Analysing Neural Network Models for Detecting Panic Attacks With Uncertainty Analysis

by

AHNAF TAHMID
20101555
RAFSAN ZAMIL
20101342
MD. MUHIMENUL MUBIN
20101112
NAFIS MOHAMMAD
20101371

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
School of Data and Sciences
Brac University
February 2024

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

AHNAF TAHMID	RAFSAN ZAMIL
20101555	20101342
MD. MUHIMENUL MUBIN	NAFIS MOHAMMAD
20101112	20101371

Approval

The thesis titled "Analysing Neural Network Models for Detecting Panic Attacks With Uncertainty Analysis" submitted by

1. AHNAF TAHMID (20101555)
2. RAFSAN ZAMIL (20101342)
3. MD. MUHIMENUL MUBIN (20101112)
4. NAFIS MOHAMMAD (20101371)

of Fall, 2023 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on February 5, 2024.

Examining Committee:

Supervisor:
(Member)

Jannatun Noor
Assistant Professor
Department of Computer Science and Engineering
School of Data and Sciences
Brac University

Program Coordinator:
(Member)

Dr. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
School of Data and Sciences
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi, PhD
Chairperson and Associate Professor
Department of Computer Science and Engineering
School of Data and Sciences
Brac University

Ethics Statement

This note is a disclaimer to indicate that the research paper in question has not violated the preservation of human rights, welfare or dignity. The honesty and integrity of the research participants are incorporated in the work, which also includes highlighted citations from various reliable sources. No discrimination or disparity has been made in the data collection, and the results have not been manipulated for any prejudiced reasons. Rest assured, we hope our work reflects our respect for all intellectual property and is useful in the long term welfare of humanity.

Abstract

In our society and around the world a lot of people suffer from panic attacks. These panic attacks can be mild or very intense physical stimulations that may incapacitate an individual at the spot when the panic attack occurs. The problem in this case is, if the person suffers from a panic attack outside their house and loses control over themselves, they might be subjected to external environmental hazard such as getting into a car accident, etc. Therefore, if we can effectively track and detect whether a person had a panic attack via their spatiotemporal and biometric data, steps can be taken to help them recover from the panic attack or send help to them, as quickly as possible. Keeping this in our mind, in this study we analysed the performance of different neural network models and techniques to detect panic attacks of individuals from their spatiotemporal and biometric data. Since detection of panic attacks is an emergency use-case, model reliability is essential. To ensure model reliability, we also represented the uncertainty analysis of these neural network models using Monte Carlo Dropout. During our study, we found that among all the models that were used, GRU (Gated Recurrent Unit) had the highest accuracy of 95.56%, and GRU also had one of the least amount of uncertainty. However, the ensemble model had the least amount of uncertainty among all the models that were used.

Keywords: Neural Network, GRU, Monte-Carlo Dropout, Uncertainty Analysis, Panic Attack, Ensemble Learning

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ANN Artificial Neural Network

CNN Convolution Neural Network

GRU Gated Recurrent Unit

$HRMAD10$ heart rate moving average deviation for 10 secs

$HRMAD30$ heart rate moving average deviation for 30 secs

$HRMAD60$ heart rate moving average deviation for 60 secs

HRV Heart Rate Variation

$LSTM$ Long Short-Term Memory Networks

MC Monte Carlo

MCD Monte Carlo Dropout

NHS National Health Service

RNN Recurrent Neural Network

Chapter 1

Introduction

Panic attack is a strong response to fear of the human body which can stem from danger, stress, or sometimes excessive excitement [34]. Hence, detection of panic attacks can be used to detect a person in danger. With more and more people working outside their home, a lot of people are susceptible to panic attacks in the streets due to external environmental hazards or stress. In cases of sever panic attack, people may loss control of their body momentarily which may lead to accidents or open up the person to other types of harm. In addition, the panic attack itself may indicate that the person is in some kind of danger. So, if it is possible to detect an individual having a panic attack, actions can be taken faster, like quick response from law enforcement organizations, that can potentially save the person before something severe can happen to them. In order to detect a person having a panic attack, we can monitor some physical parameters of that person like their spatiotemporal and biometric data.

According to the National Health Service (NHS) of the UK [36], some of the symptoms of panic attack are:

- Sudden spike in heartbeat
- Sweating
- Chest pain
- Being petrified
- Dizziness
- Numbness
- Trembling
- Shortness of breath, etc.

Some, or all, of the symptoms can be measured for panic attack detection. However, monitoring some of these symptoms can be challenging and may require use of costly and large devices. On the other hand, symptoms like a sudden spike in heartbeat,

shortness of breath, etc. can be monitored using gadgets that we use on a regular basis such as smart watches, etc.

In this study, we analyse neural network models for detecting an individual's panic attack. Our study uses models such as Artificial Neural Network (ANN), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), Long Short-Term Memory Networks (LSTM), Gated Recurrent Unit (GRU), Hybrid CNN-LSTM, Hybrid CNN-GRU neural network models, and an ensemble model consisting of RNN, CNN, LSTM, GRU and Multi-Layer Perceptron (MLP). Apart from detecting panic attacks, we also show the uncertainty of our models. We used Monte-Carlo dropout to measure the uncertainty of predictions of our model.

1.1 Motivation

In today's world, a lot of people are working outside their houses to earn a decent living subjecting them to run into different types of danger outside their homes. Often these people travel alone from place to place depending on their work, so if they get into trouble or have an accident in a remote place, it might be very late before other people figure out about it and take necessary action to help them. This late reaction by others may be detrimental to the victim and may lead to severe permanent damages. So an early detection of that person's distress can potentially help them from suffering any serious damage to their body.

Apart from the above mentioned situation, in many parts of the world, illegal military actions are taking place where military fighter jets or artilleries often drop bombs in residential areas. If during the initial bombings, an alarm system can be set up to alert the people nearby, a lot of lives can be saved during the entire ongoing illegal military activity.

Many people across the world suffer from a mental condition know as panic disorder. Panic disorder is a condition when a person suddenly gets a panic attack. During these panic attacks, they may experience very intense physical conditions such as heartbeat increase, chest pain, trembling, chills, dizziness, numbness, a choking sensation, etc. If the person experiences these symptoms in public or while doing delicate tasks like driving, they may get into an accident or open themselves up for harm to devious people leading to a life risk. So if their panic attacks can be detected very early and some mechanism can be put in place so that the person can be calmed down quickly after having a panic attack, then the chance of them getting in any sort of harm would reduce by a lot.

All of the above mentioned scenarios are linked to fear among individuals, and during fear is when a person has panic attacks, so we can detect people's panic attacks by measuring some of their physical parameters like their spatiotemporal and bio-metric data. Data like these can easily be gained from digital smart watches and smart phones. Then we can use those detection techniques for detecting panic attacks to create systems to aid people in the situations mentioned above a great deal.

1.2 Research Gap

In everyday life, a person can face dangers at any moment in time and may require the aid of others. One of the ways to detect if a person is in danger is by detecting if the person had a panic attack [26]. Some research has been conducted regarding panic attack detection by others, like Rubin et al [6], Sapounaki et al [8], Cruz et al [5] and Lazarou et al [26]. However, detection models of Rubin et al [6], Sapounaki et al [8] and Cruz et al [5] focus on detection of panic attacks of people suffering from panic disorder, which is a condition where people frequently suffer from panic attacks. Since their dataset construction, training and testing were done focusing on a specific condition, their model might not be able to detect readings that deviate from the usual readings for panic attack. In addition, they did not use machine learning techniques for detecting the panic attack, further increasing the chance of giving wrong classification.

In the case of Lazarou et al [26], they did create a dataset for panic detection without focusing on any particular cause of panic attack like in Rubin et al [6], Sapounaki et al [8], and Cruz et al [5]. But they mainly used machine learning models and just 2 neural network models. Their accuracy is fairly high but they did not show anything to justify their model's reliability which is necessary as the system they proposed was meant to be used by "Emergency Response Systems" for public safety. Therefore, in this study we analyse the performances of some neural network models in identifying panic attacks. Furthermore, we use monte-carlo dropout to find out the reliability of our models by finding the uncertainty of our models' classifications.

1.3 Research Objective

In this study, we aim to analyse neural network models to detect if a person has had a panic attack. The neural network models which we are using are: Artificial Neural Network (ANN), Convolution Neural Network (CNN), Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU), Hybrid CNN-LSTM, Hybrid CNN-GRU, and an ensemble learning model. Furthermore, our study uses Monte-Carlo Dropout to determine the certainty of predictions of the models we used. The objectives of our research are:

1. To understand how panic attacks can indicate a person being in trouble.
2. To understand how the neural network models we used works.
3. To accurately detect panic attacks using neural networks.
4. To evaluate the models we used.
5. To analyse the uncertainty of the models we used.

1.4 Thesis Organization

The remainder of the study is organized as follows: In Chapter 2, a thorough overview of the existing literature affirming the originality of the work and acknowl-

edgment of sources of help that is associated with this study is presented. Chapter 3 contains a discussion on the background that provides an overview of uncertainty analysis, and the neural network models implemented in the study, including diagrams to illustrate them more clearly. The next chapter, Chapter 4, outlines the methodology of our study by representing the stages we went through. Chapter 5 covers the implementation part, with details regarding the dataset used and the data pre-processing steps. It also covers the parameters of the neural network models used, the evaluation metrics used to measure the performance of the models, and an overview of the uncertainty analysis techniques used in this study. Then in Chapter 6, a list of figures with corresponding references to diagrams illustrating uncertainty analysis and other relevant visual representations are presented as results. In Chapter 7, distinct comparisons are made between related studies. It also covers the limitations of our study. Finally, Chapter 8 provides a conclusion and discusses scopes of future works.

Chapter 2

Literature Review

In this chapter, the research studies done by other authors are discussed.

2.1 Related Works

This section discusses the works of others who suggested any hardware and/or software models in order to detect panic attacks and also works of people who have used monte carlo dropout for uncertainty analysis. Here our goal is to review people's work to get more insight about our topic.

2.1.1 Panic Detection Methods

Rubin et al [6] proposed a panic attack detection system which comprises of a wearable device that takes in readings of physical parameters that are used to detect the panic attack. In their case, the parameters were heart rate, breathing rate, heart rate variability, and core temperature. The wearable device was connected to an application on a mobile via bluetooth. Based on the readings, an algorithm on a server or the mobile application itself could detect an oncoming panic attack. If a panic attack was detected, an intervention (some advice about what to do during the panic attack) was given as a notification on the user's mobile. They gathered data for creating their dataset from 10 individuals who had past panic disorder records. They used anomaly detection to predict the attacks by constructing Gaussian probability density distribution for every test subject and made a Gaussian fit. If a reading produced sufficiently low probability, it was considered as an anomaly/panic attack.

Sapounaki et al [8] proposed a wearable panic attack detection system. They used an open hardware/open software prototype device created by COTS products to implement their system. The wearable device takes in data from the muscle sensor and pulse sensor, and if the values exceed some threshold with respect to the values of each sensor for 150 seconds, the device considers it a panic attack and plays a melody through a speaker to make the person calm down. Their system was tested on a male and a female of 20 years of age who were previously diagnosed with a health issue related to panic attacks. They claimed their system detected panic attacks with 100% accuracy.

Cruz et al [5] proposed a combination of a mobile and a wearable system to detect panic attacks of people with panic disorder. The wearable device is attached to the user which reads and sends data via bluetooth to a mobile. The data read by the wearable system are heart rate, breathing rate, heart rate variability, core temperature, and activity. The data received by the mobile is sent to a data center where the data is stored and evaluated to predict the panic attacks by using feature vectors and anomaly detection algorithms. After detecting a panic attack, the data center sends a notification with advice about what the person should do in that situation. In order to train their system they gathered data from 7 individuals with 3 weeks of monitoring.

Ammar et al [18] proposed an effective, real-time online technique for identifying panic behavior in crowds of individuals. To recognize circumstances indicative of panic, they used an LSTM neural network together with statistical analysis of non-panic activities. This study improves the gradient of motion (GoM) feature, which computes feature values between subsequent images in a video stream. In comparison attempts, DeepROD beat a method that made use of the GoM function, providing superior results for most videos with the exception of a few.

Miranda et al [12] utilized wearable technology for emotion identification. The article provides a wearable-ready binary emotion identification method that makes use of the blood volume pulse and galvanic resistance of skin, two wearable-ready signals. These variables are used to detect fear and other negative emotions, and are derived from publicly available datasets. The study also investigates how arousal and valence ratings can be used to identify emotions within the emotional quadrant, particularly fear and other negative emotions. The study highlights the importance of conducting a thorough analysis of elements like system equilibrium, non-linear components, reduction of dimensionality, algorithm selection, and performance assessment when it comes to classifying emotions using physiological inputs in an academic setting.

Schmidt et al [16] gives a detailed review and comprehension of the theoretical foundations, methodologies, and best practices in wearable-based emotion and stress assessment. Considering techniques that employ wearable sensors, notably those that record physiological and inertial information, to detect changes in the user's emotional state. Support vector machine (SVM), k-nearest neighbor (KNN), and decision-tree (DT) classification algorithms were presented. The authors also discuss the usage of ensemble techniques for classification, such as random-forest and AdaBoost, as well as neural networks (NN).

Petrescu et al [22] uses physiological data and subjective reactions from the DEAP dataset in order to focus on the binary categorisation of the emotion of dread. To categorise fear based on physiological characteristics, they used a variety of machine learning methods, including Decision Trees, k-nearest neighbour, Support Vector Machines, and artificial networks. And in order to improve classification accuracy, these algorithms were combined with dimensionality reduction, feature selection, and hyper-parameter tweaking. By removing pertinent information from the physiological data and fine-tuning the parameters of the machine learning algorithms,

they achieved high prediction accuracy for fear categorisation, ranging from 91.7% to 93.5%.

Lazarou et al [26], in their study, analyses the performance of several machine learning models and two neural network models to detect panic attacks. For their analysis they created their own dataset. Firstly, by collecting spatiotemporal and biometric data from 2 real human subjects and then by artificially generating data for 25 more subjects. In total, their dataset consisted of 16200 rows. Apart from the collected data, they also used 3 derived features in order to train their models. Lastly, they only evaluated their models base on the accuracy score and the model that achieved the highest accuracy was Gaussian SVM with a value of 94.5% while using raw features along with a derived feature called HRMAD60 to train the model.

2.1.2 Use of Monte Carlo Dropout for Uncertainty Analysis

Islam et al [31] in their study worked on uncertainty analysis of transformers in classifying image using Monte Carlo dropout. They worked with three different transformers namely Vision Transformers (ViT), Swin Transformers (SWT), and Compact Convolutional Transformers (CCT). In order to quantify the amount of uncertainty on a specific image, they picked an image randomly from the test data and calculated the predictive entropy of that image after running the models. The higher the predictive entropy, the more uncertain the model was at predicting. In their study CCT performed the best while the ViT performed the worst.

Joshi et al [20] in their paper, show the performance of four segmentation algorithms in detecting Region of Interest (ROI) of fingerprints. Apart from this, they also performed uncertainty analysis of the best performing model using Monte Carlo dropout. Their best performing algorithm was RUnet upon which they implemented Monte Carlo dropout. They calculated the model's uncertainty by using predictive variance and they also visualised the uncertainty of the predictions using heat maps corresponding to each image input.

Abdar et al [27] in their paper worked on detection of covid-19 using chest computed tomography and x-ray of patients. They proposed a deep learning feature fusion model called UncertaintyFuseNet for the classifications. They quantified the uncertainty of their model using ensemble of the predictions after running the model with Monte Carlo dropout for multiple times.

Avci et al [19] in their study worked with U-Net with dropout layers to improve the accuracy of quantitative MRI and used Monte Carlo dropout to quantify the uncertainty of their model. To quantify the uncertainty, they generated multiple outputs by running the model multiple times and then calculated the standard deviation of the predictions.

While Deep Neural Networks (DNNs) perform well in categorizing ECG pictures, they are not able to measure the degree of uncertainty in predictions, which might cause problems with medical practitioners' decision-making. So in their study, Islam et al [29] used Convolutional Neural Networks (CNN) and Monte Carlo Dropout

(MCD) technique to analyze uncertainty in the categorization of ECG images. The mean and variance of predictions are used to detect uncertain samples, which aids in understanding the dataset and locating problems with the model.

The goal of Islam et al's study [30] is to investigate the issue of uncertainty estimates in GNN models and how it affects performance assessment. To capture the outcomes of stochastic forward passes and evaluate the prediction mean and model uncertainty, the authors employ dropout neural networks (NN). To quantify uncertainty in model parameters and assess model performance, they use GNN models, such as PPNP, APPNP, GCN, GraphSAGE, and GAT.

Islam et al [32] use Monte Carlo dropout for introducing uncertainty-aware feature to their model to reduce the risk factors and ensure robust and reliable performance in the context of disease identification using images of cells. They do this by running the Monte Carlo ensemble model 500 times on the test dataset and then selecting the most uncertain observations from the Monte Carlo prediction based on the variance from these 500 predictions. They then calculate the predictive entropy to find the reliability of these uncertain predictions. High entropy means model is less confident and vice-versa.

Islam et al [24] compare the uncertainty of Transformer-based models (BERT, XL-Net) to that of RNN variations (LSTM, GRU) in classifying text data. They use Monte Carlo Dropout to estimate uncertainty and quantify the uncertainty using entropy. They found that with smaller amounts of data, baseline BERT outperforms all the other models in terms of prediction confidence.

To the best of our knowledge there is no proper uncertainty analysis done in the context of panic attacks. Since panic attack detection is an emergency use-case, uncertainty analysis is necessary for enhancing model reliability in detecting panic attacks.

Chapter 3

Background

In this chapter, we described some of the key concepts related to our study and then we also give some idea regarding the neural network models that we implemented.

3.1 Neural Networks

Neural networks are complex systems made up of interconnected nodes, or neurons, that work together to solve complex issues. Neural networks are computational models, intended to predict outcomes, understand patterns, and operate in different kinds of applications in artificial intelligence. Neural networks are a part of machine learning. A neural network is like a machine learning model which is designed to emulate the structure and activities of the human brain. Numerous applications, such as chatbots, image identification, predictive modeling, and natural language processing heavily rely on neural networks.

An artificial neural network's node layer is made up of three layers; input, one or more hidden layers, and output. Within each artificial neuron or node, weights and thresholds are combined. There is a predefined value which is called threshold value. When a node's output exceeds this value, it activates and sends data to the network's next tier. If not, no data is sent to the next tier of the network. Neural networks are based on the artificial neuron, a simple processing unit that mimics its biological counterpart. These networks, which are composed of layers of interconnected neurons, have remarkable capacities for analysis, learning, and generalization that are strikingly similar to the cognitive functions of the human brain.

In simple terms, a neural network is fed a lot of data at first. Before being used, neural networks go through training based on rules and learning materials. Once an input layer has been established, weights are applied. The output is more significantly influenced by larger weights than by smaller ones, which helps determine the relative relevance of each variable. Then, before being joined together, each input is multiplied by its matching weight. After that, the output is obtained by applying it to an activation function. When the node's output reaches a predetermined level, it activates and starts sending data to the network's next layer. Consequently, the output of one node becomes the input of the node that comes after it. Because of the manner that information is transferred between layers in this neural network, it

is categorized as a feedforward network.

Artificial neural networks come in several types, including feed-forward neural networks, recurrent neural networks (RNN), convolutional neural networks (CNN), and so on. Similar to feedforward networks, convolutional neural networks are typically used for pattern and image recognition. Recurrent neural networks are typically used for prediction purposes, using sequential or time series data. The memory of recurrent neural networks makes it different from other neural networks as it enables them to change the input and output at any given time by utilizing data from earlier inputs. There are several different kinds of RNN architectures, including Bidirectional Recurrent Neural Networks (BRNN), LSTM, and GRU. Neural networks are at the forefront of technological innovation and data-driven decision-making because of their capacity to replicate the complex workings of the brain and adapt to a wide range of situations. They make up the architecture of deep learning models.

3.2 Uncertainty analysis

The absence of information or assurance about anything is known as uncertainty. There are two kinds of uncertainty when it comes to deep learning which are epistemic uncertainty and aleatoric uncertainty. Data uncertainty, known as aleatoric uncertainty, is irreducible since it is an intrinsic quality of the data distribution rather than a model's property. On the other side, epistemic uncertainty, also known as knowledge uncertainty, results from insufficient knowledge in training data. Adding more data can reduce this kind of uncertainty. A model will never be capable of achieving epistemic confidence of zero since nothing can offer a limitless quantity of data. For applications in the real world, where datasets may be large in size but low in quality, epistemic uncertainty is a major issue.

A model's capacity to generalize to unknown data is evaluated using uncertainty analysis, which also measures the unpredictability or lacking of data in the output of algorithms. Estimating uncertainty is particularly crucial in the context of neural networks, which have a propensity for making overly optimistic predictions [23]. In crucial use cases like autonomous vehicles or healthcare, incorrect overconfident forecasts might be hazardous. Numerous research have looked into how deep learning models may articulate uncertainty, including sparse Gaussian processes and Bayesian neural networks. These are subject to several limitations, such as the lack of expressiveness in deep Bayesian neural networks. Sparse Gaussian processes, on the other hand, are more expressive but only gather uncertainty from higher-level latent space. As a result, the deep learning model they are based on lacks understanding and fails to take into account uncertainty from the initial dataset. There are some effective methods for quantifying uncertainty, such as deep ensembles, Monte Carlo dropout, and deep Bayesian active learning.

3.3 Monte-Carlo Dropout

Monte-Carlo Dropout (MCD) is a technique used in neural networks with dropout layers. It is a regularization technique used to prevent overfitting while training a

deep learning model. While training, MCD turns off a certain number of neurons during each backward and forward pass which helps in making the model not rely too much on a particular set of neurons. Furthermore, a percentage of neurons in the dropout layer are also turned off during the inference phase making the model give different outputs for same input if ran multiple time. According to Gal Gahramani et al [7], MCD allows uncertainty quantification using which a model's robustness and reliability can be figured out. Therefore, MCD is used in risky cases where the model's predictions need to be reliable like in medical and healthcare scenarios.

3.4 Entropy

Entropy is a representation of how much information there is in a random variable. The lower the entropy, the less information it provides because it is "less surprising" and vice versa. Therefore, entropy can be a measure of uncertainty [11]. Entropy is represented by the letter H, and the formula for calculating entropy is as follows [1]:

$$H = -\sum_{i=1}^{n} p_i \log p_i$$

Where,
H is the entropy.
p is the probability of an event.

3.5 Standard Deviation

The amount of variation or dispersion in a set of values is called standard deviation. In statistics, it provides a method to quantify the amount of spread or dispersion in a distribution. Lower standard deviation means more data points are closer to the mean, and the higher standard deviation means more data points have values spread out over a wide range. For calculating standard deviation we can use the formula below.

$$s = \sqrt{\frac{\sum_{i=1}^{n}(X_i - \bar{X})^2}{n-1}}$$

Here,
s is the Sample standard deviation.
X is the Each individual data point in the sample.
$\bar{X}$ is the Mean of the sample.
n is the Total number of data points in the sample.

Firstly, the mean (average) of the dataset has to be calculated. Then, the difference between the mean and a data point is squared, it is done for all the data points and then they are summed. After that, divide the result by (n-1). And lastly, take the

square root of the result.

In a nutshell, it helps in assessing the reliability of statistical conclusions based on a particular set of data.

3.6 Artificial Neural Network

Artificial Neural Network (ANN) is a computational model that was built to imitate how neurons work in the human brain. Zupan et al [2] states ANNs are capable of forming relationships between non-linear dependent inputs and outputs. ANNs consist of an input layer, one or many hidden layers, and an output layer. Each layer consists of neurons, and neurons of adjacent layers are connected to each other. Each of the connections have a weight assigned to it, and each neuron has an activation function which gives an output based on the inputs to the neuron. The weights of the links keep changing to ultimately have values that help to accurately give output. The weights change according to the difference between the predicted outputs of the ANN and the actual outputs. ANNs are usually used for classification, speech recognition, image recognition, natural language processing, etc. The figures 3.1 and 3.2 are illustrating ANN and a neuron in an ANN.

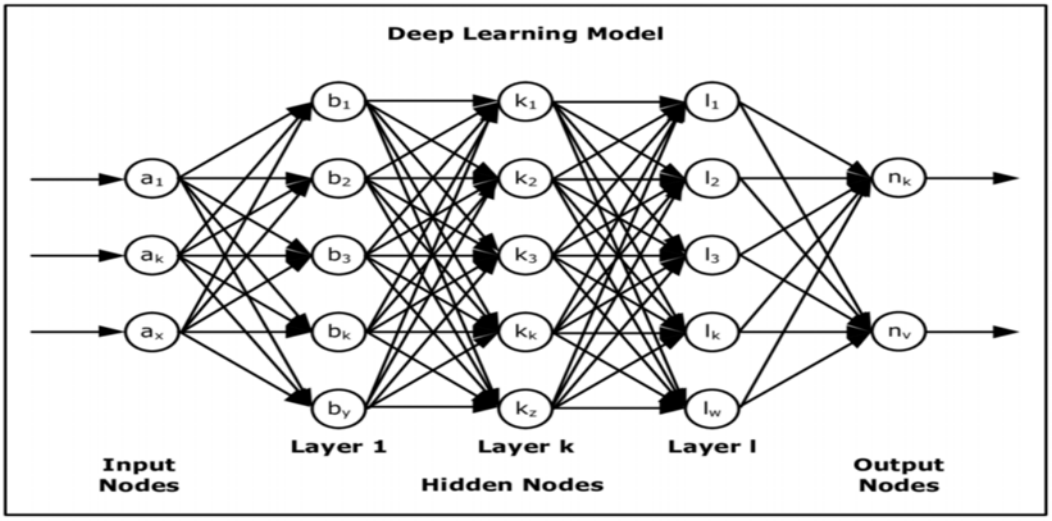

Figure 3.1: Structure of a Artificial Neural Network [9]

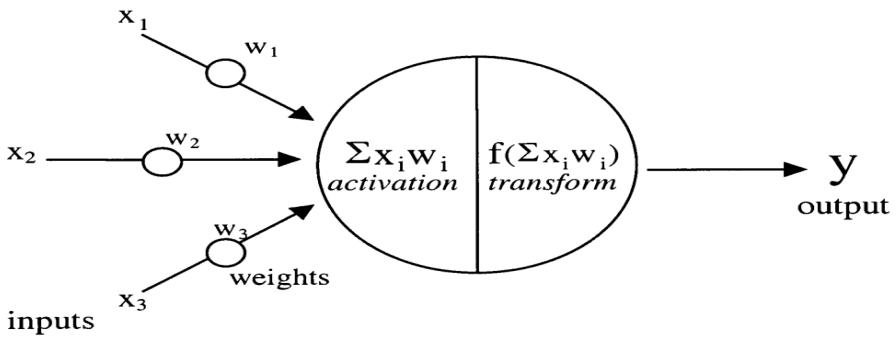

Figure 3.2: Structure of a Neuron in Artificial Neural Network [3]

3.7 Convolutional Neural Network

Similar to conventional ANNs, convolutional neural networks (CNNs) are made up of neurons that self-streamline as they learn. The main difference is that CNNs are traditionally used in image processing. CNNs are fundamentally predicated on the idea that the input will include images. Its architecture was constructed in a way that best satisfies the need to take into account a certain type of data. Three different types of layers make up the majority of CNN's architecture which are convolution, pooling, and fully linked layers. A simple CNN architecture is shown in figure 3.3.

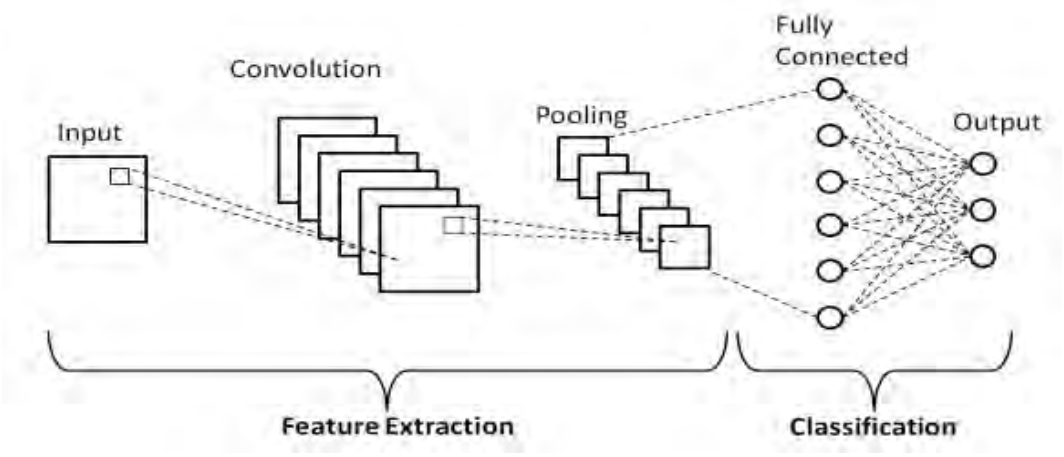

Figure 3.3: Architecture of a Convolutional Neural Network [14]

The most defining component of a CNN is the convolution layer, this is where high-level feature extraction occurs. It takes in input data, a filter, and a feature map. To put it simply, the filter will move across the receptive fields of the image extracting high-level features from them.

The pooling layer is responsible for decreasing the number of parameters in the input via the means of dimensionality reduction. It does so by sweeping a filter across the entire input not unlike what is done in the convolution layer, with the difference being that this filter lacks weights. Instead, the kernel populates the output array by applying an aggregation function (maximum or average pooling) to the values in

the receptive field.

Each node in the output layer is connected to a node in the fully-connected layer. Finally, based on the computation and feature extraction by the preceding layers' various filters, the fully-connected layer performs the classification [17].

3.8 Recurrent Neural Network

Recurrent neural network (RNN) is a type of neural network with a focus on handling sequences of data $x(t) = x(1), ..., x(\tau)$ with the time step index t ranging from 1 to τ. RNNs work well with sequential inputs, like speech. RNNs are called recurrent because they carry out identical tasks for every element of a sequence and the new output is dependent on the past outputs. To put it simply, RNNs have a "memory" which is capable of retaining past information. Figure 3.4 illustrates what an RNN looks like.

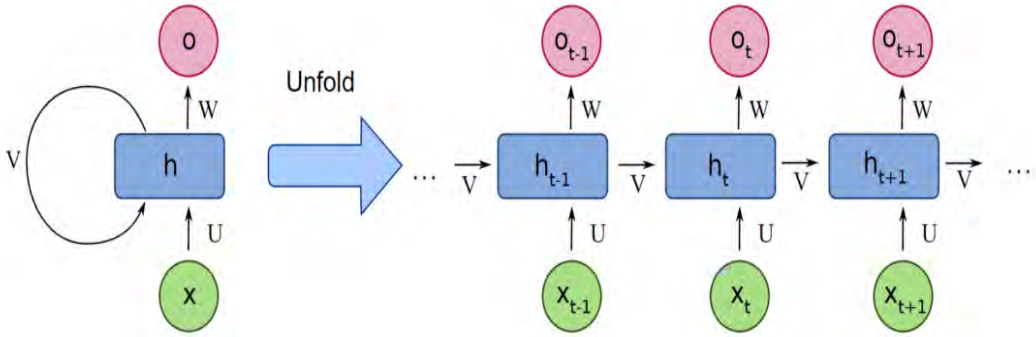

Figure 3.4: Architecture of a Recurrent Neural Network [33]

The left side of the figure 3.4 represents an RNN and the right side illustrates the RNN being "unfolded" into a complete network sequence.

$x(t)$ is the input of the network at time t.
$h(t)$ represents a hidden state at corresponding time t and acts as the aforementioned "memory" of the neural network.

Based on the current input $x(t)$ and the previous time step's hidden state $h(t-1)$, $h(t)$ is calculated: $h(t) = f(Ux(t) + Wh(t-1))$. The function f is a non-linear transformation like tanh or $ReLU$ [13].

3.9 Long Short-Term Memory Network

Long Short-Term Memory (LSTM) networks come within the category of recurrent neural networks in the context of deep learning. LSTM stands out as a very well-liked architecture that effectively tackles the vanishing gradient issue that is present in traditional RNNs. LSTMs contain memory blocks in them, which enables them to effectively remember long-term dependencies, making it better than RNN.

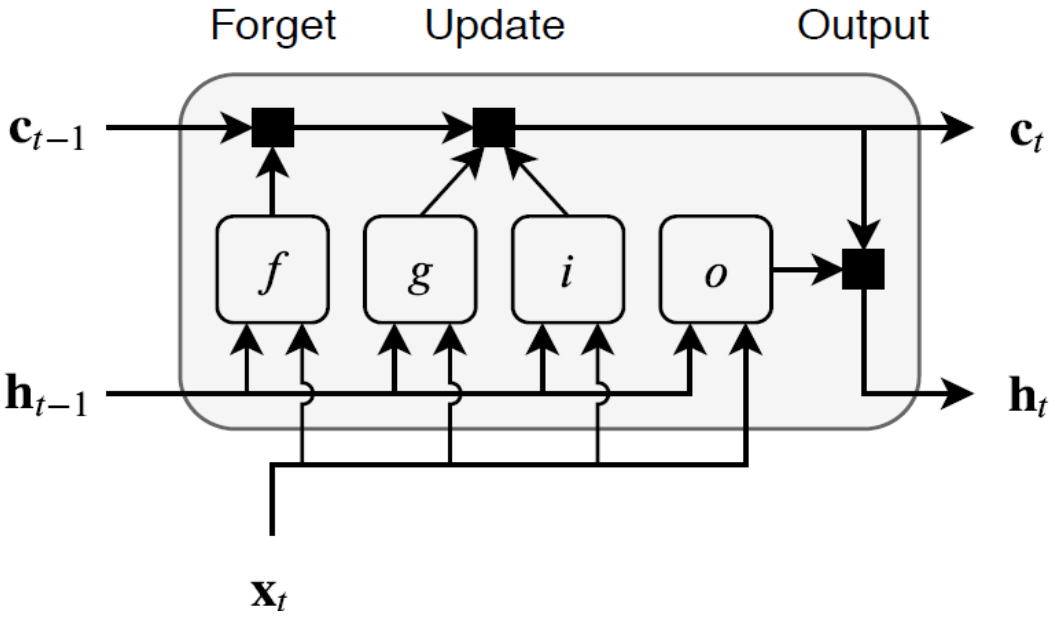

Figure 3.5: Architecture of a Long Short-Term Memory Network [35]

The structure of a LSTM cell includes input, output, and forget gates. LSTM works best with sequential data, and long sequences are able to take better advantage of LSTM's high efficiency. The memory cell managed by sigmoid gates within each LSTM unit regulates both reading and writing operations. These gates affect the input gate, forget gate, and output gate and operate as devices to enable information to pass only under certain conditions. The input gate of the LSTM controls the amount of new data to be added to the cell state and the forget gate controls the amount of data to be kept from the previous cell state. Lastly, the output gate controls the output of the cell by taking into consideration both the current input and the current cell state.

3.10 Gated Recurrent Unit

The Gated Recurrent Unit (GRU) is a more simplified variation of the LSTM. The forget and input gates of the LSTM are combined into a single update gate. Essentially, the update gate is responsible for determining the amount of past information to retain for future computations which helps eliminate the vanishing gradient problem present in RNN. GRU also has a reset gate which is essentially responsible for determining the amount of past information to forget. Figure 3.6 shows the overview of the GRU architecture.

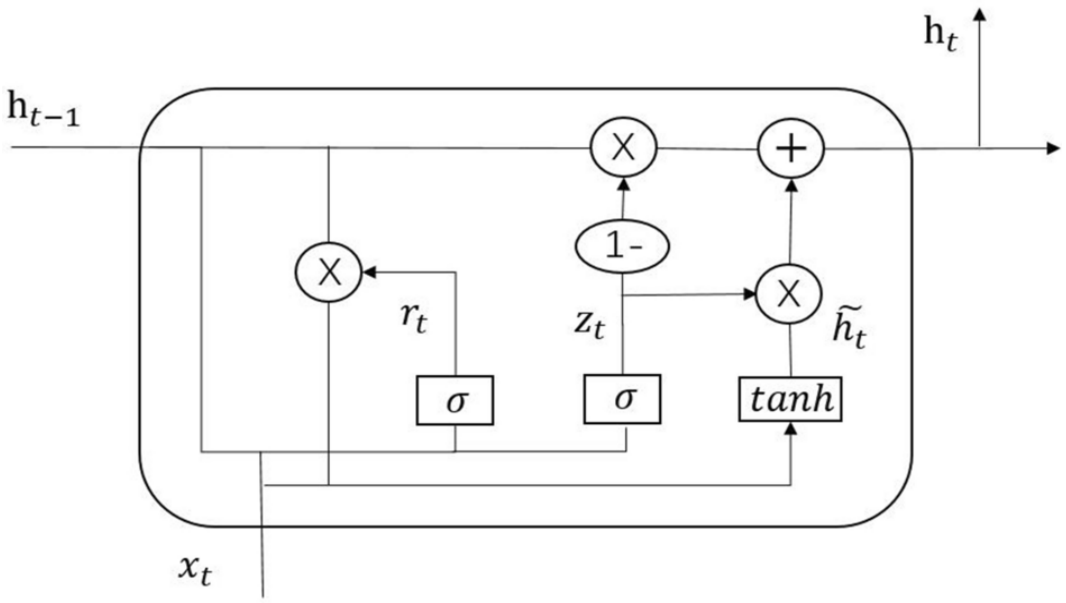

Figure 3.6: Architecture of a Gated Recurrent Unit [21]

3.11 Hybrid Model of CNN-LSTM

Hybrid models are usually the use of two different neural network models that work together to make accurate classification or regression. In the CNN-LSTM hybrid model, the convolution layers and max-pooling/average-pooling layers are used to extract high level information from the features [15]. The information extracted from the CNN part is given as input to the LSTM which computes and gives an output for the classification or regression problem. As mentioned before, an advantage of using LSTM is that they are good at capturing long-term dependencies. The figure 3.7 show how the hybrid model can be implemented.

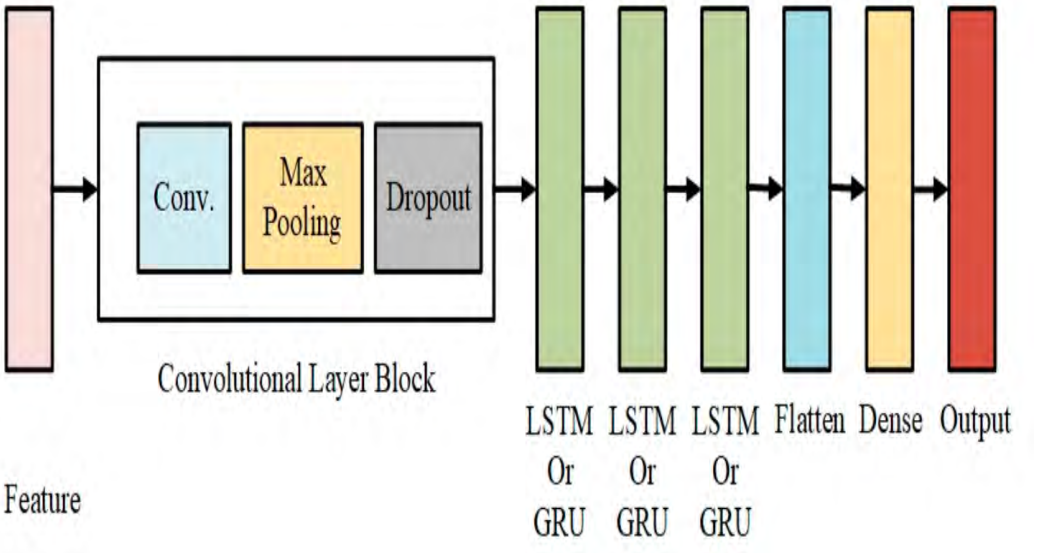

Figure 3.7: Architecture of a Hybrid CNN-LSTM/GRU [28]

3.12 Hybrid Model of CNN-GRU

As mentioned earlier, hybrid models are usually the use of two different neural network models that work together to make accurate classification or regression. In the CNN-GRU hybrid model, the convolution layers and max-pooling/average-pooling layers are used to extract high level information from the features [25], as was the case with CNN-LSTM. The information, which was extracted from the CNN part, is given as input to the GRU that computes and gives an output for the classification or regression task. As mentioned before, an advantage of using GRU is that they are good at capturing long-term dependencies and they are a variation of LSTM with fewer number of gates and hence has less computational complexity than LSTM. The figure 3.7 shows how the hybrid model can be implemented.

3.13 Ensemble Learning Model

In Ensemble Learning, multiple machine learning or deep learning models are trained using data from a dataset and then their predictions are combined by another model. There are several ways to implement ensemble learning, some of which are bagging, boosting, stacking and random subspace, etc [4]. The figure 3.8 represents how stacking ensemble model is implemented.

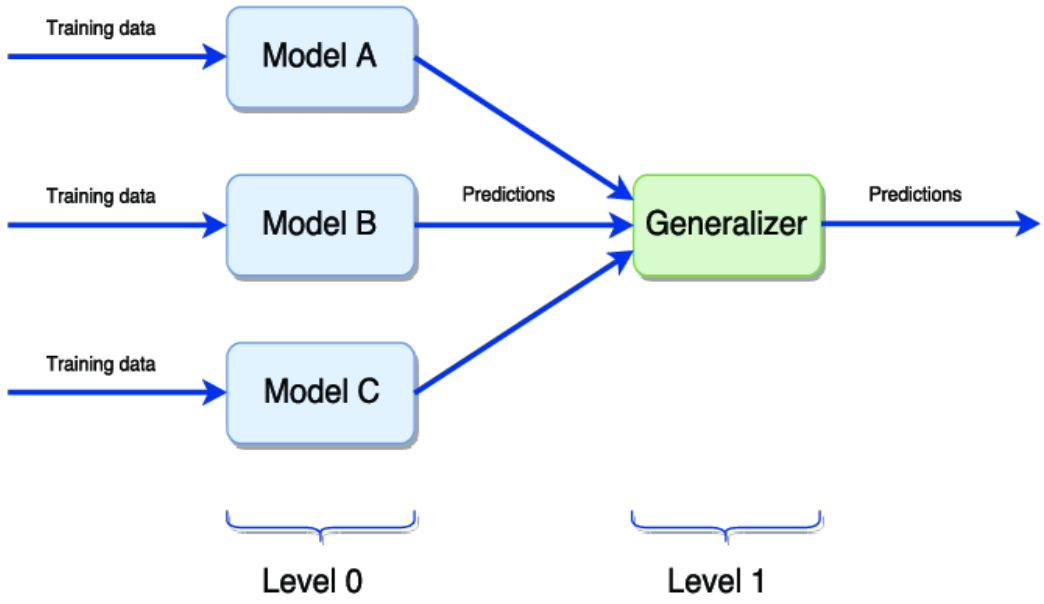

Figure 3.8: Architecture of a Stacking Ensemble Model [10]

Chapter 4

Methodology

The aim of the panic attack detection models is to identify whether the user is in panic or not. To do this, the models are trained using the training dataset, then tested using the testing dataset. The models output one of two predictions depending on if the person is in panic or not; "yes" or "no" respectively. Figure 4.1 shows an overview of how the models work.

Our proposed model is responsible for classifying and producing an appropriate prediction. It consists of four important stages:

1. Pre-processing the input data: This stage is focused on configuring the data in a way that helps the models to process them easily.
2. Training: This stage is focused on training the neural network models such as ANN, RNN, LSTM, GRU, CNN, Hybrid CNN-LSTM, Hybrid CNN-GRU and Ensemble models using the pre-processed data.
3. Prediction: This stage is focused on utilizing the trained neural network models for predicting whether the person is genuinely in panic or not.
4. Uncertainty Analysis: This stage is focused on visualisation and quantification of the uncertainty of our neural network models by using Monte-Carlo Dropout.

In the pre-processing stage, the dataset is split into two; the training data and the testing data. The training data is then passed on to the models in order to train them. The testing data is used to deduce the accuracy of the trained model in being able to detect genuine panic attacks. And finally, uncertainty analysis is done on them.

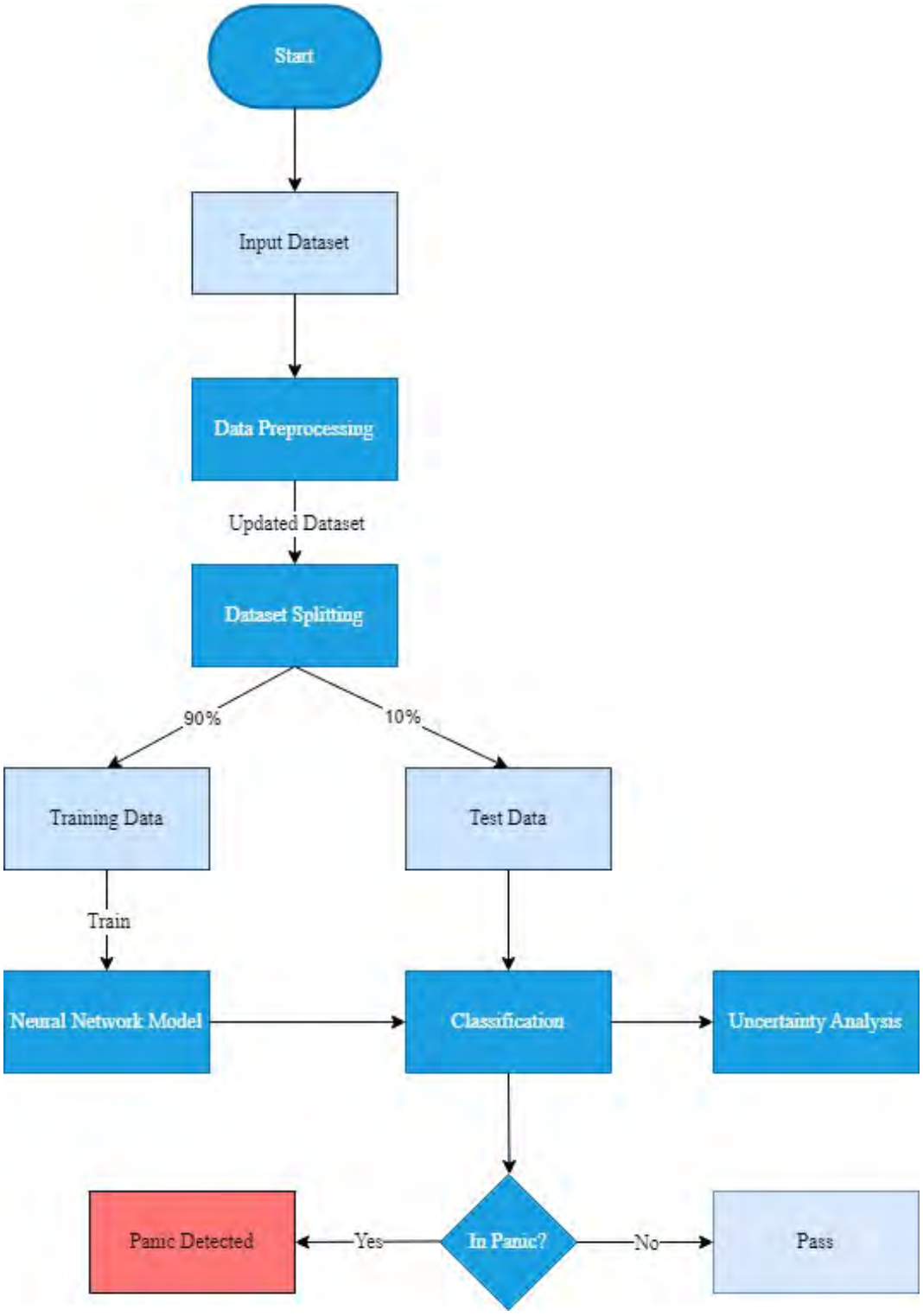

Figure 4.1: Flow Chart Representing the Training and Evaluation of our Models

Chapter 5

Implementation

5.1 Dataset

In our study, we used a dataset from the paper of Lazarou et al [26], who gave it public access so that others can use it. They made the dataset by gathering information from 27 different subjects of which only 2 were real humans and for the rest of the subjects' data was produced artificially. The 2 real subjects were monitored for a short instance of time during which, their data was being read and collected by a wearable and a smartphone. The features of the dataset are: secureid, timestamp, gender, age, weight, heartrate, hrv, speed, stepcount, activity, HRMAD10, HRMAD30, HRMAD60. And the label of the dataset is Outcome. The description of the features and label are given below:

- secureid - is an id unique to each subject which is taken from the phone.
- timestamp - is the date and time of readings.
- gender, age and weight - are self explanatory.
- heartrate - is the rate at which the heart beats per minute measured in bpm.
- hrv - is heart rate variation and is the time difference between 2 consecutive heart beats measured in milliseconds (ms).
- speed - is the speed of the person, measured in km/hr.
- stepcount - is the step count of the person, measured in steps per minute.
- activity - what the person is doing
- HRMAD10 - is heart rate moving average deviation, measured the change in heart rate for 10 seconds from a moving average rate.
- HRMAD30 - is heart rate moving average deviation, measured the change in heart rate for 30 seconds from a moving average rate.
- HRMAD60 - is heart rate moving average deviation, measured the change in heart rate for 60 seconds from a moving average rate.

- Outcome - Whether the person is in panic or not. "1" represents that the subject is in panic, and "0" represents that the subject is not in panic

Figure 5.1 shows some example rows of the dataset:

secureid	timestamp	gender	age	weight	heartrate	hrv	speed	stepcount	activity	HRMAD10	HRMAD30	HRMAD60	linearacc	Outcome
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	71	845	0	0	STILL	-1	0	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	71	845	0	0	STILL	-2	-1	-2	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	-2	-1	-2	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	-1	-1	-2	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-2	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-2	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0
wva29y8ksr6fo4j	8/18/2022 11:34	female	77	60	70	857	0	0	STILL	0	-1	-1	0	0

Figure 5.1: Example Rows of the Dataset

5.2 Data Pre-processing

In the previous section, the features of the dataset we used were mentioned. To train our Neural Network models we had to do some pre-processing to the raw features of the initial dataset. Firstly, we dropped 5 columns "secureid", "timestamp", "HRMAD10", "HRMAD30" and "HRMAD60". The columns "secureid" and "timestamp" were dropped because they are nominal categorical features. The columns "HRMAD10", "HRMAD30" and "HRMAD60" were dropped because they were derived features which we did not want to use in order to reduce time and computational complexities of the models. Secondly, the data of "gender" and "activity" were given as ordinal categorical feature so we had to apply feature encoding to these 2 columns. Lastly, we split the dataset having 16200 rows into 2 portions for training and testing our models. The ratio of train : test sets were 90 : 10.

5.3 Neural Network Parameters

Table 5.1 shows the hyperparameters of the neural network models that we used.

Table 5.1: Hyperparameters of the Neural Network Models

Neural Network Model	Hyperparameters
ANN	Dense input layer with 64 units, Activation: relu, Dense hidden layer with 32 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, Epochs: 10
CNN	Input Conv1D layer with 64 filters and kernel size of 3, Activation: relu, MaxPooling1D with pool size 2, Dense hidden layer with 32 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, Epochs: 10
RNN	Fully connected RNN input layer with 64 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, epochs: 10
LSTM	Fully connected LSTM input layer with 64 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, epochs: 10
GRU	Fully connected GRU input layer with 64 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, epochs: 10
CNN-LSTM	Input Conv1D layer with 64 filters and kernel size of 3, Activation: relu, MaxPooling1D with pool size 2, Fully connected LSTM with 32 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, epochs: 10
CNN-GRU	Input Conv1D layer with 64 filters and kernel size of 3, Activation: relu, MaxPooling1D with pool size 2, Fully connected GRU with 32 units, Activation: relu, Dense output layer with 1 unit, Activation: sigmoid, Optimizer: adam, Learning rate: 0.001, epochs: 10
Ensemble Learning	Used above mentioned CNN, RNN, LSTM, and GRU as the estimators, 10 epochs each, MLPClassifier as the final estimator, Optimizer: adam, Learning rate: 0.001

5.4 Evaluation Metrics

The performance of the models were compared based on four metrics; *accuracy*, *precision*, *recall* and $f1 - score$. The formulas for calculating each of the metrics are shown below:

$$Accuracy = \frac{CorrectPredictions}{TotalPredictions}$$

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$F1Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

5.5 Uncertainty Analysis

For uncertainty analysis, we applied Monte-Carlo dropout on all the neural network models we used for detecting panic attacks. Then we used different techniques to visualise and quantify the uncertainty of the models we used.

5.5.1 Using Entropy

During the testing process, the prediction probability of the subject being in panic or not were calculated for each model. If the probability was more than 0.5, the subject will be classified as being in panic. Whereas, if the probability was less than 0.5, the subject is classified as not being in panic. Hence, the closer the probabilities are to 0.5, the more uncertain the model is on whether the subject is in panic or not. And if the probabilities are closer to 0 or 1, then the model is more certain of the subject not being in panic or being in panic respectively. Based on this idea, we use *entropy* to quantify the uncertainty of each model. In our case, the subject has two possibilities; being in panic, or not being in panic. According to Shannon, C. E. [1], when we have two probabilities p and q where $q = 1 - p$, like in our case, entropy can be calculated using:

$$H = -(p \log_2 p + q \log_2 q)$$

Replacing q with $1 - p$, we get:

$$H = -(p \log_2 p + (1 - p) \log_2 (1 - p))$$

5.5.2 Using Standard Deviation

The second way we did uncertainty analysis is by running the models with Monte-Carlo dropout numerous times using the test data. And then we calculated the standard deviation of the predictions for each instance in the test dataset, next we calculated the mean of all those standard deviations. A lower value of mean standard deviation, represents lower uncertainty and vice versa.

Chapter 6

Results and Analysis

After training the neural network models on the train dataset, they are run on the test dataset to generate predictions and these are compared with the label of the test dataset. As mentioned before, the performance of the models was analyzed using *accuracy*, *precision*, *recall*, and $f1 - score$:

The scores of each of the models are shown below on the graphs 6.1, 6.2, and table 6.1:

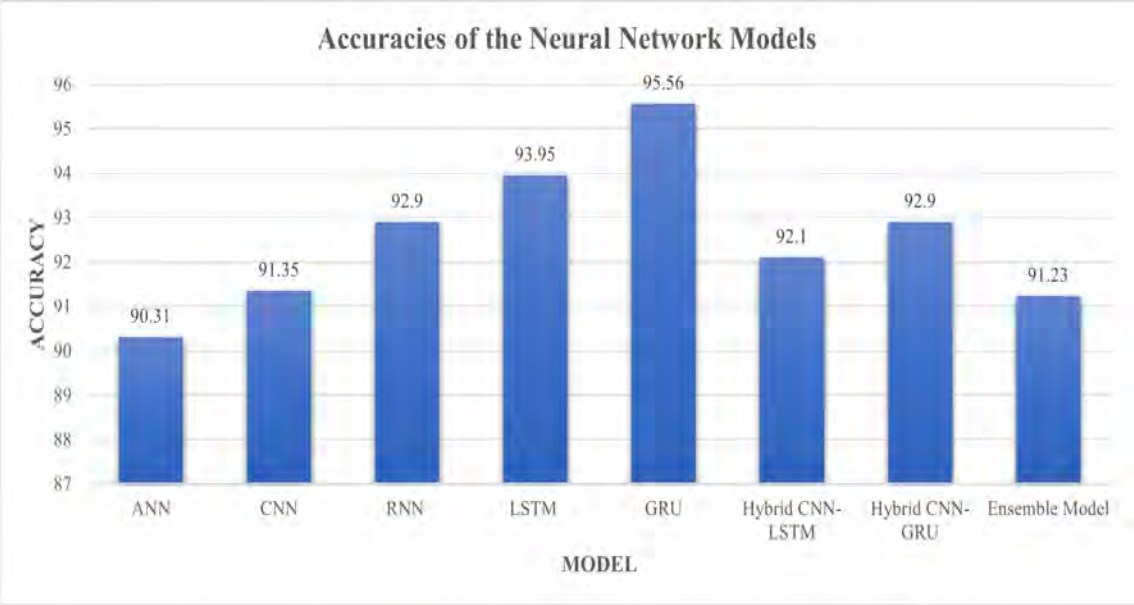

Figure 6.1: Accuracies of the our Models

Table 6.1 shows the performance of the neural network models including the following: ANN, CNN, RNN, LSTM, GRU, Ensemble model, Hybrid CNN-LSTM, Hybrid CNN-GRU, and all models after implementing Monte Carlo dropout. The range of accuracy values is 86.60% to 95.56%. With 95.56% accuracy, the GRU model achieves the best accuracy. With 86.60% accuracy, the ANN MC model has the lowest accuracy. With a 93.95% accuracy rate, the LSTM model likewise demonstrates strong performance. The accuracy of the hybrid models, which combine the best features of the CNN and LSTM/GRU models, is 92.10% for Hybrid CNN-LSTM and 92.90% for Hybrid CNN-GRU. The accuracy of the ensemble model is

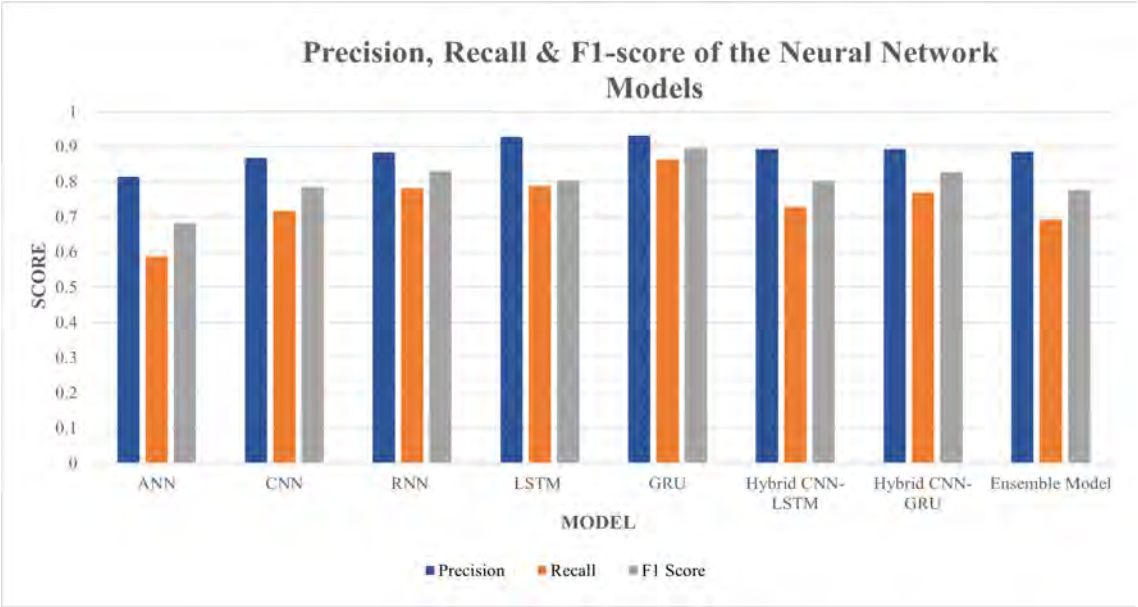

Figure 6.2: Score of our Models

Table 6.1: Scores of the Models

Neural Network Model	Accuracy	Precision	Recall	F1-score
ANN	90.31	0.8139	0.5882	0.6829
ANN MC	86.60	0.8300	0.4649	0.5960
CNN	91.35	0.8677	0.7171	0.7852
CNN MC	89.88	0.8129	0.7058	0.7556
RNN	92.90	0.8829	0.7815	0.8291
RNN MC	87.96	0.7217	0.6975	0.7094
LSTM	93.95	0.9273	0.7871	0.8028
LSTM MC	91.48	0.8323	0.7787	0.8046
GRU	**95.56**	**0.9305**	**0.8627**	**0.8953**
GRU MC	90.62	0.8359	0.7563	0.7941
Hybrid CNN-LSTM	92.10	0.8934	0.7282	0.8024
Hybrid CNN-LSTM MC	90.31	0.8485	0.6750	0.7519
Hybrid CNN-GRU	92.90	0.8928	0.7703	0.8270
Hybrid CNN-GRU MC	90.62	0.8601	0.6890	0.7651
Ensemble model	91.23	0.8853	0.6919	0.7767
Ensemble model MC	90.62	0.8648	0.6807	0.7618

91.23%.

Figure 6.2 and table 6.1 show that GRU has the best precision score at 0.9305. GRU also has the best recall and F1-score at 0.8627 and 0.8953 respectively.

Based on the above figures and table, GRU has the best overall performance in detecting panic attacks.

6.1 Uncertainty Analysis using Predictions

We plotted a scatter graph to visualise the uncertainty of each model by running each model 10 times on the test data. Then we implemented entropy on each prediction to get a numeric representation of the uncertainty of each prediction. The formula used was shown in section 5.5.1. Summing entropy of all the predictions and then dividing them by the total number of predictions gives the average entropy of each prediction. The results are shown in table 6.3.

Furthermore, we used another method to quantify the uncertainty, where we calculated the percentage of the predictions that were between 0.2 to 0.8 range. The higher the percentage of predictions within the stated range, the more uncertain the model is at predicting panic attacks. The graphs of each model are shown in figures 6.3 to 6.10, and the percentage of predictions in range 0.2 to 0.8 for each model are given in table 6.2.

Table 6.2: Percentage of Prediction within 0.2 to 0.8 range of our models

Models	Percentage of Predictions within 0.2 to 0.8
ANN MC	32.57%
CNN MC	24.16%
RNN MC	30.74%
LSTM MC	21.94%
GRU MC	18.00%
Hybrid CNN-LSTM MC	20.03%
Hybrid CNN-GRU MC	16.28%
Ensemble Model MC	**15.30%**

From the table 6.3, we can see that the GRU MC has the lowest entropy which is 0.3717 which makes it the most certain model. Apart from GRU MC, the Hybrid CNN-GRU MC model has the second lowest entropy at 0.3885. And RNN MC followed by ANN MC have the highest entropies at 0.5201 and 0.4288 respectively, making them the least certain models in predicting panic attacks by use of this uncertainty analysis technique.
From the table 6.2, we can see that the Ensemble model has the least percentage

Table 6.3: Average Entropy of Predictions of our Models

Neural Network Model	Average Entropy of Predictions
ANN MC	0.4288
CNN MC	0.4125
RNN MC	0.5201
LSTM MC	0.4028
GRU MC	**0.3717**
Hybrid CNN-LSTM MC	0.3961
Hybrid CNN-GRU MC	0.3885
Ensemble Model MC	0.3951

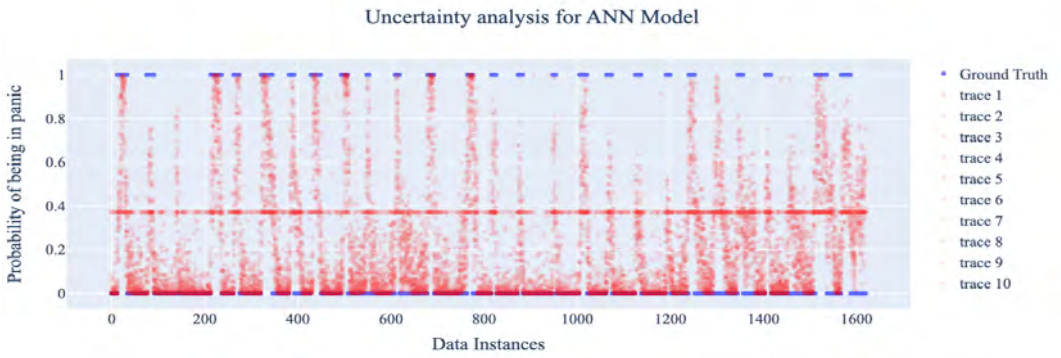

Figure 6.3: Uncertainty analysis for ANN MC

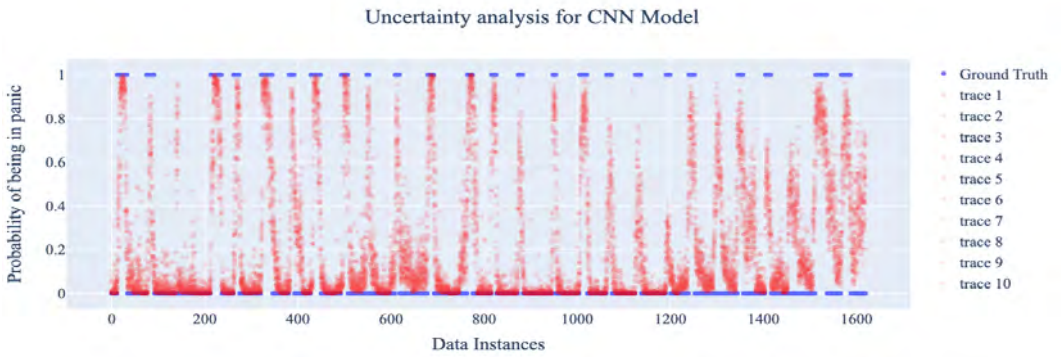

Figure 6.4: Uncertainty analysis for CNN MC

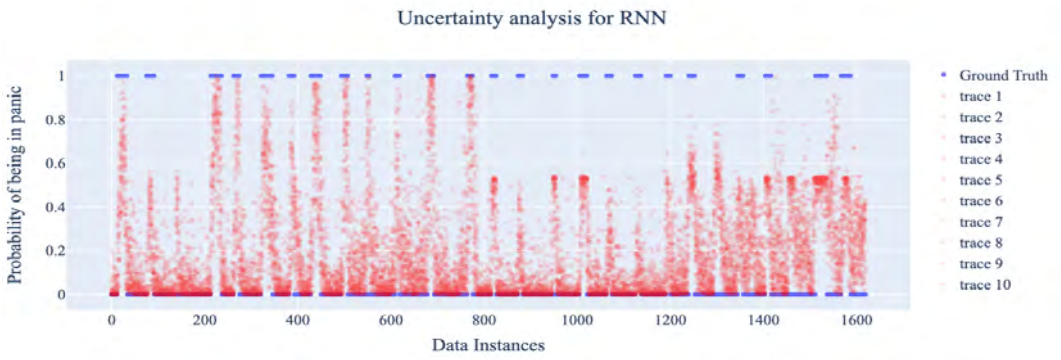

Figure 6.5: Uncertainty analysis for RNN MC

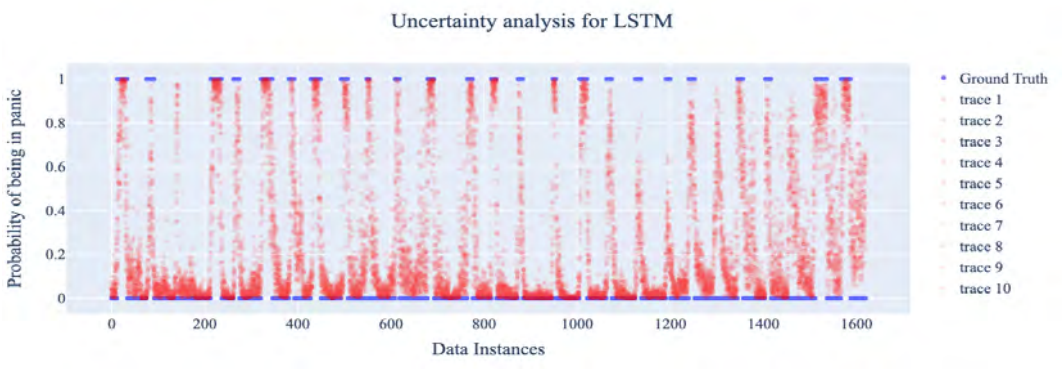

Figure 6.6: Uncertainty analysis for LSTM MC

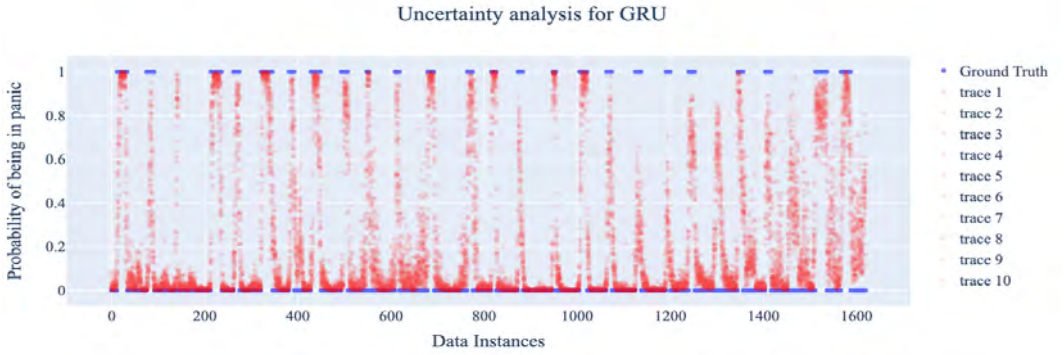

Figure 6.7: Uncertainty analysis for GRU MC

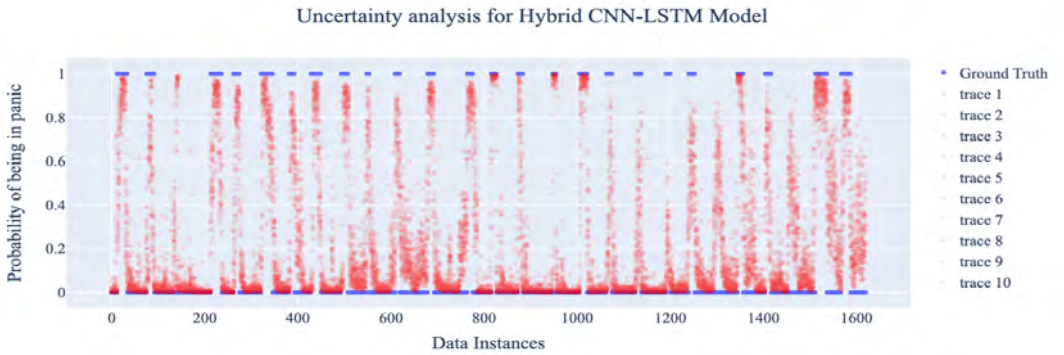

Figure 6.8: Uncertainty analysis for Hybrid CNN-LSTM MC

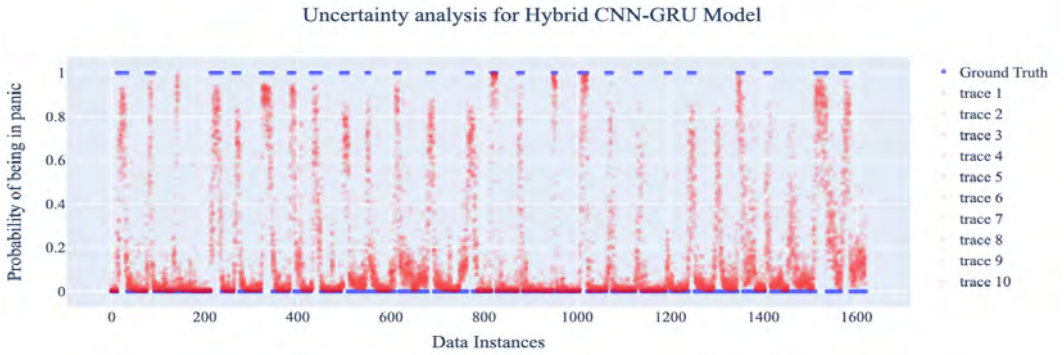

Figure 6.9: Uncertainty analysis for Hybrid CNN-GRU MC

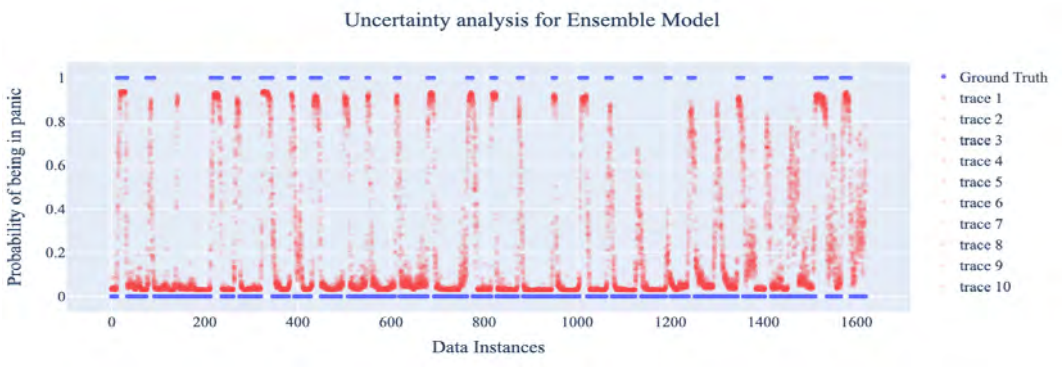

Figure 6.10: Uncertainty analysis for Ensemble Model MC

of predictions within the range of 0.2 to 0.8 which is 15.30%. That aside, Hybrid CNN-GRU MC has the second-lowest percentage of prediction at 16.28%. Apart from those, the GRU MC has the next lowest percentage of prediction at 18.00% within the stated range. And ANN MC followed by RNN MC have the highest percentages at 32.57% and 30.74% respectively, making them the least certain models in predicting panic attacks by use of this uncertainty analysis technique.

6.2 Uncertainty Analysis using Standard Deviation

We also ran the models with Monte-Carlo dropout 100 times using the test data. And then we calculated the standard deviation of each of the predictions for every instance in the test data with which we plotted another scatter graph for each of the models in order to visualise the uncertainty of the models. In the graphs, the more number of points that are near zero the more certain the predictions are for that model and the more number of points that are towards the top of the graph, the more uncertain the predictions of the model are. Also, in order to quantify the uncertainty from these graphs so that comparison of the models can be done, we computed the mean standard deviation from the standard deviations we calculated before. In case of the mean standard deviation, the lower it is the more certain the model is about its predictions and the higher it is, the more uncertain the model is about its prediction of the panic attacks. The graphs of the standard deviations are shown in figures 6.11 to 6.18. The mean standard deviation of each model is given in table 6.4.

From table 6.4, we can see that the mean standard deviation of the ensemble model is the lowest, with a value of 0.0370 which makes it the most certain model with this uncertainty analysis technique. That aside, GRU MC has the second lowest value of mean standard deviation which makes it the second-most certain model. Apart from those, Hybrid CNN-GRU MC model has a low standard deviation which is 0.0600 and is closely followed by the Hybrid CNN-LSTM MC model with a value of 0.0645. ANN MC and RNN MC are the most uncertain models with the highest mean standard deviation values at 0.1109 and 0.1051 respectively.

Table 6.4: Mean Standard Deviation of the models

Models	Mean Standard Deviation
ANN MC	0.1109
CNN MC	0.0648
RNN MC	0.1051
LSTM MC	0.0787
GRU MC	0.0562
Hybrid CNN-LSTM MC	0.0645
Hybrid CNN-GRU MC	0.0600
Ensemble Model	**0.0370**

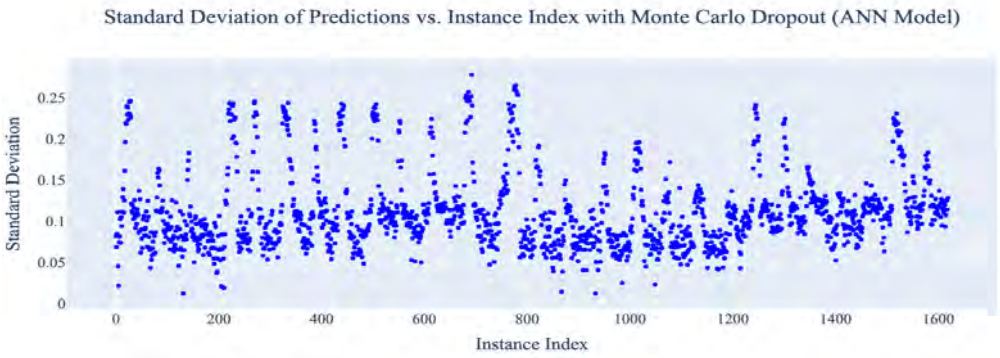

Figure 6.11: Standard Deviation Plot for ANN MC

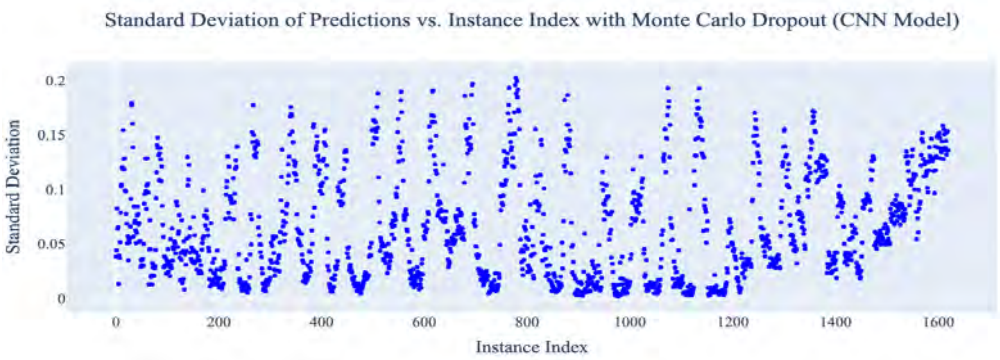

Figure 6.12: Standard Deviation Plot for CNN MC

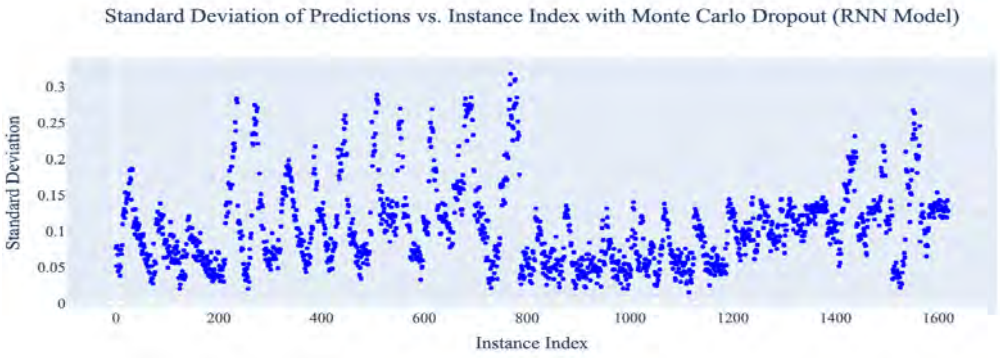

Figure 6.13: Standard Deviation Plot for RNN MC

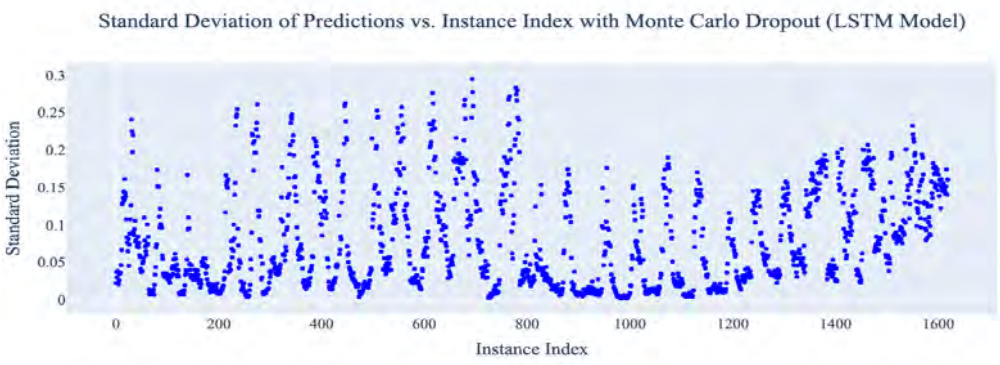

Figure 6.14: Standard Deviation Plot for LSTM MC

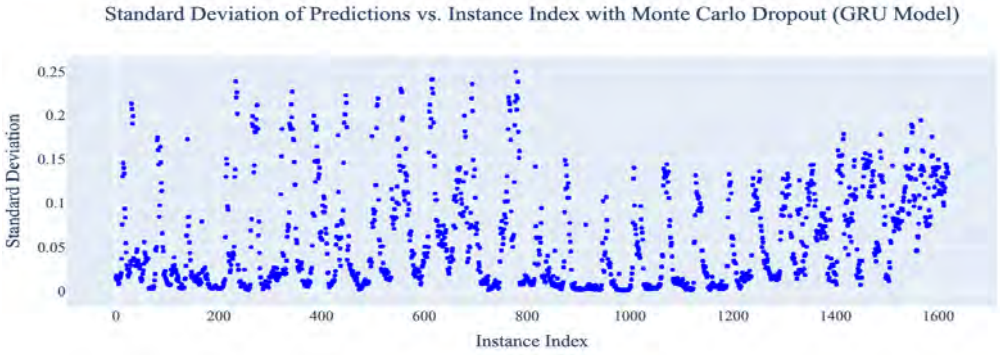

Figure 6.15: Standard Deviation Plot for GRU MC

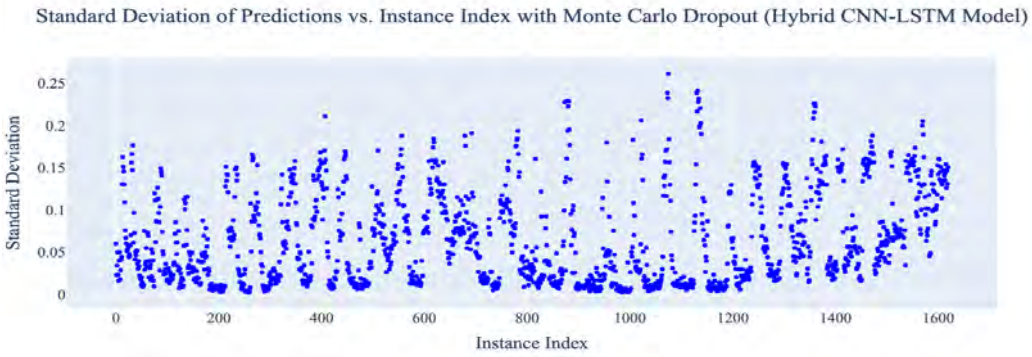

Figure 6.16: Standard Deviation Plot for Hybrid CNN-LSTM MC

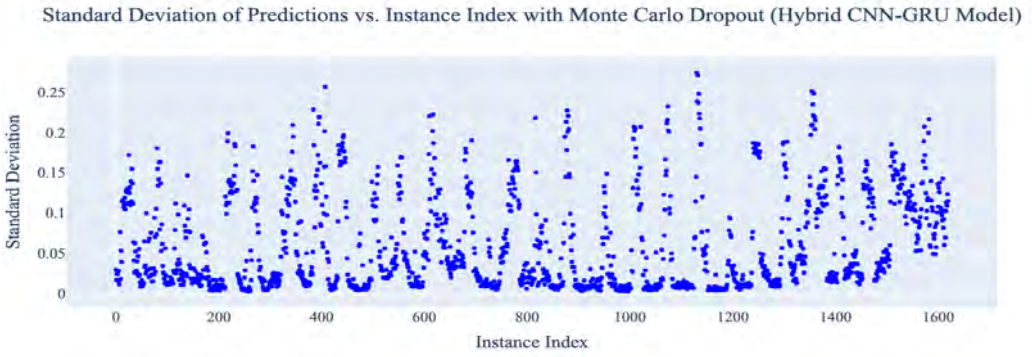

Figure 6.17: Standard Deviation Plot for Hybrid CNN-GRU MC

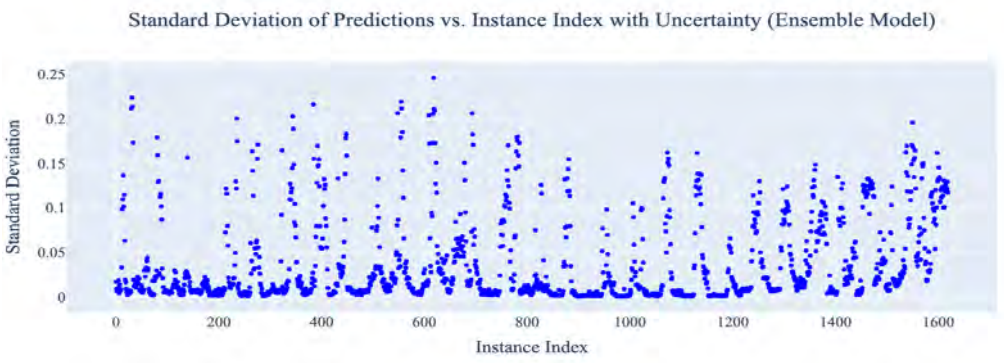

Figure 6.18: Standard Deviation Plot for Ensemble Model MC

To sum up, the Ensemble model overall performed best while evaluating uncertainty by two different ways. GRU MC model performed best while evaluating uncertainty by entropy. The Hybrid models also performed very well in all the uncertainty evaluation techniques we used. The hybrid CNN-GRU MC performed very close to GRU MC and even performed better than GRU MC when evaluated using percentage of prediction in range of 0.2 to 0.8. Finally, RNN MC an ANN MC both were the most uncertain models as they did the worst in all of the uncertainty evaluation techniques.

Chapter 7

Discussion

In this chapter, we will provide a comparative analysis of the studies that are related to ours and also point out the limitations of our study.

7.1 Comparative Analysis

The table 7.1 shows a qualitative comparison between related studies. Here we can see that many of them focuses on the panic detection of individuals [6], [8], [5], [22], [26] and others on crowds of people [18]. Some aim to detect panic attacks in general [18], [26], [8] while others focuses on specific type of panic attack [5], [6]. Also different types of approaches are taken for detecting panic attacks such as using machine learning models, deep learning models, logic-based algorithms, neural network models, ensemble learning models and Personalised models. And different kinds of data are used for the detections such as biometric/physiological data, spatiotemporal data and video feeds.

However, no existing studies focuses on uncertainty analysis of their models' predictions. Our main focus in this study was to achieve high accuracy from the models we used as well as to visualise and quantify the uncertainty of those models. The performance matrices and uncertainty analysis matrices have been discussed and presented in chapter 6. Furthermore, due to using the dataset created by [26] we discuss and compare their study with ours in details, below. The scores of their [26] models are shown in tables 7.2 and 7.3:

Table 7.2 shows the accuracy scores of the models implemented by Lazarou et al [26] from this it can be seen that the Gaussian SVM has the highest accuracy of 93.2% and this accuracy was achieved by training the model with only raw features. They also ran the models by including some derived features independently which are HRMAD10, HRMAD30 and HRMAD60 and got the highest accuracy of 94.2%, 94.4% and 94.5% respectively the accuracy are shown in table 7.3. Apart from running ML models they also ran two neural network models DNN and LSTM and they achieved the highest accuracies of 93.4% and 94.0% respectively when they trained the models with raw feature and HRMAD60. When they trained those two models with only raw features their accuracies were 91.5% for DNN and 90.6% for LSTM.

In comparison, we trained our neural network models with only the raw features in

Table 7.1: Qualitative Comparison of Related Studies

Author	**Year**	**Classification using**	**Uses Hardware**	**Focuses On**	**Detects Panic Attacks of**	**Type of Data Used**	**Uncertainty Analysis**
Lazarou et al	2022	ML and 2 Neural Network models	Yes (Only for Data Generation)	Individual People	All people, no particular target group	Biometric and Spatiotemporal	No
Ammar et al	2021	DL models	No	Crowds of People	All people, no particular target group	Video Feed	No
Petrescu et al	2021	ML models	No	Individual People	Detects Dread	Physiological	No
Sapounaki et al	2017	Logic-Based Algorithm	Yes	Individual People	All people, no particular target group	Biometric	No
Rubin et al	2015	Personalised Prediction models	Yes	Individual People	People with Panic Disorder	Biometric	No
Cruz et al	2015	Panic Prediction models	Yes	Individual People	People with Panic Disorder	Biometric	No
Our study	2024	Neural Network and Ensemble Learning models	No	Individual People	All people, no particular target group	Biometric and Spatiotemporal	Yes

Table 7.2: Accuracies of the machine learning Models of Lazarou et al [26]

ML Classifier Model	**Accuracy**
SVM kernel	90.6%
Logistic Regression	89.6%
Gaussian Naïve Bayes	79.6%
Decision Tree	91.4%
Boosted Trees	90.90%
Kernel Naïve Bayes	80.3%
Gaussian SVM	**93.2**%
DNN	91.5%
LSTM	90.6%

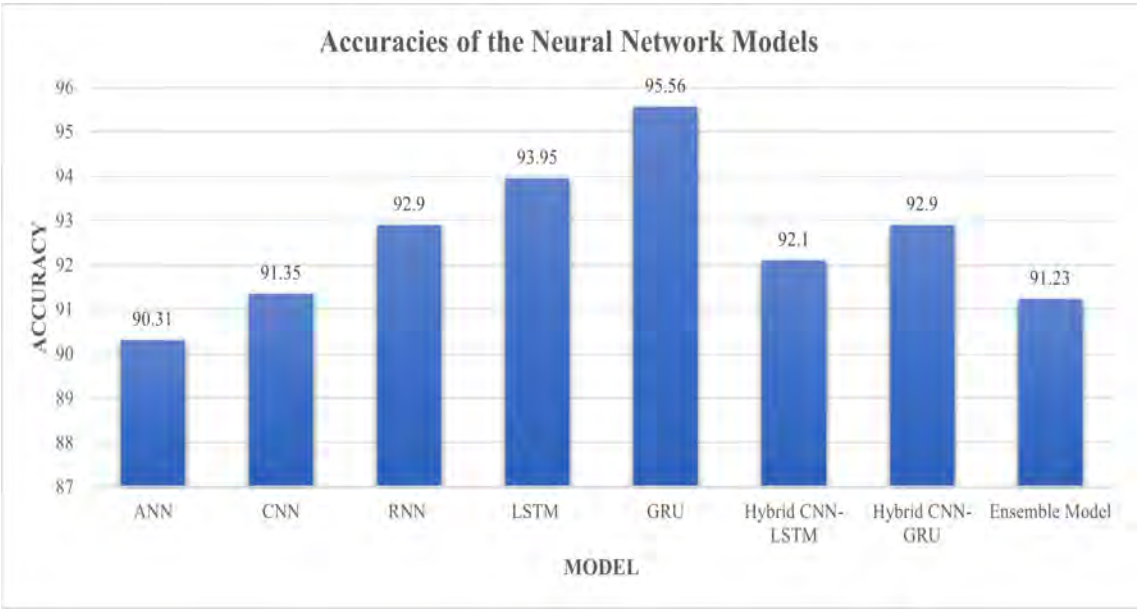

Figure 7.1: Accuracies of Our Neural Network Models

Table 7.3: Accuracies of The Models Used by Lazarou [26] with Derived Features

Classification Model	**Accuracy (Raw Features + HRMAD10)**	**Accuracy (Raw Features + HRMAD30)**	**Accuracy (Raw Features + HRMAD60)**
Decision Tree	93.3%	92.8%	92.8%
Logistic Regression	89.5%	89.0%	89.5%
Gaussian Naïve Bayes	80.4%	80.6%	81.3%
Kernel Naïve Bayes	83.7%	84.3%	85.3%
Gaussian SVM	**94.2%**	**94.4%**	**94.5%**
SVM Kernel	91.8%	92.3%	94.1%
Boosted Trees	92.3%	93.3%	93.9%
DNN	91.8%	92.3%	93.4%
LSTM	90.8%	91.4%	94.0%

order to reduce computational complexity of adding more features like HRMAD60. The highest accuracy we got was 95.56% for the GRU model, followed by LSTM with an accuracy of 93.95%. The accuracies of our models without monte-carlo dropout can be seen in figure 7.1. From this we can see that our LSTM performed better than their [26] LSTM model trained with just the raw features and actually outperformed their model with highest accuracy score which was 93.2% of the Gaussian SVM model trained with just raw features. And our GRU model's accuracy is higher than their overall highest accuracy score which was achieved by their [26] Gaussian SVM model being trained with the raw features and HRMAD60 the value being 94.5%.

In summary, our neural network models achieved higher accuracy scores than the models used by Lazarou [26] despite them using derived features to increase the accuracies. Therefore, our models performed better with less amount of features and also had reduced computational complexity due to not using the derived features.

7.2 Limitations

Our study may have certain limitations. Firstly, the dataset we used to train our models is not very big so the models could be made more robust and accurate by training them with more data. Moreover, we could not find any other datasets relevant to our study, so we could not confirm the models' performance on other datasets. Secondly, the computational cost of training neural network models may be more due to them being able to extract important features by themselves without requiring much pre-processing. Finally, for real life use of these models to detect panic attacks, devices like mobile phones and smart watches may not have the capability to run these neural network models with high computational complexity as we have not tested the models on such devices.

To overcome these challenges, in the future we plan to gather more real world data from people suffering from panic attacks and fine-tune our models' hyper parameters even more so that panic attacks can be detected from less number of features.

Finally for real life usage of our system, we may utilise cloud computing for doing the computations in order to detect panic attacks.

Chapter 8

Conclusion

Incorporating neural network models, and Monte Carlo dropout for uncertainty analysis, this study makes a crucial and new advance in the field of panic attack detection. The novelty of this research comes from a crucial need for reliable panic detection systems, particularly in situations where people's safety and security are crucial, such as on busy roadways and in metropolitan settings.

Today's society requires innovative responses to new threats against personal safety because of the unpredictable nature and inherent dangers connected with public areas, roads, and urban surroundings. Crime, accidents, and medical emergencies frequently happen without notice, leaving people exposed to unfavourable results. Because it may be able to prevent injury, save lives, and lessen the widespread anxiety brought on by the unpredictability of urban and roadside surroundings, the ability to recognise and react to panic-inducing circumstances reaches a new level of significance. Furthermore, by measuring the inherent risks associated with real-world data, the inclusion of uncertainty analysis using Monte Carlo dropout increases the robustness and credibility of panic detection. Finally, from our study we came to a conclusion that GRU was the best neural network model for panic attack detection as it had the best accuracy at 95.56% and was also among the least uncertain models compared to the others.

For our future work, we would like to collect and use more data to train the models in order to make the detection of panic attacks more robust and accurate. Furthermore, we would like to implement a wearable system which would take in data of the wearer in real time and apply our trained model with the best performance to detect panic attacks of individuals in real world situations.

Bibliography

[1] C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, no. 4, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb00917.x.

[2] J. Zupan, "Introduction to artificial neural network (ann) methods: What they are and how to use them," *Acta Chimica Slovenica*, vol. 41, pp. 327–327, 1994.

[3] S. Agatonovic-Kustrin and R. Beresford, "Basic concepts of artificial neural network (ann) modeling and its application in pharmaceutical research," *Journal of pharmaceutical and biomedical analysis*, vol. 22, no. 5, pp. 717–727, 2000.

[4] M. Sewell, "Ensemble learning," *RN*, vol. 11, no. 02, pp. 1–34, 2008.

[5] L. Cruz, J. Rubin, R. Abreu, S. Ahern, H. Eldardiry, and D. G. Bobrow, "A wearable and mobile intervention delivery system for individuals with panic disorder," in *Proceedings of the 14th International Conference on Mobile and Ubiquitous Multimedia*, 2015, pp. 175–182.

[6] J. Rubin, H. Eldardiry, R. Abreu, *et al.*, "Towards a mobile and wearable system for predicting panic attacks," in *Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, 2015, pp. 529–533.

[7] Y. Gal and Z. Ghahramani, *Dropout as a bayesian approximation: Representing model uncertainty in deep learning*, Oct. 2016. [Online]. Available: https://arxiv.org/abs/1506.02142.

[8] M. Sapounaki, I. Polychronou, M. Gkonta, A. Vogiatzoglou, and A. Kakarountas, "Wearable panic attack detection system," in *Proceedings of the 21st Pan-Hellenic Conference on Informatics*, 2017, pp. 1–2.

[9] W. Serrano, "Smart internet search with random neural networks," *European Review*, vol. 25, pp. 1–13, Feb. 2017. DOI: 10.1017/S1062798716000594.

[10] F. Divina, A. Gilson, F. Gómez-Vela, M. Garcia Torres, and J. Torres, "Stacking ensemble learning for short-term electricity consumption forecasting," *Energies*, vol. 11, p. 949, Apr. 2018. DOI: 10.3390/en11040949.

[11] S. Kwiatkowski, *Entropy is a measure of uncertainty*, Oct. 2018. [Online]. Available: https://towardsdatascience.com/entropy-is-a-measure-of-uncertainty-e2c000301c2c.

[12] J. A. Miranda, M. F. Canabal, J. M. Lanza-Gutiérrez, M. P. García, and C. López-Ongil, "Toward fear detection using affect recognition," in *2019 XXXIV Conference on Design of Circuits and Integrated Systems (DCIS)*, 2019, pp. 1–4. DOI: 10.1109/DCIS201949030.2019.8959852.

[13] J. Nabi, *Recurrent neural networks (rnns)*, Jul. 2019. [Online]. Available: https://towardsdatascience.com/recurrent-neural-networks-rnns-3f06d7653a85.

[14] Phung and Rhee, "A high-accuracy model average ensemble of convolutional neural networks for classification of cloud image patches on small datasets," *Applied Sciences*, vol. 9, no. 21, p. 4500, Oct. 2019. DOI: 10.3390/app9214500. [Online]. Available: https://doi.org/10.3390/app9214500.

[15] A. U. Rehman, A. K. Malik, B. Raza, and W. Ali, "A hybrid cnn-lstm model for improving accuracy of movie reviews sentiment analysis," *Multimedia Tools and Applications*, vol. 78, pp. 26 597–26 613, 2019.

[16] P. Schmidt, A. Reiss, R. Dürichen, and K. V. Laerhoven, "Wearable-based affect recognition—a review," *Sensors*, vol. 19, no. 19, 2019, ISSN: 1424-8220. DOI: 10.3390/s19194079. [Online]. Available: https://www.mdpi.com/1424-8220/19/19/4079.

[17] S. Vani, T. V. M. Rao, and C. K. Naidu, "Comparative analysis on variants of neural networks: An experimental study," in *2019 5th International Conference on Advanced Computing &amp Communication Systems (ICACCS)*, IEEE, Mar. 2019. DOI: 10.1109/icaccs.2019.8728327. [Online]. Available: https://doi.org/10.1109/icaccs.2019.8728327.

[18] H. Ammar and A. Cherif, "Deeprod: A deep learning approach for real-time and online detection of a panic behavior in human crowds," *Mach. Vision Appl.*, vol. 32, no. 3, May 2021, ISSN: 0932-8092. DOI: 10.1007/s00138-021-01182-w. [Online]. Available: https://doi.org/10.1007/s00138-021-01182-w.

[19] M. Avci, Z. Li, Q. Fan, S. Huang, B. Bilgic, and Q. Tian, *Quantifying the uncertainty of neural networks using monte carlo dropout for deep learning based quantitative mri*, Dec. 2021.

[20] I. Joshi, R. Kothari, A. Utkarsh, *et al.*, "Explainable fingerprint roi segmentation using monte carlo dropout," Jan. 2021, pp. 60–69. DOI: 10.1109/WACVW52041.2021.00011.

[21] N. Li, L. Hu, Z.-L. Deng, T. Su, and J.-W. Liu, "Research on gru neural network satellite traffic prediction based on transfer learning," *Wireless Personal Communications*, vol. 118, pp. 1–13, May 2021. DOI: 10.1007/s11277-020-08045-z.

[22] L. Petrescu, C. Petrescu, A. Oprea, *et al.*, "Machine learning methods for fear classification based on physiological features," *Sensors (Basel, Switzerland)*, vol. 21, 2021. [Online]. Available: https://api.semanticscholar.org/CorpusID:235790916.

[23] I. Admin, *A comprehensive introduction to uncertainty in machine learning*, Sep. 2022. [Online]. Available: https://imerit.net/blog/a-comprehensive-introduction-to-uncertainty-in-machine-learning-all-una/.

[24] M. F. Islam, F. Bin Rahman, S. Zabeen, *et al.*, "Rnn variants vs transformer variants: Uncertainty in text classification with monte carlo dropout," *2022 25th International Conference on Computer and Information Technology (ICCIT)*, Dec. 2022. DOI: 10.1109/iccit57492.2022.10055922.

[25] R. Jaiswal and B. Singh, "A hybrid convolutional recurrent (cnn-gru) model for stock price prediction," in *2022 IEEE 11th international conference on communication systems and network technologies (CSNT)*, IEEE, 2022, pp. 299–304.

[26] I. Lazarou, A. L. Kesidis, G. Hloupis, and A. Tsatsaris, "Panic detection using machine learning and real-time biometric and spatiotemporal data," *ISPRS International Journal of Geo-Information*, vol. 11, no. 11, p. 552, 2022.

[27] M. Abdar, S. Salari, S. Qahremani, *et al.*, "Uncertaintyfusenet: Robust uncertainty-aware hierarchical feature fusion model with ensemble monte carlo dropout for covid-19 detection," *Information Fusion*, vol. 90, pp. 364–381, 2023, ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2022.09.023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253522001609.

[28] M. Ashraf, F. Abid, I. U. Din, *et al.*, "A hybrid cnn and rnn variant model for music classification," *Applied Sciences*, vol. 13, no. 3, 2023, ISSN: 2076-3417. DOI: 10.3390/app13031476. [Online]. Available: https://www.mdpi.com/2076-3417/13/3/1476.

[29] M. Islam, S. Zabeen, M. H. K. Mehedi, S. Iqbal, and A. A. Rasel, "Monte carlo dropout for uncertainty analysis and ecg trace image classification," in Jan. 2023, pp. 173–182, ISBN: 978-3-031-23027-1. DOI: 10.1007/978-3-031-23028-8_18.

[30] M. Islam, S. Zabeen, F. Rahman, *et al.*, "Exploring node classification uncertainty in graph neural networks," Jun. 2023. DOI: 10.1145/3564746.3587019.

[31] M. F. Islam, S. Zabeen, M. A. Islam, *et al.*, "How certain are tansformers in image classification: uncertainty analysis with Monte Carlo dropout," in *Fifteenth International Conference on Machine Vision (ICMV 2022)*, W. Osten, D. P. Nikolaev, and J. (Zhou, Eds., International Society for Optics and Photonics, vol. 12701, SPIE, 2023, 127010K. DOI: 10.1117/12.2679442. [Online]. Available: https://doi.org/10.1117/12.2679442.

[32] M. F. Islam, S. Zabeen, F. B. Rahman, *et al.*, "Unic-net: Uncertainty aware involution-convolution hybrid network for two-level disease identification," *SoutheastCon 2023*, Apr. 2023. DOI: 10.1109/southeastcon51012.2023.10115109.

[33] D. Kalita, *A brief overview of recurrent neural networks (rnn)*, Aug. 2023. [Online]. Available: https://www.analyticsvidhya.com/blog/2022/03/a-brief-overview-of-recurrent-neural-networks-rnn/.

[34] https://www.mind.org.uk/information-support/types-of-mental-health-problems/anxiety-and-panic-attacks/panic-attacks/, Accessed: 2024-1-3.

[35] *Long short-term memory neural networks - MATLAB & simulink*, en, https://www.mathworks.com/help/deeplearning/ug/long-short-term-memory-networks.html, Accessed: 2024-1-6.

[36] *Panic disorder*, en, https://www.nhs.uk/mental-health/conditions/panic-disorder/, Accessed: 2024-1-3.