NOVICE PROGRAMMERS' EMOTION AND COMPETENCY ASSESSMENTS USING MACHINE LEARNING ON PHYSIOLOGICAL DATA

FATIMA JANNAT

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
KUALA LUMPUR

2022

NOVICE PROGRAMMERS' EMOTION AND COMPETENCY ASSESSMENTS USING MACHINE LEARNING ON PHYSIOLOGICAL DATA

FATIMA JANNAT

DISSERTATION SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF COMPUTER SCIENCE (APPLIED COMPUTING)

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY
UNIVERSITI MALAYA
KUALA LUMPUR

2022

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: ***Fatima Jannat***

Registration/Matric No.: ***17058604 (WOA180004)***

Name of Degree: ***Master of Computer Science (Applied Computing)***

Title of Dissertation: ***Novice programmers' emotion and competency assessments using machine learning on physiological data***

Field of Study: ***Artificial Intelligence***

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date: ***02/04/2022***

Subscribed and solemnly declared before,

Witness's Signature Date: ***02/04/2022***

Name:
Designation:

NOVICE PROGRAMMERS' EMOTION AND COMPETENCY ASSESSMENTS USING MACHINE LEARNING ON PHYSIOLOGICAL DATA

ABSTRACT

The technology of psycho-physiological measurement and Eye-tracking has opened up a wide range of possibilities for automating the prediction of human emotional state for a particular event. There is also growing interest in modeling machine learning and deep learning algorithms that can learn from user's data, understand and react to that individual's affective state. This research work has used novice programming learners' eye-tracking and Galvanic Skin Response (GSR) data in a novel approach. This work investigates the suitability and effectiveness of machine learning algorithms such as Multinomial Naive Bayes, KNN, Logistic Regression, Decision Tree for predicting levels of arousal intensity among the programmers and LSTM deep learning algorithm to classify the programmers according to their performance. Through experiments with the data-set, it was found that Multinomial Naive Bayes outperformed other supervised machine learning algorithms with 75.93% accuracy and 96.54% ROC while predicting levels of arousal intensity. Hyper-parameter tuning has been used in all the algorithms using k-fold cross validation to have the best accuracy and to avoid the over-fitting issue. The result implies a good connection between how a novice programmer goes through a programming problem and his/her emotional arousal at that moment. The Long Short-term Memory (LSTM) deep learning model was chosen for classifying programming learners according to their performance. LSTM model has the advantage of having internal memory suitable for longer sequences like our Eye-tracking and GSR data sequence. The LSTM model resulted in 65.71% test accuracy while classifying the students' performance.

Keywords: Emotion, Machine Learning, Deep Learning, Eye-Tracking, GSR.

PENILAIAN EMOSI DAN KECEKAPAN PENGATURCARA BARU MENGGUNAKAN PEMBELAJARAN MESIN PADA DATA FISIOLOGI

ABSTRAK

Teknologi pengukuran psiko-fisiologi dan penjejakan mata telah membuka pelbagai kemungkinan untuk mengautomasikan ramalan keadaan emosi manusia untuk peristiwa tertentu. Terdapat juga minat yang semakin meningkat dalam memodelkan pembelajaran mesin dan algoritma pembelajaran mendalam yang mampu mempejari dari data pengguna, memahami dan bertindak balas terhadap keadaan afektif individu tersebut. Penyelidikan ini telah menggunakan data penjejakan mata dan data Galvanic Skin Response (GSR) pengaturcara baru menggunakan pendekatan baharu. Penyelidikan ini menilai kesesuaian dan keberkesanan algoritma pembelajaran mesin seperti Multinomial Naive Bayes, KNN, Logistic Regression, Decision Tree bagi meramal tahap keamatan rangsangan emosi di antara pengaturcara baru dan algoritma pembelajaran mendalam LSTM untuk mengklasifikasikan tahap kecekapan pengaturcara baru mengikut prestasi mereka. Melalui eksperimen dengan set data ini, di dapati bahawa Multinomial Naive Bayes merupakan algoritma pembelajaran mesin yang terbaik dengan ketepatan 75.93% dan ROC 96.54% dalam meramalkan tahap keamatan rangsangan. Penalaan parameter hiper telah digunakan dalam semua algoritma menggunakan pengesahan silang k-kali ganda untuk memiliki ketepatan terbaik dan untuk mengelakkan masalah data berlebihan (overfitting). Hasilnya menunjukkan hubungan yang baik antara bagaimana pengaturcara baru menyelesaikan masalah pengaturcaraan dan tahap keamatan rangsangan emosinya pada ketika itu. Model pembelajaran mendalam Long Short-term Memory (LSTM) dipilih untuk mengklasifikasikan tahap kecekapan pengaturcara baru mengikut prestasi mereka. Kelebihan model LSTM adalah pada memori dalaman yang sesuai untuk urutan yang lebih panjang seperti urutan data penjejakan mata dan GSR. Model LSTM ini menghasilkan ketepatan ujian

sebanyak 65.71% dalam klasifikasi prestasi pelajar baru.

Kata kunci: Emosi, Pembelajaran mesin, Pembelajaran mendalam, Penjejakan mata (eye tracking), ‘galvanic skin response’.

ACKNOWLEDGEMENTS

All praises to Almighty Allah, the Most Glorified, the Most High, and the Benevolent for His blessings, guidance, and for giving me the strength and opportunity to finish this research work as a requirement to complete my master's degree program.

My heartiest gratitude, profound indebtedness and deep respect go to my supervisor, Dr. Unaizah Hanum Binti Obaidellah, for her constant supervision, affectionate guidance, encouragement and motivation. Her continuous support, keen interest on the topic and valuable advises throughout the study was of great help in completing this dissertation. I would also like to express my sincere gratitude to Dr. Aznul Qalid Bin Md Sabri for helping me with different machine learning approaches.

I would like to thank my parents, my parents-in-law and my siblings for their continuous support, prayer and encouragement which made my dissertation journey easier.

Lastly, I would like to dedicate my dissertation to my beloved mother, Karniz Fatema and my dearest husband, Saad Bin Bashar who inspired me every single day and believed in me even in my tough days.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AOI	:	Area of Interest.
CNN	:	Convolutional Neural Network.
DL	:	Deep Learning.
ECG	:	Electrocardiogram.
EEG	:	Electroencephalogram.
ET	:	Eye Tracking.
GSR	:	Galvanic Skin Response.
KNN	:	K-Nearest Neighbors.
ML	:	Machine Learning.
OP	:	Output.
Oper	:	Operations.
PS	:	Problem Statement.
RNN	:	Recurrent Neural Network.
ROC-AUC	:	Receiver Operating Characteristic - Area Under Curve.
SVM	:	Support Vector Machine.
Var	:	Variables.

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

The structure of this chapter is started with the background and motivation of the proposed research which gradually developed with the problem statement, the objectives of the research, the research questions, the scope of the research, and the research significance. Finally, it is completed by the organization of the dissertation.

1.1 Background & Motivation

Programming language learning is a grand challenge for the students as it requires multiple skills, thinking ability and knowledge. Programming language learners have to face different types of difficulties during their learning process specially during solving programming related problem and it is found in different researches (McGettrick et al., 2005; Robins et al., 2003) that rate of drop-out in this course is very high. Again students of different levels from novice to expert, prefer different type of methods when they are asked to solve a programming question. Therefore, learning computer programming demands a high level of cognitive capabilities across different students, especially for novices (Renumol et al., 2009). At present, almost every engineering and technology based degree require programming learning course, therefore the quantity and types of programming language learning students are increasing at high rate. It has become essential to design the programming learning course structure according to the classification of students type.

The technology of psycho-physiological measurement is getting more popular due to its implementation in various fields that require understanding and analysing human behavior. It examines how human body works in terms of cognition and behavior in any particular situation (Strumwasser, 1994). Various tools like Galvanic Skin Response (GSR), Electroencephalogram (EEG), Electrocardiogram (ECG), Eye Tracking (ET), facial expression analysis are used to gain insights in the application of neuroscience. Deep

and unique understandings in visual attention and cognitive processes can be obtained with eye-tracking data. An active, visual and sequential learning style is found in the findings by Norwawi et al. (2009) among the learners who score high in programming courses. On the other hand, GSR has been used to measure non-conscious emotional intensity. It provides the arousal level and emotional state of a participant in the controlled environments. Our skin exhibits a wide range of information whenever we are emotionally aroused. GSR which is also known as Skin Conductance (SC) or Electrodermal Response (EDR) or Psychogalvanic Reflex (PGR) or Electrodermal Activity (EDA), measures the change in our skin's electrical conductivity from the sweat glands when an emotional arousal happens.

On the other hand, Eye-tracking is a sensor technology which measures an user's eye gaze behavior such as staring, looking and reading to behavior and determines an individual's cognitive process. It can detect the amount of visual attention and focus of the user (Dalrymple et al., 2019). An eye tracker measures the movements and position the of the eyes via sending infrared light to the users eyes. Then, the eye tracker's camera records the reflection of that light in the users eyes. Thus, an eye tracker can provide the information about how long a user is looking into a particular stimuli and how the user's attention travel from one area to another.

Combination of the information from GSR and Eye-tracking data can generate more accurate and different dimensions of information while examining any behavioural performance. Together these two can identify how intensely a participant is feeling a sentiment (arousal/non-arousal) while looking (attention) at a particular place.[Figure 1.1]

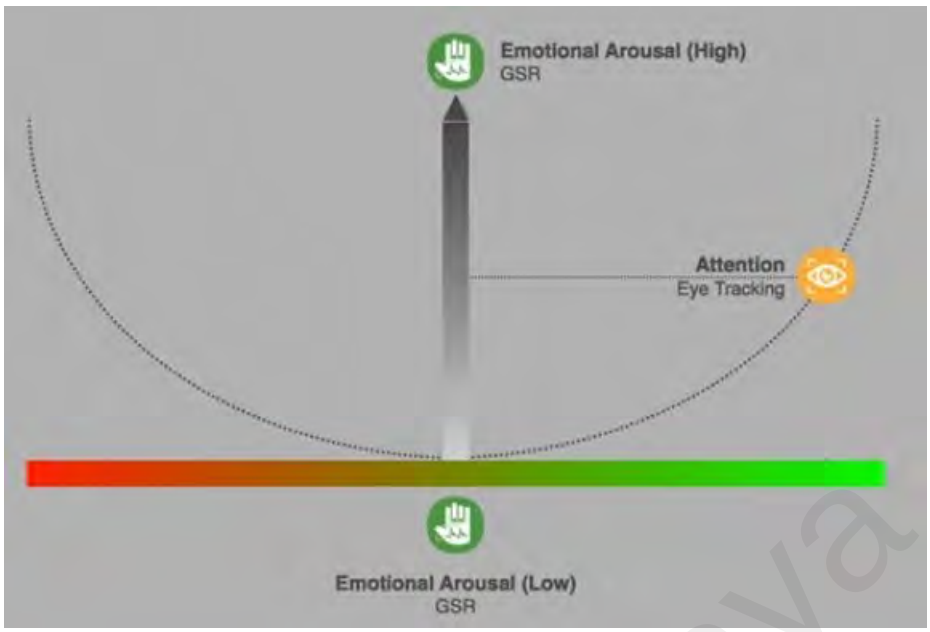

Figure 1.1: Relation between arousal and attention (*Empathy 2.0 series: How biometrics can help you understand your customers*, 2019)

Machine Learning (ML) approaches that improve automatically through experience and Deep Learning (DL) methods based on artificial neural networks are still emerging topics in the field of analyzing the data gathered from both Eye-tracking and psycho-physiological measurement studies.The use of machine learning algorithms and deep learning model can provide a better and accurate understanding of behaviours among the novice programming learners combining both type of measurements (ET & GSR). Therefore, this research work attempts to explore and contribute in the field of Eye-tracking and GSR approach for assessing novice programmers' state of emotion and competency using machine learning and deep learning techniques.

1.2 Problem Statement

In this era full of many mental health issues, the current programming language learning method in our education system lacks proper mental schema and has still many areas to get improved. The conventional learning method for novice programmers still holds some questions such as if it can identify how a novice programmer's emotional state can be while

solving a programming problem (section 1.3, Research objective 1) and if it can identify a certain category of novice programmers according to their performance (section 1.3, Research objective 2). One of the major problems for learning programming language is the generalization of all the learners and the inability to fulfill the high cognitive requirements especially for novice learners. Novice students like the first year undergraduate students attend university with various degrees of previous knowledge and experience. Hence, their level of performance is expected to vary according to their prior knowledge and ability to adapt to new techniques. A programming question that is expected to be “easy" for a group of students can be felt “hard" by other students. Therefore, the learning style should be accustomed to the type of students’ **performance** and their **emotional state**. Therefore, it would be helpful if there is a rich method that can identify students’ emotional states. Physiological data like eye-tracking and arousal measurements are considered suitable for identifying the above-raised questions but generally, these types of physiological data have larger data sizes. Although machine learning methods have more advantage in case of larger data set and more accurate in terms of prediction, most of the prior works adopted statistical methods for analysing the data (section 2.1). Keeping these issues in mind, the goals and objectives of this research work have been fixed to try to investigate students’ emotional state on programming activity with machine learning and deep learning techniques.

1.3 Research Objective

Based on the problem statement, the main objectives of this research are stated as follows-

1. To identify the best performing single supervised machine learning algorithm that produces the highest prediction performance for identifying emotional arousal

among the novice programmers using eye-tracking data.

2. To classify the programmers' level according to performance (high or low) based on eye-tracking (ET) and GSR data using Long Short Term Memory (LSTM-RNN) deep learning technique.

1.4 Research Question

This research work will try to answer the following research questions based on the research objectives mentioned in the previous section:

RQ-1: Can gaze features be used to predict a novice programmer's arousal while looking into a particular stimulus event and area of interest (AOI)?

RQ-2: What is the best type of supervised algorithm for classifying arousal among the novice programmers using Eya-tracking features?

RQ-3: Can deep learning technique find a common sequence among the ET-GSR data and categorise the novice programmers' performance?

RQ-4: Why Long Short Term Memory (LSTM-RNN) model can be estimated as an appropriate deep learning model for research objective-2?

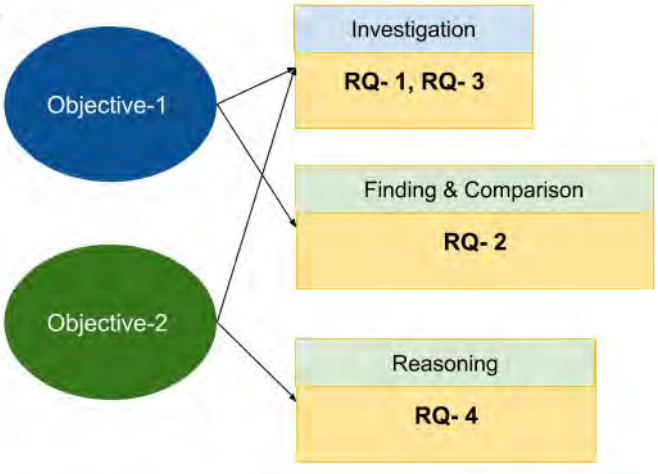

Figure 1.2: Mapping of Research Objectives and Research Questions

Figure 1.2 shows how the above mentioned research questions can be mapped with the research objectives based on the stages of a scientific study like investigation, finding, comparison and reasoning. Investigation through literature reviews and experiments on different algorithms are required to answer the RQ-1 (Research Objective- 1) and RQ-3 (Research Objective- 2). Findings from experiments and result comparison are requisite to answer the RQ-2. Lastly, the answer for the RQ-4 can be discovered by reasoning procedure through formulating logical judgment from the combination of theory, knowledge from prior related works and result analysis.

1.5 Research Scope

The scope of this research is restricted to the eye-tracking (ET) data and GSR data collected in the work (Obaidellah et al., 2019) from the 36 students of University of Malaya, exploration of the machine learning and deep learning classifiers used for optimal solution for predicting arousal and students' performance, the various findings evaluation metrics (Accuracy, Receiver Operating Characteristic - Area Under Curve (ROC-AUC), Precision, Recall, confusion matrix, F1 score etc) used for measuring the performance of the classifiers. The participants recruited to provide data for this work are considered novice programming language learners who passed only one semester and were new to solving programming problems (Figure 1.3). All of these students were majoring in computer science at the time of data collection. The duration of this research work is 14 months.

1.6 Research Significance

The main significance of this research work will be able to predict the novice programming learners' performance (high or low) basing on their emotional behaviour and visual attention. This prediction can be used to contribute to structuring programming

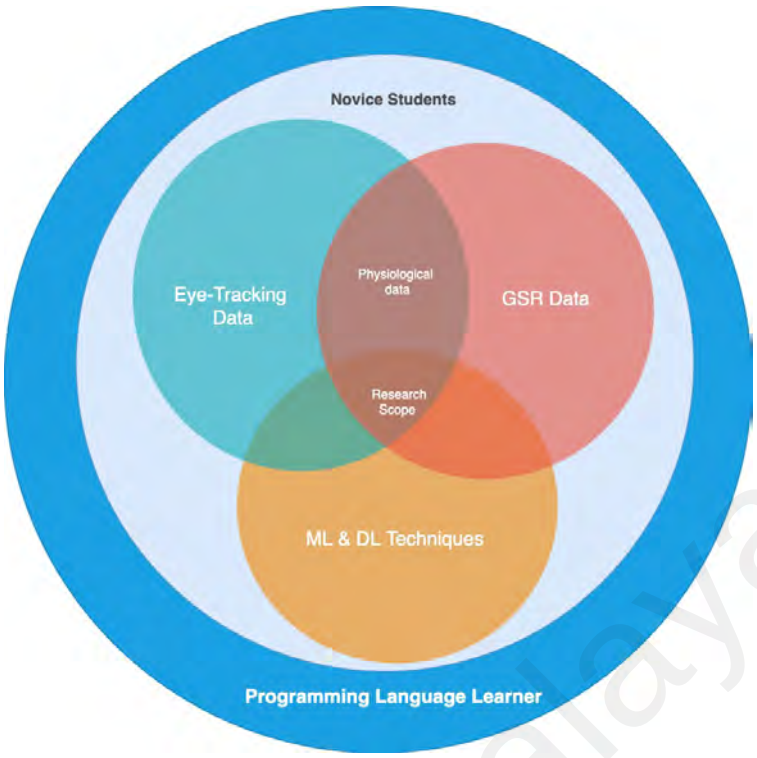

Figure 1.3: Visual representation of research scope

learning methods, designing programming problems, developing any e-learning material or programming tools suitable for students with different performance level.

The work in Renumol et al. (2009) found that, different students of various types require high cognitive capabilities during learning computer programming. This research work will contribute to establishing this existing result more strongly. Moreover, this work will focus on how we can combine the novice students' emotional behaviour and visual attention, which will add a footmark to the application of neuroscience.

Another significance of this work is, it will contribute to machine learning and deep learning fields by providing a multi-level analysis (confusion matrix and F score) of the classifiers while combining GSR and ET data. The predictive model for performance or score prediction using gaze data will also add a new dimension to machine learning applications.

1.7 Dissertation Organization

The dissertation is organised as follows: Chapter 1 presents the background and motivation, problem statement, research objective and questions, scope and, significance of this research work. A brief overview of literature reviews related to Eye-tracking, GSR (or other arousal measurements), Eye-tracking, arousal measurements and machine learning techniques, deep learning model and lastly summary of all the reviews are discussed in Chapter 2. Following this, the research methodology is presented in Chapter 3 along with the Eye tracking and Galvanic Skin Response data-set collection, raw data and data preparation. Chapter 4 explains the tools used for data analysis, correlation analysis, T-Test analysis detailed results of the experiment of supervised machine learning algorithms and deep learning models. The discussion of the experimental results is depicted in Chapter 5 including the implications of these results. Lastly, the conclusion followed by the research limitations and gaps for future research is presented in Chapter 6.

CHAPTER 2: LITERATURE REVIEW

Many recent researchers (Grawemeyer et al., 2017; Post et al., 2019) have investigated the effect of psycho-physiological measurement to compute affective states (valence, arousal, and motivational intensity) in various learning environments. To predict users' experience in a particular field, the combination of affective and cognitive measurement has provided better result (Ahn & Picard, 2014).

In this chapter, previous works related to Eye-tracking, GSR and other arousal measurements are reviewed. The usage of different machine learning algorithms and deep learning models for various applications using Eye tracking measurement or arousal measurement or combination of both types of measurements are also briefly discussed. A detailed summary has been shown in the last part of this chapter to understand the significant and relevant methods to this research topic.

2.1 Eye-tracking

In Hou et al. (2015), eye-tracking data was used to explain what are the possible reasons that make the video viewers to decide "like/dislike" decisions. Jaques et al. (2014) used eye-tracking data to predict emotions like 'boredom' and 'curiosity' related to learning for an intelligent tutoring system (ITS). The above two works show eye-tracking data contribution in other fields except programming learning. Here a brief discussion on use of eye-tracking data in case of computer programming is given.

Busjahn et al. (2014) used qualitative Analysis with ELAN and VETOOLS to analyse three types of fixation metrics- fixation count, fixation spatial density, and mean fixation duration of programming learners. In this study, the authors presented eye tracking as a way of enriching computer education research. An analysis of eye movements' linearity pattern while reading code of novice and expert programmers is conducted in Busjahn,

Bednarik, et al. (2015). Findings from Busjahn, Bednarik, et al. (2015) testify that there are differences between reading source code and natural language. The result also shows that expert programmers have increased number of non-linear reading skills than novice programmers.

Busjahn, Schulte, et al. (2015) used confirmation strategy by taking recording of novice programmers eye-tracking data three times over a period of time. Like the work in Busjahn, Bednarik, et al. (2015), this paper also proved that code reading is exceptional and if any novice learner starts with story-reading (strategy to read linearly) then later if he/she advance to code reading, he/she might be considered as becoming an expert code reader.

2.2 Arousal measurements based on GSR or similar

There are a few studies on computer programming that only worked with arousal using any of the measures like- Electrodermal activity (GSR), Electroencephalography(EEG), Electrocardiography (ECG) , Electromyography (EMG) etc.

Physiological measurements EEG and GSR are used in Yousoof and Sapiyan (2013) to estimate the cognitive load of the novice programming learners. This work by Yousoof and Sapiyan (2013) analyzed the difference between visualization and normal mood of learning programming. For this purpose, the participants were experimented with different visualizations using different tools like Jeliot, Ville and Teaching Machine. The findings indicate that visualization has much impact on programming learners.

Khan et al. (2006) found that ability of the programmer to identify and debug code errors depends on their level of arousal and valance has very less effect in this case. The work in Khan et al. (2006) used Self- Assessment Manikin (SAM) method to find the arousal-valance of the participants.

2.3 Eye-tracking and arousal measurements based on GSR or similar

The use of eye tracking and arousal measurement can be found in many real life applications especially more in the application of marketing research or advertising. The study in Edwards et al. (2017) used eye tracking and galvanic skin response recording along with facial expression analysis to monitor those students' behaviour who learn through online medium like video lectures. The observation from Edwards et al. (2017) revealed that even though gathering eye-tracking and GSR records can be a difficult process, GSR provided more accurate records about the student's behaviour.

Not a massive number of research work can be found in the domain of novice programming learners where combination ET and arousal measurement data are used. Eye-tracking, EEG and GSR are used together in Fountaine and Sharif (2017) to classify the emotional state of developers. Though Fountaine and Sharif (2017) does not directly relate with programming learners but the findings shows how code reading can affect emotional state of the developers. Fountaine & Sharif used qualitative and quantitative measurement for emotion data analysis in this research work.

2.4 Eye-tracking, arousal measurements and machine learning techniques

Using EEG and Eye Tracking Data, the authors presented a multimodal emotion recognition in Zheng et al. (2014) to classify positive, neutral and negative emotional states while watching video clips. The authors extracted different features including power spectral density (PSD), differential entropy (DE), differential asymmetry (DASM) , rational asymmetry (RASM) and ASM (concatenation of DASM and RASM) features for EEG signals and PSD, DE features for eye tracking data. Support Vector Machine (SVM) and Linear dynamic system (LDS) machine learning algorithms were trained to predict those three emotional states. This work employed feature level and decision level fusion strategies to build the models which had achieved 73.59% and 72.98% accuracy rates,

respectively.

The research work by Chmielewska et al. (2019) resulted in 68% accuracy in classification of eye-tracking data using various machine learning classifiers for visual perception of architectural spaces. Popular machine learning classifiers such as Support Vector Machines (SVM), Naïve Bayes (NB), Logistic Regression (LR) and Random Forests (RF) are trained and tested in this reviewed work.

Handri et al. (2010) used GSR data to evaluate students' physiological responses for course material of e-learning. The authors used Discriminant Analysis (DA), Support Vector Machine (SVM) and K-Nearest Neighbors (KNN) classifiers for classification where Discriminant Analysis (97.06%) and SVM(94.12%) classifier gave more accurate result than K-Nearest Neighbor (79.42%).

In Smets et al. (2015), six machine learning techniques such as LR, SVM, DT, RF, BN (static and dynamic) were tested for stress detection in a controlled environment to identify the optimal computational methods. The authors measured electrocardiogram (ECG), Galvanic skin response (GSR) and temperature and respiration for stress test. At the end of the research, they suggested using dynamic Bayesian networks (BN) and generalized SVM for stress level detection.

Fritz et al. (2014) is the most relatable work which used all the elements that this proposed work will be using in the application of researching behaviour of programmers (i,e eye-tracking, EDA-EEG (for arousal measurement) and machine learning techniques). However, the objective of this reviewed work and the proposed work are completely different. Fritz et al. (2014) classified the difficulty of programming tasks using psycho-physiological sensors and used Naive Bayes classifier, J48 decision tree, Support Vector Model machine learning algorithms.

A brief comparison among the related works is shown in table 2.1.

Table 2.1: Comparison among related works on ET or other Physiological Measures measures

Paper	Research Objective	Target group	Physiological Measures	Method/ Algorithm used for analysis	Implication / Research Gap
Khan et al., 2006	Finding significance of arousal and valence in case of improvement in programmers' task performance	Programmers	Self-Assessment Manikin (SAM)	Statistical method - paired samples t-test	Suggested that the programmer's level of arousal has a great impact on the ability to find and correct errors in programming codes.
Norwawi et al., 2009	Classifying programming language student's performance according to learning style	Programming student	No Physiological Measures	J48 decision tree , K-means cluster	**Findings**- top students' performance has a visual, active and sequential learning style. **Gap**- The classification result of students performance is not broadly described in the paper.
Busjahn et al., 2014	Case study on program comprehension and preliminary analyses	Programmers also novice	Eye-tracking	Qualitative Analysis with ELAN and VETOOLS	Only one programming program has been used for the experiment

Paper	Research Objective	Target group	Physiological Measures	Method/ Algorithm used for analysis	Implication / Research Gap
Fritz et al., 2014	Classification of the difficulty of programming tasks using psycho-physiological sensors	Programmers	Eye-tracking, EDA, and EEG	Naive Bayes classifier, J48 decision tree, Support Vector Model	The result in this work found that it is possible to predict difficulty of a programming task accurately using psycho-physiological measures. Also eye-tracker has the most predictive power than any other sensor. **Gap**- the result only can be generalized for professional programmers not for students, or novice software developers
Busjahn, Schulte, et al., 2015	Analysis of the Novice's Gaze during programming problem	Novice programmers	Eye-tracking	Qualitative and Quantitative Analysis	**Strength-** Analysed for a long period of time and collected novice eye-tracking data thrice- first week of the course (when the novice read through the code twice linearly) , middle of the course, (after learning data types), end of the course, (after learning about iteration and nested loop).

Paper	Research Objective	Target group	Physiological Measures	Method/ Algorithm used for analysis	Implication / Research Gap
Busjahn, Bednarik, et al., 2015	Analysis of eye movements pattern (by linearity) while reading code	Novice and expert programmers	Eye-tracking	Statistical method - Wilcoxon signed-rank test, Mann-Whitney test	**Strength-** Comparison between novice and expert programmers.
Huysmans et al., 2018	Classification of mental stress detection	Not specific	GSR, ECG	Self-organizing map(SOM)	Only basic two classes (relax and stress) have been estimated
Obaidellah et al., 2019	Classifying programming problem solving pattern	Novice programmers	Eye-tracking	Quantitative and qualitative methods (scarf plots)	Though the visual analysis of eye-tracking data gave an idea of gaze pattern behavior of the novice programmers during problem solving, a clearer classification of programmers strategies was difficult to identify due to overcrowded of long tasks and many participants data.

2.5 Deep learning model

This section shows the popular deep learning methods used for Eye tracking and arousal measurement data. Generally, the psycho-physiological measurement sensors and eye tracker sensor record information for each millisecond which create a larger set of data. This kind of big data set is appropriate for deep learning models. On the other hand, more accurate mean values, higher and more realistic accuracy rate and a smaller margin of error can be achieved while doing hyperparameter tuning of a deep learning model if the data set is large enough.

The amount of existing research work for predicting emotion with deep learning techniques are not huge. Several research works trained Eye-tracking data with deep learning models with various goals like- predicting confusion (LSTM, Gated Recurrent Unit (GRU); Sims et al. (2019)), predicting anger, disgust, fear, happy, sad, surprise, or neutral (LSTM; Long et al. (2017)), predicting saccade gaze (Convolutional Neural Network (CNN), LSTM - RNN; Ngo and Manjunath (2017)), identifying Autism spectrum disorder (ASD) (LSTM; Carette et al. (2017)) and visual scanpath generation on images (CNN, LSTM; Verma and Sen (2019)).

Again deep learning technique is used in many works where psycho-physiological measurement data (GSR, EEG, ECG) has been used. In Davidson et al. (2006), the authors used a LSTM Recurrent Neural Network (RNN) model to train EEG data for predicting micro-sleep behavior among the drivers. A hybrid CNN-LSTM joint learning model is used in Hong et al. (2019) to train GSR and video clips data to predict personality and classify sentiment among the users. After reviewing a good number of research works, it can be considered that Long Short Term Memory is one of the most popular and proper models in case of training eye-tracking data and psycho-physiological data.

Table 2.2: Related works using deep learning techniques

Paper	Data Type	Used DL Model	Objective
Davidson et al. (2006)	EEG	LSTM	Predicting micro-sleep behavior among the drivers
Long et al. (2017)	Eye Tracking	LSTM	Predicting anger, disgust, fear, happy, sad, surprise, or neutral
Ngo and Manjunath (2017)	Eye Tracking	CNN, LSTM	Predicting saccade gaze
Carette et al. (2017)	Eye Tracking	LSTM	Identifying Autism spectrum disorder (ASD)
Verma and Sen (2019)	Eye Tracking	CNN, LSTM	Visual scanpath generation on images
Sims et al. (2019)	Eye Tracking	CNN, GRU	Predicting Confusion
Hong et al. (2019)	GSR	Hybrid CNN-LSTM	Predict personality and classify sentiment among the users

Table 2.2 depicts a summary of related literature using deep learning techniques. The summary shows that most of the related research work preferred to use LSTM deep learning model. Most of the works related to behavior data demand the necessity of memory for long sequence of information. Their requirements can be fulfilled by LSTM deep learning model which has feedback connections to facilitate the learning of long-term dependencies. Section 4.3 will discuss briefly regarding why LSTM being one of the appropriate models for this work's dataset and objective.

2.6 Summary of related works

According to the background studies, it is found that even though many researchers worked with eye-tracking and GSR based data in various applications but very few research

works have combined both of these two with machine learning techniques (table 2.2). Though Fritz et al. (2014) has worked with eye-tracking data, EDA (Electrodermal Activity) and EEG (Electroencephalogram) data, the objective was to classify the difficulty of programming tasks. On the other hand, the main goal of this research work is to combine both eye-tracking and GSR in the application of predicting novice programming language students' arousal using fixation data during solving programming tasks and to classify their performance using eye-tracking information and GSR data. A similar work for classifying programming problem solving pattern using eye-tracking data can be seen in the work of Obaidellah et al. (2019). However, this work did not combine any arousal data and also did not use any machine learning technique. On the other hand, to our best knowledge there is no previous work that has worked with eye-tracking and GSR data to predict a novice programmer's performance using deep learning techniques. In brief, the proposed work is considered novel in the application of assessing novice programmers' state of emotion and competency by combining eye-tracking, GSR, machine learning and deep learning approach.

Table 2.3: Mapping of techniques and elements used in various related works

Topic	Davidson et al. (2006)	Khan et al. (2006)	Norwawi et al. (2009)	Fritz et al. (2014)	Huysmans et al. (2018)	Busjahn et al. (2014)	Busjahn, Bednarik, et al. (2015)	Busjahn, Schulte, et al. (2015)	Obaidellah et al. (2019)	Sims et al. (2019)	Long et al. (2017)	Ngo and Manjunath (2017)	Carette et al. (2017)	Verma and Sen (2019)	Hong et al. (2019)	Proposed Work
Research area for Programming Learners	X	X	✓	X	X	✓	✓	✓	✓	X	X	X	X	X	X	✓
Eye-tracking	X	X	X	✓	X	✓	✓	✓	✓	✓	✓	✓	✓	✓	X	✓
GSR (or other measurement for arousal)	✓	✓	X	✓	✓	X	X	X	X	X	X	X	X	X	✓	✓
ML technique	-	X	✓	✓	✓	X	X	X	X	-	-	-	-	-	-	✓
DL technique	✓	X	X	X	X	X	X	X	X	✓	✓	✓	✓	✓	✓	✓

CHAPTER 3: RESEARCH METHODOLOGY

This chapter discusses about the Eye Tracking and GSR data collection procedure, stimuli description, data preparation, data cleaning steps, down sampling the huge size of data, arousal or non-arousal data calculation from GSR Phasic response, feature selection and feature scaling method for data analysis.

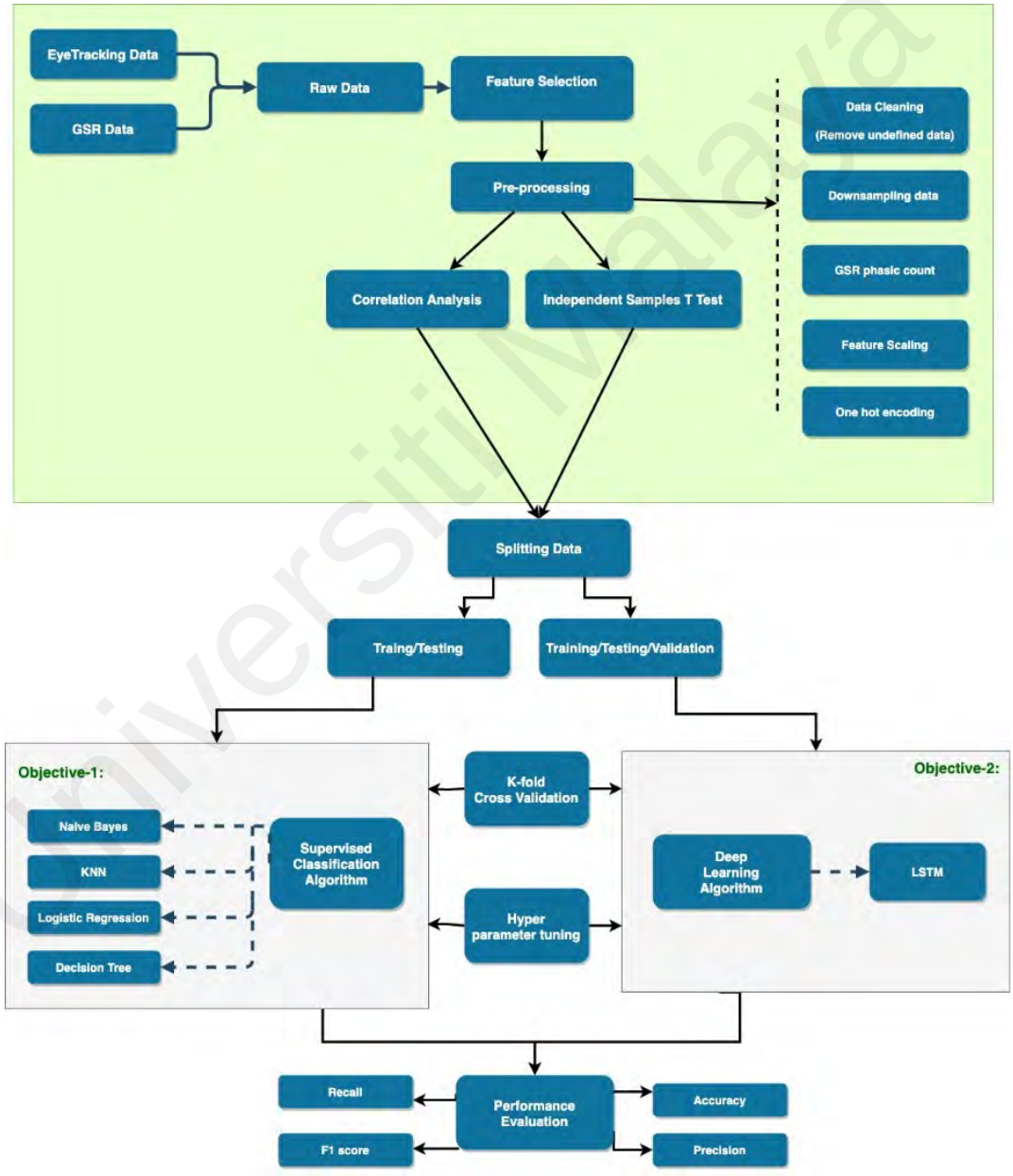

Figure 3.1: Proposed work-flow

To achieve the stated research objectives in Chapter 1, the research workflow can be

divided into few major phases: Raw data pre-processing, Feature Extraction, Correlation analysis, Training, Validation and Testing. The output or results or decisions from each phase have been used in the next phases. A brief architecture of proposed work has been shown in Figure 3.1. Each phase is described elaborately in this chapter subsequently.

3.1 Eye-tracking and GSR data collection

This section describes the data acquisition procedures for further analysis. This research work is using the eye-tracking and GSR data set that have been obtained during the data collection process of Obaidellah et al. (2019). Though the same eye tracking data has been used in Obaidellah et al. (2019), the authors did not use any GSR data in that research work. The data has been collected from total 36 first year computer science students of University of Malaya. There were total 14 male and 22 female student with age range 19-24 year (M_{age} = 21, SD_{age} = 1.2). The students were taking their fundamental programming learning course at that time. Therefore, they can be considered as novice programmers.

The flow of data collection procedures has been depicted in Figure 3.2. In the initial phase, novice programming language learners were exposed to nine programming problems with different difficulty level in a controlled environment. The nine stimulus are: Easy-1, Easy-2, Easy-3, Medium-1, Medium-2, Medium-3, Hard-1, Hard-2, and Hard-3. The novice programmers were familiar with the topics of the programming problems presented in all the stimulus.

It is very important to have a proper environment setup before collecting behavioral measurement data like Eye tracking and GSR data. A proper and controlled environment increases the validity of the research work. The data collected from one novice programmer at a time and in a room without any disturbance.

GP3 Gazepoint eye tracker with a sampling rate of 60 Hz was used for eye-tracking data and Shimmer3 GSR+ were used for collecting arousal data. Initially, the eye tracker

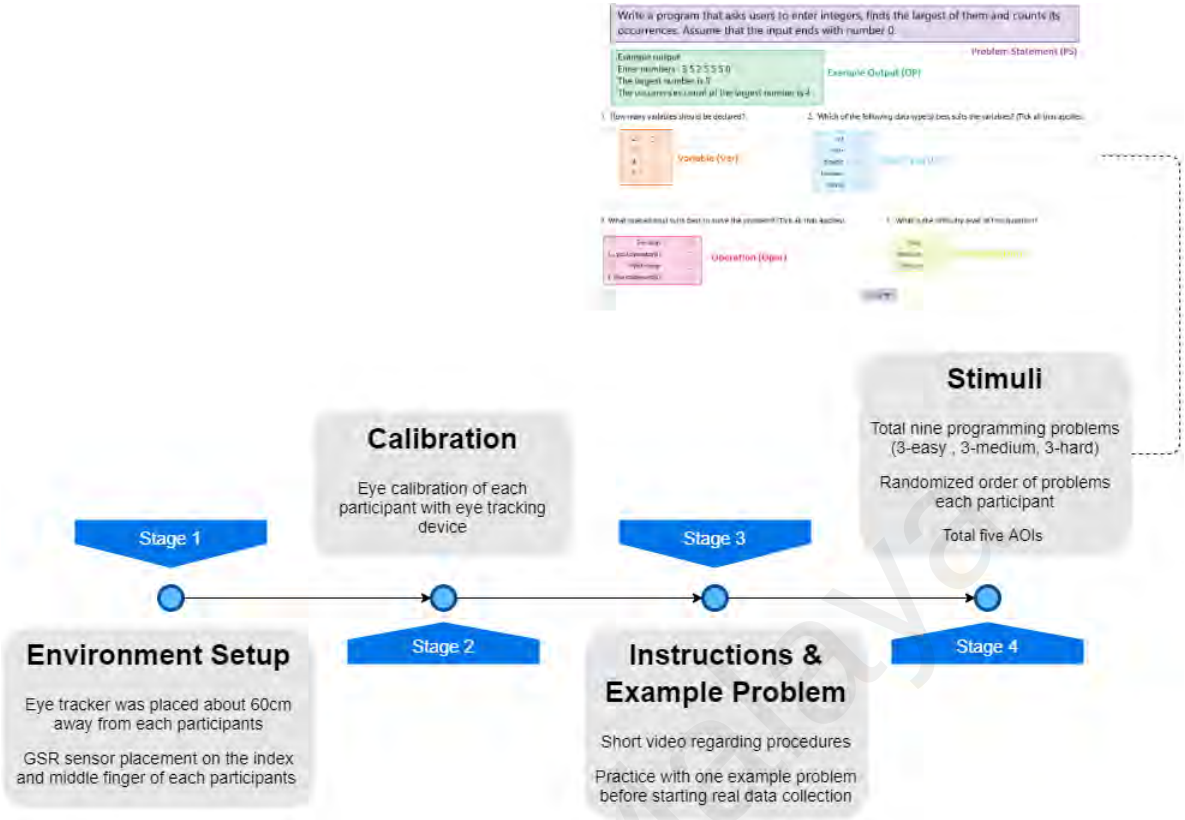

Figure 3.2: Data collection procedure

was placed about 60 cm away from each novice programmers. Then calibration between each programmer's eye and the eye tracker was performed. At the time of calibration, the eye-tracker identifies how the inferred light is reflected on each participants eyes. This is a very important step before recording gaze data. The Galvanic Skin Response sensor was placed on the index and middle finger of each participant's hand to have the recording of arousal data.

Before starting collecting Eye Tracking and GSR data, the novice programmers were given necessary instructions about the sensors and procedure and shown a demo video before they started solving the programming problems. Each participant solved one example stimuli before starting the main nine stimulus to get familiar with the procedure and environment. As GP3 Gazepoint eye tracker and Shimmer3 GSR+ were two different types of sensor with two different frequency rate, it was required to combine the recorded data of both of them with a third party software (i.e iMotions). Both the sensors were

integrated with the iMotions software which was used to streamline the timestamp for both sensors.

Each student's score and total time taken to solve each problem were also recorded. All the students had to solve total nine questions (three easy, three medium, three hard) and each contained 100 marks. The average highest marks of total nine questions obtained by a single student is 85% and lowest mark is 59%. Basing on the percentage of passing rate (> 50% marks), the students are classified into **high** (≥70% marks) and **low** (<70% marks) classes according to their performance. The grading system from University of Malaya has been followed in this case where 50% marks is required for **Pass** grade and 70%+ marks is required for **Distinction** grade. A summary of the samples' demographics is given in the Table 3.1.

Table 3.1: Samples' Demographics (Obaidellah et al., 2019)

Total no of participants	36
Male	14
Female	22
Study Qualification	1st year student
Highest Marks obtained (Avg. of total nine questions)**	85%
Lowest Marks obtained(Avg. of total nine questions)**	59%

3.1.1 Stimuli Description

Any item that is used to collect reaction from a group of respondents in any research study with a particular research objective is known as stimulus (stimuli in plural). Stimuli

can be any materials like website, question papers, audio, video or any particular product related to the research objective inside a research setting. In case of human behavior research, the materials used to evoke a reaction from a focus group or participants in the study is known as stimuli. In this research, the stimuli are various types of programming problem questions related to the students' programming learning course. The description of the stimuli is given in this subsection.

There was a total nine different programming problems where three of them were easy problem, three problems were of medium difficulty and the remaining three were hard problems. Each stimuli were divided into six Area of Interest (AOI) (Obaidellah et al., 2019): Problem Statement (PS), Example Output (OP), Variables (Var), Data Type (DT) , Operations (Oper), and Difficulty level (Diff) [Figure 3.3]. An area of interest (AOI) refers to some particular sub-regions of the stimuli which has most importance while extracting important information or metrics. It is important to select proper area of interest in such way that irrelevant data are not included and also important information is not missed during eye-tracking data collection procedure.

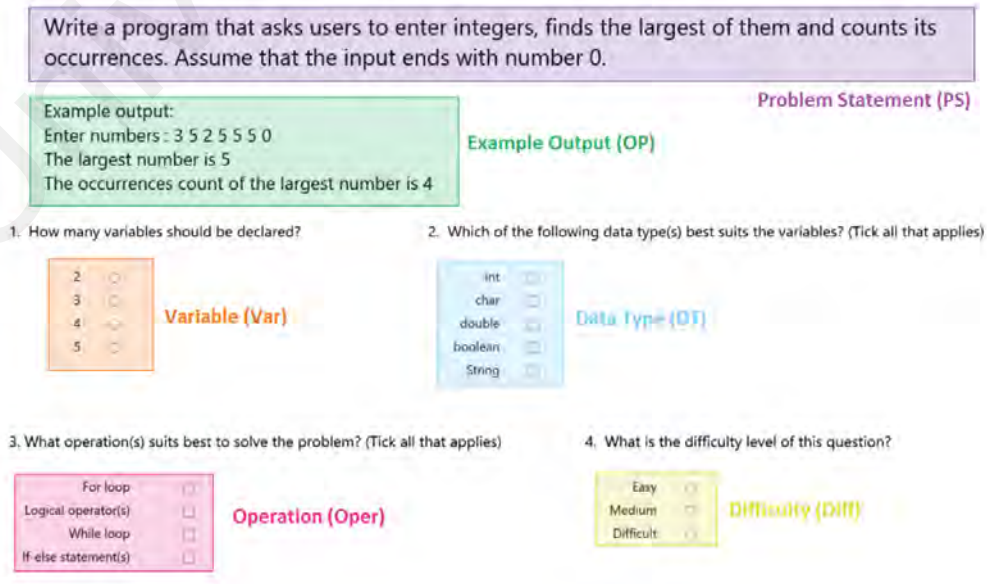

Figure 3.3: Example of a stimuli with AOIs

Area of Interests (AOI) were drawn on the major areas of the stimuli where participants were expected to focus on. The eye-tracking data identifies in which sequence of AOIs the programmers are looking into a stimuli. For example- if a programmer went through a stimuli in strictly Top-to-Bottom and Left-to-Right manner, the fixation sequence can be PS » OP » Var » DT » Oper » Diff. The sequence of AOIs is an important feature for this research work because it can be used to investigate if a particular class of programmer has a common pattern of looking sequence for research objective 2.

3.1.2 Raw Data

Different eye-trackers provide different number of features during data collection. Gaze point axis, Gaze AOI, Saccade sequence, Saccade Duration, Fixation sequence, Fixation Duration and, Fixation AOI are some of the main features of raw data collected from Gazepoint eye tracker. The gaze point reflects a snapshot of data collected from the computer screen by a tracker. This tracks a user's gaze with (x, y) co-ordinates and a timestamp which defines where the individual looked in and at what time . Again, saccades are one kind of eye movement that is used to move the eye-fovea rapidly from one point of interest to another. On the other hand, the eye is kept aligned with the target for a certain duration of time in case of fixation (*Types of eye movements*, 2015). All of these information can be put together to find a pattern for the eye movement of a particular class of novice programmers when they solve a programming problem. On the other hand, GSR signal is measured in "micro-Siemens (μS)" or "micro-Mho (μM)". The Shimmer3 provides the intensity of arousal with respect to time. The size of the initial combined (eye -tracking and GSR) raw data obtained from the iMotions software was almost **1.39 GB** with **2,920,299** number of signals for total **324 stimuli tasks (36 students X 9 stimuli each)**.

Table 3.2: Equipment (hardware & software) used for data collection

Eye tracking Data	GP3 Gazepoint
GSR Data	Shimmer3
Software to merge both data	iMotions

This large size of data is shrunk to almost 10.35% (from 2,920,299 number of rows to 302,981) of it's original size in the next data preparation phase using data cleaning (section 3.2.1) and down sampling (section 3.2.2) methods. These two methods removed the irrelevant rows of data which were gathered during data collection process from GSR sensor and eye-tracker and down sampled the data in lower frequency.

3.2 Data Preparation

In machine learning projects, the format of the data has to be prepared in a appropriate form in order to achieve the best results from the applied models. Many machine learning models require data in a specified format such as numeric. For example, some machine learning algorithms do not support null values directly, as a result the null values need to be interpreted from the original raw data set before running those algorithms. One of the major goals of data processing is to format the data-set in such a way that more than one machine learning algorithms can execute that data-set and best model out of them can be chosen.

3.2.1 Data Cleaning

Data cleaning method refers to the identification of inconsistent, incomplete and irrelevant portion of data set and then removal or modification or replacement of those data. In case of eye-tracking data collection, there can be issues like fixation contingent and attention getters. These kinds of issues can bury the important data that someone wants to analyze with a research goal among a lot of irrelevant data. Therefore, it is very

much needed to clean the messy data and keep only the relevant data for improvement of data quality and faster computation of algorithms applied on that particular data set.

In this work, initially, the raw data set is cleaned by removing unnecessary and irrelevant rows. For all the participants, a data row is defined as irrelevant row if that contains-

- gaze point axis (both x and y) value -1, has been removed. As gaze point indicates the axis landing on the stimulus, so if the value is -1 in a particular timestamp that indicates the participant has been looking outside of stimulus. Therefore, it can be said that the data for that time is irrelevant and can be removed from the data-set.
- “undefined" GazeEventType has also been removed. “undefined" GazeEventType defines that, the eye-tracker failed to characterize that particular gaze point either as Saccade or Fixation. Hence, that particular row of data is irrelevant to count.

The output data from this data cleaning phase served as the input of the next down sampling phase.

3.2.2 Down-sampling

Down-sampling is the technique that reduces some samples from a large data set in such manner that the new decreased data set represents the previous data set without much differences. This technique is very efficient to reduce the data processing time and it brings a too large data set in a manageable size.

In this research work, down-sampling method is applied to reduce the number of samples per second. The eye-tracking and GSR signal are often sampled at a much higher sampling rate than actually required. During the data collection, the frequency of the eye-tracking data was 60Hz and the frequency of the GSR data was 128Hz separately. While extracting the merged data from iMotions, the output trigger was selected for eye-tracking data. Therefore, the combined data had 60 sampling rate per second. The size of the raw data

is too big (1.39 GB) to process without down-sampling it. Therefore, the full data was down-sampled without any remarkable risk of losing important portion of the signal using Nyquist frequency (Leis, 2011) which is the minimal frequency at which a signal can be sampled without any under-sampling. The down sampled data does not have any significant difference than the actual data (Figure 3.4).

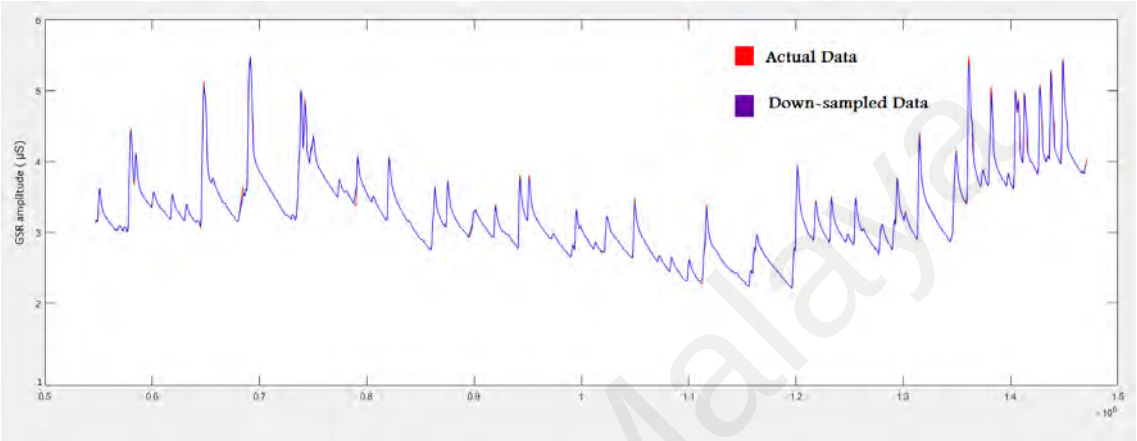

Figure 3.4: Down-sampled data of a single participant's GSR data

After data cleaning and down sampling, the number of signals or samples was decrease from 2,920,299 to **N = 302,981** for total 36 novice programmers. This was the final size of total data-set which was later used for machine learning analysis.

3.2.3 GSR phasic response (arousal / non-arousal) calculation

In this data preparation phase, the arousal and non-arousal states were calculated from the GSR signal raw data. There are main two components in GSR raw data: Skin Conductance Level (SCL) or tonic level and Skin Conductance Response (SCR) or phasic response. In the tonic level, an individual does not have any significance change of emotion or arousal for a certain event. On the other hand, the phasic response is sensitive to particular stimulus events that are emotionally arousing. When an individual is exposed to any meaningful or exciting moment, he / she has a phasic response in the GSR signal. Phasic response is always higher than the tonic level data. Therefore, this phasic response is

a very important feature for this research work. Phasic response data comes as continuous signal data. In order to calculate arousal and non-arousal states from the phasic response data, the Galvanic Skin Response (GSR) data was filtered using **Mean filter** algorithm. Mean filter algorithm is a simple filtering algorithm that smooths the signal data by replacing a center cell value with the mean (average) of all the neighbouring cells. Three basic steps had been followed for the mean filtering method. First, the data samples were gone through one by one and then for each sample, the average GSR value of the surrounding samples was calculated based on a +/-4 second time interval centered on the current sample. Lastly, the current sample value had been subtract the average. Thus, the filtered phasic data was found. After filtering the GSR data, if there was any phasic response found ($> 0.01\mu$S) in a particular timestamp, it is labeled as **1 (arousal occurred)** otherwise **0 (non-arousal)**. This is how the feature for emotional arousal detection is derived.

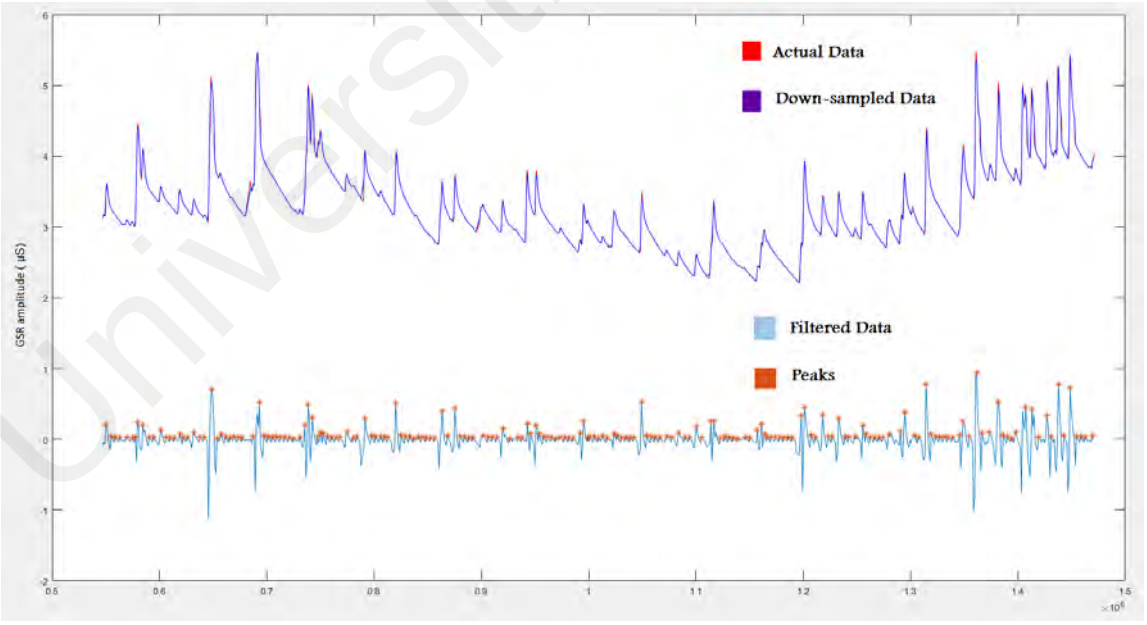

Figure 3.5: Filtered phasic response

3.2.4 Feature Selection

The goal of this phase was to select the most relevant features that are particularly distinguishing or informative while still accurately and completely describing what this study wants to obtain from the two objectives of this proposed work. Proper feature selection can improve the accuracy of the final result obtained from analysis, reduce the chance of over fitting problem for machine learning algorithms and decreases the training time. Initially, this step is done manually by understanding the nature of the features and relevance with the research objective. After primarily choosing the relevant features, the correlation analysis is conducted in section 4.2.1 to understand how strong the relations between each features (variables). Correlation analysis is a strong statistical method that reveals how much two variables are related to each other.

During the data collection phase, the eya-tracker recorded data for the following features- **GazeX, GazeY, GazeAOI, GazeEventType, SaccadeSeq, SaccadeDuration, FixationSeq, FixationDuration, FixationAOI** and the GSR sensor provided **Timestamp** and **phasic response** features. However, it is noticed that not all the features were relevant enough to be used in machine learning algorithms or deep learning algorithms for the mentioned research objectives in Chapter 1. Saccade information is such kind of feature which has less significance with the research objectives. The reason behind this statement is, Saccade is rapid eye movement data where a user looks for a very short period of time without any significance. This short period of time and rapid movement of gaze can not influence a user's mind or arousal. During saccade eye movement, there is no cognitive process among the users (Irwin & Carlson-Radvansky, 1996). On the other hand, during Fixation, the eye is aligned with an interest area for a certain period of time. Therefore, Fixation information are more important for any contribution in emotional arousal and more appropriate to use for further analysis to obtain the research objectives. Initially, using

this manual analysis, the saccade information like **SaccadeSeq** and **SaccadeDuration** features were removed from the final data set. Moreover, as only fixation information were finalized to use for analysis and as **GazeEventType** column provides information about the event type like either “saccade" or “fixation", so no longer the GazeEventType feature was needed. Hence, this feature was also removed from the final data set. The table 3.3 shows the list of selected features from the final combined data set that has been used in the machine learning algorithm for further analysis.

Table 3.3: The selected feature list from the combined data set

Individuals' data	**Eye tracking Data**	**GSR Data**
Participant	FixationSeq	Phasic Response
Stimulus	FixationDuration	
	FixationAOI	
	GazeX	
	GazeY	

3.2.5 One hot encoding and feature scaling

In this data preparation phase, one hot encoding method and feature scaling method has been applied to normalize the data distribution and prepare the data capable of processing in the machine learning or deep learning algorithms. Machine learning algorithms require numerical values in each field to support a data set. In case of raw data extracted from iMotions, most of the categorical data were defined by character or string values. Furthermore, machine learning algorithms are unable to work with this kind of categorical data directly. Therefore, the categorical data from the data set have been first transformed into numerical integer values from any string or character values and then they were represented with binary vectors using one hot encoding method. One hot encoding process the categorical

data in such a form that increase the accuracy of some of the machine learning algorithms which directly can not operate on labeled categorical data. This encoding method creates additional features for each unique category. One of the advantages of one hot encoding is it removes any order or rank among the categories. Thus while fitting the data in a machine learning model, the model treats all the categories same without any priorities.

Table 3.4: One hot encoding example for Fixation AOI

AOI Name	**PS**	**OP**	**Var**	**DT**	**Oper**
DT	0	0	0	1	0
OP	0	1	0	0	0
PS	1	0	0	0	0
OP	0	1	0	0	0

The final data set also contained some numerical features with various units, range and magnitudes. Therefore feature scaling technique has been used which is required to standardize all the data in a fixed range. Feature scaling is significant for machine learning algorithms like k-nearest neighbors (kNN) who calculates the distance between the data points. In this research, Min-Max feature scaling method is used for with the goal of standardizing the data. This scaling maps all the numerical feature values between 0 and 1. The equation 3.1 shows how the scaled value is calculated from the maximum and minimum value of a feature column.

$$X(scaled) = \frac{X(i) - X(min)}{X(max) - X(min)} \tag{3.1}$$

CHAPTER 4: ANALYSIS AND RESULT

This chapter defines the experimental tools used during the whole research duration, the procedure of experiments carried out to answer the research questions from Chapter 1 and an analysis of the finding result obtained from those experiments.

4.1 Experimental Tool

In this research work, different types of tools have been used for the data preparation and data analysis phases. Most of the data preparation algorithms were run in MATLAB while the machine learning models were run on Google Colab. IBM SPSS Statistics has been used in this work for analysing significance of relationship among the features. A summary of the used tools for this research work is given below:

- **Environment:** Google Colaboratory, MATLAB
- **Programming Language:** Python, MATLAB
- **Libraries:** PyTorch, SciKit-Learn, Numpy, Pandas, Matplotlib
- **Graphical Processing Unit (GPU):** Google Server GPU
- **Package:** IBM SPSS Statistics

4.2 Supervised Machine Learning (Objective-1)

In this section, a description of the methods to obtain the first research objective that is to identify if supervised machine learning algorithms like k-nearest neighbors (kNN), Naive Bayes (NB), Logistic regression (LR) can identify emotional arousal among the novice programmers using eye-tracking data **(RQ-1)**, has been given. Before that, a correlation analysis has been presented to understand the degree to which various features (variables) are related and and multiple independent *t*-Test analyses have been shown to understand how significantly the arousal can be different in various AOIs and Stimulus. In

this section, measurements like accuracy, precision, recall. F1 score, confusion matrix will also be shown to evaluate the performance and accuracy of the mentioned models. The comparison between multiple algorithms **(RQ-2)** is shown in the last subsection.

4.2.1 Correlation analysis

Initially, a correlation analysis has been conducted to identify the significance of relationship among the features. Using this analysis, the correlation coefficient can be evaluated which determines to what extent and to which direction two variables tend to change together. As the rate of increment / decrements of the eye-tracking and GSR data are not constant, **Spearman's rank-order** correlation has been used in this research work. The advantage of this analysis is it uses monotonic relationship, to identify the the kind of relationship among the variables / features of the data set.

Correlations Matrix

			Stimulus Name	GazeX	GazeY	FixationSeq	Fixation Duration	Fixation AOI	Phasic Data
Spearman's rho	Stimulus Name	Correlation Coefficient	--						
		Sig. (1-tailed)	.						
	GazeX	Correlation Coefficient	-.022**	--					
		Sig. (1-tailed)	<.001	.					
	GazeY	Correlation Coefficient	.024**	-.247**	--				
		Sig. (1-tailed)	<.001	.000	.				
	Fixation Seq	Correlation Coefficient	.017**	-.038**	.315**	--			
		Sig. (1-tailed)	<.001	<.001	.000	.			
	Fixation Duration	Correlation Coefficient	.010**	-.043**	.043**	.137**	--		
		Sig. (1-tailed)	<.001	<.001	<.001	.000	.		
	Fixation AOI	Correlation Coefficient	.047**	-.219**	.827**	.273**	.043**	--	
		Sig. (1-tailed)	<.001	.000	.000	.000	<.001	.	
	Phasic Data	Correlation Coefficient	-.001	.000	-.09**	-.063**	.173**	-.071**	--
		Sig. (1-tailed)	.337	.431	<.001	<.001	<.001	<.001	.

**. Correlation is significant at the 0.01 level (1-tailed).

Figure 4.1: Spearman's rank-order correlations matrix

Table in figure 4.1 shows the correlation matrix for all the features. This matrix is a table whose each cell shows the correlation coefficients between two features or variables. In the

matrix 4.1, the respective correlation coefficient cell that contains **'**'** symbol, shows that the correlation is significant at the α= 0.01 level between two variables. The alpha value is affiliated with the confidence level of this analysis. If the value of correlation coefficient (p) is less than or equal to α then the null hypothesis can be rejected. The matrix shows the arousal data (Phasic Data) has a significant correlation with Fixation Sequence, Fixation Duration, Fixation AOI ($p < \alpha$). However, the correlation coefficient value for Stimulus and Phasic data shows there is low correlation between these two variables. Therefore, a *t*-Test analysis is conducted to understand if the arousal is significantly different in each Stimuli.

4.2.2 Independent Samples *t*-Test

In this stage, an Independent Samples *t*- Test has been conducted to compare fixation duration between arousal (1) and non-arousal (0) for each nine stimuli separately.

Independent Samples *t*- Test measures the means of two independent groups and decides if the means of associated stimulus are significantly different or not. After conducting the test, the *t*- test analysis of five out of the total of nine stimuli showed a significant difference for arousal and non-arousal. For **Easy - 2** (M_0 = 147.74, SD_0 = 98.167, M_1 = 153.02, SD_1 = 99.393), **Medium - 1** (M_0 = 142.62, SD_0 = 86.129, M_1 = 144.86, SD_1 = 85.185), **Hard - 1** (M_0 = 143.92, SD_0 = 89.473, M_1 = 146.78, SD_1 = 87.269), **Hard-2** (M_0 = 155.48, SD_0 = 107.759, M_1 = 158.10, SD_1 = 111.373) and **Hard - 3** (M_0 = 147.99, SD_0 = 97.253, M_1 = 151.61, SD_1 = 99.229), the absolute ***t*** values are **4.782, 2.242, 2.807, 1.9, 3.134** respectively and ***p*** values are **0.00002, 0.025, 0.005, 0.05,0.002** respectively. For these five stimuli the significance (2 - tailed) value is not greater than 0.05 which rejects the null hypothesis. More detail information is shown into Appendix A and Table in figure 4.2. The value of N denotes the number of samples processed. The statistics implies that

Group Statistics

	PhasicData	N	Mean	Std. Deviation	Std. Error Mean
Easy-1	0	20276	147.43	85.650	.602
	1	19708	148.00	84.514	.602
Easy-2	0	16263	147.74	98.167	.770
	1	15704	153.02	99.393	.793
Easy-3	0	26538	160.23	111.334	.683
	1	25710	161.01	108.636	.678
Medium-1	0	14827	142.62	86.129	.707
	1	14547	144.86	85.185	.706
Medium-2	0	15689	158.94	105.210	.840
	1	15148	160.24	106.240	.863
Medium-3	0	17451	145.19	94.007	.712
	1	16832	146.39	88.701	.684
Hard-1	0	15388	143.92	89.473	.721
	1	14743	146.78	87.269	.719
Hard-2	0	12671	155.48	107.759	.957
	1	12550	158.10	111.373	.994
Hard-3	0	14717	147.99	97.253	.802
	1	14219	151.61	99.229	.832

Figure 4.2: Stimuli group statistics for independent samples *t* Test

the Fixation Duration during all hard questions had influence in case of emotional arousal.

Another independent Samples *t*- Test has been also conducted to compare fixation duration between arousal (1) and non-arousal(0) for each five AOIs separately. It is found that for total three AOIs among five the *t* Test showed a significant difference for arousal and non-arousal.

For **PS** (M_0 = 139.24, SD_0 = 80.247, M_1 = 141.93, SD_1 = 79.49),**OP** (M_0 = 151.96, SD_0 = 99.417, M_1 = 153.27, SD_1 = 98.978), **Var** (M_0 = 164.43, SD_0 = 120.948, M_1 = 171.11, SD_1 = 122.538), the absolute ***t*** values are **4.553, 2.401, 3.99** respectively and ***p*** values are **0.0005, 0.016, 0.00064** respectively. For these three stimuli the significance (2 - tailed) value is not greater than 0.05 which rejects the null hypothesis meaning that there **do** exist a significant difference for. More detail information is shown into Appendix B

Group Statistics

	PhasicData	N	Mean	Std. Deviation	Std. Error Mean
PS_FixationDuration	0	36826	139.24	80.247	.418
	1	36325	141.93	79.490	.417
OP_FixationDuration	0	67228	151.96	99.417	.383
	1	65455	153.27	98.978	.387
Var_FixationDuration	0	10796	164.43	120.948	1.164
	1	10431	171.11	122.538	1.200
DT_FixationDuration	0	13756	149.54	100.954	.861
	1	12964	151.29	95.361	.838
Oper_FixationDuration	0	20734	156.08	104.227	.724
	1	19968	157.97	104.338	.738

Figure 4.3: AOIs group statistics for independent samples *t*- Test

and Table in figure 4.3. The value of N denotes the number of samples processed. The statistics implies that the students had most significant change of arousal from their fixation duration while looking into Problem Statement (PS). This findings is also consistent with the statistics reported by Obaidellah et al. (2019).

4.2.3 Data Splitting

In this section, it is described how the data set has been split for the training and testing purpose. Before fitting into the supervised machine learning models, the input and target data has been separated from the full data set. For the RO- 1, the input features were Stimulus name, eye-tracking features like Fixation Duration, Fixation AOIs, Fixation Sequence, GazeX, GazeY and the target data was Phasic Data which determines arousal and non-arousal (1/0).

Proper data splitting is very important as it affects the output from the algorithms. There should be enough training data so that the machine learning models can learn proper mapping of inputs to the target outputs and evaluate the model performance efficiently. The data set needs to be split in such way that neither the training data or the testing data is too high. If the training data is too high, then the performance statistic can have greater

variance and if the testing data is too high, the parameter estimates have greater variance. For the data splitting phase, one of the advantage of this research work's data set is it's hugeness. That is why, the variance of parameter estimations or performance statistics will not be greater if the data set is spilt into a 80:20 or a 90:10 ratio.

In this research work, the spilt data went through three steps in all the machine learning models : 1) Model Training, 2) Model Validation and 3) Model Testing. K- fold cross validation approach has been used for the model validation step which mitigate the probability of over fitting issue and estimate an unbiased test error over the whole data set (Varma & Simon, 2006). For the supervised algorithms, 5- fold (Figure 4.4) cross validation has been used meaning that 20% of the total data has been used for testing and 80% data for training in each iteration. In each iteration, first 4 folds of data were taken from the total Data-set for training, keeping the remaining 1- fold unknown to the model. After training we tested the model with the remaining 1- fold data. Each iteration gives an accuracy of the model and after five iterations, the average has been calculated to get the final accuracy of the model. The same five data splits has been maintained while training in all the models to have proper comparison of results.

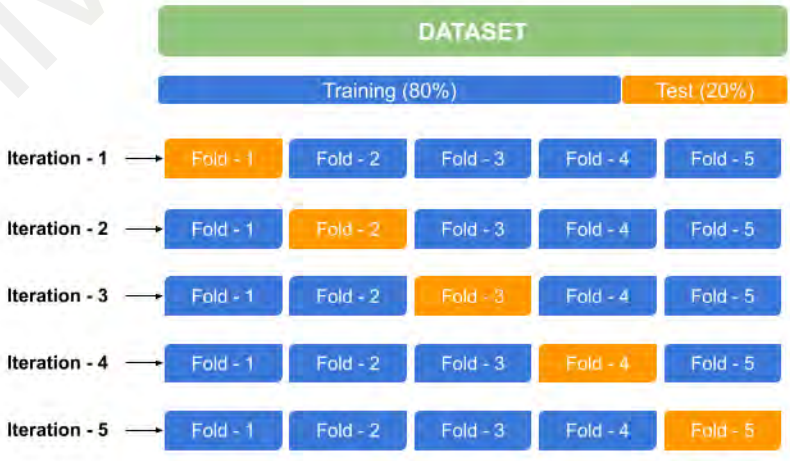

Figure 4.4: 5-fold cross validation

4.2.4 Machine Learning Models Training

In this section, it is discussed how the split data set has been fit and trained into different machine learning models and tested to examine the performance of the models. In these ML models, there are some parameters known as hyperparameters which control the learning procedure of a specific model. It is very important to find the set of right hyperparameters to calculate the best result from a machine learning algorithm. In this research work, hyperparameter tuning or optimization technique has been used repeatedly during training the data to identify the efficient combination of hyperparameters to maximize the performance of a particular model. This technique locates a tuple of hyperparameters which minimizes a predefined cost function and results in an optimal model. After locating the most appropriate combinations of hyperparameters, the accuracy, precision, recall and f1 score of each machine learning model have been calculated to analyse the models. Measuring only accuracy may not give the correct glimpse of a model's performance especially if a data set has class imbalance issue. Therefore, in this work, the precision, recall and, F1 score are also calculated for each model. Precision quantifies the rate of false positives which means the number of predictions of positive class that truly belong to the positive class. On the other hand, the number of predictions of positive class calculated out of all positive record or data point in the data set is known as recall. Lastly, F1 score judges both the issues of precision and recall in single number by weighting the average of both of their values. The description of training and testing procedures of the machine learning models which are used for Research Objective- 1 is discussed in the following subsections.

4.2.4.1 Naive Bayes

Naive Bayes classifier is a conditional probability model which works based on Bayes' Theorem (equation 4.1) with an assumption of independence among the features or predictor.

$$PosteriorProbability = \frac{ConditionProbability * PriorProbability}{PredictorPriorProbability} \tag{4.1}$$

$$P(Y|x) = \frac{P(Y) * P(x|Y)}{P(x)} \tag{4.2}$$

It predicts conditional probabilities for each target class in such a way that the probability that given data point or record belongs to a specific target class. In this way, the class that possess the highest probability is summed up as the most likely targeted class. In a short way, the posterior probability can be portrayed with this question that- What is the reconsidered probability of occurring output Y after taking new information of x into consideration (equation 4.2)?

This easy to implement classifier has low computation cost and can effectively work with bigger data sets. Naive Bayes classifier can be of three types- 1) Gaussian Naive Bayes, 2) Multinomial Naive Bayes and 3) Bernoulli Naive Bayes. In this research work, Multinomial Naive Bayes (MNB) classifier has been used as the features of the data set follow a binomial (arousal/ non-arousal) distribution. It considers a feature vector in histogram which counts the frequency of occurrence of a given event *i* in a particular instance. Class *sklearn.naive_bayes.MultinomialNB(*, alpha= 1.0, fit_prior= True, class_prior= None)* from scikit learn machine learning library has been used to implement the MNB model in python. A hyperparameter that controls the form of the Multinomial Naive Bayes (MNB) model itself is the alpha (α) parameter. This alpha parameter is Laplace or Lidstone smoothing parameter which solves the problem of zero probability

in the Multinomial Naive Bayes model. Grid Search hyperparameter tuning technique (Ghawi & Pfeffer, 2019) has been used with cross validation to choose the best alpha value and found α = 0.1 (ROC-AUC 0.988) resulted in the most optimal model (Figure 4.5).

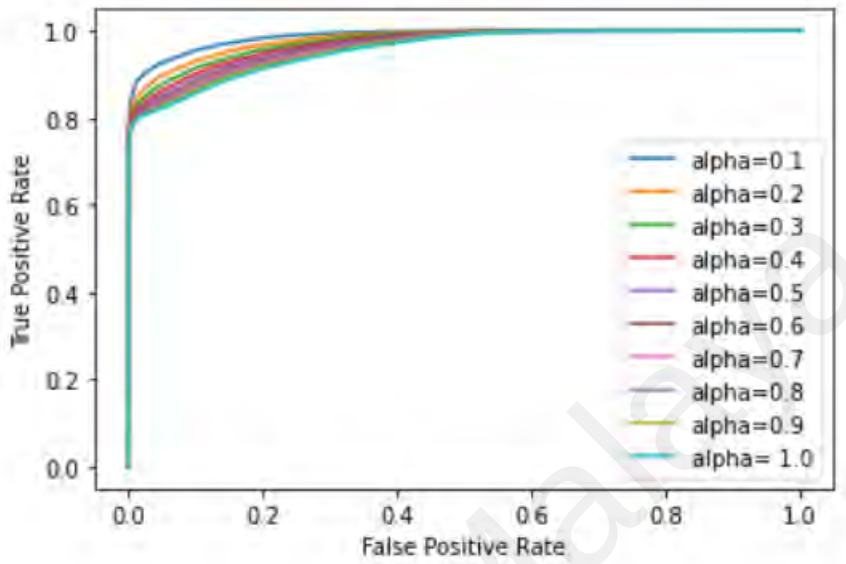

Figure 4.5: Cross validation ROC-AUC curves for finding optimal alpha value

The result of Multinomial Naive Bayes model with α = 0.1 using 5- fold cross validation is shown in Table 4.1. The average accuracy derived from the testing data set is 75.93%.

Table 4.1: Multinomial Naive Bayes algorithm result

	Fold-1 (%)	**Fold-2** (%)	**Fold-3** (%)	**Fold-4** (%)	**Fold-5** (%)	**AVG.** (%)
Accuracy	71.09	77.86	77.63	74.00	79.08	75.93
Precision	81.60	84.75	84.63	83.00	85.28	83.85
Recall	71.89	77.86	77.62	74.00	79.08	76.09
F1 score	68.62	76.83	76.55	72.23	78.16	74.48

Figure 4.6 shows a confusion matrix which is a M X M matrix (M is the number of output or target classes) for assessing the performance of a classification model. The

columns and rows of the matrix represent the actual values and predicted values of the target variable. This matrix gives a holistic view of how many True- Positive (TP), True-Negative (TN), False- Positive (FP) and False- Negative (FN) value of target variable is provided by the machine learning model.

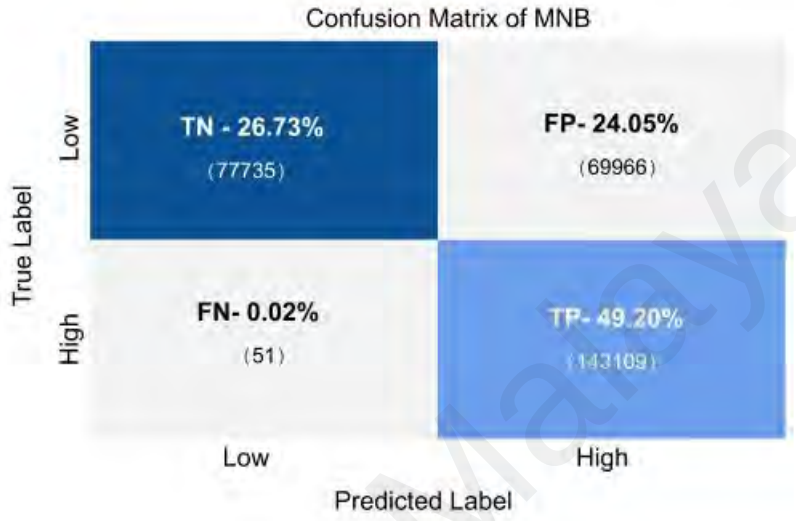

Figure 4.6: Confusion matrix of Multinomial Naive Bayes

The confusion matrix for Multinomial Naive Bayes model presents that about 49.20% samples were predicted as high arousal correctly and 26.73% samples were predicted as non- arousal correctly from the test data set.

4.2.4.2 K-Nearest Neighbor (kNN)

K-nearest-neighbor (kNN) classification is one of the most fundamental classification methods which uses Euclidean distance metrics to classify the data-set (Peterson, 2009). It finds the k-nearest neighbors to the test data, and then classify the samples by voting for the most frequent label. This classification function is computed locally and as this classifier depends on distance so normalization of data increases result of the accuracy. Class *sklearn.neighbors.KNeighborsClassifier* from scikit learn machine learning library

has been used to implement this model in python. The K value or the number of neighbors is a hyperparameter that needs to be chosen during model building.

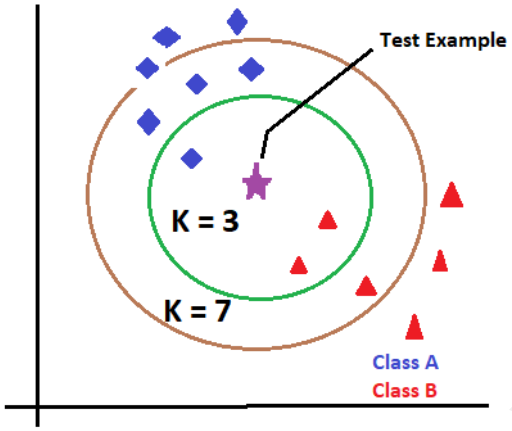

Figure 4.7: Example of importance of K value

In the kNN algorithm, k represents the number of closest neighbors that we want to compare. K value has a strong influence on KNN performance. For example, from figure 4.7, if we choose K = 3, the test example will be classified as Class B. But if we choose K=7, the test data will be classified as Class A with higher accuracy. Finding the optimal K value is a challenging task and in this research, the optimal K value is found by using hyperparameter tuning repeatedly. Using cross validation, the optimal k value has been determined by calculating the highest average accuracy. The curves with all the five folds data has been shown in Figure 4.8. First 80% data is used for training and last 20% data is used for testing in Figure 4.8 (a). Then in the same way subsequently the other graphs for remaining four folds is generated. From all the five graphs, the k=33 is pinpointed as the optimal value which resulted in with the average ROC-AUC score = 0.6806.

The average result found using 5-fold validation in k=33 is in the table 4.2. For total five fold cross validation, the measured average accuracy from this model is 64.33% with a 64.21% F1 score.

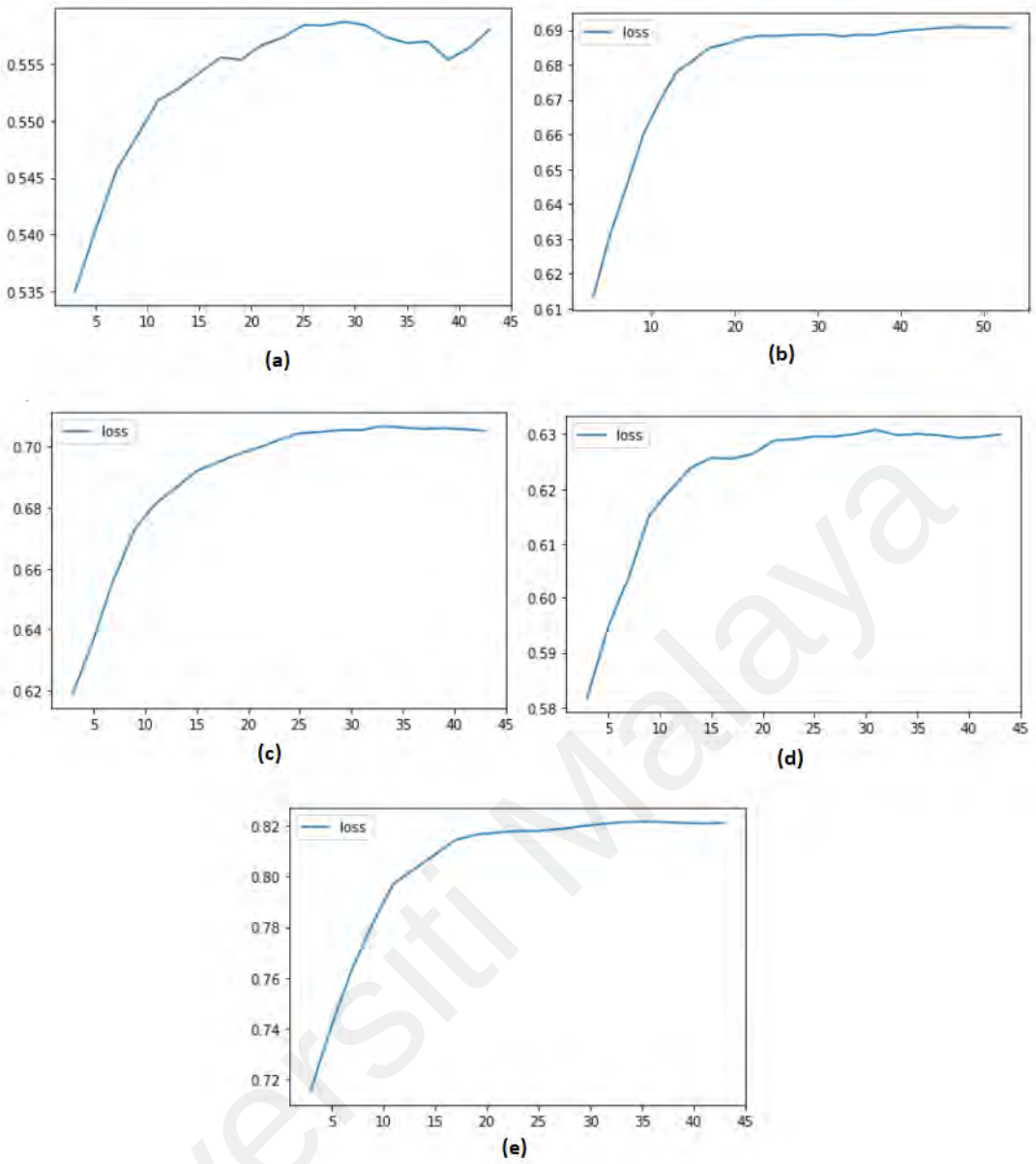

Figure 4.8: Cross validation ROC-AUC curves for finding optimal K value. [(a) Fold-1 (b) Fold-2 (c) Fold-3 (d) Fold-4 & (e) Fold-5]

Table 4.2: KNN algorithm result

	Avg. (Fold 1 - Fold 5) %
Accuracy	64.33
Precision	64.11
Recall	64.33
F1 score	64.21

The confusion matrix in Figure 4.9 shows that about 43.86% samples were predicted as

low arousal correctly and 17.93% samples were predicted as high arousal correctly from the test data set.

Figure 4.9: Confusion matrix of K-Nearest Neighbor

4.2.4.3 Logistic Regression

Logistic Regression (LR) is a predictive analysis algorithm which uses a sigmoid cost function (equation 4.3) known as 'logistic function'. This cost function gives value in between 0 to 1.

$$y = \frac{1}{(1 + e^{-x})} \tag{4.3}$$

The e in the equation is the exponential constant. Gradient Descent is used to minimize the value of cost function (Kleinbaum et al., 2002). This classification model is mostly used when the targeted dependent variable follows a binary pattern (True/ False or arousal/ non-arousal) by nature and to investigate the success and failure probability of an event. Class *sklearn.linear_model.LogisticRegression* from scikit learn machine learning library has been used to implement LR model in python. The hyperparameter **C** which is the inverse of regularization strength controls the form of the Logistic Regression model. In this work,

C value has been tested using a range of positive values (0.001, 0.01,0.1, 1, 10, 100, 1000) and it has been determined that the optimal value is C= 0.01 with a ROC-AUC score of 0.829.

The result after running the 5-fold cross validation is shown in the Table 4.3. The model could correctly classify the phasic data at 64%.

Table 4.3: Logistic Regression algorithm result

	Fold-1 (%)	**Fold-2** (%)	**Fold-3** (%)	**Fold-4** (%)	**Fold-5** (%)	**AVG.** (%)
Accuracy	60.00	62.29	64.20	58.25	74.75	64.00
Precision	56.30	58.80	60.35	55.86	70.31	60.32
Recall	60.00	62.29	64.20	58.25	74.75	64.00
F1 score	52.30	56.15	57.20	54.25	69.48	57.88

The confusion matrix in Figure 4.10 shows true positive (TP), false positive (FP), true negative (TN) and false negative (FN) values of the LR model. About 56.23% samples were predicted as low arousal correctly and 6.34% samples were predicted as high arousal correctly from the test data set.

Figure 4.10: Confusion matrix of Logistic Regression

4.2.4.4 Decision Tree

Decision Tree (DT) is one of the popular classification techniques that can be used to explicitly and visually represent decision making. A series of queries are conducted repeatedly in this algorithm in such way that the output of the last query can decide the next query. A Decision Tree is rooted directed tree with internal node, edges and leaf (Dev & Eden, 2019). An internal node indicates feature, each leaf represents the outcome and, the edges denotes a decision rule. If there is *x* number of features, then there will be *xlog(n)* comparisons. Class *sklearn.tree.DecisionTreeClassifier* from scikit learn machine learning library has been used to implement DT model in python. The optimized DT model could be found using pruning. After pruning, the best ROC-AUC score for DT model is 0.714 with criterion = 'entropy', max_depth= 4, min_samples_leaf = 3, and min_samples_split = 5. Here, the criterion is a parameter that measures the quality of a split, max_depth is the maximum depth of the decision tree, min_samples_split is the number of minimum records considered for splitting an internal node and, min_samples_leaf denotes the minimum number of records or samples considered to be at a leaf node. The criterion 'entropy' shows the randomness measurement in the data. It follows the equation 4.4 where P is the probability.

$$Entropy = -P(classA) * Log(P(classA)) - P(classB) * Log(P(classB)) \quad (4.4)$$

The result after running the 5-fold cross validation is shown in the Table 4.4. The model could correctly classify the phasic data at 64%. Figure 4.12 shows a decision tree of one fold cross validation data with the optimal parameters.

Table 4.4: Decision Tree algorithm result

	Avg. (Fold 1-Fold 5) %
Accuracy	64
Precision	49.91
Recall	64
F1 score	53.19

The confusion matrix in Figure 4.11 shows true positive (TP), false positive (FP), true negative (TN) and false negative (FN) values of the DT model. About 58.9% samples were predicted as low arousal correctly and only 3.6% samples were predicted as high arousal correctly from the test data set.

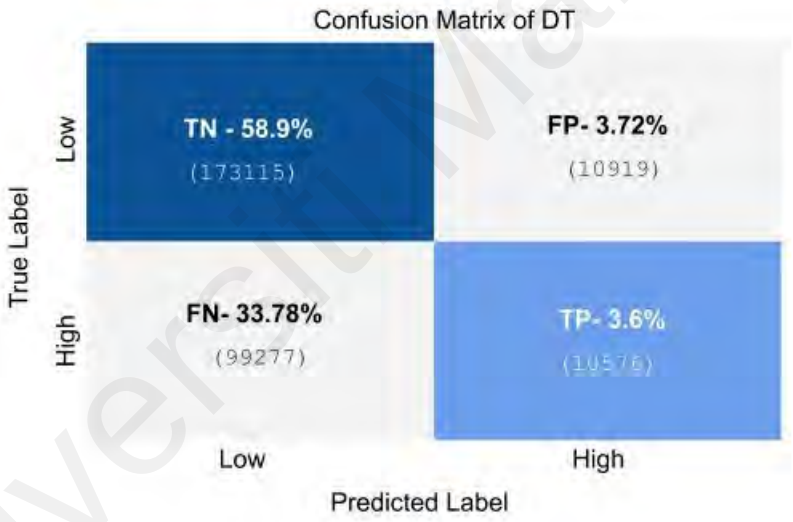

Figure 4.11: Confusion matrix of Decision Tree

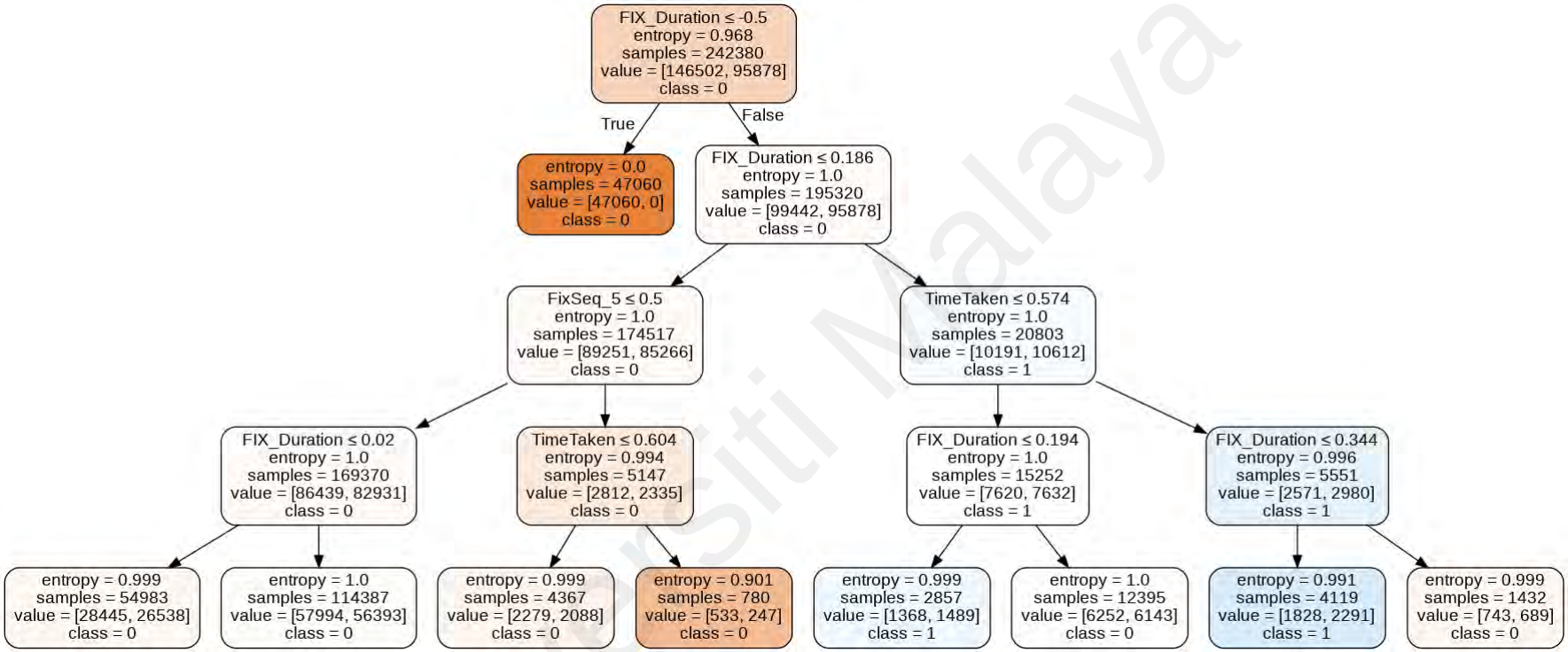

Figure 4.12: Decision Tree of one fold test data with max_depth=4

4.2.5 Analysis

After combining all the result of the supervised learning models in the previous sub-sections, it can be seen that Multinomial Naive Bayes performed best comparing to other models. The average accuracy of MNB for determining the arousal using eye-tracking data among the novice programmers' is 75.93% with precision 83.85%.

Table 4.5: All models' average ROC-AUC score

ML Model	**ROC-AUC score**
Multinomial Naive Bayes (MNB)	96.54%
K-Nearest Neighbor (kNN)	61.4%
Logistic Regression (LR)	68.56%
Decision Tree (DT)	68.3%

Table 4.5 shows the average ROC score of all four models. Multinomial Naive Bayes has the best ROC score with 96.54%. Logistic Regression has a better ROC score (68.56%) than Decision Tree and kNN. Figure 4.13 gives a visual representation of comparison of all models' ROC score for 5-Fold cross validation.

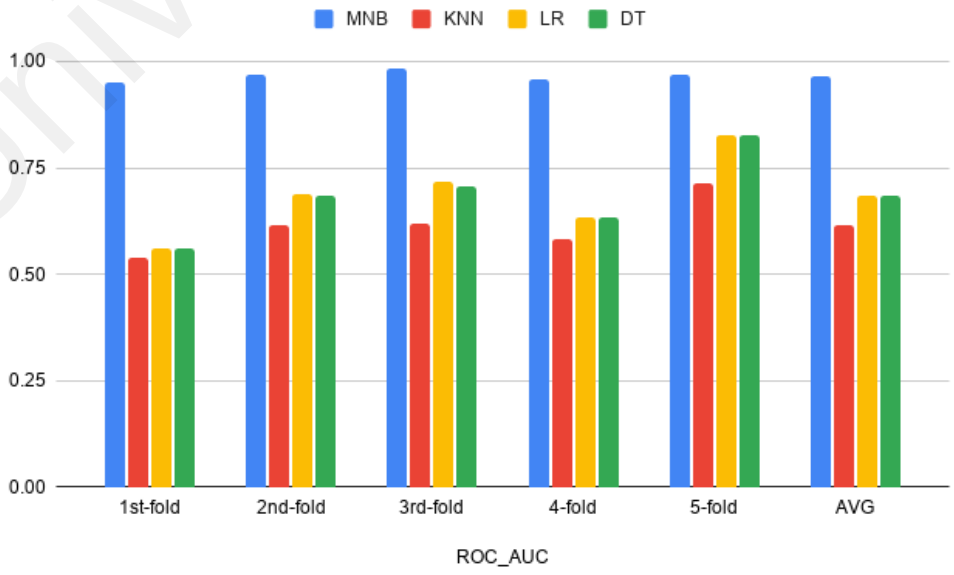

Figure 4.13: Statistic of all models' 5-fold ROC-AUC score

In Machine Learning, ROC-AUC curve is a very popular performance measurement for the classification problems. It shows how well a model is performing than the other model. The more area a ROC-AUC curve of a model covers the better the model is at predicting 0s as 0s and 1s as 1s. Figure 4.14 shows the ROC-AUC Curves for combined Models (MNB, KNN, LR & DT) with 5-fold cross-validation.

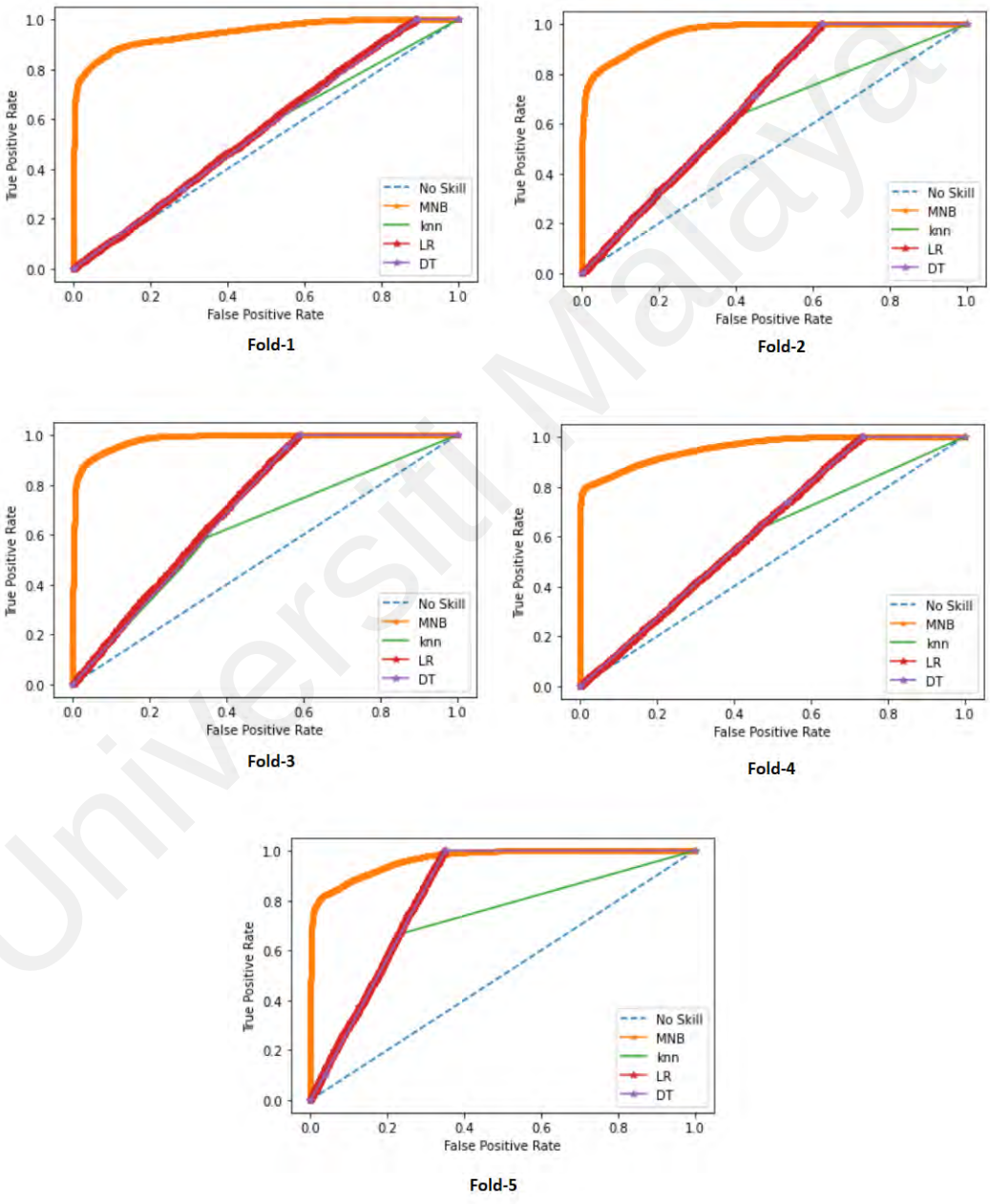

Figure 4.14: ROC-AUC Curves for Combined Models (MNB, kNN, LR & DT) with 5-fold cross-validation

4.3 Deep Learning Technique (Objective-2)

In this section, we will describe about RNN- LSTM deep learning technique used for obtaining the second objective that is to classify the novice programmers according to their performance (High performance group/ low performance group) based on eye-tracking sequence on the AOI and GSR data **(RQ- 3)**. This section will also describe about the intuition behind using LSTM on GSR and Eye Tracking data and combining them in a way that is appropriate for our dataset **(RQ- 4)**.

4.3.1 Long-Short Term Memory (LSTM)

Long-Short Term Memory (LSTM) is a very popular variation of Recurrent Neural Network (RNN) which has feedback connections to facilitate the learning of long-term dependencies and allows long term gradient flow (Schmidhuber & Hochreiter, 1997). Standard RNN is prone to issues like vanishing gradient and exploding gradient. It also fails to learn if in between input events and target there is a time lags greater than 5 – 10 discrete time steps (Gers et al., 2000). On the other hand, LSTM does not have this problem rather it can remember inputs over a long period of time. Moreover, by enforcing constant error flow using cells, this model can learn to connect minimal time lags in excess of 1000 discrete time steps (Gers et al., 2000).

Considering the input events' type of our data set, the above mentioned advantage of LSTM is the prime reason to choose LSTM as the proper model for this research work. Figure 4.15 shows example of input events and target output structure of one programmer's data. In this research, keeping track of FixationSeq is a major issue for obtaining our goal. A student with high performance can have a particular pattern of sequence while reading into the AOI. It is also important to understand the gradual sequence of increment or decrements of arousal in an individual while they solve a particular stimuli/ problem.

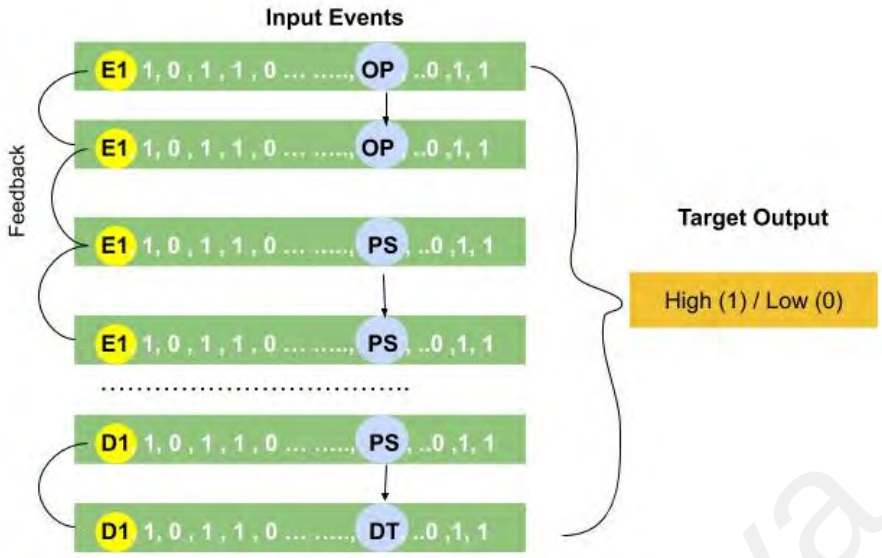

Figure 4.15: Example of input events and target output structure of one programmer's data

Just like the computer memory, LSTM cells also store information in a memory. Hence, when an input event is fed in the model, it can remember the previous input event pattern. The model later correlates this matched patterns to identify and predict a particular output.

4.3.2 Model

The RNN- LSTM model has been implemented and trained using PyTorch (Paszke et al., 2019). PyTorch is a high performance deep learning library for Python programs. It implements an object- oriented approach and has faster deep learning model development ability. This library supports dynamic computational graphs, which facilitates more efficient model optimization.

In this work, the encoded data were passed to LSTM cells first which added recurrent connections to the network and created the ability to link among the sequences of eye-tracking and GSR data. These LSTM cells or memory blocks are comprise of Forget gate, Input gate and Output gate and are responsible for remembering things and manipulations. The layer where these cells exist is known as the hidden layer. Each row of encoded record for one timestamp is fit as input in one LSTM cell. Finally, the outputs from the LSTM

cells were passed to a sigmoid output layer (Figure 4.16). This layer is known as the feedforward output layer. Sigmoid function predict output value in between 0 to 1. That is the reason, sigmoid function is used to predict the novice programmers' performance level high (1) or low (0).

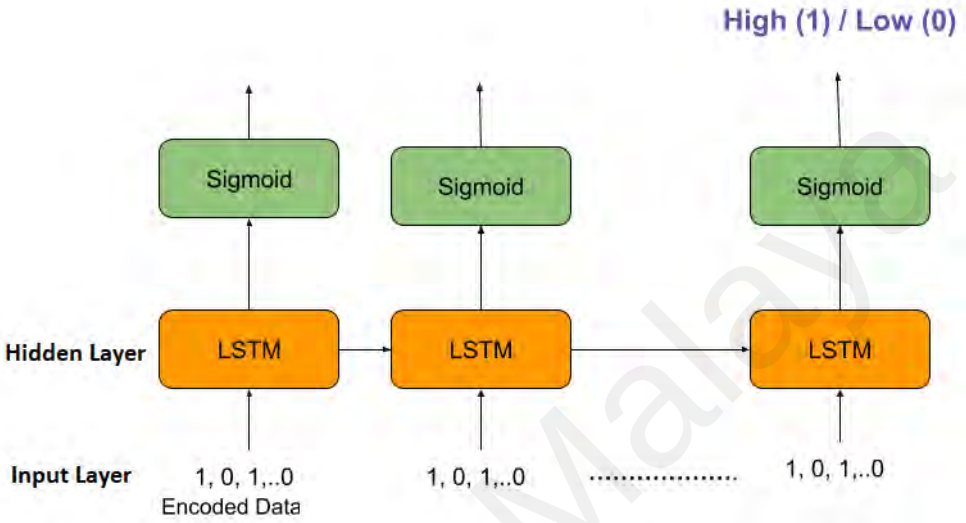

Figure 4.16: Structure of model with LSTM cells

Before feeding the data in the model, the data set for each programmer has been resized. It is important to have same size input data before training in the LSTM model. To standardise all the input length, padding mechanism (Dwarampudi & Reddy, 2019) is used for an input which has shorter size and sequence truncation technique can be used for an input which has larger size. In this work, each programmers' data has been padded or truncated to a specific length **(seq_length)** to deal with both short and very long data set. The students who have taken shorter time to solve all the programming problems (9 stimulus) have shorter length of data than the students who took longer time. The seq_length (= 8416) has been calculated from the average of the total length of all the programmers' data. For a programmer's data length shorter than seq_length, the input data set has been padded with -1 values at the end . On the other hand, for a programmer's data length longer than seq_length, the data after seq_length size has been removed or

truncated.

The LSTM network has been instantiated by defining the proper combination of hyperparameters before initiating the training procedure. Hyperparameter tuning technique has been used to investigate the correct set of hyperparameters. The model has been limited to two layers and 300 units of hidden state during hyperparameter tuning using common heuristics (Goodfellow et al., 2016). Binary Cross Entropy Loss or BCELoss has been used as the loss function with an initial learning rate of 0.001 and with the Adam optimizer (Kingma & Ba, 2014). A loss function or cost function calculates how far a predicted value is from the actual target value. Optimizer methods or algorithms are used to minimize the value of this loss function. Adam optimizer is a well known and popular gradient descent optimization algorithm that determines adaptive learning rates for each parameter. During the training of a model in deep learning, the amount that the weights are updated using back- propagation, is known as learning rate or step size. BCELoss is a loss function that is capable of working with a single sigmoid output and it is designed to apply cross entropy loss to a single value in between 0 and 1 which is appropriate for the target output class of this research work. Dropout (p = 0.5) regularization methods has been used to reduce the probability of over fitting issue and increase the model's performance. By using this method, randomly selected neurons are dropped out during the training procedure which results in the generalization of performance of a network and remove the probability of over fitting the training data.

4.3.3 Evaluation

In this subsection, the performance of RNN- LSTM model for the research objective- 2 has been assessed. To evaluate the model, 7- fold Cross Validation has been employed ensuring that no programmer's data has multiple contribution in the training , validation and testing sets of a given fold. Due to not having much data, the cross validation has

been limited into 7- folds of data. For each cross validation fold, the data set has been split into training (80%), validation (10%) and testing (10%). By splitting the data into 80-10-10 for 7-folds, some of the data were reused during training and validation in some folds. However, the reason behind splitting data like this is to keep minimum number of students' data for training (at least 28 students' data per fold). In deep learning models, the more data are feed for training, the more the model will learn properly. For this work, the training of the model was limited up to 35 epochs which was sufficient to provide the highest performance.

Table 4.6: Model's performance

Test Accuracy	**65.71**%
Test Avg. Precision Score	0.752
Test ROC score	0.725
Validation ROC score	0.587

Table 4.6 shows the performance result of the LSTM model for this research work. The model has 65.71% test accuracy using 7-fold cross validation. The Test ROC score of the model is 0.725 and Validation ROC score is 0.587. The difference between these two ROC scores (0.725 - 0.587 = 0.132) shows the model did not have high possibility of over fitting problem. This result provides empirical evidence that Long Short Term Memory (LSTM) models can be estimated to be a possible suitable choice to use for classifying programmer's level of performance based on Eye-tracking and GSR data.

CHAPTER 5: DISCUSSION

In this chapter, the experimental results and their implications are reviewed to answer the research questions presented in Chapter 1. This chapter also discusses about the importance of the findings of this research work in the real- world condition which were described in the research significance.

5.1 Relation between gaze features and arousal

In this research work, first it has been tried to determine if the eye-tracking data can be used to predict the novice programmers arousal during solving programming problems. This task has been accomplished to answer the first research question from Chapter 1; RQ-1) Can gaze features be used to predict a novice programmer's arousal while looking into a particular stimulus event and area of interest (AOI)? In the beginning, a correlation analysis and two Independent Samples *t*-Test have been conducted to understand the significance of relations among these ET and GSR data. The statistic analysis in section 4.2.1 found the arousal (Phasic Data) has a significant correlation with Fixation Sequence, Fixation Duration, Fixation Area of Interest (AOI). Even though various Stimulus and arousal had a low relation but it is also found from Independent Samples *t*-Test in section 4.2.2 that fixation duration is significantly different for arousal and non-arousal in most of the stimulus especially in case of Hard questions. For the Hard stimulus, the null hypothesis was rejected ($\rho < \alpha$) and found there are significant difference among the true mean and compared value of fixation duration during arousal and non- arousal. It implies that, the students arousal can be identified using the gaze feature in a meaningful way when they are looking (problem solving) at (fixation) different types of questions (stimulus) for a certain period of time. Figure 5.1 summaries the mean fixation duration across each stimuli for both arousal and non- arousal state. This bar chart has been derived from the table 4.2 in

section 4.2.2 . It can be observed from this Figure 5.1 that the novice programmers spend longer period of time gazing (Fixation Duration) on a particular position of stimuli or a certain part of programming problem question while looking during their arousal period. It can be also witnessed from the data that the programmers tend not to look into the non-exciting part of stimuli for a longer time. This non- exciting part of programming problem can be a certain part which has less importance to them for solving the question or may be a part which is very easy to understand for them that it does not create any arousal during reading.

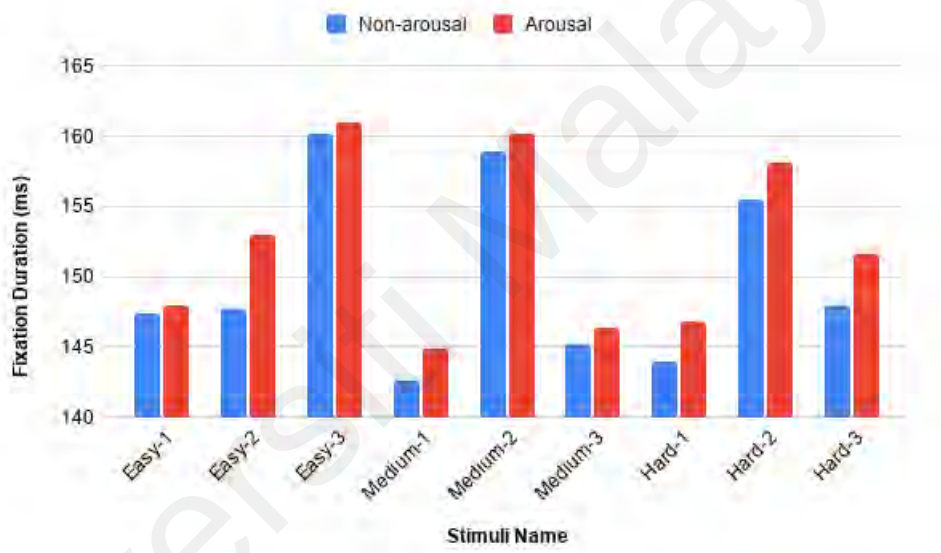

Figure 5.1: Mean Fixation Duration Across Stimulus for Arousal and Non-arousal state

Again, the second Independent Samples *t*-Test statistics in section 4.2.2 shows how significantly the fixation duration can vary for arousal and non- arousal for different AOIs like Problem Statement (PS), Output (OP) and Variables (Var). The result implies that at the time of problem solving, most of the students exhibit an evident aroused state of emotion during looking into the Problem Statement, Output of the sample problem and Variables. This statement is also supported by the findings of the work by Obaidellah et al. (2019). The statistic from the data of this proposed work showed the fixation duration does not change significantly during arousal or non-arousal state in case of Data Type

(DT) and Operation (Oper) ($\rho > \alpha$). One thing to note that, the DT and Oper AOIs were in the bottom part of each stimuli (Figure 3.3). This also implies that most of the novice programmers went through the programming problems' question in a Top-Down approach with significant fixation duration during high intensity arousal. This finding or flow of viewing during the problem solving activity is consistent with those reported by Obaidellah et al. (2019).

Figure 5.2, which is derived from the table 4.2 in section 4.2.2, summaries the mean fixation duration across each AOI for both arousal and non-arousal state. The bar chart also depicts that arousal state can be triggered among the novice programmers when they observe certain area of interest (AOI) for longer fixation duration.

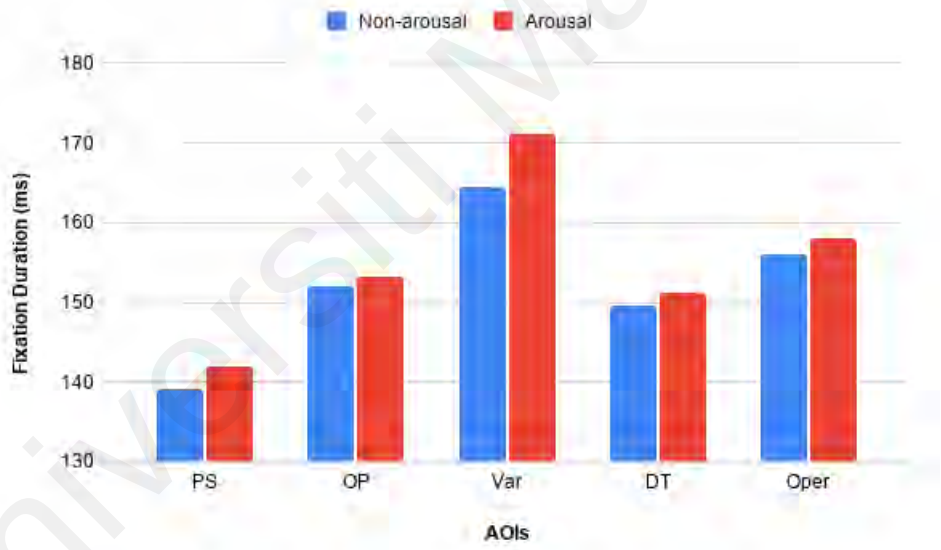

Figure 5.2: Mean Fixation Duration Across AOIs for Arousal and Non-arousal state

All these statistics and analysis in this section indicates that the eye-tracking features have a significant relation with the arousal (Galvanic Skin Response - GSR) data. This finding can be used to predict the arousal among the students using gaze feature and this also answers the first research question in Chapter 1.

5.2 Most effective supervised machine learning algorithm

Combining the information found from correlation analysis and *t*-Test analysis, the Eye-tracking (ET) data has been used to predict the novice programmers arousal using different supervised machine learning algorithms. The goal of this task is to answer the second research question from Chapter 1; RQ- 2 (What is the best type of supervised algorithm for classifying arousal among the novice programmers using Eya-tracking (ET) features?) This part of work has investigated Multinomial Naive Bayes (MNB) [section 4.2.4], K-Nearest Neighbor (kNN) [section 4.2.5], Logistic Regression [section 4.2.6], Decision Tree [section 4.2.7] algorithms and compared the overall results. Five fold cross validation method has been used for each machine learning models to avoid the over-fitting issues and to have the result more accurate across the whole data set. The result from the all models' performance analysis and comparison shows that, Multinomial Naive Bayes (MNB) classifier performed best to classify the arousal among the novice programmers using fixation information with accuracy rate 75.93%, precision 83.85% and recall 76.09% (RQ-2).

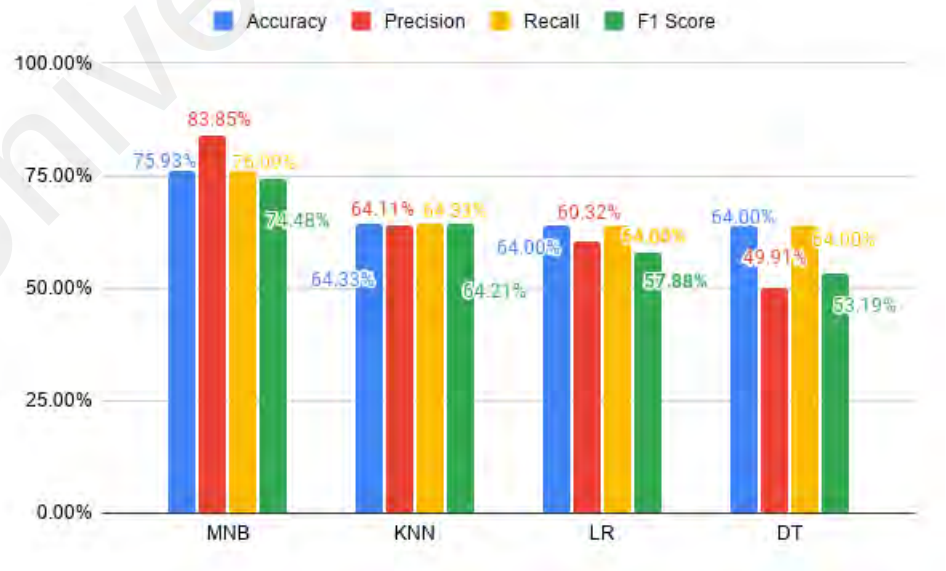

Figure 5.3: Overall overview of the performance of the machine learning algorithms

Figure 5.3 shows the comparison of performance of all the four algorithms in the form of

bar chart. This figure is presented to answer the second research question that is proposed to identify the best type of supervised algorithm for classifying arousal among the novice programmers using Eye-tracking (ET) features.

The validation of the dominance of Multinomial Naive Bayes (MNB) classifier over the other models is shown in Section 4.2.8 using ROC-AUC curves (Figure 4.14). The ROC-AUC curve of MNB covered more area under it than any other algorithms. Also the ROC-AUC score in all folds were higher in case of Multinomial Naive Bayes (ROC-AUC_{avg} = 0.9654) than K-Nearest Neighbor (ROC-AUC_{avg} = 0.614), Logistic Regression (ROC-AUC_{avg} = 0.6856) or Decision Tree (ROC-AUC_{avg} = 0.683).

In this research, the significance of finding the fact that MNB (a variant of Naive Bayes) yields as the most effective prediction performance is similar with the findings of (Fritz et al., 2014) which demonstrate Naive Bayes is an effective classifier to train Eye-tracking and psycho-physiological measurement data for classifying difficulty level of programming tasks. Moreover, Abbas et al. (2019) also showed how MNB can be used as a very efficient algorithm for emotional classification. Decision Tree (LR) performed lowest in terms of predicting the arousal or non-arousal state. The superiority of Multinomial Naive Bayes (MNB) classifier over the other models is potentially due to its advantage to work better with non-linear data and the target feature of the data set of this research follow a binomial (arousal and non-arousal) distribution.

5.3 Deep learning model for predicting programmers' performance

This section will discuss the procedure of investigation and the result of the analysis that were presented in section 4.3. In this research work, Long Short Term Memory (LSTM) RNN model has been used to fulfill the requirements to answer the Research Question-3 and to investigate if deep learning techniques can find a common sequence among the ET-GSR data and categorise the novice programmers' performance. The

LSTM model of this work, which was implemented and trained using PyTorch, could classify the programmers according to their performance with 65.71% test accuracy by answering the Research Question-3. The result of this finding implies that application of deep learning model is able to find a common sequence of combination of visual attention (Eye Tracking data) and emotional arousal (GSR data) among the high performing novice programmers and low performing novice programmers. This result is consistent in terms of some previous related works (Akram et al., 2018; Dien et al., 2020) where LSTM model is used to classify a particular group of people's performance or stealth based on attention or sequence of activities.

One of the important reasons for choosing LSTM as an appropriate model for this research is its ability to remember a long data sequence and it has feedback connections to promote the learning of long-term dependencies. The elaborate answer for Research Question-4 which seeks the reasoning behind choosing LSTM model as an appropriate deep learning model for this research work, has been discussed in section 4.3.1. The findings from the brief literature review in Section 2.5 on deep learning models used on Eye Tracking or any arousal measurement data proves that Long Short Term Memory (LSTM) model is appropriate for this kind of long sequence behavior analysis data. The scope of this research is to find if a good deep learning model like LSTM is capable of finding a relationship among the sequence of attention of novice programmers and if the model can predict programmers' performance from that relation. However, there is a wide range of future scope to investigate with hybrid deep learning model to increase the accuracy of prediction of novice programmers' performance.

5.4 Implications of the Research

This section describes the overall implications of the whole research work and mentions some of the real life application of those findings.

The finding of this research that has been derived from various measures and shows that programming problems with different levels of difficulty have different effect on an individual's cognitive ability, problem reading time and sequence of gaze pattern. The result of this research also suggests that staring for a longer duration at a specific problem area may have greater difficulty to understand at that moment for a novice programmer and that can cause high arousal. This kind of trigger of emotional arousal has an impact on programmers' performance and activation and this statement is also supported by the work of Mäntylä et al. (2017).

Furthermore, the result derived from the supervised machine learning algorithms shows that it is possible to classify arousal / non-arousal using Eye-tracking data of novice programmers using proper supervised machine learning algorithm. This finding can be informative for the researchers or developers while determining and understanding a novice programmer's real time arousal/non-arousal in any E-learning platform.

Lastly, the observations from deep learning model analysis prove that Deep learning technique like Long Short Term Memory (LSTM) has huge possibility in the field of combination of Eye-tracking and Psycho-physiological data. It is capable of classifying high and low performer students by determining a common sequence of behaviour, combined with attention and arousal, though out their problem solving activity. This kind of prior classification of novice learners according to their performance from their Eye Tracking (ET) and Galvanic Skin Response (GSR) information can help the programming language instructors to choose a specific learning approach for each group upon the identification of the learners' state of arousal and performance.

CHAPTER 6: CONCLUSION

This chapter presents a concise version of the full dissertation along with the limitations of this work and the new directions that can be implemented in the future.

6.1 Summary

Students who are new to programming learning always face some difficulties to find a proper way to start solving a programming problem. Different students or novice programmers have different method of defining a programming problem and that can effect their performance also. This research study tried to investigate this hypothesis that the novice programmers' reading (gaze) pattern can be related to their performance as well as their cognitive process. Chapter 1 of this dissertation discussed about background and motivation behind choosing this research title. It has been discussed in the same chapter that how the relation between attention and arousal can be combined to understand an individual's behaviour towards a particular situation. The underlying problems of the novice programmers that they face during learning a programming language are discussed briefly which contributed to structure the problem statement, research objective, research questions, scope of this research, and significance of this research.

A detailed review of the related previous works in the field of programming learning, Eye-tracking, psycho-physiological measurement, machine learning algorithms and deep learning techniques is shown in Chapter 2. These studies gave a profound knowledge about various ranges of research techniques and frameworks used in the prior works. A brief summary of these studies has been also shown in this chapter which helped us to develop the conceptual framework of this research and choose the proper mechanism.

A comprehensive explanation of the complete work-flow of this work has been depicted in Chapter 3. As this work has used the same data-set from Obaidellah et al. (2019),

in this phase, the Eye-tracking data and Galvanic Skin Response (GSR) data collection procedure of the paper has been described with the samples' demographics details. A detailed description is given in this chapter about how the raw data has been prepared through a good number of steps such as data cleaning, down-sampling, Galvanic Skin phasic response calculation for arousal and non-arousal data, feature selection, one hot encoding and feature scaling.

In chapter 4, the data analysis and findings carried out for each research objective, as mentioned in the Introduction chapter. In this phase, we have analysed correlation between fixation data and arousal data of AoI sequence collected from undergraduate computer science students who completed a set of programming problems (Obaidellah et al., 2019). A correlation analysis has been presented to understand the degree to which various features (variables) are related and and multiple independent *t*-Test analyses have been shown to understand how significantly the arousal can be different in various Stimulus and AOIs. Our main goals for both research objectives were to investigate if eye-tracking information can classify the emotional arousal states of the programmers and if the eye-tracking and GSR data sequence can classify the programmers according to their performance. To obtain our goals we have examined different supervised machine learning approaches like Naive Bayes, KNN, Logistic Regression, Decision Tree and Long short-term memory (LSTM) deep learning technique.

The result from this research study shows that Multinomial Naive Bayes classifier performed best to classify the arousal among the novice programmers using fixation information with accuracy rate 75.93%, precision 83.85% and recall 76.09%. Also, the Long Short Term Memory (LSTM) model of this study could classify the programmers' according to their performance with 65.71% test accuracy. In Chapter 5, all these results has been summarised in a manner to answer all the research questions mentioned in Chapter

1. The implications of this research work has been also described in this Chapter 5. The approaches that have been presented in this combining both eye-tracking (ET) and GSR data in this research work are novel and have not been used previously in any other works to the author of this research work's best knowledge.

6.2 Research Limitations & Future Work

There are few limitations to be aware of while considering this research work. The data set for the deep learning model was small only 36 students' data and also the data set was collected from only one institution's students. A larger data set from different sources or institutions could let us to have a more intense research result.

In case of Galvanic Skin Response (GSR), the data needs to be recorded from the fingers or palm or feet. The sensor needs to be connected with the skin all the time which can create a little discomfort for the users especially for a novice programmer who is trying to solve a programming problem. Moreover, wrong placement of the sensor can provide wrong record during data collection. Therefore, it is obligatory to be extra careful during data collection with GSR sensor. Another limitation of this work is lack of comparison of the result of LSTM model with other types of deep learning techniques.

For further future work, researchers can include investigating Gated Recurrent Unit (GRU) which is another variation of Recurrent Neural Network (RNN) with the same data set. Also Phased Long Short Term Memory (Neil et al., 2016), which has significant advantage on modeling long sequence of data, can be explored as a potential deep learning model for classifying programmers group. In future, the current LSTM deep learning model can also be used with larger data set collected from more novice programmers' eye-tracking and GSR data from various sources and institutions to validate the current research result. The future researchers can work more on this high potential research

area of the novice programmers' emotions and eye-movement which can contribute in the programming problem setting style for the novice programmers and assist the trainer or the teachers to understand students' performance in an efficient way. Now that this research work has shown that this classifier can be used for pre-identifying students' performance and researchers can use them to develop any e-learning material or programming tools suitable for different types of students.

REFERENCES

Abbas, M., Memon, K. A., Jamali, A. A., Memon, S., & Ahmed, A. (2019). Multinomial naive bayes classification model for sentiment analysis. *IJCSNS Int. J. Comput. Sci. Netw. Secur*, *19*(3), 62.

Ahn, H.-i., & Picard, R. W. (2014). Measuring affective-cognitive experience and predicting market success. *IEEE Transactions on Affective Computing*, *5*(2), 173–186.

Akram, B., Min, W., Wiebe, E., Mott, B., Boyer, K. E., & Lester, J. (2018, july). Improving stealth assessment in game-based learning with lstm-based analytics. In *International conference on educational data mining* (pp. 208–218).

Busjahn, T., Bednarik, R., Begel, A., Crosby, M., Paterson, J. H., Schulte, C., ... Tamm, S. (2015). Eye movements in code reading: Relaxing the linear order. In *2015 ieee 23rd international conference on program comprehension* (pp. 255–265).

Busjahn, T., Schulte, C., Sharif, B., Begel, A., Hansen, M., Bednarik, R., ... Antropova, M. (2014). Eye tracking in computing education. In *Proceedings of the tenth annual conference on international computing education research* (pp. 3–10).

Busjahn, T., Schulte, C., Tamm, S., & Bednarik, R. (2015). Eye movements in programming education II: Analyzing the novice's gaze. In *2nd ed. berlin: Freie university.*

Carette, R., Cilia, F., Dequen, G., Bosche, J., Guerin, J.-L., & Vandromme, L. (2017). Automatic autism spectrum disorder detection thanks to eye-tracking and neural network-based approach. In *International conference on iot technologies for healthcare* (pp. 75–81).

Chmielewska, M., Dzieńkowski, M., Bogucki, J., Kocki, W., Kwiatkowski, B., Pełka, J., & Tuszyńska-Bogucka, W. (2019). Affective computing with eye-tracking data in the study of the visual perception of architectural spaces. In *Matec web of conferences* (Vol. 252, p. 03021).

Dalrymple, K. A., Jiang, M., Zhao, Q., & Elison, J. T. (2019). Machine learning accurately classifies age of toddlers based on eye tracking. *Scientific reports*, *9*(1), 1–10.

Davidson, P., Jones, R., & Peiris, M. (2006). Detecting behavioral microsleeps using eeg and lstm recurrent neural networks. In *2005 ieee engineering in medicine and*

biology 27th annual conference (pp. 5754–5757).

Dev, V. A., & Eden, M. R. (2019). Gradient boosted decision trees for lithology classification. In *Computer aided chemical engineering* (Vol. 47, pp. 113–118). Elsevier.

Dien, T. T., Luu, S. H., Thanh-Hai, N., & Thai-Nghe, N. (2020). Deep learning with data transformation and factor analysis for student performance prediction. *Int. J. Adv. Comput. Sci. Appl*, *11*(8), 711–721.

Dwarampudi, M., & Reddy, N. (2019). Effects of padding on lstms and cnns. *arXiv preprint arXiv:1903.07288*.

Edwards, A. A., Massicci, A., Sridharan, S., Geigel, J., Wang, L., Bailey, R., & Alm, C. O. (2017). Sensor-based methodological observations for studying online learning. In *Proceedings of the 2017 acm workshop on intelligent interfaces for ubiquitous and smart learning* (pp. 25–30).

Empathy 2.0 series: How biometrics can help you understand your customers. (2019, December). Retrieved from `https://www.boardofinnovation.com/blog/how-biometrics-can-help-you-understand-your-customers/`

Fountaine, A., & Sharif, B. (2017). Emotional awareness in software development: Theory and measurement. In *2017 ieee/acm 2nd international workshop on emotion awareness in software engineering (semotion)* (pp. 28–31).

Fritz, T., Begel, A., Müller, S. C., Yigit-Elliott, S., & Züger, M. (2014). Using psycho-physiological measures to assess task difficulty in software development. In *Proceedings of the 36th international conference on software engineering* (pp. 402–413).

Gers, F. A., Schmidhuber, J., & Cummins, F. (2000). Learning to forget: Continual prediction with lstm. *Neural computation*, *12*(10), 2451–2471.

Ghawi, R., & Pfeffer, J. (2019). Efficient hyperparameter tuning with grid search for text categorization using knn approach with bm25 similarity. *Open Computer Science*, *9*(1), 160–180.

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). *Deep learning* (Vol. 1)

(No. 2). MIT press Cambridge.

Grawemeyer, B., Mavrikis, M., Holmes, W., Gutiérrez-Santos, S., Wiedmann, M., & Rummel, N. (2017). Affective learning: improving engagement and enhancing learning with affect-aware feedback. *User Modeling and User-Adapted Interaction*, *27*(1), 119–158.

Handri, S., Yajima, K., Nomura, S., Ogawa, N., Kurosawa, Y., & Fukumura, Y. (2010). Evaluation of student's physiological response towards e-learning courses material by using gsr sensor. In *2010 ieee/acis 9th international conference on computer and information science* (pp. 805–810).

Hong, T., Sun, X., Tian, F., & Ren, F. (2019). Sentiment classification and personality detection via galvanic skin response based on deep learning models. In *2019 5th international conference on big data computing and communications (bigcom)* (pp. 313–317).

Hou, Y., Xiao, T., Zhang, S., Jiang, X., Li, X., Hu, X., ... others (2015). Predicting movie trailer viewer's "like/dislike" via learned shot editing patterns. *IEEE Transactions on Affective Computing*, *7*(1), 29–44.

Huysmans, D., Smets, E., De Raedt, W., Van Hoof, C., Bogaerts, K., Van Diest, I., & Helic, D. (2018). Unsupervised learning for mental stress detection-exploration of self-organizing maps. *Proc. of Biosignals 2018*, *4*, 26–35.

Irwin, D. E., & Carlson-Radvansky, L. A. (1996). Cognitive suppression during saccadic eye movements. *Psychological Science*, *7*(2), 83–88. Retrieved from `http://www.jstor.org/stable/40062915`

Jaques, N., Conati, C., Harley, J. M., & Azevedo, R. (2014). Predicting affect from gaze data during interaction with an intelligent tutoring system. In *International conference on intelligent tutoring systems* (pp. 29–38).

Khan, I. A., Hierons, R. M., & Brinkman, W.-P. (2006). Programmer's mood and their performance. In *Proceedings of the 13th eurpoean conference on cognitive ergonomics: trust and control in complex socio-technical systems* (pp. 123–124).

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., & Klein, M. (2002). *Logistic regression*. Springer.

Leis, J. (2011). *Digital signal processing using matlab for students and researchers*. Wiley Online Library.

Long, Y., Lu, Q., Xiang, R., Li, M., & Huang, C.-R. (2017). A cognition based attention model for sentiment analysis. In *Proceedings of the 2017 conference on empirical methods in natural language processing* (pp. 462–471).

Mäntylä, M. V., Novielli, N., Lanubile, F., Claes, M., & Kuutila, M. (2017). Bootstrapping a lexicon for emotional arousal in software engineering. In *2017 ieee/acm 14th international conference on mining software repositories (msr)* (pp. 198–202).

McGettrick, A., Boyle, R., Ibbett, R., Lloyd, J., Lovegrove, G., & Mander, K. (2005). Grand challenges in computing: Education—a summary. *The Computer Journal*, *48*(1), 42–48.

Neil, D., Pfeiffer, M., & Liu, S.-C. (2016). Phased lstm: Accelerating recurrent network training for long or event-based sequences. *arXiv preprint arXiv:1610.09513*.

Ngo, T., & Manjunath, B. (2017). Saccade gaze prediction using a recurrent neural network. In *2017 ieee international conference on image processing (icip)* (pp. 3435–3439).

Norwawi, N. M., Abdusalam, S. F., Hibadullah, C. F., & Shuaibu, B. M. (2009). Classification of students' performance in computer programming course according to learning style. In *2009 2nd conference on data mining and optimization* (pp. 37–41).

Obaidellah, U., Raschke, M., & Blascheck, T. (2019). Classification of strategies for solving programming problems using aoi sequence analysis. In *Proceedings of the 11th acm symposium on eye tracking research & applications* (pp. 1–9).

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., . . . others (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, *32*, 8026–8037.

Peterson, L. E. (2009). K-nearest neighbor. *Scholarpedia*, *4*(2), 1883.

Post, L. S., Guo, P., Saab, N., & Admiraal, W. (2019). Effects of remote labs on cognitive, behavioral, and affective learning outcomes in higher education. *Computers & Education*, *140*, 103596.

Renumol, V., Jayaprakash, S., & Janakiram, D. (2009). Classification of cognitive difficulties of students to learn computer programming. *Indian Institute of Technology, India*, *12*.

Robins, A., Rountree, J., & Rountree, N. (2003). Learning and teaching programming: A review and discussion. *Computer science education*, *13*(2), 137–172.

Schmidhuber, J., & Hochreiter, S. (1997). Long short-term memory. *Neural Comput*, *9*(8), 1735–1780.

Sims, S. D., Putnam, V., & Conati, C. (2019). Predicting confusion from eye-tracking data with recurrent neural networks. *arXiv preprint arXiv:1906.11211*.

Smets, E., Casale, P., Großekathöfer, U., Lamichhane, B., De Raedt, W., Bogaerts, K., ... Van Hoof, C. (2015). Comparison of machine learning techniques for psychophysiological stress detection. In *International symposium on pervasive computing paradigms for mental health* (pp. 13–22).

Strumwasser, F. (1994). The relations between neuroscience and human behavioral science. *Journal of the experimental analysis of behavior*, *61*(2), 307–317.

Types of eye movements. (2015). Retrieved from `https://www.tobiipro.com/learn-and-support/learn/eye-tracking-essentials/types-of-eye-movements/`

Varma, S., & Simon, R. (2006). Bias in error estimation when using cross-validation for model selection. *BMC bioinformatics*, *7*(1), 1–8.

Verma, A., & Sen, D. (2019). Hmm-based convolutional lstm for visual scanpath prediction. In *2019 27th european signal processing conference (eusipco)* (pp. 1–5).

Yousoof, M., & Sapiyan, M. (2013). Measuring cognitive load for visualizations in learning computer programming-physiological measures. *Ubiquitous Computing and Communication Journal*, *8*(3), 1410.

Zheng, W.-L., Dong, B.-N., & Lu, B.-L. (2014). Multimodal emotion recognition using eeg and eye tracking data. In *2014 36th annual international conference of the ieee engineering in medicine and biology society* (pp. 5040–5043).