UNSUPERVISED/SEMI-SUPERVISED LEARNING-BASED STRESS LEVEL DETECTION SYSTEM BY USING UNOBTRUSIVE WEARABLES IN THE WILD

by

Osman Tugay Başaran

B.S., Electronics&Communication Engineering, Istanbul Technical University, 2018

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2022

ACKNOWLEDGEMENTS

I would like to express my gratitude to Professor Cem Ersoy, who trusted me and accepted me into the research group at the beginning of this adventure. Throughout my research, he has always been with me with his understanding, subtle, guiding, and supportive comments. Apart from research, he has greatly contributed to me with his vision and perspective on life. I will always be honored to be his student. Also, I would like to express my heartfelt thanks to my thesis co-advisor Dr. Yekta Said Can. He always took me a few steps forward with his constructive and perspective-expanding comments on the difficulties I faced. I am aware that we have received a very distinguished education thanks to the efforts of my esteemed Bogazici University Computer Engineering Department professors and academic staff; I would like to thank each of them for their visionary contributions.

My most special thanks to my family, who stand by me in every decision I make, support me endlessly, and have the most outstanding effort in getting me to where I am today. My mother, who taught me the existence of unconditional love and being happy with simple things; my father, who taught me the value of pursuing unconventional, ground-breaker, fearless ideas and a strong father-son friendship; my smart brother, whose ideas are older than his age, I am glad that you are my family and you were with me in this process.

Finally, I dedicate this thesis study to my warmhearted grandmother, who has made a priceless contribution to my life and success. I hope you are at peace where you are; I love you...

Brussels, May 2022

ABSTRACT

UNSUPERVISED/SEMI-SUPERVISED LEARNING-BASED STRESS LEVEL DETECTION SYSTEM BY USING UNOBTRUSIVE WEARABLES IN THE WILD

Stress is one of the most important problems of today. Although it seems to be a part of modern human life, it is known to cause serious health problems. Many researchers from different disciplines have been working on this subject, which has personal and social effects, for many years. Psychologists, behavioral scientists, and psychiatrists continue their research in the clinical setting. However, when the stress factor is considered as a part of daily life, clinical environments or controlled experimental areas may be insufficient in terms of stress classification. Thanks to developing sensor technologies, wearable devices, and machine learning methods, stress classification has become an area of interest for computer scientists. Although developments in wearable sensors, ubiquitous computing, and machine learning continue, they bring new challenges to this field. The data labeling burden is one of these challenges. It requires significant effort and resources to have the subjects who have stress problems fill out questionnaires periodically in their daily life and to synchronize the physiological data with the questionnaire results. Being aware of this labeling burden, we aimed to find a new solution by using a less amount of labeled data from the multi-sensor physiological dataset that we collect in daily life. For this reason, this thesis focuses on what will be the performance of a system using a less amount of labeled data and semi-supervised learning techniques.

ÖZET

GÜNLÜK HAYATTA GÖZETİMSİZ/YARI GÖZETİMLİ ÖĞRENME TEMELLİ STRES DÜZEYİ TESPİT SİSTEMİ

Stres günümüzün en önemli problemlerinden biridir. Modern insan yaşamının bir parçası gibi görünse de ciddi sağlık sorunlarına neden olduğu bilinmektedir. Farklı disiplinlerden birçok araştırmacı bireysel ve sosyal etkileri olan bu konu üzerine uzun yıllardır çalışmaktadır. Psikologlar, davranış bilimciler ve psikiyatristler klinik ortamda araştırmalarını sürdürmektedir. Fakat stres faktörü günlük hayatın bir parçası olarak düşünülünce klinik ortamlar veya kontrollü deney alanları stres tanılama açısından yetersiz kalabilmektedir. Gelişen sensör teknolojileri, giyilebilir cihazlar ve makine öğrenmesi metodları sayesinde stres tanılama konusu bilgisayar bilimcilerinde ilgi alanı haline gelmiştir. Giyilebilir sensörler, yaygın bilişim ve makine öğrenimi konularında gelişmeler devam etse de bu alan yeni zorlukları beraberinde getirmektedir. Veri etiketleme yükü bu zorluklardan biridir. Özellikle stres problemi yaşayan deneklere günlük hayat içerisinde düzenli aralıklarla anket doldurtmak, veriler ile bu anket sonuçlarını senkronize etmek önemli efor ve kaynak gerektirmektedir. Biz de bu etiketleme yükünün farkında olarak günlük hayatta topladığımız çok tipli sensör fizyolojik veriseti içerisinden az miktarda etiketli veri kullanarak yeni bir çözüm yolu bulmayı hedefledik. Bu nedenle tez çalışması yarı-gözetimli öğrenme teknikleri kullanılarak eldeki az miktardaki etiketli veri kullanılarak nasıl sonuçlar elde edilebileceğine odaklanmaktadır.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES ix
LIST OF TABLES xi
LIST OF SYMBOLS xiii
LIST OF ACRONYMS/ABBREVIATIONS xiv
1. INTRODUCTION 1
1.1. Motivation 2
1.2. Contributions 3
1.3. Thesis Outline 4
2. BACKGROUND 5
2.1. Stress and Its Origin 5
2.2. Stress Signals 7
2.2.1. Brain Signals 8
2.2.2. Heart Signals 9
2.2.3. Muscle Signals 9
2.2.4. Electrodermal Activity(EDA) 10
2.2.5. Blood Volume Pulse(BVP) 10
2.2.6. Acceleration Data 10
2.2.7. Skin Temperature(ST) 10
2.2.8. Speech Signals 11
2.2.9. Facial Mimicry 11
2.2.10. Keyboard and Mouse Usage 11
2.3. Stress Experiment Environments 12
2.3.1. Restricted Environments 12
2.3.2. Semi-Restricted Environments 13
2.3.3. Non-Restricted Environments (Daily Life) 13
2.4. Stress Data Collection Challenges 13

2.4.1. Noisy and Distorted Signals 14
2.4.2. Data Fusion Principles 14
2.4.3. Unobtrusive Design 14
2.4.4. Battery Life 14
2.4.5. Data Labeling Process 15
2.5. Questionnaires and Tests for Stress 15
2.5.1. Perceived Stress Scale (PSS) 15
2.5.2. NASA Task Load Index (NASA-TLX) 16
2.5.3. The State-Trait Anxiety Inventory (STAI) 17
3. LITERATURE REVIEW 19
3.1. Stress Prediction with Supervised Learning (SL) Models 20
3.2. Stress Prediction with Semi-Supervised Learning (SSL) Models 23
3.3. Stress Prediction with Unsupervised Learning (UL) Models 25
4. PROPOSED SEMI-SUPERVISED LEARNING ARCHITECTURES 27
4.1. Experiment Design 27
4.1.1. Data Collection Unit - Empatica E4 Wristband 27
4.1.2. Ground Truth Collection 29
4.1.3. Ethics 30
4.2. Label Propagation Algorithm 32
4.2.1. Theoretical Formulation and Preliminaries 32
4.2.2. Algorithm Implementation 33
4.3. Autoencoder Architecture 34
4.3.1. Theoretical Formulation and Preliminaries 34
4.3.2. Algorithm Implementation 35
4.4. Experimental Results & Discussion 36
4.4.1. LP Algorithm Hyperparameter Tuning Stages and Results . . . 37
4.4.2. Autoencoder Hyperparameter Tuning Stages and Results 40
5. EXPERIMENTS WITH SUPERVISED LEARNING ARCHITECTURES . . 51
5.1. LSTM Networks 51
5.1.1. Theoretical Formulation and Preliminaries 52
5.2. CNN-LSTM Networks 53

5.3. Experimental Results & Discussion 55
5.3.1. LSTM Network Hyperparameter Tuning Stages and Results . . 56
5.3.2. CNN-LSTM Network Hyperparameter Tuning Stages and Results 58
6. EXPERIMENTS WITH UNSUPERVISED LEARNING ARCHITECTURES 62
6.1. K-Means 62
6.2. BIRCH 63
6.3. DBSCAN 64
6.4. Experimental Results & Discussion 65
7. CONCLUSION 69
7.1. Future Work 70
REFERENCES 72
APPENDIX A: COPYRIGHT PERMISSION GRANTS 81

LIST OF FIGURES

Figure 2.1. Representation of the hypothalamic-pituitary-adrenal(HPA) axis . 6

Figure 2.2. PSS-10 Questions 16

Figure 3.1. Empatica E4 Smartband Physiological Data. Electrodermal Activity (EDA), Blood Volume Pressure (BVP), Accelerometer (ACC), Interbeat Interval (IBI) and Skin Temperature (ST) 21

Figure 4.1. The Overview of the Stress Detection System with Three Different Learning Model Types 28

Figure 4.2. PSS-5 Survey used for Ground Truth 31

Figure 4.3. Label Propagation Through Labeled Data Samples. 34

Figure 4.4. Autoencoder Architecture 35

Figure 4.5. Accuracy Scores of Different Classifiers After LP Algorithm 39

Figure 4.6. Visualization of Stress and Nonstress Classes via t-SNE 41

Figure 4.7. Learning Curves of Our Final Deep Autoencoder Model 44

Figure 4.8. Problematic Learning Curves of Experimental Autoencoder Model 46

Figure 4.9. Latent Representation of Stress and Nonstress Classes via t-SNE . 46

Figure 5.1. Simple Representation of Recurrent Neural Network 51

Figure 5.2. Simple Representation of LSTM 52

Figure 5.3. Our Architecture Design of the LSTM and CNN–LSTM Neural Networks 54

Figure 5.4. Performance Results of CNN-LSTM Network 61

Figure 6.1. Visualization of K-means Classification Result via PCA 64

Figure 6.2. Silhouette Scores for different number of clusters. 66

LIST OF TABLES

Table 3.1. Literature Review on Stress Detection via Supervised Learning Models 19

Table 3.2. Literature Review on Stress Detection via Semi-Supervised Learning Models 23

Table 3.3. Literature Review on Stress Detection via Unsupervised Learning Models 25

Table 4.1. The Sampling Frequencies of Empatica E4 Sensors 29

Table 4.2. Classification Report of Label Propagation Algorithm (Selected LP Kernel = kNN) 38

Table 4.3. Classification Report of Label Propagation Algorithm (Selected LP Kernel = RBF) 38

Table 4.4. RF Classifier's Accuracy Results for Variable *max_depth* and *n_estimators* Parameters 40

Table 4.5. Final Deep Autoencoder Model Summary with Parameters 43

Table 4.6. First Experiments of Deep Autoencoder Model Summary 45

Table 4.7. Classification Report of Logistic Regression Classifier(solver=lbfgs) 47

Table 4.8. Classification Report of Logistic Regression Classifier (solver=saga) 47

Table 4.9. Classification Report of RF Classifier(*max_depth*, *n_estimators*) . . 48

Table 4.10. Classification Report of MLP Classifier (default parameters) 48

Table 4.11. Parameter Grid via GridSearchCV 49

Table 4.12. Classification Report of MLP Classifier (Hyperparameterized) . . . 49

Table 5.1. Number of Trainable Parameters of LSTM Network 55

Table 5.2. Layers and Input Sizes of LSTM Network 56

Table 5.3. Number of Trainable Parameters and Training Time of Networks . 58

Table 5.4. Number of Trainable Parameters of CNN-LSTM Network 60

Table 5.5. Classification Results of LSTM and CNN-LSTM Networks 61

Table 6.1. Silhouette Score for dynamically varying number of clusters 66

Table 6.2. Runtime and Resource Utilization 67

Table 6.3. Accuracy Results of SL & SSL & UL Architectures 68

LIST OF SYMBOLS

C	Cluster Center
C_t	Cell State
$\hat{C}_t$	Cell Update
f_t	Forget Gate
h_t	LSTM Output
i	Graph Node
i_t	Input Gate
j	Graph Node
$J(p, C)$	Objective Function
o_t	Output Gate
p	Cluster Indicator
w_{ij}	Weights
T_{ij}	Transition Matrix
X	Entire Dataset
(x_n, y_n)	Labeled Data Samples
Y_N	Known Labels
Y_T	Unknown Labels

LIST OF ACRONYMS/ABBREVIATIONS

ACTH	Adrenocorticotropic Hormone
ACC	Acceleration
AG	Adrenal Gland
ANS	Autonomic Nervous System
ARC	Ames Research Center
AUC	Area Under Curve
Bi-LSTM	Bidirectional Long Short-Term Memory
BIRCH	Balanced Iterative Reducing and Clustering Using Hierarchies
BVP	Blood Volume Pulse
CE	Conformitè Europëenne
CF	Clustering Feature
CNN-LSTM	Convolutional Neural Network-Long Short-Term Memory
CRH	Corticotropin-Releasing Hormone
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DT	Decision Tree
EMA	Ecological Momentary Assesment
EMG	Electromyography
ECG	Electrocardiogram
EDA	Electrodermal Activity
EEG	Electroencephalogram
FCC	Federal Communications Commission
FCL	Fully Connected Layer
FGSR	Foot Galvanic Skin Response
GSR	Galvanic Skin Response
HGSR	Hand Galvanic Skin Response
HPA	Hypothalamic-Pituitary-Adrenal
HR	Heart Rate
HRV	Heart Rate Variability

IBI	Interbeat Interval
kNN	k-Nearest Neighbor
LDA	Linear Discriminant Analysis
LED	Light-Emitting Diode
LR	Logistic Regression
LSTM	Long Short-Term Memory
MLP	Multilayer Perceptron
MRI	Magnetic Resonance Imaging
NASA	National Aeronautics and Space Administration
NASA-TLX	Nasa Task Load Index
OGSSL	Optimal Graph Coupled Semi-Supervised Learning
PCA	Principal Component Analysis
PG	Pituitary Glands
PNS	Parasympathetic Nervous System
PILAE	Pseudoinverse Learning Algorithm Based Autoencoder
PPG	Photoplethysmograph
PSS	Perceived Stress Scale
PTSD	Post-Traumatic Stress Disorder
RDSR	Robust Discriminative Sparse Regression
ReLU	Rectified Linear Unit
RESP	Respiration
RF	Random Forest
RLSR	Rescaled Linear Square Regression
ROC	Receiver Operator Characteristic
UL	Unsupervised Learning
SCR	Skin Conductance Response
semiLSR	Semi-supervised Linear Square Regression
semiPCAN	Semi-supervised Projected Clustering with Adaptive Neighbors
semiSVM	Semi-supervised Support Vector Machine
SNS	Sympathetic Nervous System
SRAD	Stress Recognition in Automobile Drivers

SS	Silhouette Score
SSL	Semi-Supervised Learning
SSR	Sympathetic Skin Response
SSRS	Stress Self-Rating Scale
SL	Supervised Learning
ST	Skin Temperature
STAI	State-Trait Anxiety Inventory
SVM	Support Vector Machine
TSS	Trier Social Stress
WHO	World Health Organization

1. INTRODUCTION

One of the generally accepted definitions of stress describes it as the effects of environmental demands on the organism at a level that may cause physiological or psychological illnesses [1]. Schneiderman et al. [2] examined the relationship of stress with human physiology, psychology, and mental health. They explained in detail how stress is associated with illnesses such as cancer, immune system problems, cardiovascular diseases, personality disorder, post-traumatic stress disorder (PTSD), and depressive symptoms. Many internal and external factors such as traumatic memories, long-lasting illnesses, family problems, inconvenience of the workplace environment, economic concerns, and the recent Covid-19 pandemic increase the stress level of individuals in the society. Vital processes such as decision making, communication quality with the social environment, and mental health are seriously damaged on the way to treatment.

World Health Organization (WHO) has defined stress as the health epidemic of the 21st century [3]. It is clear that such an important issue, affecting human mental health in every aspect, will have different economic and social consequences. According to the Global Organization for Stress, 80% of American workers experience stress in their work environment, 442.000 British workers believe they are ill due to the stress they face in the work environment [4]. According to the American Institute of Stress, 63% of American employees say they are ready to quit their job because of stress at work [5]. It is also necessary to take into account the workers who do not dare to leave the job but are inefficient due to stress. In the literature, the inefficient workforce created by this type of employee is called "Presenteeism" [6]. The negative economic impact it will create around the world should be taken into account.

When evaluated from many perspectives, it is evident that the stress research needs to be addressed more broadly. In the field of clinical psychiatry, different studies are carried out for the diagnosis and treatment of this problem [7]. Computer scientists also develop smart sensor technologies and machine learning algorithms for identify-

ing stress. Electroencephalogram (EEG) electrodes [8], electrocardiogram (ECG) and electromyography (EMG) electrodes [9], ECG harness [10], both ECG and galvanic skin response (GSR) electrodes [11], photoplethysmography (PPG), smartphones [12] and smartwatches [13] are preferred as wearable sensors. With the help of these wearable smart devices, data directly related to stress such as movements, brain, muscle, heart and electrodermal activities can be collected. There are also studies that focus on image, video, and speech data with the help of camera setups and microphones without using wearable sensors. However, multimedia sensors are not preferred due to privacy concerns and incompatibility with unrestricted environments. In addition to the experimental setups used, the conditions and environment of the experiment are also very important. Different experimental environments and setups are designed to monitor stress: Restricted, Semi-Restricted, Unrestricted Environments. Considering the results of the research, the effective use of sensors and the selection of the appropriate experimental environment are very vital. The results obtained by exposing the subjects to stress in restricted and semi-restricted environments are weak in terms of feasibility and generalizability. For this reason, it seems appropriate to follow the subjects in their daily lives with unobtrusive wearables.

1.1. Motivation

Tracking subjects' physiological data continuously and labeling this data for specific periods brings new problems to the surface: Noisy and Distorted Signals, Data Fusion Requirements, Battery Life, and Label Collection Process. Among these problems, we focused to the Ground Truth Collection and Label Collection Processes. Although the subjects in restricted and semi-restricted environments are reminded by the researchers for the labeling processes, it becomes a bigger problem in the unrestricted environment. Collecting tests and questionnaires from the subjects at certain hourly intervals in a 24-hour daily life creates difficulties in many ways. Expecting people to make stress assessments at work, school or dinner reduces the quality of labels. On the other hand, it should not be forgotten that the success of supervised learning models comes from the labels used as the ground truth in the dataset. Despite our costly

and resource-consuming methods of obtaining the ground truth, we should also consider that the labels collected with the help of surveys, questions, and reports reflect personal experience. Stress ground truths that the subjects presented to us with the help of questionnaires can create unique situations. For instance, two subjects with the same physiological signals may evaluate the situation as stressful or non-stressful based on their own experiences [14]. Therefore, this adherence of supervised models to the label also limits the overall performance and generalizability of machine learning models. After assessing the challenges encountered in stress sensing classification systems, that can work with less labeling burden will be preferred. One solution for this labor-intensive process is to use with Semi-Supervised Learning (SSL). Establishing an architecture that learns from a small number of labeled data can reduce the labeling burden. In particular, the fact that SSL techniques have not been investigated in depth by using multi-sensor physiological raw data in the literature creates room for improvement.

1.2. Contributions

14 different subjects were tracked continuously for one week with the Empatica E4 smartband. Multi-sensor raw data was carefully collected and prepared for use in SSL models. When working with SSL models, the results of reducing the labeling burden in terms of performance were examined. In order to make this examination clear, a comparison was made with implemented Supervised/Unsupervised Learning models. In light of the evaluations we have made so far, we are the first to implement SSL models with raw multi-modal physiological sensor data collected in unrestricted everyday life and compare it with Supervised and Unsupervised models. The progress of our work is as follows:

- Multi-sensor physiological raw dataset is prepared. The data is preprocessed according to the data fusion techniques and input shape of models.
- An unobtrusive data collection setup has been designed. Thus, the subjects could be followed independently of heavy, static sensors that can only be used in the laboratory. On the other hand, this system also allowed the experiment to be

carried out at a much lower cost.

- The subjects were not in any laboratory or controlled experimental environment. The data were collected in an unrestricted environment in daily life.
- The SSL algorithms we have designed are applied and performance metrics are obtained.
- State-of-the-art supervised learning-based deep learning architectures are implemented, and performance metrics are obtained.
- Unsupervised learning clustering algorithms are applied, and performance metrics are obtained.
- By evaluating the performance metrics, the advantages and the shortcomings of the presented SSL models are expressed.

1.3. Thesis Outline

The remaining chapters of the thesis are as follows. In Chapter 2, the formation of physiological stress, its origins, and stress signals are mentioned. Thus, the concepts are clearly expressed before moving on to the methods proposed in the thesis. Supervised, Semi-Supervised, and Unsupervised Learning architectures used for stress classification in the literature are discussed in Chapter 3. In Chapter 4, we explained our proposed SSL architecture and data collection setup as a solution to the labeling problem of the Stress Classification Problem. We implemented supervised learning and unsupervised learning architectures to see how successful SSL architectures are in terms of performance while eliminating the labeling burden. We explained these SL and UL architectures in Chapter 5. In Chapter 6, the performance results of all models will be compared and inferences will be made about SSL models. In Chapter 7, the conclusions and future work will be given.

2. BACKGROUND

Stress detection systems are formed by the combination of different components and functionalities. A clear definition of these sub-components is crucial to understanding the overall architecture. In this chapter, we will begin to discuss the physiological formation and origin of stress. Stress seems like an abstract concept even though we hear it a lot in our daily life. Expressing stress in a physiological sense will help to understand other sections in this chapter. When our perspective on the concept of stress becomes clear, we will talk about the physiological indicators that allow us to perceive stress, then the stress perception tests based on the experiences of the individuals, the stress prediction experimental environments, and finally the challenges experienced while creating these systems.

2.1. Stress and Its Origin

The word "homeostasis" is derived from the Greek words homeo and statis. When these two words, which mean same and steady, are combined, it means to stay stable. The word was first used by Walter Bradford Cannon in the 1920s-1930s [15]. It refers to the survival of organisms in the face of changing conditions and external factors. In the 1960s, Hans Selye defined "stress" as factor that could disrupt homeostasis and affect it negatively [16]. He later expanded his research on factors that stress organisms and the response of organisms to these stressors. After these first definitions, the Hypothalamic-pituitary-adrenal (HPA) axis system was revealed, and it focused on the body's response to stress. This neuroendocrine system, which is based on a feedback mechanism, is essential for us to cope with stress physiologically. The HPA axis consists of three parts: the hypothalamus, pituitary glands (PG), and adrenal glands(AG). With the corticotropin-releasing hormone (CRH), the hypothalamus stimulates PG. PG, which secretes adrenocorticotropic hormone (ACTH), activates the adrenal gland (AG). As a result, the cortisol level in the body increases. Thanks to the feedback mechanism, the HPA axis returns to its normal function as the stress factor decreases.

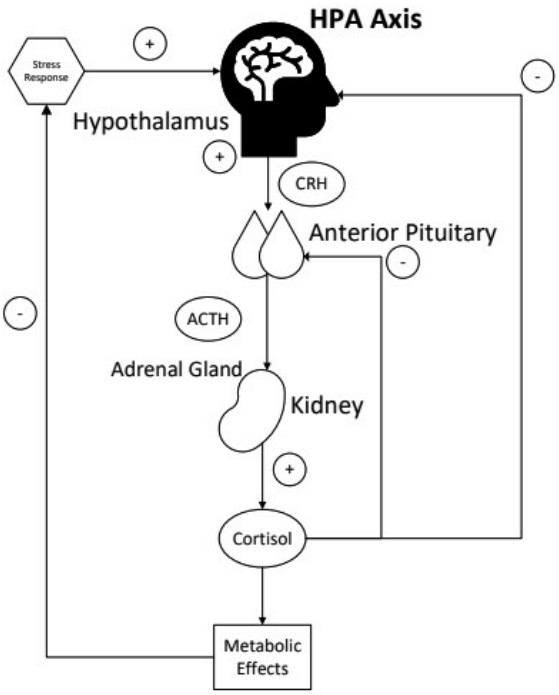

Figure 2.1. Representation of the hypothalamic-pituitary-adrenal(HPA) axis

So what are the results for people who are constantly exposed to stress or who are exposed to chronic stress? Chronic stress refers to the long-term exposure of a person to stress. For example, problematic carriers, severe chronic diseases, and problems with family in life can cause chronic stress. Since the HPA axis will be continuously stimulated in people experiencing chronic stress, a high amount of cortisol will be secreted into the blood via AG. Excessive cortisol release in the body has serious physical and mental consequences. The feedback mechanism of the HPA axis can be seen in Figure 2.1.

The situation is slightly different in acute stress, which is another type of stress. It occurs as a result of stressors that we are exposed to for a shorter time in daily life. Final exams, project deadlines, and short-term discussions are examples of acute stress. Similar to the previous process, the HPA axis is activated again, and the hormones are activated in the neuroendocrine system, but since it is not continuous, the cortisol level in the body returns to its original state. An adequate amount of

cortisol hormone secreted in the face of acute stress is very important for the body's fight-flight-or-freeze response (also known as acute stress response). In research on the acute stress response, the autonomic nervous system (ANS) is vital and should be well understood. ANS is formed by the combination of the sympathetic nervous system (SNS) and the parasympathetic nervous system (PNS). The diffusion of stress-related hormones within the neuroendocrine system occurs with the help of the SNS and PNS. The amygdala, which processes emotional stimuli in brain, triggers the HPA axis by stimulating the hypothalamus against external factors that cause stress. In addition to the above-mentioned cortisol diffusion, epinephrine (adrenaline hormone) is also secreted in the body through AG. The SNS assists in the release of cortisol and adrenaline hormones. The blood is pumped to the muscles, the volume of the lungs increases [17], the heartbeat accelerates, the digestion slows down or even comes to a halt, and the blood pressure rises. Basically, SNS puts the person in a form that can fight against the threats perceived by the different senses. PNS, on the other hand, helps restore body functions after the threat is gone. It is clear how crucial acute stress response is in threatening moments. This response, which affects our decisions to survive, get injured, and to fight, not only protects us but also helps us have fun. For instance, extreme athletes enjoy the situation they are in, even if the surrounding conditions are acutely stressful. Here, stress is far from being chronic and is short-lived.

2.2. Stress Signals

The autonomic nervous system(ANS) enables the body to create a physiological response to stress through sympathetic and parasympathetic nerves. In this case, if researchers follow specific physiological changes that occur in the human body, they obtain essential data regarding stress prediction. In addition, we are able to obtain physiological and psychological outputs thanks to developing sensor technologies, new generation imaging techniques, and psychological tests. Stress-associated signals can be obtained with complex devices such as magnetic resonance imaging (MRI), EEG, EMG, ECG, and customized electrodes. However, 3-axis accelerometer, infrared thermopile, galvanic skin response (GSR), and photoplethysmography (PPG), which are more small

scale sensors, are also used. There are also studies on stress in areas such as image processing, signal processing, and computer vision. These areas generally work with the subjects' audio and video data by using microphones, cameras, and voice recorders. When the subjects are exposed to stress, changes in voice tone and facial expressions can be followed in video and audio recordings. However, studies on mimic and voice may be biased because individuals have different ways of coping with stress. For example, a person may prefer to laugh in a very stressful moment, or he can hide the expression on his face in a different way. Considering these situations, designing an experiment provides more reliable results.

The recent increase in the use of smartphones and tablets has revealed a new approach. Personal identification can be made by keystroke dynamics [18]. Keyboard and mouse usage dynamics of people, frequency, and intensity of touching the screen while using a tablet can also be studied as stress signals.

Finally, people can verbally express their stress levels. This form of verbal expression, called "perceived stress" in the literature, is obtained by questionnaires, surveys, and self-reports. The point to be noted here is that these reports are personal. Factors that cause stress for someone may not be a problem for another.

2.2.1. Brain Signals

Researchers in clinical stress studies prefer EEG signals. Primarily as power spectrum features, delta (0.5-4 Hz), theta (4-8 Hz), alpha (8-12 Hz), beta (12-30 Hz), and gamma (above 30 Hz) bands are used. These frequencies show different functionalities in the stress response. Also, the fractal dimension feature is used to understand the complexity and irregularity properties of time-series signals [19]. When using an EEG device, it is necessary to determine the correct number of channels. Thus, the accuracy and quality of the signals obtained are not compromised. Recently, researchers have started to use EEG devices with more unobtrusive, wearable, and wireless designs.

2.2.2. Heart Signals

The heart continues to work with the help of electrical impulses it produces. Electrical signals are generated in a part of the heart called the sinus node (SN). This signal is then transmitted to different parts of the heart. The devices in which impulses are monitored with the help of electrodes are called an electrocardiogram (ECG). ECG devices, which usually consist of 4 electrodes, can be placed on the body differently. Chest, shoulder heads, both arms and legs are the parts used to obtain heart signals. Accurate acquisition of signals is ensured by using gel between the ECG electrodes and the body connection points. So ECG is usually used with wet electrodes. However, as a result of increasing technological developments, new sensor technologies have allowed us to obtain heart signals in different ways. Photo-Plethysmography (PPG) is a very light and portable sensor that can work dry. It has also found a wide range of use in smartwatches and wristbands. Thanks to the LED inside, it sends light into the tissue and examines the degree of absorption with a photosensitive sensor. It is widely used in pervasive health applications. Heart signals consist of heart rate variability (HRV), heart rate (HR), and RR intervals (IBI). When the studies in the literature are examined, HRV is seen as an informative signal in stress studies [20]. Studies have shown that HRV is associated with post-traumatic stress disorder (PTSD) [21]. SNS and PNS directly regulate heart rate. Therefore, HRV provides an important biofeedback in stressful conditions.

2.2.3. Muscle Signals

Our muscles are stimulated through the nervous system. This creates an electrical potential just like in the heart. Electromyography (EMG) measures this electrical potential in muscles. Signals are collected by placing electrodes on muscle groups. Luijcks *et al* [22] showed the relationship between stress and muscle stimulation by looking at the Mean EMG results in the baseline, pre-stimulus, and post-stimulus periods.

2.2.4. Electrodermal Activity(EDA)

Electrodermal activity (EDA) can be expressed as the change of electrical properties of human skin against certain factors. Researchers have named this electrical characteristic of the skin as galvanic skin response (GSR), skin conductance, skin conductance response (SCR), or sympathetic skin response (SSR). Electrical conductivity varies depending on perspiration on the skin surface. With the help of the GSR sensors, skin electrical conductance change can be detected. In the literature, ECG signals, HRV, and EDA are highly preferred biomarkers for identifying stress [23].

2.2.5. Blood Volume Pulse(BVP)

Blood volume pressure signal is obtained with the PPG sensor. PPG is a non-invasive optical sensor. With the green and red LED light sources used in the sensor, the variability in blood flow is tried to be measured. Experiments have also been carried out in the finger, toe, and ear lobes, which are the regions with high vascular density in the literature. Alternatively, studies have been published recently on the hand and wrist.

2.2.6. Acceleration Data

The rate of change in the velocity of the subject is obtained by acceleration sensors. They collect data in a 1-axis, 2-axis or most commonly 3-axis. It is frequently used in research on motion and vibration measurement. The relationship between emotional change and body movements has been demonstrated by Ekman *et al.* [24].

2.2.7. Skin Temperature(ST)

Body temperature is a significant indicator for diagnosing diseases and understanding the course of treatment. In addition, the body temperature must be stable within a certain range for the body functions to work correctly. To understand the body temperature, we usually measure it from the skin surface with the help of a ther-

mometer. Since the body temperature is directly controlled by the nervous system in mammals, studies have been conducted on its relationship with stress. It has been showed that there is a temperature change on the skin surface with sympathetically mediated vasoconstriction in a person experiencing acute stress [25, 26].

2.2.8. Speech Signals

Speech is a type of signal that carries essential personal information. Besides its meaning, it has paralinguistic features. These features are body language, facial expression, emotion, dialect, and accent [27]. On the other hand, it has features such as tone of voice, emphasis, pause, and breathing pattern. Using these features, stress detection studies are carried out from the speech signal [28].

2.2.9. Facial Mimicry

Developing technologies in the fields of computer vision and image processing provide important opportunities to understand people's emotional changes from the expression on their faces [29]. Similarly, with the help of advanced lenses and cameras, the mimic variations of people can be displayed better. Studies on stress in this area should be scrutinized. Because people can hide their emotions through their facial expressions.

2.2.10. Keyboard and Mouse Usage

Physiological user authentication and personality characterization studies were carried out using keyboard and mouse usage dynamics [30, 31]. As can be understood from these studies, the keyboard and mouse usage dynamics of people are different from each other; it is possible to use this information as a personal signature. Further studies have shown that when people are stressed, pressing the keyboard keys and clicking the mouse becomes more intense [32].

2.3. Stress Experiment Environments

Stress research has been carried out in many different scenarios and environments since its inception. Experimental designs and research outputs played an substantial role in determining these environments. It also affected the setup of the experiments in the areas where the researchers came from. For example, psychology and psychiatry researchers mostly preferred clinical settings. In fact, studies in these fields have influenced many computer scientists, and controlled experiments have been put forward. However, later studies wanted to construct more realistic experimental scenarios. This has accelerated the shift of research from controlled and restricted environments to semi-restricted environments. Although researchers gather more challenging data, they are moving towards non-restricted environments because these experiments are more suitable for daily life.

2.3.1. Restricted Environments

A specific environment for the experiment is determined, and the experiment is carried out under those conditions. For example, studies were carried out in an office, inside a car, or laboratory. The general motivations for experiments conducted in the office are to measure the stress that employees are exposed to through their workload. With the help of special sensors and cameras, the stress experiences of individuals can be measured more easily. For example, some researchers collected and labeled the physiological data of the subjects under special conditions such as important meetings or project deadlines. Today, the rate of car usage is increasing day by day depending on the population rate. This means that traffic accidents are increasing day by day. Fatal accidents occur for many different reasons such as stress-related inattention, sudden change of decision, and anger. Researchers also use the car as an experimental environment to prevent such accidents. With the help of wearable wristbands, portable ECG devices, and cameras, research was carried out on the stress status of drivers [33]. The limitation of the driver's movements and the collection of stress data in a city with heavy traffic create a limited experimental environment.

2.3.2. Semi-Restricted Environments

It is an environment in which subjects interact with different factors more than they would in a restricted environment. Constraints are less than restricted environments. Experiments carried out in the university environment can be given as an example of these environments [34]. Subjects tracked at the university can sometimes be found in restricted environments such as classrooms or lecture halls. However, the same subjects have the right to roam around the campus. They can also be found in different areas within the university, such as a gym, cafe, or stadium. Since this creates a less constrained environment, they are called semi-restricted environments. It is an essential step in the transition to daily life experiments and produces results closer to non-restricted environments.

2.3.3. Non-Restricted Environments (Daily Life)

Non-restricted environment is based precisely on the monitoring of subjects in daily life. Unobtrusive wearable devices are used. In this way, the experimental environment is set up in an unconstrained manner. Since it is more challenging to obtain data in these wild environments, there are fewer studies in the literature [35, 36]. Seeing the gap, we focused our research on this experimental environment. In addition to wearable designs, there are also research made with mobile phones. Since people exposed to stress follow specific patterns in terms of mobility, research has also progressed in that direction.

2.4. Stress Data Collection Challenges

In this thesis study, the research was completed by collecting daily life data. However, collecting daily life data poses many challenges. These challenges should be carefully studied, and the experimental setup should be designed accordingly. Otherwise, a decrease in the overall system performance may be observed.

2.4.1. Noisy and Distorted Signals

It mainly occurs due to insufficient quality of the sensors, sensor positioning error, and sudden movements. Processes such as filtering and artifact removal ensure that the signal is free of noise.

2.4.2. Data Fusion Principles

Since there can be more than one sensor on the wearable devices, multi-sensor raw data is obtained. The synchronization and integration of this data must be done with precision. For this, the sampling frequency of the sensors and the timestamps of the sensor data must be used correctly.

2.4.3. Unobtrusive Design

Complex sensors and imaging devices are used in clinical settings. However, these designs remain heavy and static for pervasive health applications. Unobtrusive and wearable devices are more advantageous in terms of fast sensing, networking, connectivity, and data fusion [37]. In addition, individuals who have problems such as stress and anxiety are not willing to come to the clinical environment when they are going through difficult periods. To solve this problem, researchers try to expose the subjects to stress in different experimental scenarios, but this poses a problem in terms of reality. As a result, it will be helpful to follow the subjects in daily life with an unobtrusive design.

2.4.4. Battery Life

Wearable devices are all battery-dependent. It is inevitable that the sensor battery will run out in studies where subjects are continuously followed in daily life. The important thing here is that the researchers can set up the battery charging process without interrupting the data collection and prevent the time gap between the data.

2.4.5. Data Labeling Process

Conventional machine learning algorithms need ground truth when seeking solutions for tasks such as classification or regression. Especially during supervised learning-based training, the target variable must be given to the models as the input. In such cases, labeling the data requires serious effort and resources. Considering today's deep learning and machine learning problems, even the smallest recommendation systems require large amounts of data and ground truth labels. Robust labeling principles should be used to see the success of models accurately. In the stress domain, questionnaires such as NASA-TLX [38], Perceived Stress Scale (PSS) [39], and State-Trait Anxiety Inventory (STAI) [40] are preferred.

2.5. Questionnaires and Tests for Stress

It was stated in Section 2.4 that one of the most critical challenges of stress estimation research is the ground truth collection. Physiological data is easier to label in controlled experimental environments. For example, in studies on the stress level of students taking the exam, it is accepted that the physiological data represents the stress class during the exam. However, as research moves towards less controlled experiments and even daily life data, it becomes vital to collect compelling ground truth. Researchers have designed many questionnaires and stress tests with different features to solve this problem.

2.5.1. Perceived Stress Scale (PSS)

Perceived stress is the verbal or written expression of people's feelings, thoughts, and perceptions in the face of the stressor. The Perceived Stress Scale (PSS) is widely used to measure perceived stress in the literature. Cohen *et al.* published PSS in 1983 to measure perceived stress universally [39]. Correlation studies between PSS and stress measurement were also published. PSS wants to analyze the stress created by the uncontrollable and unpredictable parameters in life with questions. Although some of the questions in the test are similar to each other, it is basically aimed to measure

a) In the last month, how often have you been upset because of something that happened unexpectedly?

b) In the last month, how often have you felt that you were unable to control the important things in your life?

c) In the last month, how often have you felt nervous and stressed?

d) In the last month, how often have you felt confident about your ability to handle your personal problems?

e) In the last month, how often have you felt that things were going your way?

f) In the last month, how often have you found that you could not cope with all the things that you had to do?

g) In the last month, how often have you been able to control irritations in your life?

h) In the last month, how often have you felt that you were on top of things?

i) In the last month, how often have you been angered because of things that happened that were outside of your control?

j) In the last month, how often have you felt difficulties were piling up so high that you could not overcome them?

Figure 2.2. PSS-10 Questions

the perceived stress consistently. It is based on the emotional changes and thoughts that the individual has experienced in the past month. It is evaluated on a scale from zero to four. It has different variations like PSS-10 [41] or PSS-5. You can see the PSS-10 questions in Figure 2.2. The PSS score prepared according to these questions gives precious information about the individual. For example, a high PSS score may indicate severe depression due to stress, difficulty quitting smoking, or diabetes due to changes in blood sugar.

2.5.2. NASA Task Load Index (NASA-TLX)

NASA-TLX is a subjective mental assessment for reporting the workload perceived by individuals. It emerged from theoretical research conducted at NASA Ames Research Center (ARC) in 1988 [38]. While determining the workload score, it uses the following six weighted averaged subscales:

- Mental Demand,
- Physical Demand,
- Temporal Demand,
- Performance,
- Effort,
- Frustration.

The first tests with pen and paper have now been moved to more technological environments such as computers and mobile apps. The fact that it can be used as software provides excellent benefits in terms of fast analysis and reporting. It has found a wide range of uses in different fields, from aircraft cockpit to process production and control areas.

2.5.3. The State-Trait Anxiety Inventory (STAI)

STAI is a measurement method used for anxiety disorder and trait anxiety problems [40]. It is instrumental in clinical studies of depression or studies on anxiety. Through different answers, results on trait anxiety and state anxiety are obtained. Answers are evaluated on the same 4-point scale used in the PSS test. The following answers are used to measure trait and state anxiety:

- I worry too much over something that really doesn't matter.
- I am content.
- I am a steady person.
- I'm tense.
- I'm worried.
- I feel calm.
- I feel secure.
- I feel at ease.
- I am presently worrying over possible misfortunes.
- I feel satisfied.

- I feel frightened.
- I feel indecisive.
- I feel nervous.
- I have disturbing thoughts.
- I feel like a failure.
- I feel nervous and restless.
- I lack self-confidence.
- I am jittery.
- I feel content.

When the results obtained from the subjects are evaluated, a high STAI score indicates high anxiety.

3. LITERATURE REVIEW

The studies in the literature are branched out in terms of different sensor setups, experimental environments, and machine learning / deep learning techniques. From a general perspective, survey articles on stress classified the studies under the laboratory, restricted, semi-restricted, and unrestricted environments [42]. However, since we focused on the labeling problem in our research, it would be more advantageous for us to examine the literature studies under the Supervised, Semi-Supervised, and Unsupervised Learning models. In Chapter 2, we talked about the sensors and experimental environments used. In this direction, before we talk about our SSL model design, it is valuable to classify what kind of studies have been done on the Supervised Learning models.

Table 3.1. Literature Review on Stress Detection via Supervised Learning Models

Article	**Stress Sensor**	**Stress Signal**	**Stress Test**	**Environment**	**Unobstrusive**	**Number of Participants**	**Duration**	**Method**
Mozos et al. [43]	Electrode Wristband Sociometric Sensor	EDA PPG ACC Speech	TSST	Laboratory	No	18	17 Minutes	SVM kNN Adaboost
Seo et al. [10]	Zephyr Bioharness	ECG Respiration	Visual Analogue Scale(VAS)	Laboratory	No	16	1 Hour	CNN+LSTM SVM RF kNN LR DT
Garcia-Ceja et al. [44]	Samsung Galaxy SIII Mini Smartphone	ACC	Oldenburg Burnout Inventory (OLBI)	Office	No	30	8 Weeks	Naive Bayes DT
Can et al. [34]	Samsung Gear S1 S2 S3 Empatica E4	PPG EDA ACC ST	NASA TLX	University	Yes	21	9 Day	PCA+LDA PCA+SVM kNN LR RF MLP
Gjoreski et al. [45]	Empatica E4	BVP HRV ST EDA RR	Ecological Momentary Assessment (EMA) STAI	Laboratory Real Life	Yes	26	55 Days	SVM
Seo et al. [46]	Zephyr Bioharness HDR-CX450 Camcorder	ECG RESP Facial Expressions	Stroop Task	Laboratory	No	24	45 min	CNN-LSTM
Our Work	Empatica E4	EDA BVP ACC ST	PSS-5 Questionnaire	Daily Life	Yes	14	989 Hours	LSTM CNN+LSTM

3.1. Stress Prediction with Supervised Learning (SL) Models

Pioneering supervised learning studies were initiated by research groups in which the stress levels of subjects were closely monitored in laboratory settings. In these controlled environment studies, results were obtained using different sensors, signals, stress factors, stress tests, experiment durations, and different conventional supervised learning algorithms. Table 3.1 shows the summary of the studies in the literature and our SL implementation. Mozos *et al.* [43] have established a robust laboratory environment by combining physiological EDA and PPG sensor data with speech and accelerometer data. They shared their experimental results with the controlled trier social stress test (TSS), SVM, k-nearest neighbor, and AdaBoost classifiers. Accordingly, the AdaBoost algorithm gave the best accuracy performance with 94% for two-class classification (Stressful and Neutral Situation). In another lab study, Seo *et al.* [10] implemented different supervised models using ECG and respiration data. The data was collected with a wearable device called the Zephyr Bioharness. They tracked 18 subjects as they relaxed or solved different levels of math and stroop tasks. Time and frequency domain features are extracted from ECG and respiration signals. A solution to the binary classification problem was sought with CNN-LSTM based deep neural network. In addition, the results were compared in the experiment section by implementing conventional algorithms such as support vector machine (SVM), random forest (RF), k-nearest neighbors(kNN), logistic regression (LR), and decision tree (DT). According to accuracy, F1 Score, and area under the ROC curve (AUC) metrics, their CNN-LSTM network provided the best performance. They surpassed the studies using similar neural network architectures with an accuracy score of 83.9%. Further studies were carried out in restricted areas such as offices and cars. In the office environment, Garcia-Ceja *et al.* [44] reported an overall accuracy of 71% using smartphone accelerometer data and classifiers such as Naive Bayes and DT. Researchers continued to work on semi-restricted and unrestricted environments to make experiments more realistic. University campuses were mostly chosen as the semi-restricted environment. The reason for this is to be able to create a less intrusive experiment that is close to real life. Can *et al.* [34] conducted a study on the stress levels of university students in a summer camp. The data was collected from 21 university students participating

in a nine day algorithm competition. The designed three-class automatic stress detection system includes capabilities such as modality-specific artifact removal and feature extraction.

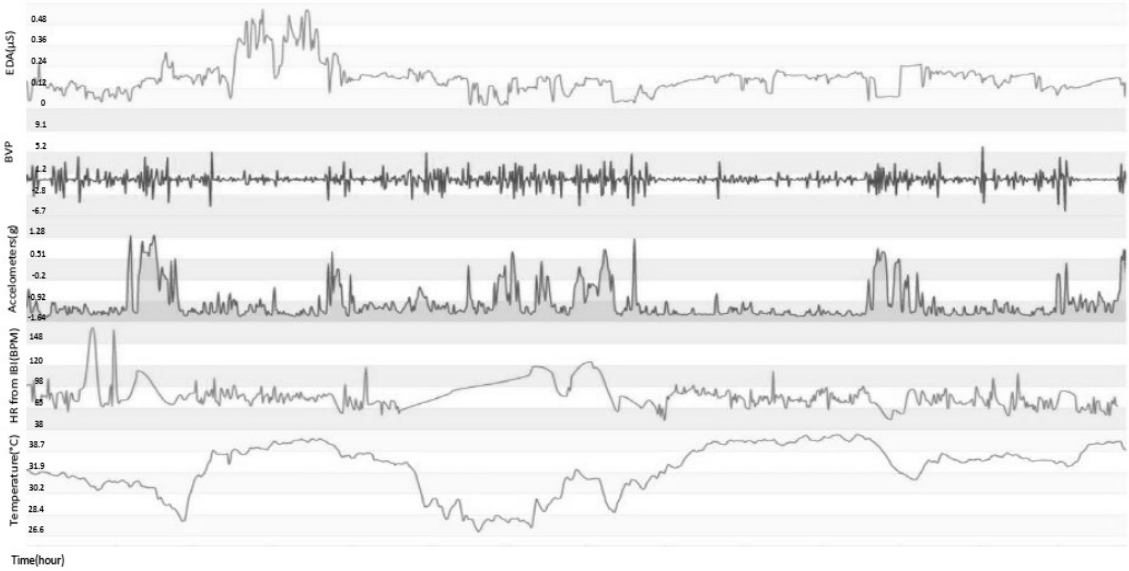

Figure 3.1. Empatica E4 Smartband Physiological Data. Electrodermal Activity (EDA), Blood Volume Pressure (BVP), Accelerometer (ACC), Interbeat Interval (IBI) and Skin Temperature (ST)

Electrodermal activity (EDA), photoplethysmography (PPG), skin temperature (ST) and the accelerometer (ACC) physiological signals were obtained using different smartwatches such as Samsung Gear S1, S2, S3, and Empatica E4. Two separate ground truth creation methods are applied; the first was the context labeling for three known classes. For example, the labels were determined that the students were very stressed during the competition, less stressed but still stressed in the lesson compared to the competition, and not stressed in their free time. Secondly, the NASA-TLX questionnaire was used. A maximum of 98% accuracy score was achieved with person-specific RF algorithm among supervised learning algorithms such as PCA+LDA, PCA+SVM, kNN, LR, RF, and MLP. In recent days, researchers have preferred unrestricted environments because they fully cover the stress factors in real life. Due to the unconstrained and uncontrolled conduct of the research, it creates a challenge in terms of ground truth gathering, and lower accuracies are obtained. Considering its challenges, stress detection in everyday life has been studied less than in other experimental en-

vironments. One of the few studies was carried out by Gjoreski *et al.* [45], following the subjects with the Empatica E4 smartwatch both in the laboratory and in real life. They carried out labeling with stress logs and the Ecological Momentary Assessment (EMA) prompt implemented on the smartphone. Different features are extracted using BVP, HRV, ST, EDA, and inter-beat (RR) intervals signals. Their model consists of three parts as activity recognizer, base stress detector, and context-based stress detector. While the base stress detector enables the extraction of features from the raw physiological data, the activity recognizer determines the activity of the subjects with the data received from the accelerometer. The last stress detector module performs two-class classifications every 20 minutes. In this module, the SVM classifier was trained with 55-day real life data, and an accuracy score of 92% was obtained. In their recently published study, Seo *et al.* [46] focused on the stress of workload. A sample dataset consisted of 24 healthy individuals and the experiments were carried out at the Pohang University of Science of Technology (POSTECH), South Korea. It was confirmed that the subjects did not have any heart disease and had not participated in the stress experiment before. In the experimental environment, they prepared as a GUI, the subjects' stress levels were increased or decreased in a controlled manner by giving them tasks with certain difficulties. Electrocardiogram (ECG), respiration (RESP), and video data were used in the controlled experiment performed in the laboratory. Stroop tasks were used to generate changes in the stress level. Using Zephyr Bioharness as a sensor, 1 kHz ECG signal and 25 Hz RESP signal were obtained. Subjects sitting in front of a Hewlett Packard laptop were followed with a HDR-CX450 camcorder at a resolution of 1280 $\times$ 720 and 30 fps. Preprocessed ECG, RESP, and facial expression features were fed to the neural network as input. They used 68 landmarks around the eyes, nose, and mouth while extracting the facial features. The models consist of two parts. First, physiological signals were processed with the CNN-LSTM model. In the second part, facial feature sequences are processed using Bidirectional Long Short-Term Memory (Bi-LSTM). After completing the feature level fusion, the highest accuracy score of 73.3% was obtained by using RESP, and facial landmarks. However, when using ECG, RESP and facial expression features, the accuracy score decreased to 54.4%, which means an almost 50% probability of correct stress classification result.

Although the study focuses on workload stress, the fact that the subjects were students and the experiment was far from the working environment makes the results limited.

Table 3.2. Literature Review on Stress Detection via Semi-Supervised Learning Models

Article	Stress Sensor	Stress Signal	Stress Test	Environment	Unobstrusive	Number of Participants	Duration	Method
Wampfler et al. [47]	Smartphone Smarttablet	Hand Movements	Self Reports	Laboratory	Yes	70	70 Minutes	Variational Auto-Encoder(VAE)
Lin et al. [48]	Garmin Smartband	Heart Rate HRV Stress Sequence Intervals	Surveys	Office	Yes	574	8 Months	Autoencoder + LSTM
Peng et al. [49]	62-Channel Electrode Cap	EEG	Videos	Laboratory	No	15	72 Minute	OGSSL
Our Work	Empatica E4	EDA BVP ACC ST	PSS-5 Questionnaire	Daily Life	Yes	14	989 Hours	Label Propagation Deep Autoencoder

3.2. Stress Prediction with Semi-Supervised Learning (SSL) Models

As researchers started to work with daily life data, even a few innovative SSL model trials began to work to optimize the labeling problem, as we discussed earlier. Although these studies are new, they differ in terms of the experimental environment, data type, and algorithmic models. At ETH Zurich University, Wampfler *et al.* [47] studied an experimental group of 70 undergraduate and graduate students. They exposed the subjects to different stress situations with their own 70-minute Skype messaging chat conversation in a controlled laboratory environment. While messaging was carried out with smartphones via Skype, they also collected self-reports with smart tablets. In order to predict stress, they produced the dataset as heat maps based on the intensity of movement on the touch screen of smartphones. Self-reports collected via the tablet consist of two parts. In the first part, they infer valence, arousal, and dominance scores. In the second part, they ask people to choose emojis that reflect their emotions. During the experiment, they were exposed to shocking, sad, rude, exciting, and confusing events in their communication. Variational auto-encoder was used as the SSL model, and low-dimensional embeddings were extracted from the 2D heat maps they created. Then, classification was made by giving low-dimensional em-

beddings as input to the fully connected layers. The network with fully connected layers consists of a pre-trained model that has been trained using labeled data. Accuracy and AUC scores for valence, arousal, and dominance were obtained by using heat maps of hand movements such as Pressure, Down-down, Up-down, and Combination obtained during the conversation. According to their performance evaluation, AUC metric results reported; valence achieved 84%, arousal 82%, and dominance 82%. In [48], Lin *et al.* studied multi-label human psychology anomaly detection (MBEAD framework). Multivariate temporal sensor data were consisting of minutely heart rate, minute-to-minute HRV signals, and three-minute stress sequence intervals were collected via Garmin bands. The encoder part consist of ReLu activation function, five CNN layers stack, reweighting mechanism with different size of kernels. The decoder part is symmetric to the encoder. The framework is completed with the relevance learning module and the LSTM network-based temporal relevance module. The obtained results were compared using seven different state-of-the-art frameworks. They detected the Affect, Stress, and Work Performance classes as anomalies and reported F1, Recall, and Precision metrics. The results show that better performance scores were obtained compared with the other seven studies. One of the recent SSL studies has focused on emotion recognition using EEG signals [49]. They proposed a model called Optimal Graph coupled Semi-Supervised Learning (OGSSL), which combines the concepts of adaptive graph learning and emotion recognition. The SEED-IV public dataset prepared by Shanghai Jiao Tong University was used. Different video clips were watched in three sessions to the subject group consisting of 15 healthy people. According to the video content, it was aimed to create four emotional states (sad, fear, happy and neutral) in the subjects. EEG data were sampled at 1000Hz with a 62-channel electrode cap. Their OGSSL model was compared with the Semi-supervised Projected Clustering with Adaptive Neighbors (semiPCAN) [50], the semi-supervised support vector machine (semiSVM), the semi-supervised Linear Square Regression (semiLSR), the Rescaled Linear Square Regression (RLSR) [51, 52], and the Robust Discriminative Sparse Regression (RDSR) [53] models were compared. The OGSSL model has an average of 76% accuracy score. Outperformed the benchmark models with an improvement of around 5%. When the literature is carefully examined, it is seen that SSL

architectures are used more widely in human action recognition and motion recognition fields [54–56]. Therefore, stress recognition is still a new field for SSL architectures. Table 3.2 shows the summary of the studies in the literature.

Table 3.3. Literature Review on Stress Detection via Unsupervised Learning Models

Article	Stress Sensor	Stress Signal	Stress Test	Environment	Unobstrusive	Number of Participants	Duration	Method
Wu et al. [35]	Empatica E4	EDA BVP	STAI	Laboratory	Yes	169	10-20 Minutes	K-Means
Wang et al. [36]	ECG Electrodes EMG Electrodes Respiration Sensor Skin Conductivity Sensor	ECG EMG GSR HR RESP	Comparative Questionnaire	Car	No	7	65-93 Minutes	Autoencoder+AdaBoost
Our Work	Empatica E4	EDA BVP ACC ST	PSS-5 Questionnaire	Daily Life	Yes	14	989 Hours	K-Means DBSCAN BIRCH

3.3. Stress Prediction with Unsupervised Learning (UL) Models

Table 3.3 shows the summary of the studies in the literature and our UL implementation. Wu *et al.* [35] set up a simulation environment for the experimental group of 169 fifth and sixth grade medical students in Presage Training Center. Although ACC, BVP, EDA, IBI, and ST data are collected with the Empatica E4 smartwatch, especially EDA and BVP signals are used. Subjects are called to collect 42, 43, 40, and 44 samples daily for 11 days in 4 different simulation rooms. The subjects deal with the patients by pretending to be doctors working in the clinic in a simulation environment. In this case, the created simulation environment expressed the stressed class, and the 2-3 minutes before the simulation formed the baseline class. Physiological signals are passed through segmentation, filtering, and feature extraction stages. K-Means clustering algorithm is trained with the obtained features. Algorithm success is measured using the Silhouette Score (SS). When the SS is examined by giving a different number of clusters to the algorithm, the highest performance result of 0.49 is obtained for two clusters (baseline and simulation). They compared the obtained results with some of the studies they went through in the related work section. However, these compari-

son results are far from benchmarking. In addition, this study, which proceeds on the assumption that the stress level of the subjects will increase in the simulation environment, is not realistic. However, it is a study that we examined in order to compare the Silhouette Score results of the K-means algorithm that we are using. Another unsupervised learning study conducted in a semi-restricted environment is to recognize the stress level of drivers [36]. Researchers preferred to use the previously prepared Stress Recognition in Automobile Drivers (SRAD) dataset. The experiment took place on the streets of Boston, Massachusetts. It comes out of different scenarios such as Rest, City-driving and Highway-driving. The SRAD dataset consists of different physiological signals such as EMG, ECG, GSR, HR, and RESP. The proposed model consists of two parts. The first is the Pseudoinverse Learning Algorithm based Autoencoder (PILAE) revealed in their previous research, and the other is the ensemble classifier using the AdaBoost algorithm. They reported the ROC curves of the three classes (Low, Medium, and High) classified for ECG, EMG, Foot Galvanic Skin Response (FGSR), Hand Galvanic Skin Response (HGSR), HR, and RESP. The most successful results were obtained in classification using the FGSR signal. They also extended their experiments with multiple signals obtained by fusion. In terms of performance metrics such as test accuracy and training time, better results were obtained compared with the results of the studies in the literature.

4. PROPOSED SEMI-SUPERVISED LEARNING ARCHITECTURES

In Chapter 1, we talked about how difficult to collect daily life stress data and obtain the corresponding ground truth. Based on this problem, we stated that SSL architectures would be a vital solution. SSL models provide significant advantages by working with a less amount of labeled data. The architecture we propose via SSL consists of two parts, as seen in Figure 4.1. The multi-sensor raw physiological data is prepared by fusion techniques, and then the raw data is used as input in our Label Propagation and Deep Autoencoder models. This section will detail our data collection principles, wearable device framework, ground truth collection, ethical consent, theoretical formulation, and implementation of SSL models.

4.1. Experiment Design

When the studies in the literature are examined, few studies are using daily life data. Stress detection studies are mostly concentrated in restricted environments and semi-restricted environments. We focused on non-restricted environments, as we saw the deficiency here and wanted to deepen the research. The primary purpose of experiments in non-restricted environments is to follow the subjects in their daily lives. Unobtrusive, light wearable designs are preferred because it is necessary for the subjects to continue their daily lives uninterruptedly.

4.1.1. Data Collection Unit - Empatica E4 Wristband

After analyzing the studies in the literature, we deduced that it is not realistic to use incommodious sensor designs in unrestricted areas. We decided to use the unobtrusive Empatica E4 smartband as a sensor to observe the stress situation of individuals under real-life conditions. With the help of its multi-sensor design, it can collect data such as BVP, EDA, ACC, and TMP. Wristband also can work with IOS

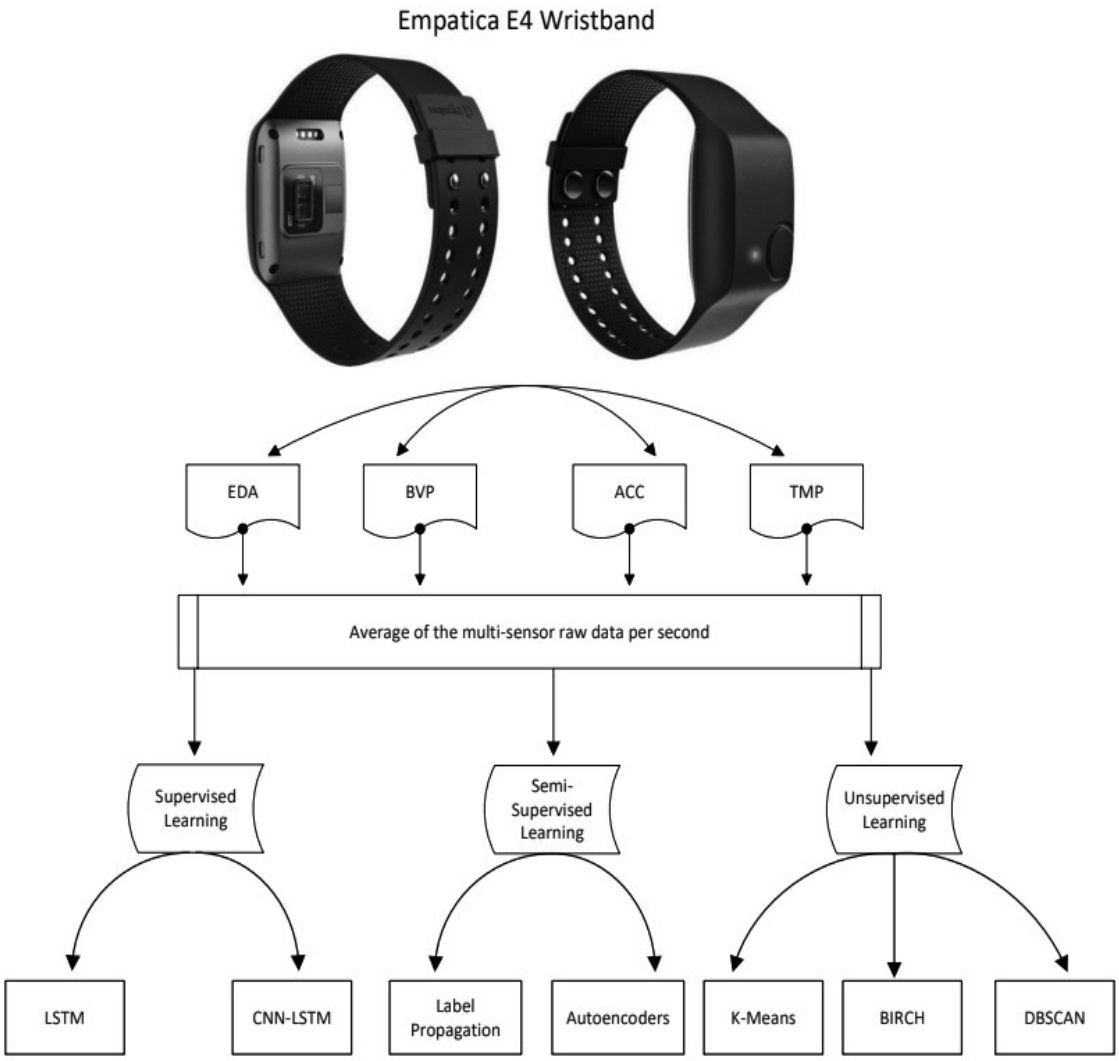

Figure 4.1. The Overview of the Stress Detection System with Three Different Learning Model Types

Table 4.1. The Sampling Frequencies of Empatica E4 Sensors

Physiological Signal	**Sampling Frequency**
EDA	4 Hz
BVP	64 Hz
ACC	32 Hz
ST	4 Hz

and Android operating systems, and send data to Empatica Cloud in real time. The obtained signals were used in the designed models without feature extraction. Signals with different sampling frequencies from multi sensors are synchronized with a python script according to start and end timestamps. Windowing or artifact removal methods were not used on the physiological data. Instead, the dataset was created by taking the averages of the BVP, EDA, ACC, and TMP data per second. The sample was created for that second by averaging 64 elements of the BVP signal with a sampling frequency of 64Hz. The exact process was applied to other physiological signals, considering their sampling. Frequencies are shown in Table 4.1. Since Empatica E4 smartband battery life supports 24+ hours in streaming mode and 48+ hours in memory mode, the subjects charged the watch every three days. While this charging process was taking place, users were requested to upload the collected data to the cloud. Since the subjects were followed entirely in their daily lives, they were not exposed to any test scenarios or restrictions. Similarly, they were not exposed to unrealistic stressors. The obtained dataset consisted of Subject IDs, Session IDs, used Empatica E4 Device ID, Timestamp, UTC Start Time, UTC End Time, and the Perceived Stress Score to synchronize with the physiological data.

4.1.2. Ground Truth Collection

The daily life experiment was carried out on 14 participants who were university students aged between 20 and 25 (Nine male and five female). Empatica E4 smartbands are given to all participants for one week. They were instructed to wear these

smartbands for twelve hours a day, between 9 a.m. and 9 p.m in their daily routine. These days were not specifically chosen consecutively. Ecological Momentary Assessments (EMAs) were collected. Thus, ground truth was gathered about the stress levels of the subjects. Participants are required to fill in the questionnaire every three hours. The three-hour intervals are called the session. To ensure the gathering of self-reports, reminder e-mails and questionnaire link were sent. Survey questions can be seen in Figure 4.2. A survey app was used to deliver the questionnaire to the participants. Finally, 989 hours of physiological sensor data and 332 EMAs obtained. More details about the data collection procedures and dataset can be found in the previous study of our research group [57].

4.1.3. Ethics

The procedure of the methodology used in this study was approved by the Institutional Review Board for Research with Human Subjects of Boğaziçi University with the approval number 2018/16. Prior to the data acquisition, each participant received a consent form, which explains the experimental procedure and its benefits and implications to both the society and the subject. The procedure was also explained vocally to the subject. The data collection procedure and all of the interventions in this research fully meet the 1964 Declaration of Helsinki [58]. All of the data are stored anonymously.

1-) How 'cheerful' were you in this period? *

	1	2	3	4	5	
Very low	○	○	○	○	○	Extremely

2-) How 'happy' were you in this period? *

	1	2	3	4	5	
Very Low	○	○	○	○	○	Extremely

3-) How 'Angry/Frustrated' were you in this period? *

	1	2	3	4	5	
Very Low	○	○	○	○	○	Extremely

4-) How 'Nervous/Stressed' were you in this period? *

	1	2	3	4	5	
Very Low	○	○	○	○	○	Extremely

5-) How 'Sad' were you in this period? *

	1	2	3	4	5	
Very Low	○	○	○	○	○	Extremely

Figure 4.2. PSS-5 Survey used for Ground Truth

[57]

4.2. Label Propagation Algorithm

Label propagation is an SSL technique based on the graph theory. In an approach where nodes represent data samples, edges represent the similarity between nodes. Propagation provided through nodes with known labels enables unlabeled nodes to turn into labeled nodes similar to them [59]. The red and blue samples that appear in Figure 4.3 actually illustrate the less amount of labeled data the LP algorithm uses. Classes are tried to be obtained by propagation technique between labeled samples (red, blue) and unlabeled samples (white).

4.2.1. Theoretical Formulation and Preliminaries

The LP algorithm was developed by Zu and Ghahramani [60]. The mathematical formulation of this study express the labeled data as

$$(x_1, y_1)...(x_n, y_n), where \quad Y_N = (y_1...y_n) \in \{1...C\}, \tag{4.1}$$

where Y_N refers to the labels for the different classes we have available in our dataset. While C expresses the number of classes, it is accepted that there are samples from each class in the dataset. Unlabeled data and the entire dataset are expressed as

$$(x_{n+1}, y_{n+1})...(x_{l+t}, y_{l+t}), where \quad Y_T = (y_1...y_t), \tag{4.2}$$

$$X = \{x_1...x_{l+t}\} \in R^D. \tag{4.3}$$

The ultimate goal is to estimate Y_T using X and Y_N. A fully connected graph infrastructure has been designed to solve the problem. While each data sample in the dataset represents a node, the relationship between nodes is weighted using the Eu-

clidean distance. The weights can be expressed mathematically as

$$w_{ij} = exp\left(-\frac{d_{ij}^2}{\sigma^2}\right), \tag{4.4}$$

where labels are propagated to all unlabeled data samples over the edges determined by weights. σ is used as a control parameter for calculating weights. The probabilistic transition matrix is used to assign the labels correctly. Thus, the labels of nodes can be updated over the edges with higher weights. The matrix can be expressed as

$$T_{ij} = P(j \rightarrow i) = \left(\frac{w_{ij}}{\sum_{k=1}^{l+t} w_{kj}}\right). \tag{4.5}$$

The probability of transition from node j to node i is calculated through the T_{ij} matrix defined in $(l+t)(l+t)$. As can be seen from the formula, this calculation changes according to the weight of the edges between the nodes. Matrix Y, whose size is defined as $(l+t)C$ depending on the number of classes, keeps the label probabilities of the nodes. The working principle of the algorithm consists of three basic steps:

(i) All nodes propagate labels using the probabilistic transition matrix Y.
(ii) Y matrix rows are normalized to provide the class probability interpretation.
(iii) Run Step 2 until Y converges.

4.2.2. Algorithm Implementation

The algorithm is implemented using the scikit-learn library [61]. The strategy followed during implementation is as follows:

(i) Split the dataset into training and test sets.
(ii) Split the training dataset into labeled and unlabeled sets.
(iii) Predict the labels of unlabeled samples with the label propagation algorithm.
(iv) The pseudo-labels, which are the outputs of the Label Propagation algorithm, are replaced with unlabeled samples in training dataset.

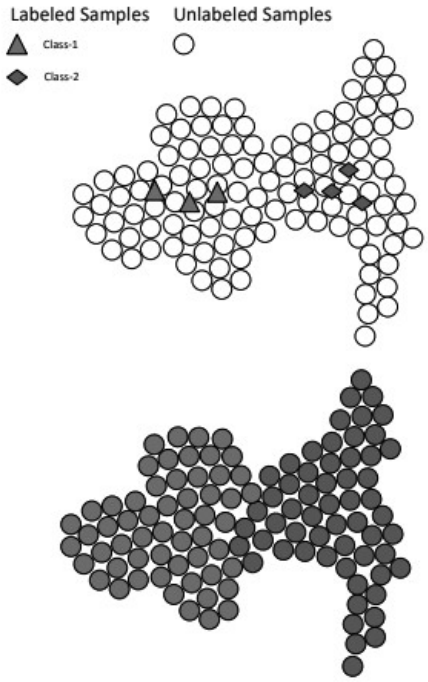

Figure 4.3. Label Propagation Through Labeled Data Samples.

(v) Training the classifier with the new augmented dataset of labeled and pseudo-labeled samples.

(vi) Use this model to predict test data.

4.3. Autoencoder Architecture

Autoencoder studies were first published in 1986 [62]. They proceeded with the perspective of unsupervised learning to learn about the internal representation of the data. Basically, the input, which is encoded with the help of a neural network, is tried to be reconstructed by extracting the informative parts of the data.

4.3.1. Theoretical Formulation and Preliminaries

Autoencoder was expressed mathematically by Pierre Baldi [63]. The encoder and decoder functions can be expressed as

$$Y : \mathbb{R}^n \to \mathbb{R}^p \quad (encoder), \tag{4.6}$$

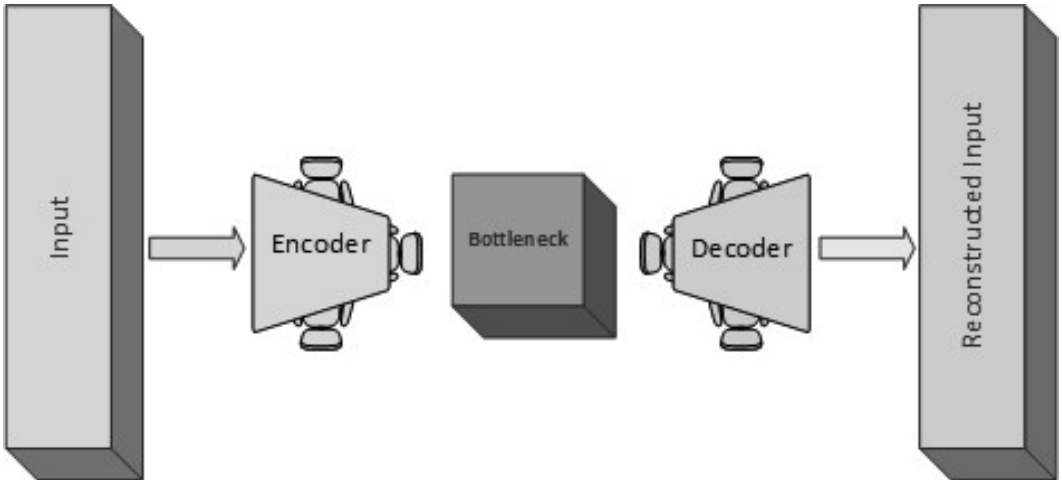

Figure 4.4. Autoencoder Architecture

$$Z : \mathbb{R}^p \rightarrow \mathbb{R}^n \quad (decoder). \tag{4.7}$$

While learning the above functions, it is necessary to consider some constraints. One of these constraints is expressed as

$$arg\, min_{Y,Z} E[\Delta(x, Z\ o\ Y(x)]. \tag{4.8}$$

In this expression, expectation over the distribution of x is calculated with the help of operator E. Δ operator expresses the reconstruction loss function by calculating the distance between the encoder input and the decoder output.

4.3.2. Algorithm Implementation

The autoencoder, which is an UL technique, was designed to create an SSL architecture in our study. By showing only a less amount of the non-stress samples to the autoencoder model, the best representation of the non-stress class will be learned by the model. Then, with the same model, stress samples will be generated differently from non-stress samples. Thus, the autoencoder will be able to distinguish the automatically generated stress samples. The flow of the algorithm is as follows

(i) Create the autoencoder network with input and output layers.
(ii) Apply Min-Max Normalization.
(iii) Training the autoencoder model with a less amount of non-stress samples.
(iv) Create a new network consisting of the weights of the trained network (This will create a network of latent representations of non-stress samples).
(v) Predicting raw non-stress and stress samples' hidden representation.
(vi) Hyper-parameter tuning in parameter space (hidden layer sizes, activation function, solver, alphas, learning rate).
(vii) Training and validating the classifier with the dataset containing the latent representation with the best parameters.

4.4. Experimental Results & Discussion

Since we used three different learning techniques and different models in our research, we thought it would be beneficial to use other performance metrics. The Silhouette Score metric has been chosen for clustering algorithms. It will be discussed in detail in the relevant section.

- **Accuracy**: A ratio of correctly classified observations to the total observations

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{4.9}$$

- **Precision**: A ratio of correctly classified positive observations to the total classified positive observations

$$Precision = \frac{TP}{TP + FP}. \tag{4.10}$$

- **Recall**: A ratio of correctly classified positive observations to the all observations

$$Recall = \frac{TP}{TP + FN}. \tag{4.11}$$

- **F-Measure**: Weighted average of precision and recall

$$Precision = \frac{TP}{TP + \frac{1}{2}(FP + FN)}. \tag{4.12}$$

- **Silhouette Score**: Metric that measures separation quality among clusters

  Let's assume:

  p = *Mean distance to the points in the nearest cluster*

  q = *Mean intra-cluster distance to all the points.*

$$SS = \frac{(p - q)}{max(p, q)}. \tag{4.13}$$

4.4.1. LP Algorithm Hyperparameter Tuning Stages and Results

The LP algorithm has two different kernel functions. Therefore, before starting the performance evaluation, we experimented with which kernel we would use in the algorithm. Thus, the best performance result of the LP algorithm was obtained by choosing the correct kernel function. LP algorithm has kNN or RBF options as its kernel. Table 4.2 and Table 4.3 show the performance metrics obtained using these two kernels. LP algorithm with kNN kernel, we observe that the precision score is high for both classes. f-Measure is also acceptable (0.9 for non-stress class and 0.7 for stress class). The important point is that the original dataset is partially imbalanced toward non-stress class value count being three times the value count (or the number of samples) of stress class (Almost 1.5 million class-0 and 0.5 million class-1 samples). Thus obtaining a high f1-score for class-0 is pretty natural, but still we obtained 0.75 f1 score for class-1, which is acceptable. However, the performance results of the LP algorithm with the RBF kernel are lower than the kNN kernel version. If Table 4.3 is sifted through, f-measure and recall scores are lower. Especially in the non-stress class, the success of the RBF kernel function has decreased. Considering the performance metrics, it has been confirmed that using kNN as a kernel function in the LP algorithm will perform better. Now we have new labels successfully classified by the LP algorithm. After adding those observations to the training data (by replacing the unlabeled data

Table 4.2. Classification Report of Label Propagation Algorithm (Selected LP Kernel = kNN)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.86	0.99	0.92
Stress	0.97	0.72	0.75
Macro Average	0.92	0.86	0.84
Weighted Average	0.89	0.92	0.88

Table 4.3. Classification Report of Label Propagation Algorithm (Selected LP Kernel = RBF)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.86	0.85	0.83
Stress	0.88	0.62	0.68
Macro Average	0.87	0.74	0.76
Weighted Average	0.86	0.80	0.79

with the new predictions) on which we are moderately confident. These are called as pseudo-labeled as contrasted to labeled data. Then, we trained this new (augmented) dataset using different classifiers and used these models to predict the accuracy of the test data (with 651600 samples). Performance comparison between these classifiers was made by examining the accuracy score. In Figure 4.5, although classifier performance results are close to each other, the highest accuracy score was obtained with the random forest classifier. With such a small set of labeled and unlabeled datasets, with the help of LP algorithm we obtained an acceptable accuracy of 75% on this test set. After this stage, hyperparameter tuning was performed to improve the performance of the random forest classifier. The RF algorithm consists of many parameters. We used *max_depth* and *n_estimators* parameters in the hyperparameter tuning process. Because these two parameters are directly related to the learning ability of the classifier, if they are tuned correctly, the results can be improved. Therefore, we can enhance the overall classification performance with these two parameters:

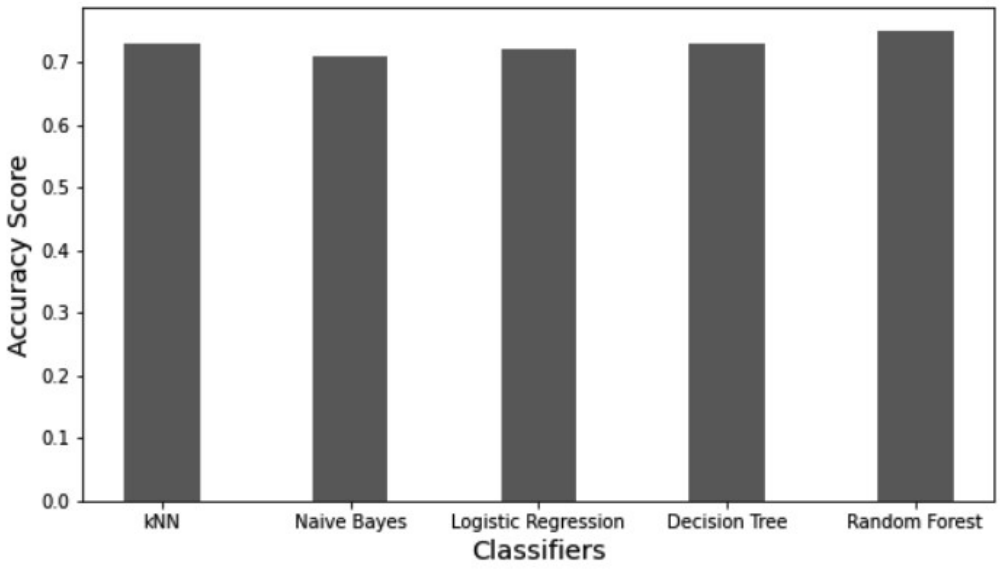

Figure 4.5. Accuracy Scores of Different Classifiers After LP Algorithm

- *max_depth*: Maximum depth of tree,
- *n_estimators*: Number of trees in the forest.

Accuracy scores of the RF classifier were obtained for these parameter values that change relative to each other. The results are shared in Table 4.4. Three different forest scenarios (100 , 300 , 500) were examined for increasing the maximum depth of tree values. Increasing the *max_depth* parameter too much causes the model to overfit. It has been observed that the model overfits the training data for values of *max_depth* greater than 20. That is why it is bounded to *max_depth=20*. Ascending values of the *n_estimators* can contribute to better learning of the data as it will increase the trees in the forest. However, higher values will increase the model's computational complexity. Growing it in a controlled way prevents the training time from getting too long. When the accuracy scores were evaluated, the best performance was obtained with *max_depth=20* and *n_estimeators=300*. Before the *n_estimator* parameter reached its maximum value, the model's performance reached saturation. In other words, the random forest model consisting of 300 trees with a depth of 20 units allowed us to achieve the best overall performance. In order to understand the success of the augmented dataset obtained with the LP algorithm, it will be useful to train the RF

Classifier (with the same parameters) with the real labeled dataset and compare the results. When RF Classifier is trained with augmented dataset, 77% Accuracy Score is obtained. When the RF Classifier is trained with the real dataset without pseudo-labels, 81% Accuracy Score is obtained. Considering the actual size of the dataset and the labeling burden that the pseudo-labels obtained with the LP algorithm save us; a performance difference of 4& can give researchers a new direction in line with their priorities.

4.4.2. Autoencoder Hyperparameter Tuning Stages and Results

We also obtained results with our other model, the autoencoder. It will be useful to visualize the data before hyperparameter tuning. The t-Distributed Stochastic Neighbor Embedding (t-SNE) nonlinear statistical method is very useful for high-dimensional data visualization.

Table 4.4. RF Classifier's Accuracy Results for Variable *max_depth* and *n_estimators* Parameters

max_depth	n_estimators		
	100	300	500
2	75.23	75.23	75.23
4	75.49	75.49	75.49
6	75.78	75.78	75.78
8	76.00	75.99	75.99
10	76.11	76.11	76.11
15	76.49	76.50	76.50
20	76.86	**76.88**	76.86

The t-SNE algorithm was developed in 2008 [64]. Based on the nonlinear dimensionality reduction method, high-dimensional data is reduced to two or three low-dimensional maps. The working logic of the algorithm is that similar objects in high-dimensional space are assigned a high probability, while dissimilar ones have a low probability distribution. Then, a similar process is performed in the low-dimensional

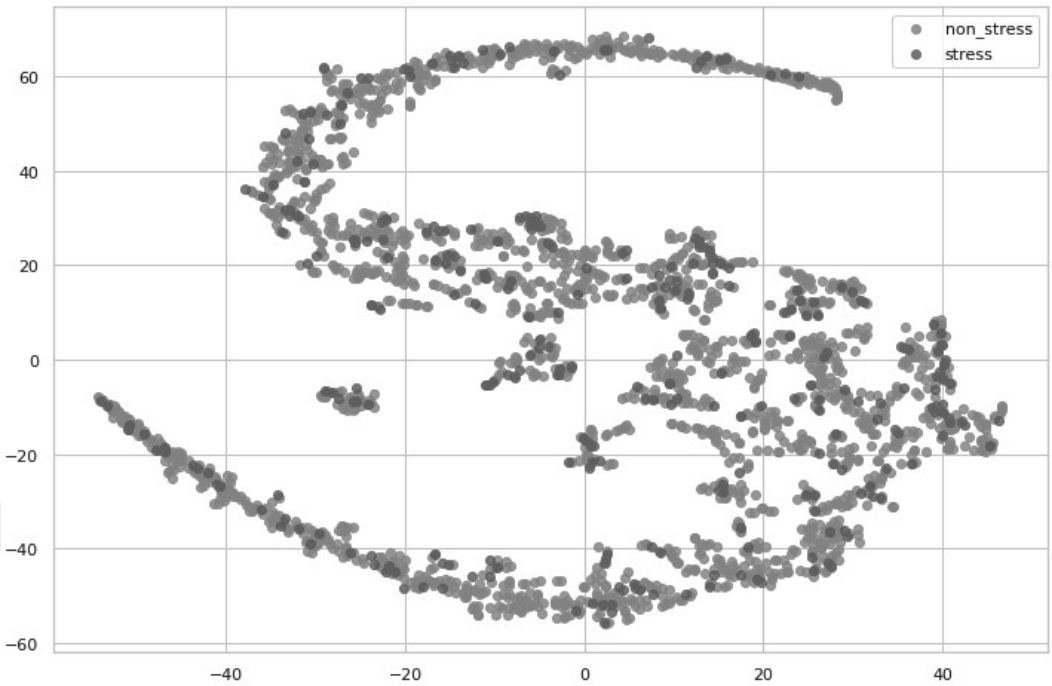

Figure 4.6. Visualization of Stress and Nonstress Classes via t-SNE

space, and the data points are mapped by considering the probability distribution in these two spaces. The mapping operation is performed using Kullback–Leibler divergence (KL divergence). As seen in Figure 4.6, we reduced the dimensions of the data using t-SNE and obtained the two-component representation. While the red dots represent the stress class, the green dots belong to the non-stress class. It is clearly seen that the samples of the two classes are very close to each other. Therefore, it is a very challenging dataset to work with simple models.Considering the challenges of our data, the autoencoder design was carried out. Working with a high-dimensional and imbalanced dataset on a single layer autoencoder makes it impossible to learn from the data. Therefore, we designed a Deep Autoencoder architecture using stacks of layers. In Figure 4.4, Deep Autoencoder architecture is created symmetrically on the encoder and decoder parts. Encoder and decoder consist of shallow layers and are connected with a bottleneck. We have conducted many experiments to determine the input nodes of the shallow layers in the encoder. Considering that we have four features, the dense layer, which started with a hundred nodes, was gradually reduced to three nodes when it reached the bottleneck. The bottleneck design decision is quite significant. We have

seen in the experiments that when the bottleneck size is designed much larger than the number of features, the network becomes light and flexible. It copies the input's low dimensional representation exactly instead of compressing it. On the other hand, if the bottleneck is designed too narrow, this time the network loses its ability to learn as it will experience a massive amount of information loss. For this reason, the number of nodes has been gradually reduced from one hundred to three. In the encoder part, the layers are designed with a width of 100, 75, 50, 25, and 3 nodes, respectively, until the end. Since the decoder is symmetric of the encoder, it is designed to reconstruct the input with 3, 25, 50, 75, and 100 nodes incrementally. Also, the reason why the width of the nodes in the encoder layers is decreasing and the decoder is increasing is that we have come to the conclusion that the network can learn much better with this design. Adding noise to the encoder side of the network provides better learning. We decided that it would be useful to add L1 regularization to the encoder's initial layer so that the features can be learned better. As a result of the experiments, the L1 regularization value was chosen as 0.00001. We used MSE Loss (Mean Squared Error Loss) to measure the error between the actual input and the reconstructed input. Different activation functions can be used within the layers of the Autoencoder model. As a result of experiments using only *tanh* or only ReLU, we have seen that making a mixed design results in much better performance. In this direction, we used *tanh* in the first two layers of the encoder, and we used *tanh* in the last two layers of the decoder. The remaining layers were created using the ReLU activation function. Deep autoencoder consists of many stacked layers. Training it recursively involves a large number of parameters. For this reason, the probability of the designed models being overfit increases. In addition, it takes a lot of time to converge for models with thousands of parameters. Batch normalization is a fundamental method for reliable network design and adequate convergence time. Moreover, it helps to create less reaction to sudden changes in the input and hidden layers. We also used of batch normalization to avoid overfitting and shorten the long training time during layer designs. We created a batch normalization layer after each dense layer. Since the encoder and decoder are symmetrical, we used batch normalization in both parts. As a result of the experiments, thanks to batch normalization, overfit possibility of our model was eliminated. We also

Table 4.5. Final Deep Autoencoder Model Summary with Parameters

Autoencoder Layers	**Output Shape**	**Parameters**
input_1 (InputLayer)	(None, 4)	0
dense (Dense)	(None, 100)	500
batch_normalization (BatchNormalization)	(None, 100)	400
dense_1 (Dense)	(None, 75)	7575
batch_normalization_1 (BatchNormalization)	(None, 75)	300
dense_2 (Dense)	(None, 50)	3800
batch_normalization_2 (BatchNormalization)	(None, 50)	200
dense_3 (Dense)	(None, 25)	1275
batch_normalization_3 (BatchNormalization)	(None, 25)	100
dense_4 (Dense)	(None, 3)	78
dense_5 (Dense)	(None, 3)	12
batch_normalization_4 (BatchNormalization)	(None, 3)	12
dense_6 (Dense)	(None, 25)	100
batch_normalization_5 (BatchNormalization)	(None, 25)	100
dense_7 (Dense)	(None, 50)	1300
batch_normalization_6 (BatchNormalization)	(None, 25)	100
dense_8 (Dense)	(None, 75)	3825
batch_normalization_7 (BatchNormalization)	(None, 75)	300
dense_9 (Dense)	(None, 100)	7600
dense_10 (Dense)	(None, 4)	404
Total Params: 28,081		
Trainable Params: 27,275		
Nontrainable Params: 806		

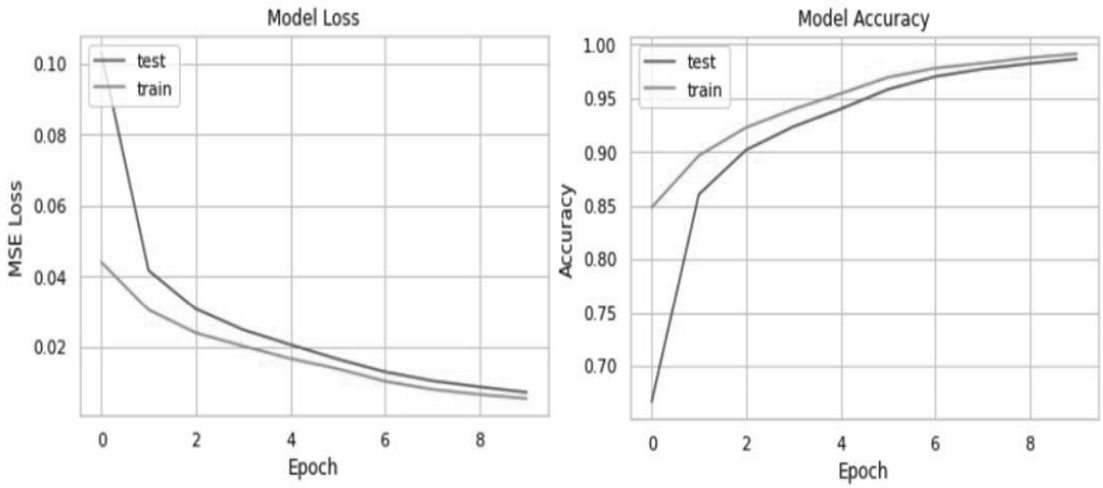

Figure 4.7. Learning Curves of Our Final Deep Autoencoder Model

achieved sustainable training time. Considering the size of the dataset we have, the *batch_size* was tried gradually as 32, 64, 18, and 256 during the autoencoder (only non-stress samples) training phase. As a result, it was seen that the network achieved the best performance when *batch_size=256*. Many different optimizers can be used while designing the deep autoencoder model. Stochastic Gradient Descent (SGD) Adam, Adagrad, and RMSprop are widely used. However, as a result of the experiments we carried out, we decided that the Adadelta [65] optimizer functions most effectively in the network. Adadelta, a stochastic gradient descent method with an adaptive learning rate, gave more successful results. Primarily, the optimizer's ability to adapt itself without depending on the initial learning rate provided convenience in the hyperparameter tuning phase. A detailed summary of the deep autoencoder model can be seen in Table 4.5. While making this design, the decision was taken according to the best training performance result by examining the loss and accuracy curves in Figure 4.7.

Our autoencoder network is trained with non-stress class samples. Thanks to the accessible weights of the autoencoder model, it is possible for us to access the latent representation of the non-stress input. To use these weights of the trained network, we can create a new network with hidden layers. Thus, we will predict raw stress and non-stress data through these sequential weight layers. Since the network is trained with

Table 4.6. First Experiments of Deep Autoencoder Model Summary

Autoencoder Layers	**Output Shape**	**Parameters**
input_1 (InputLayer)	(None, 4)	0
dense (Dense)	(None, 100)	500
dense_1 (Dense)	(None, 50)	5050
dense_2 (Dense)	(None, 50)	2550
dense_3 (Dense)	(None, 100)	5100
dense_4 (Dense)	(None, 4)	4004
Total Params: 13,604		
Trainable Params: 13,604		
Nontrainable Params: 0		

the non-stress class, it will sense the difference while predicting the samples of the stress class. While designing the network consisting of weights, the number of layers of the autoencoder model should be taken as a reference. Otherwise, the latent representation of input cannot be used correctly. The encoder consists of five dense layers and four batch normalization layers (same structure in the decoder side). There are weights of ten separate layers in total that we can access together with the input layer. In this case, it would be appropriate to make the latent representation network with ten layers. It is vital to correctly parameterize the encoder, decoder, and bottleneck layers of the deep autoencoder model. The model's inability to learn leads to its inability to reconstruct input samples. In our scenario, the model that cannot learn non-stress samples will not be able to predict stress samples. Such a design cannot be expected to perform well. To explain this better, it would be meaningful to share the results of the simpler autoencoder model we designed before finding the final model parameters. Table 4.6 shows the simpler autoencoder model we designed first. There are no batch normalization layers, less dense layers, the bottleneck size is much larger than the size of the input, and the number of trainable parameters that the model can learn is almost half of our final model. We talked about the importance of tuning the batch normalization layers and bottleneck size correctly to eliminate the overfitting problem.

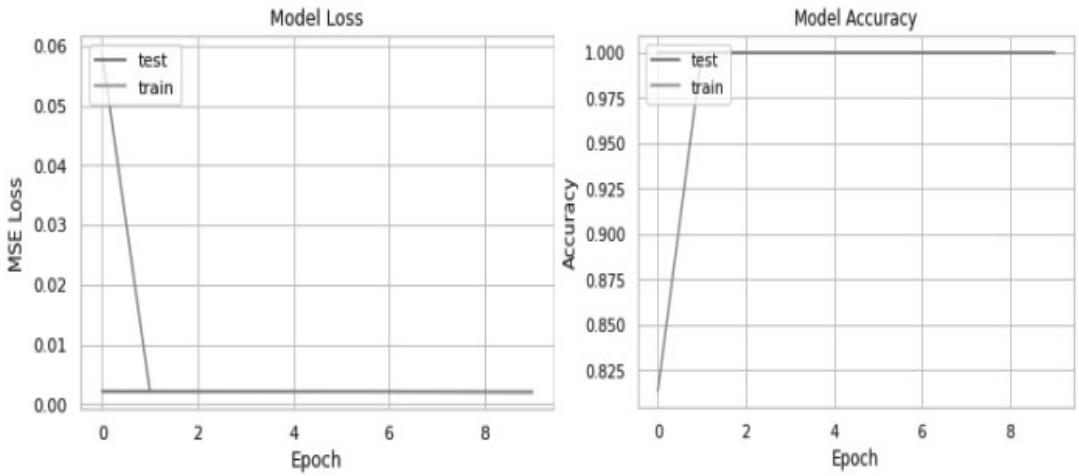

Figure 4.8. Problematic Learning Curves of Experimental Autoencoder Model

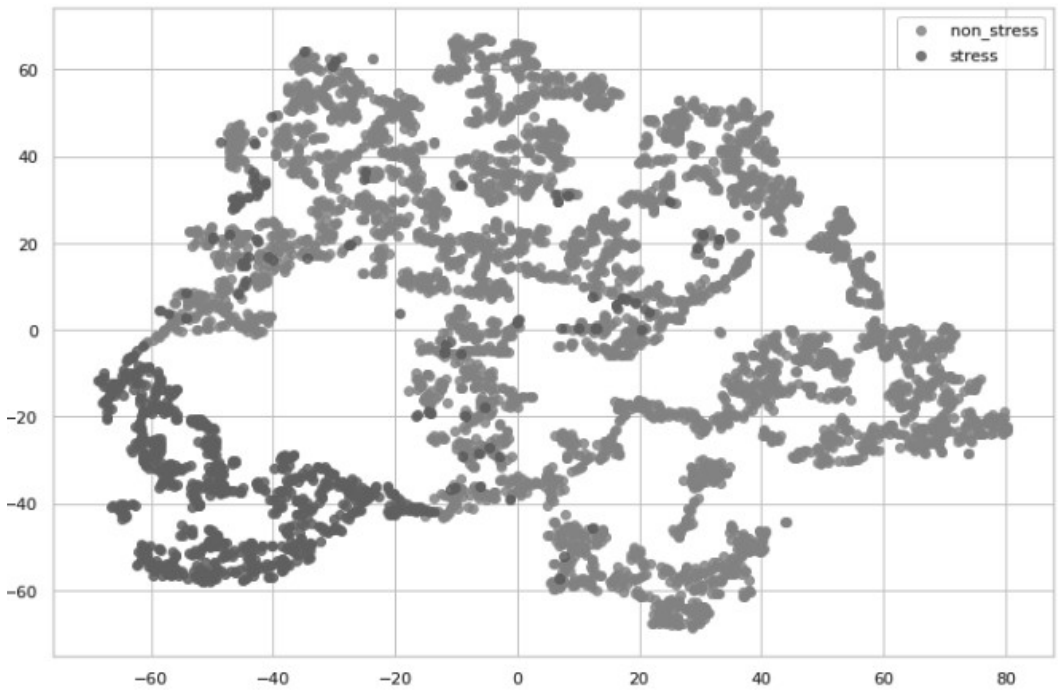

Figure 4.9. Latent Representation of Stress and Nonstress Classes via t-SNE

Table 4.7. Classification Report of Logistic Regression Classifier(solver=lbfgs)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.62	0.67	0.64
Stress	0.58	0.54	0.56
Accuracy	0.61		

Table 4.8. Classification Report of Logistic Regression Classifier (solver=saga)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.63	0.68	0.65
Stress	0.60	0.56	0.57
Accuracy	0.63		

Let's look at the learning curves in Figure 4.8 of this experiment where these parameters could not be tuned precisely. As can be seen from the curves, it is clear that the model directly overfits the training data. For this reason, making predictions behind the input reconstruction with such an autoencoder model will produce fallacy and biased results. Figure 4.9 shows the success of the classification of stress and non-stress samples reconstructing the latent representation of non-stress input. After this stage, we can examine the success of the dataset we obtained with the deep autoencoder model using different classifiers. Considering Figure 4.9, we started with the linear classifier. The performance results are obtained when the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) algorithm is used as a solver. However, when Table 4.7 is examined, the classifier prediction results are not very satisfying. The overall accuracy 0.61 is slightly above 50%, which has a lot of room for improvement. L-BFGS solver is only capable of L2 regularization, and this can create a constraint. So we repeated the experiments using the SAGA optimizer. SAGA is a solver that offers fast linear convergence rates in big datasets. However, the performance results obtained with SAGA are slightly better than the L-BFGS solver results. In Table 4.8, the overall classifier accuracy increased to around 0.63. When these results are evaluated, expected performance scores are still not achieved. The augmented dataset we created through the LP algorithm was reported with the RF classifier. In order

Table 4.9. Classification Report of RF Classifier(*max_depth*, *n_estimators*)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.64	0.84	0.72
Stress	0.77	0.53	0.62
Accuracy	0.68		

Table 4.10. Classification Report of MLP Classifier (default parameters)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.65	0.73	0.69
Stress	0.69	0.61	0.65
Accuracy	0.68		

to compare the dataset we created with our deep autoencoder model with the LP, RF classifier was used with similar parameters(*max_depth=20*, *n_estimators=300*). In Table 4.9, RF classifier accuracy score was obtained as 0.68. Although it seems to be more successful than the LR classifier, it is still a result that is open to improvement. Moreover, when the RF Classifier was trained for similar parameters, it was revealed that the augmented dataset created by the label propagation algorithm gave better results than the dataset created with the deep autoencoder. As the next step, it would be useful to examine how close we are to the performance of the label propagation algorithm by training a more strong classifier. For this reason, experiments were carried out with the MLP classifier. First, the classifier was trained with its default parameters, and performance results were obtained. The parameters used during the initial training are:

- Hidden Layer Size = (100,),
- Activation Function = ReLU,
- Solver = Adam,
- Alpha = 0.0001,
- Learning Rate = Constant.

Table 4.11. Parameter Grid via GridSearchCV

Parameters	**Parameter Values**
Hidden Layer Size	[(50, 50, 50), (50, 100, 50), (100,)]
Activation Function	['lbfgs', 'sgd', 'adam']
Solver	['logistic', 'tanh', 'relu']
Alpha	[0.00001, 0.0001, 0.05]
Learning Rate	['constant', 'invscaling', 'adaptive']

Table 4.12. Classification Report of MLP Classifier (Hyperparameterized)

Class	**Precision**	**Recall**	**f-Measure**
Non-Stress	0.81	0.73	0.77
Stress	0.72	0.80	0.76
Accuracy	0.76		

The MLP classifier performance results, which were trained using the above parameters, are given in Table 4.10. The maximum performance of the RF Classifier has been achieved with the basic MLP Classifier without parameter tuning. In this case, it will be useful to evaluate the results by examining a more comprehensive parameter grid. Hyperparameter tuning is performed in a wide parameter space, and the parameters that provide the best estimation of the model are obtained with the scikit-learn library GridsearchCV tool. The best parameters were obtained after hyperparameter tuning with the parameter space in Table 4.11:

- Hidden Layer Size = (50, 100, 50),
- Activation Function = tanh,
- Solver = Adam,
- Alpha = 0.0001,
- Learning Rate = Adaptive.

Performance results are obtained with these tuned parameters. Table 4.12 shows the performance results of the model after hyperparameter tuning. Compared to the

simpler autoencoder design, much more successful and improvable results were obtained with the deep stacked autoencoder model. The final model achieved 81% precision score for class-0 (non-stress) and 72% for class-1 (stress). When f-Measure was examined, 77% score was achieved for non-stress, 76% score was achieved for stress classes. After all, we were able to achieve 76% accuracy from the classifier we trained with the new augmented dataset that prepared with the deep autoencoder model. Promising results were obtained using a less amount of labeled samples from an imbalanced dataset with 75.17% nonstress class and 24.83% stress class. In order to understand the success of the augmented dataset we obtained with the deep autoencoder model, we trained the MLP classifier with the real dataset (with the same parameters). It can be seen above that the accuracy score obtained with the Augmented dataset is 76%. The accuracy score of the MLP Classifier trained with the real dataset is 79%. There is a 3% difference between the performance of the models trained with augmented and real dataset. This proves that the deep autoencoder model creates an important trade-off with the SSL perspective.

5. EXPERIMENTS WITH SUPERVISED LEARNING ARCHITECTURES

There are many different conventional machine learning algorithms used in this field; we summarized them in Table 3.1. When the literature is examined, we have seen that working with raw sensor data and using deep learning architectures are more challenging and rare. For this reason, we decided to use LSTM and CNN-LSTM deep learning architectures. Since physiological sensor data is time-series, it would be advantageous to start the design with an architecture like LSTM that uses sequential input.

5.1. LSTM Networks

The recurrent neural network (RNN) is an artificial neural network with reasoning capability. They form repetitive cells with directed or undirected connections to each other. This creates a temporal sequence via a loop.

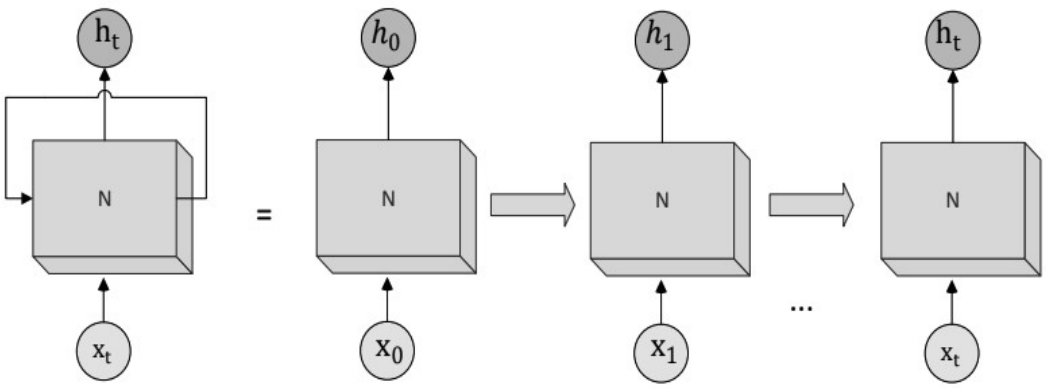

Figure 5.1. Simple Representation of Recurrent Neural Network

When Figure 5.1 is examined carefully, it can be seen that a *t-long* list can be created when the main loop is opened. Basically, it is desired to create a memory by using the information from the previous neural network cell as an input. It has been

used in various fields such as image captioning and speech recognition. But when RNN creates a memory using the past context, it can not go back very far. This issue is called the long-term dependency deficit. *Hochreiter et al.* [66] designed the Long Short-Term Memory (LSTM) to solve this problem. Although LSTM is basically RNN, it is a new generation version of RNN that solves the long-term dependency problem. In LSTM cells, different from RNN, new layers are added, and information is carried longer.

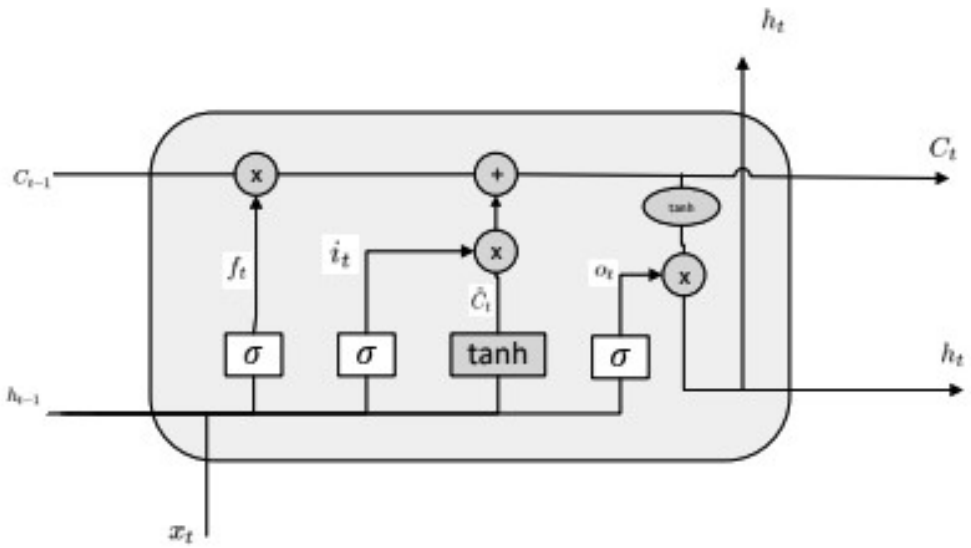

Figure 5.2. Simple Representation of LSTM

5.1.1. Theoretical Formulation and Preliminaries

The mathematical representation of the LSTM cell in Figure 5.2 is as follows:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f), \tag{5.1}$$

where forget gate is a sigmoid function. For 0 and 1 states, it is determined whether the input stays in the cell or not. The remaining input gate, cell update, cell state

layers are expressed as

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i), \tag{5.2}$$

$$\hat{C}_t = tanh(W_C.[h_{t-1}, x_t] + b_C), \tag{5.3}$$

$$C_t = ft * C_{t-1} + i_t * \hat{C}_t. \tag{5.4}$$

The input gate (i_t) layer and cell state (C_t) layers help to decide and store new information in the cell. In C_t, the old state C_{t-1} is deleted from memory by multiplying with f_t. Output gate and final output are expressed as

$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o), \tag{5.5}$$

$$h_t = o_t * tanh(C_t). \tag{5.6}$$

Finally, the required parts of the current cell state are extracted with the last sigmoid function. C_t is scaled with the *tanh* function and the h_t is obtained.

5.2. CNN-LSTM Networks

In Chapter 5.1, we talked about LSTM networks. They offer a special memory capacity through the layers they contain in their cells. The convolutional neural network (CNN) is an artificial neural network mostly used on visual data [67]. It allows extracting valuable features from the data using the convolution operation. CNN basically consists of 3 separate layers:

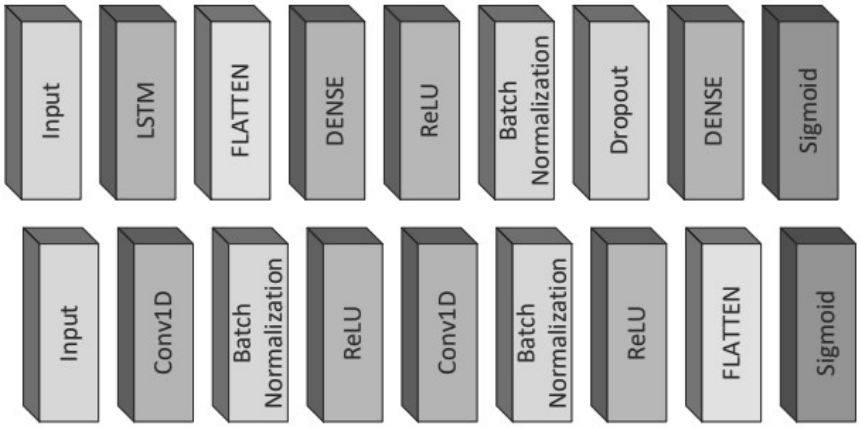

Figure 5.3. Our Architecture Design of the LSTM and CNN–LSTM Neural Networks

- **Convolutional Layer:**
  The dot product of the learnable parameters matrix of the data and the kernel function matrix is performed. Most of the computational load on the network is handled here. The kernel creates the activation map by moving in two dimensions, taking into account the height and width of the data. The sie of this sliding movement of the kernel is called stride.

- **Pooling Layer:**
  The pooling layer helps to eliminate the computational burden by reducing the size of the feature maps obtained through the convolutional layer. It reduces the representation size to a reasonable level by summarizing feature maps.

- **Fully Connected Layer:**
  It provides a convenient representation between input and output. In fully connected layer (FCL), input with distributed representations passing through different layers is converted to a single vector form.

Video and image data do not have a linearity principle. For this reason, nonlinear functions such as sigmoid, *tanh*, or ReLU are used to perform nonlinear operations in the network. These functions are selected during model design. For example, the

Table 5.1. Number of Trainable Parameters of LSTM Network

Modules	**Parameters**
lstm.weight_ih_l0	80
lstm.weight_hh_l0	400
lstm.bias_ih_l0	20
lstm.bias_hh_l0	20
fc.0.weight	12000000
fc.0.bias	1000
fc.2.weight	1000
fc.2.bias	1000
fc.3.weight	1000
fc.3.bias	1
Total Trainable Params:	12,004,521

sigmoid function takes a value between 0 and 1, while the *tanh* function takes a value between -1 and 1. In recent studies, hybrid versions of CNN and LSTM networks have been used [68, 69]. The CNN-LSTM network extracts preliminary features from the data via CNN and focuses on temporal features with the help of LSTM. We can define CNN-LSTM networks as an LSTM network variant with CNN layers. This new hybrid model also gives effective results on time-series. That's why we wanted to see the results by experimenting with this model.

5.3. Experimental Results & Discussion

In Figure 5.3, you can see the LSTM and CNN-LSTM based artificial neural networks we have designed. One of the problems encountered in the literature is that the time-series data is split by shuffling during training. After such an ill-judged training, the network performance will be biased as the LSTM network will see the data sequence ahead and behind. To prevent this, the time-series data split method is used. In our training sessions, time-series split techniques were used where the k-fold

Table 5.2. Layers and Input Sizes of LSTM Network

Layers	**Input Size**
LSTM	(128, 600, 4)
Flatten	(128, 600, 20)
Dense	(128, 12000)
ReLU	(128, 1000)
BatchNormalization1D	(128, 1000)
Dropout(p=0.5)	(128, 1000)
Dense	(128, 1000)
Sigmoid	(128, 1)

cross-validation's k is five. Thus, we will have four splits in training and one split in validation. Since the features were not extracted beforehand with raw data, studies were carried out on the CNN-LSTM architecture. CNN layers provide better feature extraction before data is fed to the LSTM layer. We planned to extract both spatial and temporal features from our raw dataset with the hybrid CNN-LSTM deep neural network.

5.3.1. LSTM Network Hyperparameter Tuning Stages and Results

The models were implemented using the PyTorch Deep Learning Tensor Library and Google Colaboratory Notebook. Thus, while the models are being trained with our large dataset, the GPU processing capability of the existing server is utilized by using the CUDA API. When evaluating the current NVIDIA Tesla T4 GPU performance and competence, the maximum initial batch size is set to 128. The problem we are working on is binary classification. In this respect, we decided to make hyperparameter tuning according to the accuracy score. Moreover, in the final discussion, we can easily compare the performances of our SSL and UL architectures with the accuracy score. The parameters we tune while training the LSTM model are:

- Hidden Size,

- Learning Rate,
- Activation Function,
- Batch Normalization.

The hidden size parameter is the number of features in the hidden state. While determining this parameter, it was increased gradually, and its final value was obtained. In experiments:

- $hidden_size < 20$:
  The accuracy score was lower because it was seen that the model was underfitting.
- $hidden_size > 20$:
  The probability of overfitting has increased. It has been clearly seen that the accuracy score is insufficient.

That is why it is set to hidden_size = 20. The learning rate was used as 0.0001. For larger learning rate values, the probability of over-shooting the global optimum increases. When lower learning rate values are selected, the learning capacity of the network decreases considerably, and the training time is long. Different activation functions have been tried. Better performance was obtained with ReLU compared to the others. Also, ReLU does not allow a negative gradient, which is good for efficient and fast training. As we used in our deep autoencoder model, the batch normalization layer was also used in the LSTM network. Thus, we made the LSTM network more reliable. During training, Binary Cross Entopy (BCE) loss is used instead of MSE loss. Due to the nature of our problem, it is thought that binary classification can be better analyzed with the BCE loss. The output layer is designed with sigmoid function in accordance with the BCE loss. Kingma *et al.* [70] suggested the advantages of using Adam optimizer in situations that require sparse and noisy gradients, so Adam was chosen as the optimizer for both LSTM and CNN-LSTM models. In the models established through the PyTorch framework, gradients are set to zero for each batch in the training phase. This is done before starting backpropagation. Because PyTorch models tend to keep these gradients. To prevent this, we reset the gradients by calling the zero gradient method with the help of the optimizer. By performing the gradient cleaning

Table 5.3. Number of Trainable Parameters and Training Time of Networks

Network	**Parameters**	**GPU (seconds)**	**CPU (seconds)**
LSTM	12,004,521	220	5000
CNN-LSTM	1,030,753	115	8000

after each backward pass, we ensured that the parameters were updated correctly. If this zero gradient method is not applied correctly, it causes the old used gradients to interfere with the newly computed gradients. After the hyperparameter tuning is completed, the layers of the finalized model and the number of trainable parameters formed in these layers are shared in Table 5.1. The final LSTM network architecture and input sizes are given in Table 5.2. To understand the virtue of using GPU, we also trained the LSTM model via CPU. We shared the training times in Table 5.3. In training with the GPU, each iteration took 2.2 seconds, while the CPU took 50 seconds per iteration. Each iteration consists of one forward pass and one backward pass. The GPU enabled 23x faster training iterations.

5.3.2. CNN-LSTM Network Hyperparameter Tuning Stages and Results

As in the LSTM network, the hyperparameter tuning in the CNN network is based on the accuracy metric. In addition to accuracy, loss curve, f-measure, recall and precision metrics were also checked simultaneously. In the training phase of the LSTM network, the capacity of the current GPU was evaluated and the maximum batch size was selected as 128, and this value was continued to be used in the CNN-LSTM network. Since the overall network consists of CNN and LSTM parts, the parameters are tuned in two separate parts. The tuned CNN network parameters are:

- Kernel Size,
- Stride,
- Hidden Size.

The dimensions of the dataset is taken into account when determining the dimensions

of convolution layers, kernel size, and stride. Considering that there are four attributes in the convolution layer, the kernel size and stride have been tested from one to three to extract these features. Although the CNN model extracts more information when the kernel size is selected as one, choosing two provides better performance in the case of overfitting. When the kernel size is chosen as three, it is observed that the accuracy score decreases, and the final decision is made as *kernel_size=2*. Since we use 1D convolution layers and the kernel size is selected as two, *stride=1* is set. While determining the stride, this result was reached by testing the mutual values with the kernel size. While the CNN network hidden size parameter was determined, similar findings were observed in Section 5.3.1, so this parameter was kept the same with its value in the LSTM network.

Layers of the LSTM network part have been designed considering Section 5.3.1. Input sizes are rearranged according to the output of the CNN network part. The layers of the finalized CNN-LSTM model and the number of trainable parameters formed in these layers are shared in Table 5.4. The CNN network part of the CNN-LSTM model has put a severe load on the CPU. Training time with CPU takes 70x longer. The training duration of the CNN-LSTM network was lower than the LSTM network. Training the CNN-LSTM model with the GPU took 1.1 seconds per iteration. It is even 2x faster than training the LSTM network with a GPU. Considering the performance metrics in Table 5.5, a more successful result was obtained with the CNN-LSTM network. The time-series split method was used as a cross-validation strategy, thus preventing the model from being trained with samples from the future sequence. In the literature, this often leads to biased performance results. Thus, our LSTM and CNN-LSTM models were built with a correct cross-validation strategy. The results obtained are at a level to compete with the performance results in the literature. Accuracy, Precision, Recall and f-measure results of the CNN-LSTM model were obtained slightly better performance than those of the LSTM model. Figure 5.4 shows the results of these metrics prepared per iteration. Even though we performed 100 epoch trainings, the model was saturated around the 80th iteration. Very similar curves are also obtained in the LSTM model for these performance metrics.

Table 5.4. Number of Trainable Parameters of CNN-LSTM Network

Modules	**Parameters**
cnn.0.weight	307200
cnn.0.bias	256
cnn.1.weight	256
cnn.1.bias	256
cnn.3.weight	65536
cnn.3.bias	128
cnn.4.weight	128
cnn.4.bias	128
lstm.weight_ih_l0	40
lstm.weight_hh_l0	400
lstm.bias_ih_l0	20
lstm.bias_hh_l0	20
fc.0.weight	655360
fc.0.bias	256
fc.2.weight	256
fc.2.bias	256
fc.3.weight	256
fc.3.bias	1
Total Trainable Params:	1,030,753

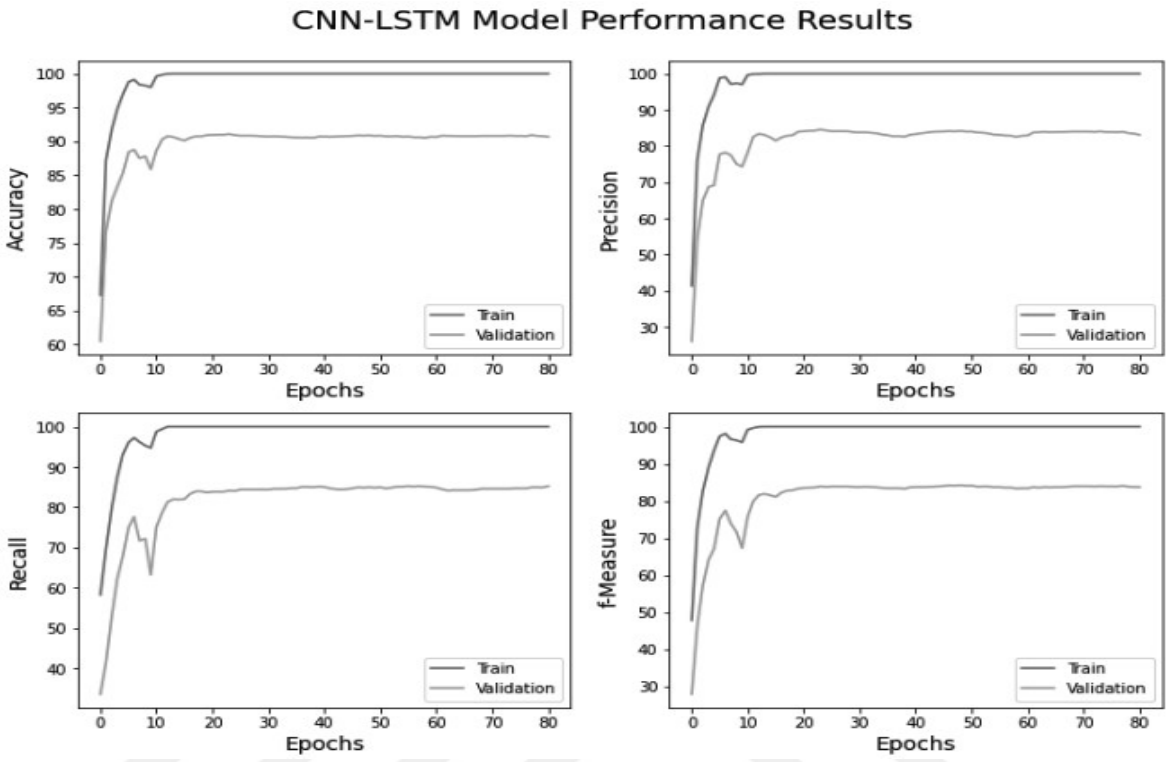

Figure 5.4. Performance Results of CNN-LSTM Network

Table 5.5. Classification Results of LSTM and CNN-LSTM Networks

Algorithm	**Accuracy**	**f-Measure**	**Precision**	**Recall**
LSTM	90.38	82.60	83.18	82.01
CNN-LSTM	**91.35**	83.84	85.70	82.09

In Table 5.5, the performance results of the LSTM and the CNN-LSTM models are given together. Although the results were close to each other, CNN-LSTM model ensured improvement.One of the main reasons for this is that thanks to the CNN layers, more informative features were extracted from the raw data and fed to the LSTM layers. This increased the overall performance. Besides this performance improvement, there is another very valuable output provided by the CNN-LSTM model. Training using the GPU took much less time with the CNN-LSTM network, these type of models will make it easier to work with a large dataset.

6. EXPERIMENTS WITH UNSUPERVISED LEARNING ARCHITECTURES

In this learning technique, the model is expected to learn the internal representation of the data without the use of labels. Clustering is in this category. Clustering algorithms look for solutions by considering the similarity of the samples and their distance from each other in the feature space. It aims to separate homogeneous subgroups in the dataset by using this statistical similarity. The clustering algorithms to be used may vary depending on the difficulty of the problem. We worked with three different clustering algorithms. Some of these algorithms must first be given parameters such as the number of clusters or the minimum distance between observations. With the hyperparameter tuning algorithm, we enabled the model to choose the most appropriate number of clusters itself. To eliminate the initialization for the number of clusters, the optimal number of clusters is taken from the algorithm by checking the Silhouette Score metric during the hyperparameter tuning phase.

6.1. K-Means

K-means is among the most known and used clustering algorithms. It works on minimizing the average squared distance of the samples in the same dense region (cluster). When the problem is defined mathematically, the dataset can be defined as

$$X = \{x_1, ..., x_u\} \in R^t, \tag{6.1}$$

$$C = \{c_1, ..., c_v\} \in k = (1, ..., v), \tag{6.2}$$

$$p = [p_{ik}]_{t \times v}, where \quad p_{ik} \in \{0, 1\}, \tag{6.3}$$

where C represent Cluster Center, p represents Cluster Indicator. In t-dimensional Euclidean space, whether any data point belongs to the k-th cluster is checked by eq. (6.3). We can express the objective function of the algorithm as

$$J(p, C) = \left(\sum_{i=1}^{u} \sum_{k=1}^{v} p_{ik} \|x_i - c_k\|^2 \right). \tag{6.4}$$

The Euclidean distance between data points and cluster centers is updated to minimize the iteratively calculated objective function.

6.2. BIRCH

Since traditional clustering algorithms such as K-means are open to improvement in areas such as running time, memory management and processing performance, we thought that BIRCH, defined as Balanced Iterative Reducing and Clustering using Hierarchies, could yield more effective results. BIRCH, which creates a meaningful and informative summary dataset from a large original dataset, aims to eliminate the disadvantages of traditional clustering algorithms by applying clustering on the new summary dataset. Zhang *et al.* [71] expressed their algorithm flow as follows:

- Data Loading,
- Initial Clustering Feature (CF) Tree,
- Smaller CF Tree,
- Good Clusters,
- Better Clusters with Cluster Refining(Optional).

The CF specified here refers to the dense information fields obtained in the transition to the smaller dataset. The CF consists of the number of data points in the cluster, the linear sum of the data samples, and the squared sum of data samples. The CF Tree defines the structure whose leaf nodes are conjugated to information-carrying sub-clusters.

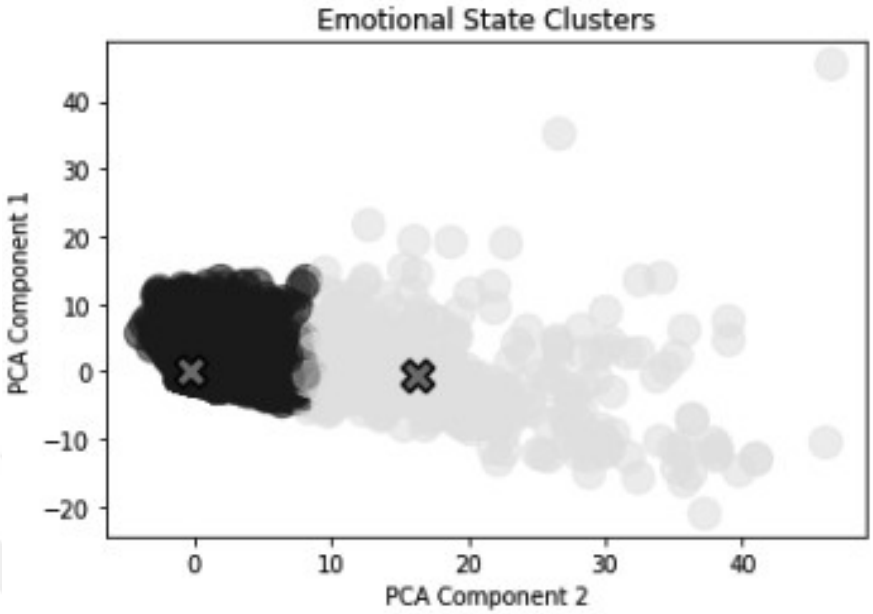

Figure 6.1. Visualization of K-means Classification Result via PCA

6.3. DBSCAN

In 1996 Ester *et al.* [72] published the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm. Basically, the algorithm examines the high density and low-density regions of the data points in the data space. It uses epsilon (*eps*), which expresses the distance between data points and the minimum number of data points (*minPts*) that can form a cluster as parameters. The algorithm working steps are as follows:

- Random data points are selected until all data points are classified,
- If there are at least *minPts* data points within the *eps* radius, this creates a cluster,
- After the neighborhood calculation is made according to the neighborhood points, the existing clusters are expanded until the final number is found.

Each data point is classified as Core Points, Boundary Points, and Noise Points with the help of *minPts* and *eps* parameters. There are two important advantages of using the DBSCAN algorithm compared to K-means. First of all, DBSCAN can work with-

out knowing the number of clusters a priori. Secondly, while the K-means algorithm can include the data point that has a weak relationship with the cluster, DBSCAN's clustering by detecting noise points gives more effective results in terms of performance.

6.4. Experimental Results & Discussion

Since the size units of the features are different from each other, standardization was performed using the scikit-learn *StandardScale* method before the three clustering algorithms were run. Performance results are obtained for K-means, DBSCAN, and BIRCH clustering algorithms. Principal Component Analysis (PCA) is used to visualize how clustering algorithms classify data. After the high-dimensional data is reduced to two components with the help of PCA, the visualization of the K-means algorithm is shared in Figure 6.1. One of the points to be considered is that important information about the data can be lost after PCA is applied to clustering algorithms, so it would be more accurate to use it for data visualization purposes. During the hyperparameter tuning phase, Silhouette Score was checked, and the models could determine the number of clusters themselves. In Figure 6.2, silhouette scores were obtained for the different number of clusters of the K-means algorithm. The highest SS was obtained from the algorithm for two clusters. We already knew that there were two classes in our data, but we validated the K-means algorithm to cluster these two classes correctly with the SS metric. Similar results were obtained by examining SS for BIRCH algorithms. In Table 6.1, the silhouette score results for the different clustering algorithms are displayed. Here, silhouette score results are reported by dynamically parameterizing the number of clusters. Although the results are very close to each other, it can be said that the BIRCH algorithm clusters the two classes better with a minimal difference. We theoretically stated that the BIRCH algorithm creates an advantage in terms of running time, CPU utilization, and memory management. It will be useful to test this theoretical information and report it. The server where runtime and resource utilization experiments are performed has NVIDIA RTX A2000 GPU, 11th Gen Intel(R) Core(TM) i7-11850H @ 2.50GHz (16 Cores), 32 GB RAM. While calculating the runtime, this metric was calculated for each function block used

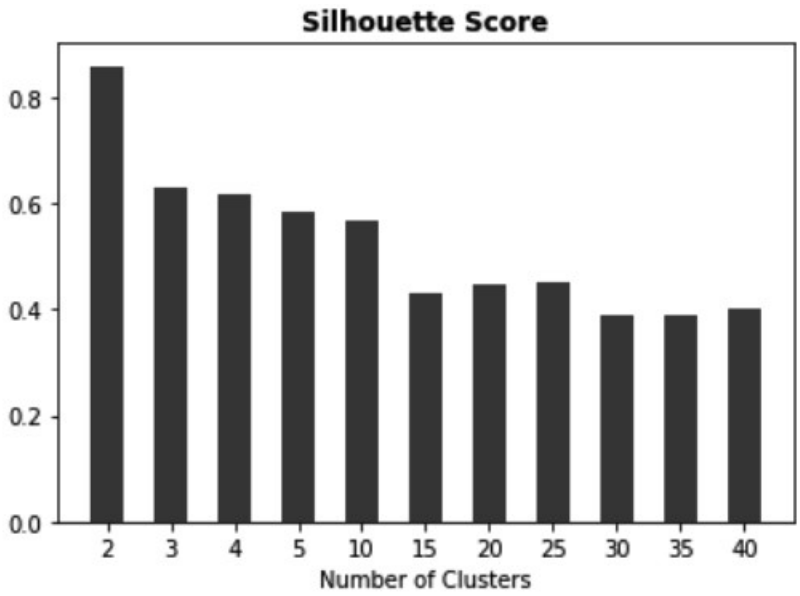

Figure 6.2. Silhouette Scores for different number of clusters.

Table 6.1. Silhouette Score for dynamically varying number of clusters

Algorithm	**2-Cluster**	**3-Cluster**	**4-Cluster**
K-Means	0.85	0.62	0.61
BIRCH	**0.86**	0.63	0.62

in the algorithm, and the cumulative sum was obtained. When Table 6.2 is examined, it is seen that CPU and RAM utilization of the BIRCH algorithm is more efficient than other algorithms. In addition, a shorter runtime was obtained than the others. It has been observed that the DBSCAN algorithm's attempt to handle even the smallest density regions causes it to run longer and to be more demanding in terms of resources. At this point, it would be right to make the final decision by using the ground truth samples available. The accuracy metric of the algorithms was calculated using the data samples labeled by the clustering algorithms and the existing ground truth samples, and their accuracy scores are shared in Table 6.3. K-means algorithm clustered stress and non-stress samples more successfully than BIRCH and DBSCAN. When the labels obtained with K-means were compared with the original labels, the overall accuracy score was 73%. The BIRCH algorithm also performed close to the K-means result. BIRCH presented a vital trade-off in terms of runtime and resource usage. It can be

Table 6.2. Runtime and Resource Utilization

Algorithm	**Runtime (ms)**	**CPU Utilization (%)**	**RAM Utilization (%)**
K-Means	85809	0.77	0.12
DBSCAN	101286	0.81	0.14
BIRCH	65052	0.73	0.10

of great advantage when used with high-dimensional datasets. Variable density regions in the dataset may have caused DBSCAN to tackle. For this reason, the DBSCAN accuracy score tells us that it makes an almost random prediction.

Table 6.3. Accuracy Results of SL & SSL & UL Architectures

Model	**Hyperparameter Tuning**	**Accuracy**
Supervised Learning Results		
LSTM	Yes	90%
CNN-LSTM	Yes	91%
Semi-Supervised Learning Results		
Label Propagation (RF Classifier)	Yes	**77%**
Autoencoder (LR Classifier-lbfgs)	No	61%
Autoencoder (LR Classifier-saga)	No	63%
Autoencoder (RF Classifier)	Yes	68%
Autoencoder (MLP Classifier)	No	68%
Autoencoder (MLP Classifier)	Yes	**76%**
Unsupervised Learning Results		
K-Means	Yes	73%
BIRCH	Yes	70%
DBSCAN	Yes	56%

7. CONCLUSION

In this study, we focused on the semi-supervised classification of mental stress in daily life. Unlike previous studies, we collected multi-sensor physiological data of the subjects in their routine. Our main goal was to provide a solution to the labeling problem of sensory data. In this direction, we designed LP and deep autoencoder models and compared their performance with the existing SL (LSTM, CNN-LSTM) and UL (K-means, BIRCH, DBSCAN) algorithms. Especially for the time-series data, instead of the conventional train&test split model used in the cross-validation phase, we obtained the performance of our LSTM and CNN-LSTM models correctly by using the time-series split method. It is the first study conducted on semi-supervised mental stress classification using daily life physiological data with the label propagation algorithm and deep autoencoder model.

Although our performance metrics vary since we apply three different learning techniques, we ultimately decided to report our models with the accuracy metric. Table 6.3 shows the percentage accuracy score of our models. SL architectures are leading in terms of performance since they have the ground truth gathered beforehand. However, the results of the Label Propagation algorithm, which we designed with the very limited labeled data, are also promising. Our Label Propagation algorithm achieved 86% precision score for class-0 (non-stress) and 97% for class-1 (stress). When f-Measure was examined, class-0 achieved 92%, class-1 72% results. Afterward, we were able to achieve 77% accuracy from the classifier we trained with the new augmented dataset that prepared with the label propagation algorithm. Precision, accuracy, and f-measure performance metrics obtained per class are close to those of the SL models. The deep autoencoder, which was initially under the umbrella of UL, was used in our study with an SSL perspective. Such a study has not yet been conducted using raw daily life data in the literature. When the deep autoencoder with logistic regression classifier performance result is examined, it is seen to have a lower accuracy score than some clustering algorithms. One of the reasons could be imperfect decoding. In this case, the lossy reconstruction phase may cause a decrease in performance, or the augmented

dataset we obtained with the deep autoencoder might not be compatible with the classifier running in the continuation of the model. Therefore, we conducted experiments with different classifiers after the deep autoencoder model. First, we trained the RF classifier using the same parameters to compare with the label propagation algorithm. When the deep autoencoder was trained with the RF classifier, it performed 5% better than the LR classifier. Although this result is an improvement for the deep autoencoder model, it was found to be almost 10% lower when compared to the label propagation algorithm. On the other hand, results closer to clustering algorithms were obtained. After these results, it was decided to test the autoencoder model with a more advanced classifier. For this reason, by choosing the MLP classifier, the best parameters were obtained with hyperparameter tuning. As a result, the overall accuracy score almost approached the LP algorithm.

Consequently, when the performance results are examined, the superiority of the SSL architectures to UL architectures is also evident. Moreover, the ground truth is needed to ensure accuracy when measuring the performance of existing clustering algorithms. Similarly, although very high performance is achieved with LSTM and CNN-LSTM models, the labeling burden cannot be ignored. In such a case, it will be much more advantageous to obtain results with the minimum labels using SSL architectures.

7.1. Future Work

We aimed to create a performance trade-off with SSL architectures by focusing on the labeling problem of physiological data collected with unobtrusive wearable wrist bands. In the next step, hybrid models can be built with CNN or LSTM layers, especially on the autoencoder side, and better results can be obtained there. In addition, the Label Propagation algorithm was tested with different kernels such as kNN and RBF. As a next step, a new autoencoder kernel can be implemented into the LP algorithm and the results can be examined. In UL Clustering algorithms, we provided accuracy measurements with the ground truth samples and raw features. One of the new research areas is deep clustering in UL. Researchers aim to use clustering

algorithms more effectively by obtaining the features of the data with the help of neural networks. We can continue future research by redesigning the autoencoder model we used in this thesis to work as a hybrid model with clustering algorithms. While working in this field, it is necessary to be aware of the difficulties as well as the benefits of the deep clustering algorithm. It provides feature extraction, processing large datasets, and advanced parameter estimation. But the neural network side of deep clustering algorithms is data-hungry. A massive amount of data is required to obtain informative features. Accordingly, the hyperparameter tuning phases require enormous computational resources.

REFERENCES

1. Cohen, S., R. C. Kessler and L. U. Gordon, *Measuring Stress: A Guide for Health and Social Scientists*, Oxford University Press on Demand, Oxford, 1997.

2. Schneiderman, N., G. Ironson and S. D. Siegel, "Stress and Health: Psychological, Behavioral, and Biological Determinants", *Annual Review of Clinical Psychology*, Vol. 1, No. 1, pp. 607–628, 2005.

3. Fink, G., "Stress: Definition and History", *Encyclopedia of Neuroscience*, pp. 549–555, Academic Press, Oxford, 2009.

4. Global Organization of Stress, "Stress Facts", http://www.gostress.com/stress-facts/, 2022, accessed at May 10 2022.

5. The American Institute of Stress, "What is Stress?", https://www.stress.org/daily-life/, 2022, accessed at May 9 2022.

6. Alberdi, A., A. Aztiria and A. Basarab, "Towards an Automatic Early Stress Recognition System for Office Environments Based on Multimodal Measurements: A Review", *Journal of Biomedical Informatics*, Vol. 59, pp. 49–75, 2016.

7. Maercker *et al.*, "Diagnosis and Classification of Disorders Specifically Associated with Stress: Proposals for ICD-11", *World Psychiatry*, Vol. 12, No. 3, pp. 198–206, 2013.

8. Kalas, M. S. and B. Momin, "Stress Detection and Reduction Using EEG Signals", *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, pp. 471–475, Chennai, India, 2016.

9. Cho, H.-M., H. Park, S.-Y. Dong and I. Youn, "Ambulatory and Laboratory Stress Detection Based on Raw Electrocardiogram Signals Using a Convolutional Neural

Network", *Sensors*, Vol. 19, No. 20, pp. 171–189, 2019.

10. Seo, W., N. Kim, S. Kim, C. Lee and S.-M. Park, "Deep ECG-Respiration Network (DeepER Net) for Recognizing Mental Stress", *Sensors*, Vol. 19, No. 13, pp. 207–222, 2019.

11. Sriramprakash, S., V. D. Prasanna and O. R. Murthy, "Stress Detection in Working People", *Procedia Computer Science*, Vol. 115, pp. 359–366, 2017.

12. Garcia-Ceja, E., V. Osmani and O. Mayora, "Automatic Stress Detection in Working Environments From Smartphones' Accelerometer Data: A First Step", *IEEE Journal of Biomedical and Health Informatics*, Vol. 20, No. 4, pp. 1053–1060, 2016.

13. Can, Y. S., N. Chalabianloo, D. Ekiz, J. Fernández-Álvarez, C. Repetto, G. Riva, H. Iles-Smith and C. Ersoy, "Real-Life Stress Level Monitoring Using Smart Bands in the Light of Contextual Information", *IEEE Sensors Journal*, Vol. 20, No. 15, pp. 8721–8730, 2020.

14. Liapis, A., C. Katsanos, D. Sotiropoulos, M. Xenos and N. Karousos, "Stress Recognition in Human-Computer Interaction Using Physiological and Self-Reported Data: A Study of Gender Differences", *Proceedings of the 19th Panhellenic Conference on Informatics*, p. 323–328, Association for Computing Machinery, New York, NY, USA, 2015.

15. Cannon, W. B., *Bodily Changes in Pain, Hunger, Fear and Rage: An Account of Recent Researches into the Function of Emotional Excitement*, D. Appleton, Eastford, 1922.

16. Selye, H., *The Stress of Life*, McGraw-Hill Companies, New York, 1956.

17. Harvard, "Understanding the Stress Response", http://www.health.harvard.edu/, 2018, accessed at May 13 2022.

18. Samura, T. and H. Nishimura, "Influence of Keyboard Difference on Personal Identification by Keystroke Dynamics in Japanese Free Text Typing", *2012 Fifth International Conference on Emerging Trends in Engineering and Technology*, pp. 30–35, Himeji, Japan, 2012.

19. Hou, X., Y. Liu, O. Sourina, Y. R. E. Tan, L. Wang and W. Mueller-Wittig, "EEG Based Stress Monitoring", *2015 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 3110–3115, Hong Kong, China, 2015.

20. Kim, H.-G., E.-J. Cheon, D. Bai, Y. Lee and B. H. Koo, "Stress and Heart Rate Variability: A Meta-Analysis and Review of the Literature", *Psychiatry Investigation*, Vol. 15, 2018.

21. Tan, G., T. Dao, L. Farmer, R. Sutherland and R. Gevirtz, "Heart rate variability (HRV) and Posttraumatic Stress Disorder (PTSD): A Pilot Study", *Applied Psychophysiology and Biofeedback*, Vol. 36, pp. 27–35, 2011.

22. Luijcks, R., H. Hermens, L. Bodar, C. Vossen, J. van Os and R. Lousberg, "Experimentally Induced Stress Validated by EMG Activity", *PloS one*, Vol. 9, No. 4, pp. 1–8, 2014.

23. Zontone, P., A. Affanni, R. Bernardini, A. Piras and R. Rinaldo, "Stress Detection Through Electrodermal Activity (EDA) and Electrocardiogram (ECG) Analysis in Car Drivers", *27th European Signal Processing Conference (EUSIPCO)*, pp. 1–5, Coruña, Spain, 2019.

24. Ekman, P., ""Differential Communication of Affect by Head and Body Cues", *Journal of Personality and Social Psychology*, Vol. 2, No. 5, pp. 726–735, 1965.

25. Marks, A., D. M. Vianna and P. Carrive, "Nonshivering Thermogenesis without Interscapular Brown Adipose Tissue Involvement During Conditioned Fear in the Rat", *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, Vol. 296, No. 4, pp. R1239–R1247, 2009.

26. Oka, T., K. Oka and T. Hori, "Mechanisms and Mediators of Psychological Stress-Induced Rise in Core Temperature", *Psychosomatic Medicine*, Vol. 63, No. 3, pp. 476–486, 2001.

27. Hansen, J. H. L. and S. Patil, *Speech Under Stress: Analysis, Modeling and Recognition*, pp. 108–137, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

28. Wang, Y., N. Botros and I. Shahin, "Speech Recognition under Stress.", *International Conference on Bioinformatics & Computational Biology*, pp. 855–859, Las Vegas Nevada, USA, 2009.

29. Wang, Y., H. Ai, B. Wu and C. Huang, "Real Time Facial Expression Recognition with Adaboost", *Proceedings of the 17th International Conference on Pattern Recognition (ICPR)*, Vol. 3, pp. 926–929, IEEE, Cambridge, UK, 2004.

30. Banerjee, S. and D. Woodard, "Biometric Authentication and Identification Using Keystroke Dynamics: A Survey", *Journal of Pattern Recognition Research*, Vol. 7, pp. 116–139, 2012.

31. Pusara, M. and C. E. Brodley, "User Re-Authentication via Mouse Movements", *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security*, p. 1–8, Association for Computing Machinery, New York, NY, USA, 2004.

32. Hernandez, J., P. Paredes, A. Roseway and M. Czerwinski, "Under Pressure: Sensing Stress of Computer Users", *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 51–60, Association for Computing Machinery, Toronto, Ontario, Canada, 2014.

33. Healey, J. and R. Picard, "Detecting Stress During Real-World Driving Tasks Using Physiological Sensors", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 6, No. 2, pp. 156–166, 2005.

34. Can, Y., N. Chalabianloo, D. Ekiz and C. Ersoy, "Continuous Stress Detection Using Wearable Sensors in Real Life: Algorithmic Programming Contest Case Study", *Sensors*, Vol. 19, p. 1849, 04 2019.

35. Wu, Y., M. Daoudi, A. Amad, L. Sparrow and F. D'Hondt, "Unsupervised Learning Method for Exploring Students' Mental Stress in Medical Simulation Training", *Companion Publication of the 2020 International Conference on Multimodal Interaction*, ICMI '20 Companion, p. 165–170, Association for Computing Machinery, New York, NY, USA, 2020.

36. Wang, K. and P. Guo, "An Ensemble Classification Model With Unsupervised Representation Learning for Driving Stress Recognition Using Physiological Signals", *IEEE Transactions on Intelligent Transportation Systems*, Vol. 22, No. 6, pp. 3303–3315, 2021.

37. Zheng, Y., X. Ding, C. Poon, B. Lo, H. Zhang, X. Zhou, G.-Z. Yang, N. Zhao and Y.-T. Zhang, "Unobtrusive Sensing and Wearable Devices for Health Informatics", *IEEE Transactions on Biomedical Engineering*, Vol. 61, No. 5, pp. 1538–1554, 03 2014.

38. Hart, S. G. and L. E. Staveland, "Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research", *Advances in Psychology*, Vol. 52, pp. 139–183, Elsevier, 1988.

39. Cohen, S., T. Kamarck and R. Mermelstein, "A Global Measure of Perceived Stress", *Journal of Health and Social Behavior*, pp. 385–396, 1983.

40. Spielberger, C., R. Gorsuch, R. Lushene, P. Vagg and G. Jacobs, *Manual for the State-Trait Anxiety Inventory (Form Y1 – Y2)*, Palo Alto, CA: Consulting Psychologists Press, Redwood City, USA, 1983.

41. New Hampshire, "Perceived Stress Scale", https://www.das.nh.gov/wellness/docs/, 2022, accessed at May 15 2022.

42. Can, Y. S., B. Arnrich and C. Ersoy, "Stress Detection in Daily Life Scenarios Using Smart Phones and Wearable Sensors: A Survey", *Journal of Biomedical Informatics*, Vol. 92, p. 103139, 2019.

43. Mozos, Ó. M., V. Sandulescu, S. Andrews, D. Ellis, N. Bellotto, R. Dobrescu and J. M. Ferrández, "Stress Detection Using Wearable Physiological and Sociometric Sensors", *International Journal of Neural Systems*, Vol. 27, No. 02, p. 1650041, 2017.

44. Garcia-Ceja, E., V. Osmani and O. Mayora, "Automatic Stress Detection in Working Environments From Smartphones x2019; Accelerometer Data: A First Step", *IEEE Journal of Biomedical and Health Informatics*, Vol. 20, No. 4, pp. 1053–1060, July 2016.

45. Gjoreski, M., H. Gjoreski, M. Luštrek and M. Gams, "Continuous Stress Detection Using a Wrist Device: In Laboratory and Real Life", *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct*, UbiComp '16, p. 1185–1193, Association for Computing Machinery, New York, NY, USA, 2016.

46. Seo, W., N. Kim, C. Park and S.-M. Park, "Deep Learning Approach for Detecting Work-Related Stress Using Multimodal Signals", *IEEE Sensors Journal*, Vol. 22, No. 12, pp. 11892–11902, 2022.

47. Wampfler, R., S. Klingler, B. Solenthaler, V. R. Schinazi and M. Gross, "Affective State Prediction Based on Semi-Supervised Learning from Smartphone Touch Data", *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, CHI '20, p. 1–13, Association for Computing Machinery, New York, NY, USA, 2020.

48. Lin, S., L. Faust, S. D'Mello, G. Martinez and N. V. Chawla, "MBead: Semi-supervised Multilabel Behaviour Anomaly Detection on Multivariate Temporal Sensory Data", *2020 IEEE International Conference on Big Data (Big Data)*, pp.

1089–1096, Atlanta, Georgia, 2020.

49. Peng, Y., F. Jin, W. Kong, F. Nie, B.-L. Lu and A. Cichocki, "OGSSL: A Semi-Supervised Classification Model Coupled With Optimal Graph Learning for EEG Emotion Recognition", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, Vol. 30, pp. 1288–1297, 2022.

50. Nie, F., X. Wang and H. Huang, "Clustering and Projected Clustering with Adaptive Neighbors", *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, p. 977–986, Association for Computing Machinery, New York, NY, USA, 2014.

51. Chen, X., F. Nie, G. Yuan and J. Z. Huang, "Semi-Supervised Feature Selection via Rescaled Linear Regression", *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, p. 1525–1531, AAAI Press, Melbourne, Australia, 2017.

52. Chen, X., G. Yuan, F. Nie and Z. Ming, "Semi-Supervised Feature Selection via Sparse Rescaled Linear Square Regression", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 32, No. 1, pp. 165–176, 2020.

53. Song, P., W. Zheng, Y. Yu and S. Ou, "Speech Emotion Recognition Based on Robust Discriminative Sparse Regression", *IEEE Transactions on Cognitive and Developmental Systems*, Vol. 13, No. 2, pp. 343–353, 2021.

54. An, S., A. Medda, M. N. Sawka, C. J. Hutto, M. L. Millard-Stafford, S. Appling, K. L. S. Richardson and O. T. Inan, "AdaptNet: Human Activity Recognition via Bilateral Domain Adaptation Using Semi-Supervised Deep Translation Networks", *IEEE Sensors Journal*, Vol. 21, No. 18, pp. 20398–20411, 2021.

55. Liu, D. and T. Abdelzaher, "Semi-Supervised Contrastive Learning for Human Activity Recognition", *2021 17th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, pp. 45–53, Coral Bay, Pafos, Cyprus, 2021.

56. Ding, Y., B. Jin, J. Zhang, R. Liu and Y. Zhang, "Human Motion Recognition Using Doppler Radar Based on Semi-Supervised Learning", *IEEE Geoscience and Remote Sensing Letters*, Vol. 19, pp. 1–5, 2022.

57. Can, Y. S., D. Gokay, D. R. Kılıç, D. Ekiz, N. Chalabianloo and C. Ersoy, "How Laboratory Experiments Can Be Exploited for Monitoring Stress in the Wild: A Bridge Between Laboratory and Daily Life", *Sensors*, Vol. 20, No. 3, 2020.

58. World Medical Association, "World Medical Association Declaration of Helsinki: Ethical Principles for Medical Research Involving Human Subjects", *JAMA*, Vol. 310, No. 20, pp. 2191–2194, 2013.

59. Bengio, Y., O. Delalleau and N. Le Roux, *Label Propagation and Quadratic Criterion*, pp. 193–216, MIT Press, Semi-Supervised Learning edn., January 2006.

60. Zhu, X. and Z. Ghahramani, "Learning from Labeled and Unlabeled Data with Label Propagation", http://www.cs.cmu.edu/zhuxj/pub/CMU-CALD-02-107.pdf, 2002, accessed at May 16 2022.

61. Pedregosar *et al.*, "Scikit-learn: Machine Learning in Python", *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830, 2011.

62. Rumelhart, D. E. and J. L. McClelland, *Learning Internal Representations by Error Propagation*, pp. 318–362, MIT PRESS, 1987.

63. Baldi, P., "Autoencoders, Unsupervised Learning, and Deep Architectures", *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, Proceedings of Machine Learning Research, pp. 37–49, PMLR, Bellevue, Washington, USA, 2012.

64. Van Der Maaten, L. and G. Hinton, "Visualizing Data Using T-SNE", *Journal of Machine Learning Research*, Vol. 9, pp. 2579–2605, 2008.

65. Zeiler, M. D., “ADADELTA: An Adaptive Learning Rate Method”, *arXiv preprint arXiv:1212.5701*, 2012.

66. Hochreiter, S. and J. Schmidhuber, “Long Short-Term Memory”, *Neural Computation*, Vol. 9, No. 8, pp. 1735–1780, 1997.

67. Lecun, Y., L. Bottou, Y. Bengio and P. Haffner, “Gradient-Based Learning Applied to Document Recognition”, *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

68. Zhao, J., X. Mao and L. Chen, “Speech Emotion Recognition Using Deep 1D & 2D CNN LSTM Networks”, *Biomedical Signal Processing and Control*, Vol. 47, pp. 312–323, 2019.

69. Yang, C.-H. and P.-Y. Chang, “Forecasting the Demand for Container Throughput Using a Mixed-Precision Neural Architecture Based on CNN–LSTM”, *Mathematics*, Vol. 8, No. 10, p. 1784, 2020.

70. Kingma, D. P. and J. Ba, “Adam: A Method for Stochastic Optimization”, Y. Bengio and Y. LeCun (Editors), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

71. Zhang, T., R. Ramakrishnan and M. Livny, “BIRCH: An Efficient Data Clustering Method for Very Large Databases”, SIGMOD ’96, p. 103–114, Association for Computing Machinery, New York, NY, USA, 1996.

72. Ester, M., H.-P. Kriegel, J. Sander and X. Xu, “A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise”, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD’96, p. 226–231, AAAI Press, Portland, Oregon, 1996.

APPENDIX A: COPYRIGHT PERMISSION GRANTS

For [57], the figure has been reused with permission from members of our research group and copyright holders Can, Y. S., D. Gokay, D. R. Kılı ç, D. Ekiz, N. Chalabianloo and C. Ersoy, "How Laboratory Experiments Can Be Exploited for Monitoring Stress in the Wild: A Bridge Between Laboratory and Daily Life", Sensors, Vol. 20, No. 3, 2020.