020.

List of Publications

Published:

1. **Prashanth KVTKN**, Tene Ramakrishnudu, "A novel method for detecting psychological stress at tweet level using neighborhood tweets", *Journal of King Saud University - Computer and Information Sciences*, Volume 34, Issue 9, October 2022, Pages 6663-6680.
2. **Prashanth KVTKN**, Tene Ramakrishnudu, "Sarcasm-based tweet-level stress detection", *Expert Systems* , (2024): e13534.
3. **Prashanth KVTKN**, Tene Ramakrishnudu, "Semi-supervised approach for tweet-level stress detection ", *Natural Language Processing Journal, volume 4, Volume 4, September 2023, 100019*

Communicated:

1. **Prashanth KVTKN**, Tene Ramakrishnudu, "Multi-task Learning Approach for Tweet-level Stress Detection Using Deep Learning ", *SN Computer Science*

ॐ शान्तिः शान्तिः शान्तिः

Transliteration

Ōṃ śānti śānti śānti

Translation

Ōṃ Peace, Peace, and Peace to all, everywhere, in all circumstances !