A COMPARATIVE STUDY OF THE IMPACT OF DATA AUGMENTATION IN MACHINE LEARNING BASED CLASSIFICATION ACCURACY

by
Arif Jahangir
Master of Science in Physics, Quaid-e-Azam University, 1992
Bachelor of Science in Physics, Applied Mathematics, and Pure Mathematics, Punjab University, 1989

A thesis
presented to Ryerson University
in partial fulfillment of the
requirements for the degree of
Master of Science in the program of
Computer Science
Toronto, Ontario, Canada, 2019

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

Abstract

Arif Jahangir
Master of Science - Computer Science
Ryerson University, Toronto, Canada, 2019

A COMPARATIVE STUDY OF THE IMPACT OF DATA AUGMENTATION IN MACHINE LEARNING BASED CLASSIFICATION ACCURACY

Traumatic Brain Injury is the primary cause of death and disability all over the world. Monitoring the intracranial pressure (ICP) and classifying it for hypertension signals is of crucial importance. This thesis explores the possibility of a better classification of the ICP signal and detection of hypertensive signal prior to the actual occurrence of the hypertensive episodes.

This study differ from other approaches as time series is converted into images by Gramian angular field and Markov transition matrix and augmented with data. Due to unbalanced data, the effect of smote extended nearest neighbour algorithm for balancing the data is examined.

We use various machine learning algorithms to classify the ICP signals. The results obtained shoe that Ada boost performance is the best among compared algorithms. F1 score of the Ada boost is 0.95 on original dataset, and 0.9967 on balanced and augmented dataset. Quadratic Discriminant Analysis F1 score is 1 when data is augmented and balanced.

ACKNOWLEDGMENTS

I graciously thank my supervisor and mentor, Professor Alireza Sadeghian, for his encouragement, technical and personal insight and steadfast support throughout this work. His enthusiasm, vision, and experiences have been the most significant source and motivation to make possible the accomplishments herein.

I would also like to thank my supervisor for the financial support provided through the Big Data Research, Analytics, and Information Network (BRAIN) Alliance established by the Ontario Research Fund - Research Excellence Program (ORF-RE). Financial assistance provided by the School of Graduate Studies and Department of Computer Science are highly appreciated as well.

I would also like to acknowledge the collaboration with Dr. Michael Cusimano and his neuroscience research group (our research partner at St. Michaels' Hospital) that was established and facilitated through the Institute for Biomedical Engineering, Science and Technology (iBEST).

I would like to recognize my family for all the love and support they have provided over the years. My wife Soniiya and my children Rehan, Maha, and Saher.

Table of content

List of Figures

List of Tables

List of Acronyms

ABP	Arterial Blood Pressure
CHARIS	Cerebral Haemodynamic Auto-regulatory Information System Database
CNN	Convolutional Neural Network
CPP	Cerebral Perfusion Pressure
CPR	Cerebrovascular Pressure Reactivity
CSFPPW	Cerebrospinal Fluid Pulse Pressure Waveform
DTW	Dynamic Time Warping
ECG	Electrocardiogram
ENN	Extended Nearest Neighbour
GAF	Gramian Angular Field
ICP	Intracranial pressure
K-NN	K Nearest Neighbour
LDA	Linear Discriminant Analysis
MAE	Mean Absolute Error
MCNN	Multi-Scale Convolutional Neural Networks
MIMIC-II	Multi-parameter Intelligent Monitoring In Intensive Care II
MTF	Markov transition fields
MTF+GAFI	Markov transition fields and Gramian Angular Field images
PRx	Pressure-Reactivity Index
QDA	Quadratic Discriminant Analysis
RAP	Pressure Coefficient
RBF	Radial Basis Function
SVM	Support Vector Machine
SICU	Surgical Intensive Care Units
TBI	Traumatic Brain Injury

List of Appendices

Chapter I - Introduction

1.1 Problem Statement

Traumatic Brain Injury (TBI) is the primary cause of death and disability all over the world among children, youth and adult. Around 1.5 million Canadians live with the consequences of an acquired brain injury. The yearly occurrence of TBI is more significant than that of Breast Cancer, HIV/AIDS, Spinal Cord Injury, and Multiple Sclerosis combined [1].

Stabilizing a patient after traumatic brain injury is crucial for saving a patient from dead and further deterioration of brain tissues. Monitoring the intracranial pressure (ICP) and classifying it for hypertension signals in ICP becomes of crucial importance in this regard. This thesis will explore the possibility of better classification of the ICP signal and detection the pattern of hypertensive ICP signal prior to the actual occurrence of the hypertensive episodes [2].

1.2 Motivation

Traumatic brain injury (TBI) causes human to deviate from their normality. The first step from preventing this is to save lives and stop further deviation. Monitoring and classifying ICP becomes of utmost importance in this regard [1][2].

Different algorithms have different inner workings. The motivation of this thesis is to study whether an inner algorithm working is sensitive to data representation or not. This study is conducted to explore this. Information contained in different data representations is the same, but different algorithms process data representation differently. Some algorithms may find it easy to classify the data if data is presented in a certain manner while other algorithms may find it challenging to classify if data is presented in that same manner.

The approach of this thesis is to present data in its original form to different algorithms and compares its results to the results if the data is augmented with different representation. The machine learning algorithms selected for this study are Nearest Neighbors, Naïve Bayes, Gaussian Process, Linear Support Vector Machine - Classification (SVM), RBF Support Vector Machine - Classification (SVM), Quadratic Discriminant Analysis (QDA), Decision Trees, Ada Boost, and Multi-Scale Convolutional Neural Networks (MCNN). Data is represented in its original time series form as well as in Gramian Angular Plots representation and Markov Transition Fields representation. Gramian Angular Plots representation and Markov Transition Fields representation has been used before; this study explores these representations effects on above-mentioned algorithm's inner workings. Additionally, this study is conducted in the context of ICP signals.

Importance of this study is to explore whether data representation aids in classification or not particularly on ICP signals. Another motivation of this thesis is to study the effects of balancing the dataset with SMOTE ENN algorithm (section 4.3) and comparing how classifying abilities of above-mentioned algorithms changes for both original dataset and augmented dataset with different data representation.

1.3 Objectives and Methodology

The objective is to compare above-mentioned algorithms for four datasets listed below. To compare the results of various above-mentioned algorithms, four datasets are passed through various algorithms. Finally, comparison and analysis of all the results are investigated as follows:

- Dataset is taken from MIMIC-II and CHARIS database.
- Dataset is cleaned, and hypertensive signals are identified.
- Hypertensive signal's past six minutes segments are used as positive examples.
- Rest of the signal are identified as negative examples.
- Following datasets are constructed.
    - Data without SMOTE ENN algorithm and without augmenting it with Gramian angular field images and Markov transition matrix.
    - Data with SMOTE ENN algorithm but without augmenting it with Gramian angular field images and Markov transition matrix.
    - Data without SMOTE ENN algorithm but augmenting it with Gramian angular field images and Markov transition matrix.
    - Data with SMOTE ENN algorithm and augmenting it with Gramian angular field images and Markov transition matrix.
- Four datasets are then passed through the following algorithms.
    - Nearest Neighbors
    - Naïve Bayes
    - Gaussian Process

    - Linear Support Vector Machine - Classification
    - RBF Support Vector Machine - Classification
    - Quadratic Discriminant Analysis
    - Decision Trees
    - Ada Boost
    - Multi-Scale Convolutional Neural Networks
- Results of the above algorithms are compared and investigated.

MCNN is selected because it showed promising results in classifying various time-series datasets [1]. MCNN is a multichannel convolutional neural network it is well suited to handle Gramian Angular Plots images and Markov Transition Fields images as it is a convolutional neural network. Gramian Angular Plots images and Markov Transition Fields images are different representations of our original dataset. Figure 1.1 presents the schema of the thesis.

1.4 Contribution of this thesis

As illustrated above, ICP signals have been studied with many different methodologies. This thesis contribution is to classify the hypertensive signal before the hypertensive state occurs by passing the original data and data augmented with different data representation to various classification algorithms mentioned above. The effect of balancing of data with Smote ENN is also studied. Finally, the best algorithm among Nearest Neighbors, Naïve Bayes, Gaussian Process, Linear Support Vector Machine - Classification, RBF Support Vector Machine - Classification, Quadratic Discriminant Analysis, Decision Trees, Ada Boost, and Multi-Scale Convolutional Neural Networks on the given dataset is evaluated.

Though the neural network is applied before on ICP signal. ICP signals were treated as time series. The contribution of this thesis is converting time series into images by using Markov transition fields and Gramian angular field and using MCNN and other algorithms to classify hypertensive and non-hypertensive signals. The Gramian angular field is essential in the analysis of ICP signal because once we have transformed the rescaled time series into polar coordinates, we can easily exploit the correlations within different time intervals. Markov transition field is vital in this regard because it is similar to a framework for encoding dynamical transition statistics; the concept is extended by representing the Markov transition probabilities serially, to preserve statistics in the time domain.

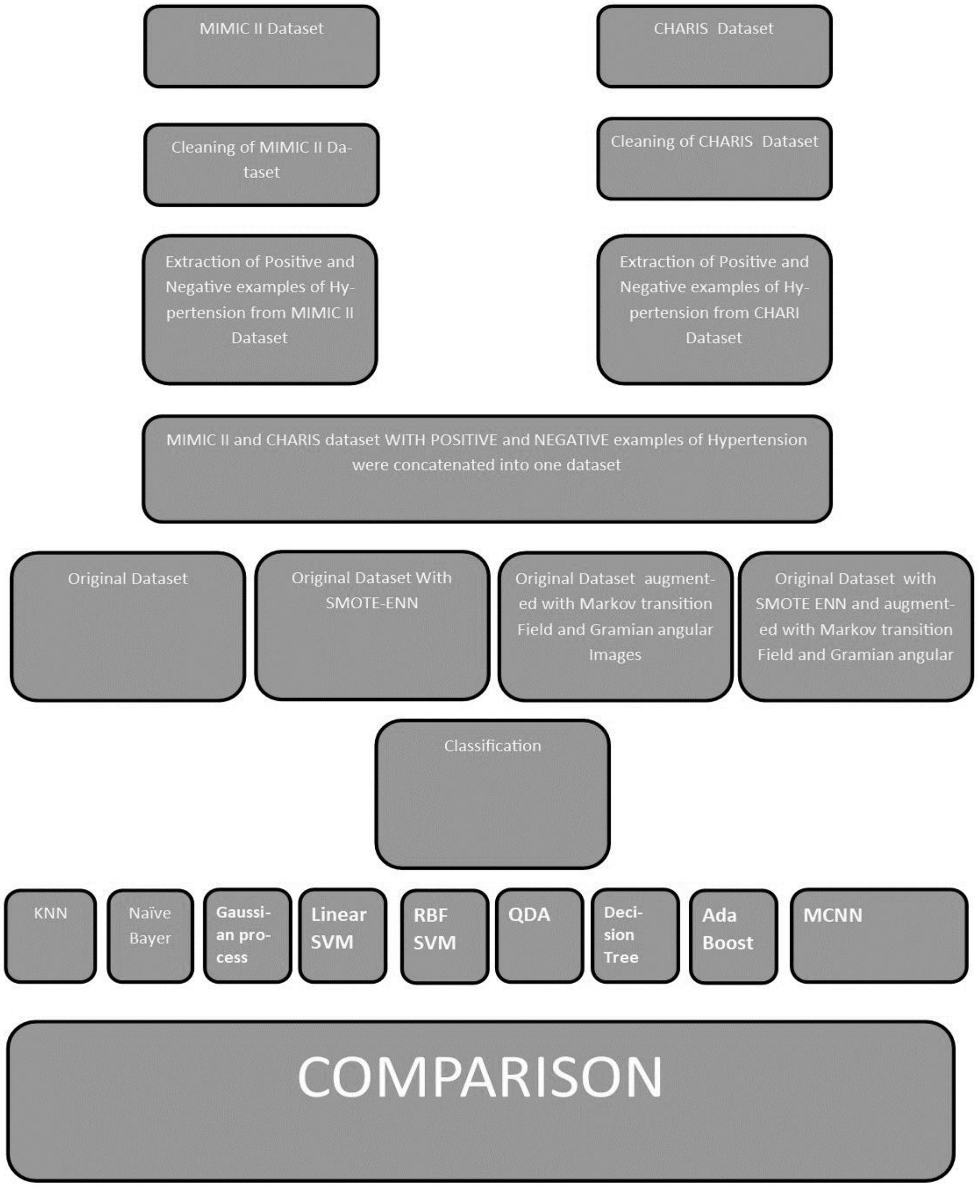

MIMIC II Dataset
CHARIS Dataset
Cleaning of MIMIC II Dataset
Cleaning of CHARIS Dataset
Extraction of Positive and Negative examples of Hypertension from MIMIC II Dataset
Extraction of Positive and Negative examples of Hypertension from CHARI Dataset
MIMIC II and CHARIS dataset WITH POSITIVE and NEGATIVE examples of Hypertension were concatenated into one dataset
Original Dataset
Original Dataset With SMOTE-ENN
Original Dataset augmented with Markov transition Field and Gramian angular Images
Original Dataset with SMOTE ENN and augmented with Markov transition Field and Gramian angular
Classification
KNN
Naïve Bayer
Gaussian process
Linear SVM
RBF SVM
QDA
Decision Tree
Ada Boost
MCNN
COMPARISON

Figure 1.1. The Schema of the Thesis

1.5 Structure of the Thesis

In chapter 2, the background information related to intracranial pressure is presented. Chapter 3 explains Gramian angular plots, Markov transition field and multi-scale convolutional neural network. In chapter 4, the steps taken in order to clean and pre-process the datasets are illustrated along with the theoretical underpinning of algorithms. In chapter 5, the results of our investigation are presented. Chapter 6 provides a discussion about the results and the connection with the theoretical structure of the algorithms. Finally, in chapter 7, the conclusion and future directions are presented.

Chapter II - Background

Intracranial pressure is referred to as the pressure inside the skull due to brain tissues and cerebrospinal fluid. This pressure is increased due to brain/head injury, bleeding in the brain, brain tumour, swelling of the brain, excessive cerebrospinal fluid aneurysm or infections such as encephalitis or meningitis. With the increase of intracranial pressure, an auto-regulatory process exerts to produce equilibrium initially. If the volume is expanded beyond a certain point due to the reason mentioned above, the auto-regulatory process breaks down, and ICP rises. The normal range of ICP values is from 0-25 mm Hg. In children, the range is between 3 and 7 mm Hg, and in infants, it is 1.5 to 6.0 mm Hg beyond these ranges ICP is considerate to be abnormal [1][2].

The increase in ICP can be divided into three stages. In stage 1, the autoregulatory mechanism pulls processes towards equilibrium. Stage 2 represents intracranial hypertension. Key characteristics of this stage include less neuronal oxygenation, and arteriolar vasoconstriction to increase Cerebral perfusion pressure CPP. CPP is the remaining pressure gradient producing cerebral blood flow to the brain. It must be upheld within thin limits since tiny pressure could cause brain tissue to develop ischemic (having insufficient blood flow), and excessive blood flow could raise intracranial pressure (ICP). Stage 3 is characterized when small changes in volume correspond to substantial changes in ICP. In this stage, it becomes increasingly difficult to squeeze blood into internarial space, which leads to ischemia and brain infarction. The further increase can eventually be fatal [1][2].

Analysis of ICP signals for these stages is essential because they can be used to take timely appropriate actions to remedy the problem. In effect, a predictive model -- that can forecast the occurrence of different phases based on the past history of ICP -- will be beneficial from diagnostic and treatment point of view. ICP signals have already been analyzed with various mathematical techniques [1] [2] [5] [6] [7] [8] [9] [10] [11] [12]. In time-domain approaches, typically a number of windows are created, and their mean, standard deviation and variance are calculated accordingly, and then sing these parameters, regularities and irregularities cand classified. In the frequency domain, different harmonics are ranked for the same. ICP signals have also been subjected to wavelets analysis. Other methods of ICP signal analysis include approximate entropy analysis and fractal analysis [27][28][29].

2.1 Background of Intracranial Pressure

The main aim of understanding background information about ICP is to investigate the present limitations and capabilities of past and current methods. Also, to observe and understand the ICP time series and extract their inherent features around the homeostasis state perturbations. The variations within physiological system signals are analyzed and captured through nonlinear dynamics analysis by exploring various mathematical models and methods. This is done to ascertain and investigate the reliability of different approaches in order to encode ICP information in time and frequency domain. Another aim of presenting previous and current approaches is to further translational research to explore whether these techniques can provide qualitative and quantitative data that can be encoded into a small number of parameters or signals that can be interpreted easily by clinicians.

The brain has an auto-regulatory mechanism to deal with a matter that accumulates abnormally. It shifts cerebrospinal fluid volume along with cerebral blood volume to the outside of the brain. This mechanism has a limit. When matter accumulates beyond this threshold system disruption happens and break down is the result. When equilibrium vanishes, any further swelling shows itself as an upsurge in ICP. It cannot be ascertained whether the upsurge is a primary or secondary cause of the pathological mechanism. Nevertheless, the rise in ICP is associated with cerebral ischemia, neurological deficit, and death. Majority of neurological intensive care exertions are designed to treat intracranial hypertension [3] in order to boost patient health [4]-[6].

Pressure transducers are put in the intracranial spaces, particularly subdural, intraparenchymal, epidural, and lateral ventricular regions. The bedside monitor is used to display the values from pressure sensors. Table 2.1.1 shows the normal variations of the ICP signal with respect to age.

Table 2.1.1. Age vs Normal range of ICP signal

Age	The normal range of ICP signal
Infants	1.5 to 6.0 mm Hg
Young Child	3 to 7 mm Hg
Adults	0 to 15 mm Hg

According to The Brain Trauma Foundation 2007, 20 to 25 mm Hg should be considered as the upper limit beyond which management of ICP should be introduced [1][2].

The Valsalva maneuver, sneezing, coughing, and specific forced body movements against resistance cause changes in ICP values. These changes need to be filtered out to obtain the correct value of the ICP signal. Physicians primarily measure ICP signals in patients with the comatose condition to see deviations from normal related to various kind of brain damages particularly stroke, trauma and deterioration of neurological health which are difficult to observe by the methods of clinical checks.

Investigators primarily in the field of physics and engineering have studied various mathematical models, techniques, and approaches to abstract prognostic and diagnostic parameters which are discussed in subsequent sections of this chapter. These approaches are limited to research applications. Recently ICP has been included in the list of physiological signals to be studied by deep learning and traditional machine learning techniques.

2.2 Constituents of ICP Signal

ICP waveforms are a superposition of three waveforms. The pulse waveform, the respiratory waveform, and the slow waveform. These waveforms are considered as harmonics components. The pulse waveform is due to the cardiac cycle and contain:

a. Arterial pressure waves which consist of the percussion peak, the tidal peak, and the dichotic peak.
b. Venous pressure waves

Peak amplitudes may be considered to be connected to variations in systemic arterial pressure, brain tissue compliance, or the closure of the aortic valve. Slow waves are an important indicator as it is shown that they are related to the fatal consequence in patients with traumatic brain injury [7]. The frequency range of slow waves is between 0.05 to 0.0055 Hz [8].

2.2.1 What causes ICP waveform

The simultaneously interacting physiological process causes the emergence of ICP waveforms as time-series data signal. The shape of the waveform is studied in the following two ways.

1. Averaging values over time
2. Harmonics detailed analysis [9]

Many mathematical techniques, methodology and approaches can be applied to find the rate of occurrences and frequencies of key pattern within the signal. These patterns can then be used for predicting and anticipating adverse events. If such patterns can be found, then this pattern can guide clinicians to desired therapies.

O'Phelan et. al. [10] reached a conclusion that ICP display characteristic patterns in the time domain. They also suggested the requirement of finding ICP patterns and attaching a predictive value on its manifestation in time. Lundberg classified the waveform[2]. Lundberg classification consists of classifying a series of ICP wave into A, B, C waveforms:

- ***A waveform:*** is characterized by the sudden surge in ICP signal and maintenance of ICP signal over 50 mm Hg for a duration of 5 to 20 minutes [2]. They indicate diminished brain compliance [11] and are due to inadequate cerebral perfusion pressure and cerebral blood flow.
- ***B waves form:*** are slow waves consisting of a range of amplitude between 10 to 20 mm Hg. They are allied with respiratory cycles.
- ***C waves form:*** characterized as resulting from the interaction of cardiac and respiratory waves signals. They are characterized as consisting of a frequency of 5 to 9 oscillation per 60 seconds [2].

2.2.2 ICP waveform superimposed on the respiratory variation waves

It has been indicated that superimposition of respiratory variation waves on ICP signal in traumatic brain injury patients gives insights into brain compliance. It has been shown that the waveforms for inspiration and expiration are different in traumatic brain injury patients.

2.2.3 Mean ICP values

In the traditional monitor, the mechanism of displaying ICP signal consists of averaging over ICP signal measured in divisions of millimetres of mercury that is in mm Hg during the small duration of time. Standard deviation, variances and mean are computed for these durations [12]. After surgery and traumatic brain injury, these simple statistics usually show dangerous trends.

Therefore it becomes imperative to control and reduce these simple ICP statistics values in patients with elevated ICP [13].

The problem with mean ICP values is that they lack the dynamic features of the component signals [14]. Environmental variations and methods of measurements also affect these mean values. To overcome these difficulties [15], it is proposed that research should focus on identification of pulse wave inside the ICP signal to calculate the average amplitude of the wave. It is suggested that it will probably give a more conclusive and accurate description of intracranial compliance and will have more predictive power [14].

2.3 Frequency domain

In the frequency domain of ICP signal analysis, we are mainly concerned with harmonics and its interpretation [8]. Fast Fourier transform is used to identify and classify the harmonics of the signal. Historically fast Fourier transform is used in the analysis of ICP waveform. This methodology is shown to have a more reliable analysis as compared to ICP averaged over time [12].

Wavelet analysis is also used to find wavelet components in the ICP waveforms. Wavelet analysis gives us a result both in time and frequency domain. Wavelets are convolute over the entire signal. This methodology encompasses features that are nonstationary [12].

2.4 Limitations of morphological analysis of ICP

In the morphological analysis of waveform, some technical factors introduce errors. Researchers find great difficulty in isolating discreet ICP signal and peaks inside waveforms which are usually irregular. There is a superposition of reference ICP wave and pulse wave. This morphs the overall wave into a wave which has more rounded peaks [16].

Morphological analysis techniques are unable to differentiate the signal distortions caused by postural variations, the interaction between the adjacent intracranial tissue and the pressure transducer, patient motion, the monitor’s drift, disturbance of hardware connections, and volume changes. This upsurges the noise to signal ratio and analysis of true signal becomes difficult [17].

Present-day research in the morphological analysis of ICP waveform concentrates in enhancing the signal to noise ratio and more accurately identifying the peaks in the signal. Peak formations are the foundation of analysis, and therefore, they are vital to identify these distinctly [18]-[19]. Identification of peak signal gives insight into changes in the form of sub-peaks that are very helpful both in time and frequency domain. Identification of the shape and periodicity of the waveform correctly forms the foundation of analysis and interpretation of ICP signals.

2.5 ICP Waveform-Derived Indexes

Secondary indexes that may be helpful in identification of the onset perturbation in ICP waveform has been investigated [20]. Analysis of intracranial ICP waveform decompensation and its elastance capacity has been explored [20]. Researchers have suggested exploiting the standard deviations regression plot and slope of mean ICP [20]. They have also proposed to exploit the intersection of ICP pulse amplitude and slope of baseline ICP waveform [20]. Researchers believe that these may indicate the perturbations in the underline mechanisms that cause perturbations in the ICP waveform.

Derivation of secondary indices uses mean wave amplitude and mean values. It is suggested that the pressure-volume graph may be drawn by pressure coefficient (RAP) and regression of amplitude. It is also proposed that this can be considered as a correlational measure between the amplitude of the pulse waveform and ICP signal. This can be considered as an indicator of the cerebrospinal compensatory reserve [8].

Cerebrovascular pressure reactivity (CPR), which indicates the ability of smooth muscle tone within the confined of cerebral arteries to correspond with the perturbations in transmural pressure may also be an indicator of neurological compensation [21]. Czosnyka and Pickard [8] suggested the pressure-reactivity index (PRx) indicate the correlation coefficient of mean ICP and arterial blood pressure (ABP). (PRx) is used as a predictive parameter of deteriorated outcome after traumatic brain injury [20].

2.6 Neural Networks

Presently neural networks and deep learning are being used to analyze and classify ICP waveforms. It has been found that neural networks and deep learning approaches are more dependable techniques as far as forecasting of future mean values of ICP signals is concerned. Neural networks can dig into tiny time frames of ICP waveform, which are extracted from a complete time series. For example, Deep Learning algorithms have been used to model the association between waveform morphology and hypertension in [22]. In another study [23], deep belief networks (DBNs) were demonstrated to be adept at learning generative and discriminative characteristics from the models; Arguments has been provided in favour of the suitability of these approaches to classify ICP waveform with various features.

In [24], a neural network based approach was used to train to classify 60 cerebrospinal fluid pulse pressure waveform (CSFPPW) in four dissimilar classes conferring to their morphology. The efficacy of this method was compared to a proficient inspector's classification. The morphology of CSFPPW was logged in 60 patients. Results showed the general concordance in CSFPPW classification between an artificial neural network, and the expert examiner was 88.3 %.

In another approach [25], the breakdown of an ICP signal into clinically relevant dimensions, to permit the identification of important ICP waveform, was studied. It. The authors divided their analysis into three stages. They started with multi-resolution convolution analysis for the main signal decomposition. Afterwards, they created an impulse function with multiple factors that can characterize any form in the signal under investigation. Lastly, they used localized optimization technique to discover morphologies of concern in the decomposed ICP signals. It was shown that the method works with performance receiver operator characteristic area below the curve figures for each of the waveform categories, *B* wave, A wave, and C wave's low, and high compliance states of 0.694, 0.936, 0.698, and 0.676 were found respectively.

In another paper [26], ICP monitoring by means of texture features have been suggested. A mixture of image processing approaches and a decision tree algorithm is used to assess ICP of traumatic brain injury patients non-invasively. Furthermore, a visual analytics tool is applied to conduct an outlier detection and interactive visual feature analysis.

2.7 ICP and Entropy

Entropy approaches are different from other approaches in the sense that they try to find regularity in the signals while other approaches try to find an irregularity in the signal [27]. Approximate entropy is defined as "the negative normal logarithm of the conditional probability that a dataset of extent *N* will replicate itself again." It examines a set of data series for emerging new patterns and the frequency of their reappearance within the waveform. The ability of approximate entropy to generate waveform randomness and to form a pattern is exploited [28]. Estimated entropy, along with Lempel–Ziv (LZ) compression entropy quantity [28] is used to measure drastic perturbations in the ICP waveform in young patients suffering from austere traumatic brain injury. This is used to explore the correlation with the mean ICP waveform.

The LZ compression entropy quantity is well-defined as “a nonparametric measure of complexity for one-dimensional signal connected to the number of separate substrings and the rate of their occurrence along with a given series.” The LZ recognizes new emerging patterns and replaces the series with a reduced reference in order to compress the whole series into reduced sets as this compression can reproduce the original signal from its reduced form, and it is considered as lossless. The LZ can identify and differentiate the regularities in a time series. Researchers conclude that the complexity of ICP waveform decreases during intracranial hypertension period. These periods are labelled as “plateau wave” and are indicative of persistent elevations of ICP waveform. The rise of pattern’s regularity in durations of intracranial hypertension indicate secondary brain injury.

2.8 Non-Invasive method for ICP

In one non-invasive based approach for ICP [30], it was concluded that Blood Pressure, Mean Arterial, Respiration, Diastolic Arterial, Heart Rate, ECG ST-segment levels, and Pulse are strongly correlated with ICP and have the potential of predicting intracranial hypertension. Elevated ICP periods and related physiological signals were then extracted from given datasets. Based on ICP levels, “Severe Intracranial Hypertension,” “Intracranial Hypertension,” and “Intracranial hypertension onset” events were identified. It was suggested that physiological signals in TBI patients have the predictive power to predict undesirable events in the ICP waveform.

In a study conducted on 31 traumatic brain injury patients [31], the principal component analysis was applied to extract non-correlated feature selection. They concluded that two components could be extracted from continuously gathered physiological signals of traumatic brain injury patients. The result was examined by utilizing parallel analysis, Scree test, and Kaiser’s Criterion. The power of prediction of these two components was confirmed as it achieved an error of 0.025 on mean absolute error (MAE) score.

Another study [32] was done on 20 patients with TBI. K-means clustering investigation centred on a wavelet was applied to identify configurations of physiological signals. They investigated variations in these configurations. Researchers concluded that this unsupervised method could be a possible technique to identify patterns in the ICP waveform.

2.9 Main advantage of the proposed approach for classification of ICP signal

As discussed above, ICP signals are studied in both the time domain and the frequency domain. They have also been investigated using wavelet and fractal analysis. Investigators have applied ICP waveform-derived indexes to get insights into ICP signal. The exploitation of irregularities by means of entropy exploration has also been studied. Researchers have also considered non-invasive methods and neural networks.

The main advantage of the this thesis approach is in converting ICP signal into images, particularly Gramian angular plots and Markov transition field images (section 3.1 and 3.2) . The Gramian angular field is significant in the study of ICP signal for the reason that once we have transformed the time series into polar coordinates, we can straightforwardly exploit the correlations inside different time intervals. Markov transition field is central in this regard because it is comparable to a framework for encoding dynamical transition statistics; the concept is drawn-out by representing the Markov transition probabilities serially to preserve statistics in the time domain.

By using different data representation (images) of ICP signal, classification accuracy, precision, recall and F1 score are investigated on the various machine learning algorithms such as Nearest Neighbors, Naïve Bayes, Gaussian Process, Linear Support Vector Machine - Classification (SVM), RBF Support Vector Machine - Classification (SVM), Quadratic Discriminant Analysis (QDA), Decision Trees, Ada Boost, and Multi-Scale Convolutional Neural Networks (MCNN)

Chapter III - Gramian Angular Plots, Markov Transition Field and Multi-Scale Convolutional Neural Network (MCNN)

3.1 Gramian Angular Field

The Gramian angular field is contracted in the following way [4]. Assume we have a time series $X = \{x_1, x_2, ..., x_n\}$ of n real-valued observations, To make all values fall in the interval [-1,1] we rescale the values by:

$$x_i^{\sim} = \frac{(x_i - \max(X) + (x_i - \min(X))}{\max(X) - \min(X)}$$

(3.1.1)

In polar coordinates, we can characterize the rescaled time series X by taking a timestamp as the radius and by converting the $x_i^{\sim}$ as the angular cosine with the equation below [4]

$$\varphi = \arccos(x_i^{\sim}), -1 \leq x_i^{\sim} \leq 1, x_i^{\sim} \in X^{\sim}$$

$$r = \frac{t_i}{N}, t_i \in N$$

(3.1.2)

In the above equation, N and time stamp t_i are a constant element to normalize the span of the polar. φ is the polar angle and r is the radius of polar coordinates.

The polar coordinate grounded illustration is a new method to comprehend time series. As time upsurges, corresponding values twist among various angular points on the spanning circles, like water rippling. The coding map has two essential properties.

First, due to monotonicity of cos() function in the interval [0, p_i], it is bijective. This transformation produces only one map in polar coordinates with a distinctive inverse function.

Second, in the Cartesian coordinate system area does not depend on the absolute value of time. It only depends on the time interval. While in polar coordinates it depends on both. In Cartesian coordinates, the area $S_{i,j}$ in i, j dimensions are defined by.

$$S_{i,j} = \int_{x(i)}^{x(j)} f\big(x(t)\big)dx(t)\ .$$

(3.1.3)

In the Cartesian coordinate system, the area is the same

$$S_{i,i+k} = S_{j,j+k}$$

(3.1.4)

In the above equation *f*(*x*(*t*)) has the identical values on $[i,\ i\ +\ k]$ and [*j*, *j* + *k*]. While in polar coordinates, the area $S'_{i,j}$ is defined by way of

$$S'_{i,j} = \int_{\emptyset(i)}^{\emptyset(j)} r[\emptyset(t)]^2 d(\emptyset(t))$$

(3.1.5)

where $\emptyset(i) and\ \emptyset(j)$ are two angles, and we are interested in the area between the two. *r* is the radius then

$$S_{i,i+k} = S_{j,j+k}$$

(3.1.6)

the matching area from time stamp *i* to time stamp *j* is not only reliant on the time duration |*i j*|, but also specified through the absolute value of *j* and *i*.

Once we have transformed the rescaled time series into polar coordinates, we can easily exploit the correlations within different time intervals. By taking the trigonometric sum between each point in time, we can calculate the correlation inside different time duration. The Gramian angular plot G is contracted in the following way.

$$G = \begin{bmatrix} \cos(\emptyset_1 + \emptyset_1) & \cdots & \cos(\emptyset_1 + \emptyset_n) \\ \vdots & \ddots & \vdots \\ \cos(\emptyset_n + \emptyset_1) & \cdots & \cos(\emptyset_n + \emptyset_n) \end{bmatrix}$$

(3.1.7)

The Gramian Angular Field has numerous benefits.

- It delivers a method to preserve the temporal dependence, as time rises as per the location moves from top-left to bottom-right.
- The Gramian Angular Field encompasses temporal correlations
- The chief diagonal Gi,i is the singular situation when $k = 0$, which comprises the original value/angular info. With the central diagonal, we will almost rebuild the time series data from the high-level features learned by the deep neural network.
- The Gramian Angular Field is significant since the scope of the Gramian matrix is n ⇥ n when the distance of the unprocessed time series is n. To decrease the dimension of the Gramian Angular Field, we apply Piecewise Aggregation Approximation [36] to even the time series and while preserving the trends.

3.2 Markov Transition Field

Markov Transition Field is similar to a framework of [37] for encoding dynamical transition statistics; The concept is extended by representing the Markov transition probabilities serially, to preserve statistics in the time domain [4].

If we have a time series X, we construct its Q quantile bins and allocate individually x_i to the conforming bins q_j ($j\ \mathcal{E}\ [1, Q]$). In this manner, we build a Q by Q weighted adjacency matrix W by calculating transitions midst quantile bins in the method of a first-order Markov chain along the time axis. Elements of matrix W are denoted by $w_{i,j}$ is specified by the frequency through which a point in the quantile qi trails a point in the quantile q_j.

After this we normalize $w_{ij} = 1$, W is the Markov transition matrix. It is oblivious to the spreading of X and temporal dependence on time steps which are denoted by t_i. There is a lot of information loss in getting rid of temporal dependence in matrix W. To overcome this disadvantage, Markov Transition Field (MTF) matrix M is defined as follows [4]:

$$M = \begin{bmatrix} w_{ij|x_1 \in q_i, x_1 \in q_j} & \cdots & w_{ij|x_1 \in q_i, x_n \in q_j} \\ \vdots & \ddots & \vdots \\ w_{ij|x_n \in q_i, x_1 \in q_j} & \cdots & w_{ij|x_n \in q_i, x_n \in q_j} \end{bmatrix} \tag{3.2.1}$$

We construct a $Q \rightarrowtail Q$ Markov transition matrix by separating the data into Q quantile bins. The quantile bins that encompass the data at time stamp i and j (temporal axis) are qi and q_j ($q \mathcal{E} [1, Q]$). M_{ij} in MTF signifies the transition probability of q_i -> q_j . By considering the temporal positions, MTF matrix is built, which contain the transition probabilities at time locations.

By conveying the probability commencing from the quantile at time footstep i to the quantile at time footstep j at each pixel Mij , the matrix M essentially captures the multi-span transition probabilities of the time series

Markov transition matrix has the following properties.

- $Mi_{,j||ij|=k}$ signifies the transition probability amid the points with time interval k. For instance, $M_{ij|ji=1}$ demonstrates the transition progression along the time axis with a skip footstep.
- The central diagonal M_{ii}, which is a distinct situation when $k = 0$ engulfs the probability from each quantile to itself
- Blurring kernel is applied to create the image extent controllable and computation more effectual, Aggregation of the transition probabilities is done in every subsequence of length m.

3.3 Time Series classification and why Multi-scale Convolutional Neural Network

ICP signal is a time series. Researchers have studied and tried to solve the problem of forecasting time series' class labels for nearly a couple of decades within the data mining and machine learning community. There has been some significant advancement in the field and applications were found in clinical prediction and biomedical field. These advancements still fall short of giving good accuracy and efficiency. In the past approaches mostly involved pulling out discriminative features within a time series by utilizing dynamic time warping (DTW) or wavelet transformations. These discriminative features were then fed to the classifier. In these methods, feature extraction and classification are two separate parts, which limits the accuracy of these methods. Features also exist at different time scales, and this aspect is ignored in these methods.

There are two categories in which most time series classification approaches fall, distance-based methods and feature-based methods. In the distance-based method, a similarity metric is evaluated. Once we have the similarity metric, we can classify the time series with k- nearest neighbours or support vector machine. Most notably, the similarity metric used is dynamic time warping (DTW). In DTW, two time series are aligned with dynamic warping to find the best fit. Dynamic programming is used to do this.

In feature-based methods, feature vectors are formulated, and then these feature vectors are fed into classification algorithms to generate classification. In clinical prediction time series is divided into windows and features are extracted from these windows. Features can be as simple as mean and standard deviation as well as sophisticated features derived from detrended fluctuation analysis and spectral analysis.

In another approach to feature, extraction wavelets are used. In these methods, shapelets are defined and considered as a signature subsequence. These shapelets are used in various ways; for example, they are viewed as a dictionary with each shapelet as a word and time series is regarded as a bag of words model. A more recent study [35] builds the feature vector by the minimum distance between the shapelet and time series containing the shapelet.

A disadvantage of the shapelet method is that we have to search an ample space for discriminative shapelets. To overcome this difficulty, Grabocka et al. [38] propose to jointly explore a collection of shapelets simultaneously along with the classifier. The drawback of their method is that it can only separate linear planes.

The convolutional neural network has shown promising results in the field of object recognition [39], face detection [40], audio and speech analysis [41] and natural language processing. A key reason for the success of the convolutional network is its ability to learn complex features by its convolutional layers. It is natural to ask this question can convolutional neural networks be applied to time series where CNN learns complex feature representations by itself and classify the time series.

Cui et al [3] proposed a novel method of an end to end neural network model. Multi-Scale Convolutional Neural Network (MCNN) integrates classification part and feature extraction part into one scheme. They created a multi-branch layer and a convolutional layer that can be learned. In the portion of the algorithm that runs the multi-branch layer, different scales and frequencies are processed, and features are extracted, which shows superior features representations. We propose to extend this framework by adding more branches to Multi-Scale Convolutional Neural

Network (MCNN). We intend to add Gramian angular plots and Markov transition fields as two new branches. Gramian angular plots and Markov transition fields all are images. The convolutional neural network is suited to extract features from these images which are fed to the full convolutional network.

The intuition behind this thesis is if we feed many more perspectives to full convolutional neural network, it become easier for the network to classify the time series.

3.4 Multi-scale Convolutional Neural Network (MCNN)

Multi-scale Convolutional Neural Network (MCNN), a convolutional neural network precisely planned for classifying time series. A characteristic feature of MCNN is that its first layer encompasses several branches that perform various transformations of the time series, covering those in the frequency and time domains, excavating features of various types and time scales. After that dot product is applied to the transformation layer and 1-D learnable filter in the convolutional layer. This is the typical method to acquire features in the convolutional neural network. The result of feature knowledge in the branches is then concatenated and fed to the full convolutional neural network. Figure 3.4 is the depiction of the overall architecture of MCNN by Cui and el [3].

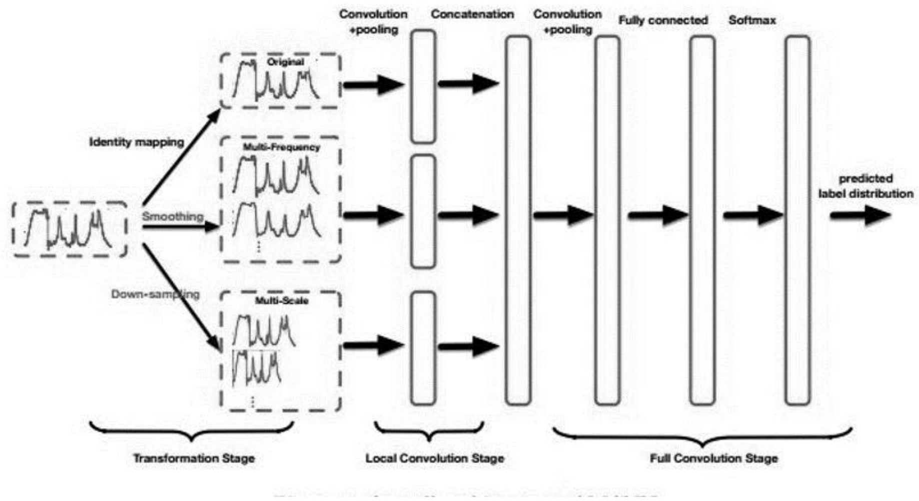


Figure 3.4. Overall Architecture of MCNN Zhicheng Cui and al [3]

MCNN is one of our algorithm that is compared with other algorithms. There are three stages in MCNN.

1) **Transformation Stage**

   In the transformation stage, the signal is passed in three ways. First, the signal is given as it is and is called the identity transformation. Secondly, the signal is downsized to capture temporal patterns at different time scale. Overall all trends are reflected in long-term features and while delicate features are present in short-term signals. Both are necessary for classifying a time series. Transformation stage has three branches.

   a) Identity Branch

      In Identity branch, the signal is passed as it is.

   b) Down-sampling Branch

      If we have a time series

$$T = \{t_1, t_2, t_3, t_4, \dots \dots \dots ., t_n\} \tag{3.4.1}$$

   And if k is the down-sampling rate, then we select only the kth and its multiple elements of the series to form a new series.

$$T^k = \{t_{1+k*i}\}, \qquad i = 0,1,2, \dots \dots \dots ., [\frac{n-1}{k}] \tag{3.4.2}$$

   c) Multi-frequency branch

      As noises correspondence to high-frequency signals, a varying degree of smoothness to ICP signal in this branch is applied. Variation of time series signal can be reduced in this manner. In this branch, various moving averages are applied to obtain various degree of smoothness.

      If we have a time series T

$$T = \{t_1, t_2, t_3, t_4, \dots \dots \dots ., t_n\} \tag{3.4.3}$$

We obtain new series T^l by taking moving average according to the following formula, where l is the window size of the moving average.

$$T^l = \frac{t_i + t_{i+1} + t_{i+1} + \cdots \ldots \ldots \ldots .. + t_{i+l-1}}{l}$$

(3.4.4)

2) **In the local convolution stage**, numerous convolutional layers are used to excerpt the features for each branch. In this stage, the convolutions are independent for different branches from each other. Max pooling technique with multiple sizes is applied to the output of convolutional layers.

3) **In the full convolution stage**, all extracted features are concatenated, and several more convolutional layers (each followed by max-pooling) are applied, after this fully connected layers, and a softmax layer to produce the concluding output. In this end-to-end system, parameters are trained via backpropagation.

3.5 Architecture of the MCNN algorithm augmented by Gramian angular plot branch and Markov transition field branch

Figure 3.5.1 is the depiction of the architecture of the MCNN algorithm augmented by Gramian angular plot branch and Markov transition field branch. As in this thesis, we are combining MCNN, Gramian angular field, Markov transition field.

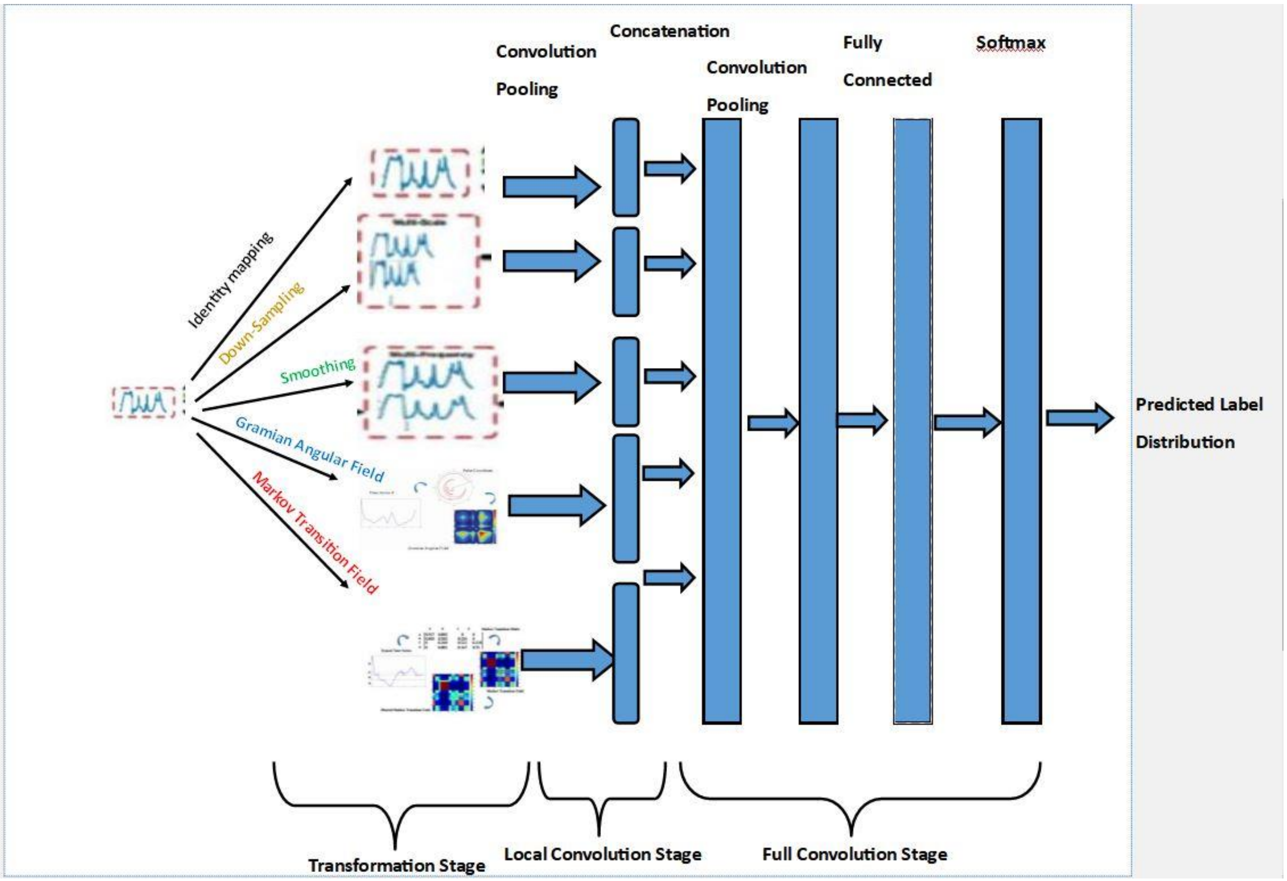


Fig. 3.5.1. Architecture of proposed modification of MCNN. In the above figure first three branches are from Cui and el [3]. Fourth and fifth branch are Gramian Angular Field (GAF) and the Markov Transition Field (MTF) [4]. In the transformation stage and local convolution stage features are extracted which are fed to full convolution stage which consists of convolution pooling, fully connected layer and softmax layer.

Chapter IV - Data Preparation and Theoretical Underpinning

4.1 Data Acquisition and Data Preparation

4.1.1 Mimic II (Multi-parameter Intelligent Monitoring In Intensive Care II)

The MIMIC-II database consists of high-resolution signals. These are the time-series signal of vital signs. It also contains static clinical records. The purpose of the database is to support epidemiologic research and assessment of clinical decision support systems in the domain of critical care medicine. The dataset is collected from Beth Israel Deaconess Medical Center. It consists of 25,328 adult patients surgical and cardiac records. It was obtained from 2001 to 2007. According to health act standards, the data is entirely de-identified. If the same patient was admitted after 24 hours, the data, in that case, was recorded with a different ID. Time series are updated at 1 Hz.

All data that do not contain ICP signal over their entire range was discarded. The dataset contains only a small fraction of the data that includes ICP signals. Please see the appendix A for file names that were selected from the MIMIC II database. The segments were cross-checked with static patient information to eliminate the duplicate data of the same patient.

The definition of intracranial hypertension is an elevation of ICP over 20mm Hg. The dataset was imbalanced as only 10% positive labels were among the instances. This imbalance was handled by the SMOTE+ENN algorithm (please see section 4.3) to create more similar examples of the positive label.

4.1.2 CHARIS (Cerebral Haemodynamic Auto-regulatory Information System Database)

The CHARIS database consists of multi-channel records of arterial blood pressure (ABP), ECG, and ICP (intracranial pressure) of individuals diagnosed with TBI (traumatic brain injury). The purpose of the distribution of data to researchers is to systematize the analyses of appropriate physiological signals and construct algorithms driven by data in search of possible predictors of critical clinical events for individuals with significant brain injury.

The apparatus was installed in Robert Wood Johnson Medical Center of Rutgers University's surgical intensive care units (SICU). Data gathering units were coupled with patient monitors. The apparatus was activated as soon as a patient arrived with a diagnosis of brain injury that requires an ICP bolt/ventriculostomy. Individuals were mostly ventilated and sedated. ICP was uninterruptedly monitored with ventriculostomy.

General Electric TRAM-rac 4A was used to acquire the outputs from clinical monitors. The rate of sampling was 50 Hz with a resolution of 1.41 mV at plus-minus 5 V, which corresponds to the pressure resolution of 0.14 mmHg and a range of plus-minus 500 mmHg. ICP was uninterruptedly recorded with micro-transducers (Camino Direct Pressure Monitor, Camino Laboratories, San Diego, CA) that were implanted intra-parenchyma into the frontal cranium.

4.2 Data Cleaning

Data was collected from MIMIC-II and CHARIS databases. They were first scanned for any value higher than 100 Hg or negative values. Values higher than 100 Hg or negative values are caused by the movement of apparatus and does not give us information about ICP. If any value was found higher than 100 or negative, a NaN value replaced it. NaN values were subsequently replaced by interpolating the values 50 steps before the NaN value and 50 steps after the NaN value. 50 steps prior and after were chosen for obtaining a smooth spline for interpolation. The interpolated value was placed where NaN values occur. From the dataset, sampling was done at 50 Hz.

4.2.1 Identification of Hypertension Signals

Hypertension signals were identified by scanning for any value that is greater than 20 Hg. It identified 6 minutes of data before the designated point and were extracted and labelled as positive examples. This was done for both datasets MIMIC II and CHARIS. These extracted positive examples were then deleted from the original data. The remaining data was divided into 6 minutes of duration signals and labelled as negative examples. The 6 minutes was found suiTable due to the following considerations:

1. The length of ICP signals collected from the MIMIC and CHARIS datasets such that enough data is collected for training and testing phase.
2. The clinical response time required for medical staff to take necessary actions to avoid the hypertensive state of patients.

Positive and negative examples were then combined into one dataset for both MIMIC II and CHARIS dataset. Figure 4.1 shows the pre-processing steps.

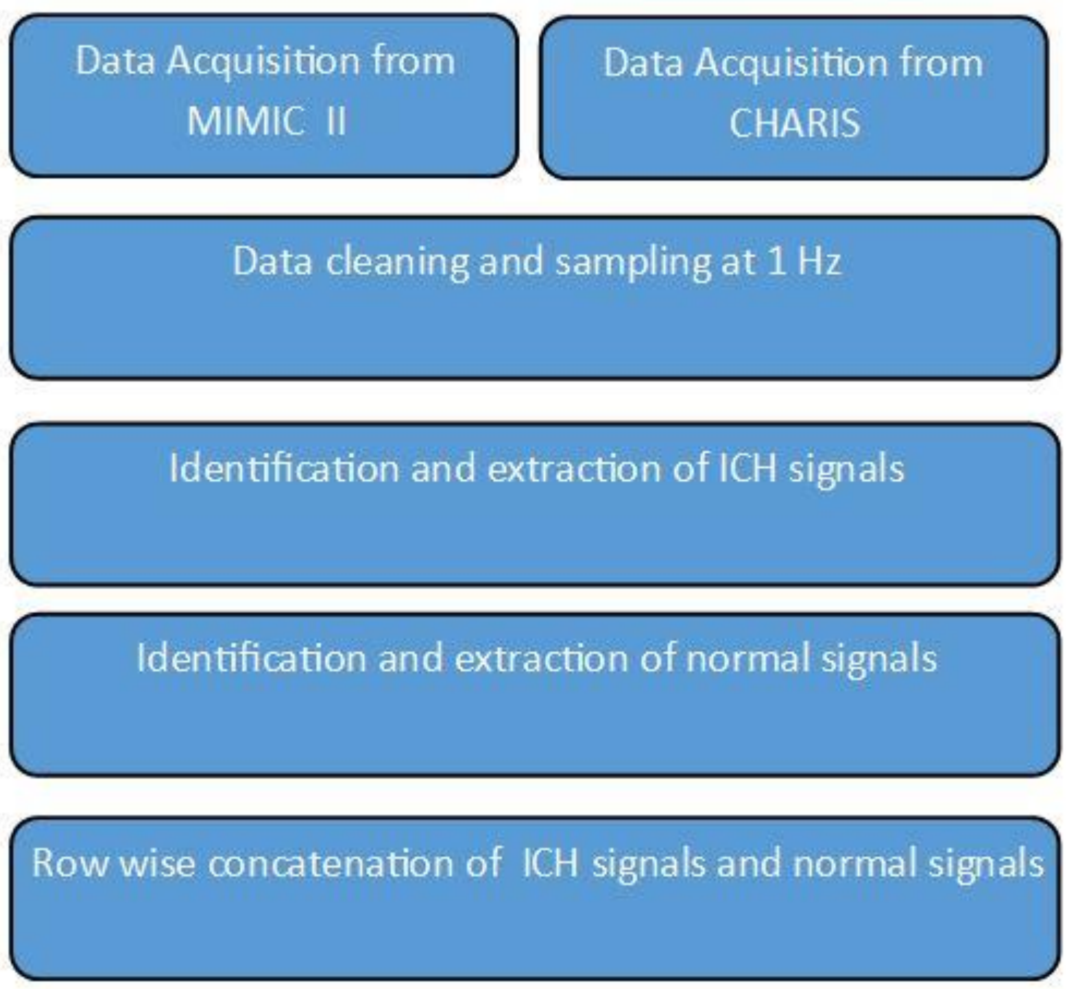

Data Acquisition from MIMIC II
Data Acquisition from CHARIS
Data cleaning and sampling at 1 Hz
Identification and extraction of ICH signals
Identification and extraction of normal signals
Row wise concatenation of ICH signals and normal signals

Fig. 4.1. Pre-Processing Steps

4.3 SMOTE+ENN

The resulting dataset was an imbalanced dataset with many more negative examples as compared to positive examples. The dataset was then treated with a SMOTE+ENN algorithm to balance the data. Smote oversamples the dataset by creating a synthetic data point for the minority class. It takes a vector from one of the minority class k neighbours. It then multiplies a random number x which is in between 1, and 0. Complement this to the current data point to create a new artificial data point.

For individual point p in rear class:

1. Calculate its k nearest neighbours in rear class.
2. Arbitrarily choose $r \leq k$ of the neighbours, and it is done by replacement.
3. Pick a random point p along the line and then join p and each of the r selected neighbours.
4. Complement these fictitious points to the dataset with rear class.

After SMOTE algorithm data was passed through the ENN (Extend natural neighbour) method. ENN creates a forecast in two-way communication. It takes into account the nearest neighbours of the test sample, and it considers the test sample as their nearest neighbours. It exploits the universal class-wise statistics from all training data by iteratively assuming every probable class memberships of a test sample. Extended nearest neighbour learns from the global distribution.

4.3.1 ENN Algorithm

Given an unknown sample X to be classified, it is iteratively assigned to each class j $j = 1, 2, \ldots, N$, and predicts the class membership according to the formula.

$$c_{ENN,V1} = \arg max_{j \in 1,2,\ldots\ldots N} \left\{ \left(\frac{\Delta n_j^i + k_i - kT_i}{(n_i + 1)k} \right)_{i=j} - \sum\nolimits_{i \neq j}^{N} \left(\frac{\Delta n_j^i}{n_i k} \right) \right\} \tag{4.3.1}$$

where

k = user-defined parameter for nearest neighbours

n_i = number of training data for class i

k_i = number of nearest neighbours of test sample X for class i

T_i = generalized class wise statistics of class i

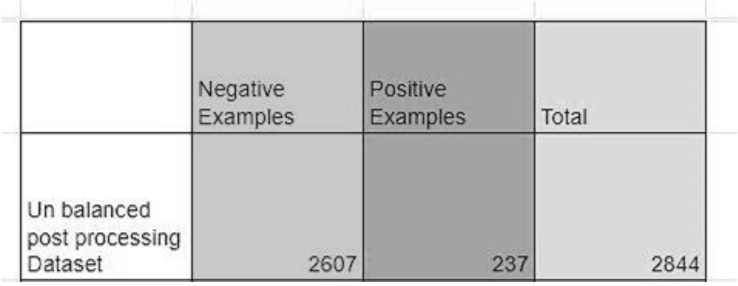
Δn_i^j = change of k nearest neighbours for class I when test sample X is labelled as belonging to class j

After the balancing the data, Markov transition matrix is created for each example. The matrix is flattened and concatenated with each example. The same process is repeated with Gramian Angular field images. Each example is taken, and the corresponding Gramian Angular field image is created. Each image is then concatenated with the same example. Finally, the data is fed to classifying algorithms for classification.

4.4 Data Statistics

After eliminating all the signals that did not contain a full six minutes duration from the data, we were left with 2844 examples. From these examples, 2607 were negative examples containing normal activities of ICP signal and 237 were positive examples containing precursor signal of hypertensive activity. Table 4.4.1 illustrates this distribution.

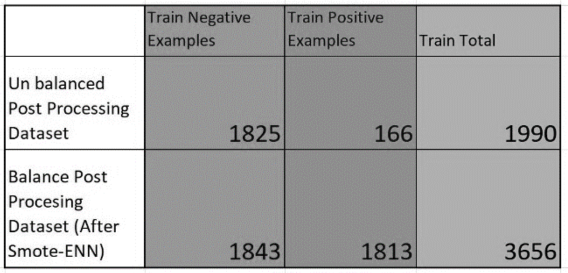
Table 4.4.1 Data distribution before Somte-ENN algorithm



	Negative Examples	Positive Examples	Total
Un balanced post processing Dataset	2607	237	2844

From the above dataset train and test dataset were created. 70% of data was selected for training purpose and 30% of data was selected for test purpose. Train dataset contained a total of 1990 examples. From which 1825 were negative examples and 166 were positive examples. After balancing the dataset with Smote ENN, there were 1843 negative examples and 1813 positive examples.

Table 4.4.2 Data distribution for training before and after Somte-ENN algorithm



	Train Negative Examples	Train Positive Examples	Train Total
Un balanced Post Processing Dataset	1825	166	1990
Balance Post Procesing Dataset (After Smote-ENN)	1843	1813	3656

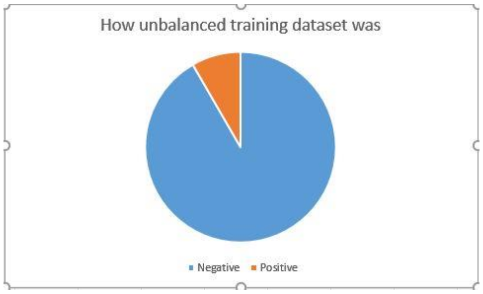


Figure 4.4.1 Unbalanced training dataset

Figure 4.4.1 gives us a sense of how much data was unbalanced, and Figure 4.4.2 provides a sense of how the dataset was balanced by Smote ENN algorithm.

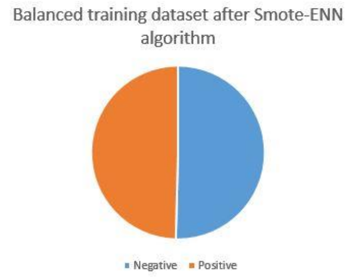
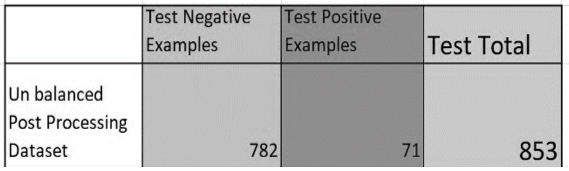


Figure 4.4.2 Balanced training dataset after Smote-ENN algorithm

Test dataset set contained a total of 853 examples from which 782 were negative examples, and 71 were positive examples. Table 3.4.5 illustrates this.

Table 4.4.3 Test data distribution



	Test Negative Examples	Test Positive Examples	Test Total
Un balanced Post Processing Dataset	782	71	853

Figure 4.4.3 gives us a sense of unbalanced test dataset

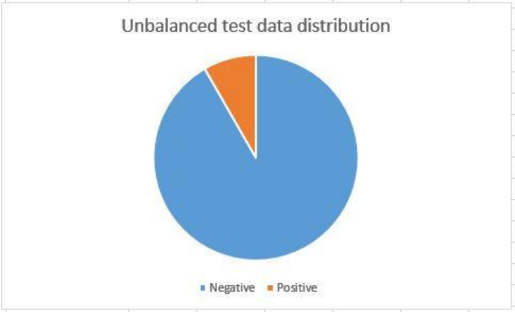


Figure 4.4.3 Unbalanced test data distribution

4.5 Machine Learning Algorithms used for classification

Following algorithms were used for classification:

- Nearest Neighbors
- Naïve Bayes
- Gaussian Process
- Linear Support Vector Machine - Classification (SVM)
- RBF Support Vector Machine - Classification (SVM)
- Quadratic Discriminant Analysis (QDA)
- Decision Trees
- Ada Boost
- MCNN

4.5.1 Nearest Neighbors

K nearest neighbour's (K-NN) algorithm can be classified as a non-parametric classifier. In feature space, the input of the classifier is k nearest neighbours. The output is a class membership. A voting strategy is applied for classifying class membership. Every neighbour votes for its class and a class is assigned to the observation according to majority vote. The K-NN algorithm is regarded as the simplest classification method in machine learning and can be considered as lazy learning or instance-based classification method. In this type of classification, method weights are assigned to each vote according to its distance from the observation. That is a weight of 1/*d* is attached to each vote where *d* is the distance of observation to the voting data point.

$$d(p,q)=\sqrt{\sum_i (p_i - q_i)^2}$$

(4.5.1.1)

where p and q are coordinates of point p and q

$$w \equiv \frac{1}{d(p,q)^2}$$

(4.5.1.2)

where x_q and x_i are coordinates of q and i th point.

A grid search was applied on *K*-Neighbors Classifier with 10 fold cross-validation, and k was searched from 2 to 7, and the best results were obtained for k =3 for each dataset.

4.5.2 Naïve Bayes

Naïve Bayes is a straightforward classifying algorithm. It is based on the Bayes theorem. The term naïve comes from the assumption that the value of any features does not depend on any other feature. That is all the features are independent of each other.

The conditional probability can be divided into

$$p(C_k|x) = \frac{p(c_k)p(x|C_k)}{p(x)}$$

(4.5.2.1)

where C_k is the k^{th} class, x is the observations, $p(C_k |x)$ is the posterior probability, $p(x| C_k)$ is the likelihood, $p(C_k)$ is the prior probability, and $p(x)$ is the evidence. And $p(Ck, x_1, x_2, x_3, \ldots, x_n)$ is defined as:

$$p(C_k, x_1, x_2, x_3, \ldots\ldots..x_n) = p(x_1, x_2, x_3, \ldots\ldots..x_n, C_k)$$
$$= p(x_1|C_k) * p(x_2|C_k) * p(x_3|C_k) * \ldots\ldots\ldots\ldots\ldots\ldots p(x_n|C_k)$$
$$= p(C_k)\prod_{i=1}^{n} p(x_i|C_k)$$

(4.5.2.2)

Under the naïve assumptions that all features are independent of each other. Classifier from the probability model

$$\overbrace{y}^{hat} = argmax_{k\epsilon\{1,2,3,\ldots\ldots..,k\}} p(C_k)\prod_{i=1}^{n} P(x_i|C_k)$$

(4.5.2.3)

The maximum a posteriori can be derived from the above formula. The Bayes classifier is the classifier that assigns a class label $\overbrace{y}^{hat} = C_k$ according to the above formula. No grid search was applied on naïve Bayes algorithm only ten-fold cross-validation was applied.

4.5.3 **Gaussian Process**

A Gaussian random variable X is specified by its standard deviation σ and mean μ. The density function is defined as:

$$P[X = x] = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(\frac{-(x-\mu)^2}{2\sigma^2})$$

(4.5.3.1)

A multi-variate Gaussian random variable X is specified by its covariance matrix $\sum$ and its mean μ. The density function is then defined as:

$$P[X = x] = \frac{1}{(2\pi)^{\frac{k}{2}}|\sum|^{2}} \exp(-\frac{1}{2}(x-u)'\sum(x-u))$$

(4.5.3.2)

A Gaussian process f(x) is an ensemble of random variables of which any finite set is a joint Gaussian distribution. A Gaussian process is defined by its covariance function K(x, y) and its mean μ. For n ∈ N and $x_1, x_2 \ldots \ldots \ldots .. x_n$, we have

$$(f(x_1), f(x_2), \ldots \ldots \ldots .. f(x_n) \sim N((\mathrm{\mu}_{x1,} \mathrm{\mu}_{x2,} \ldots \ldots .. \mathrm{\mu}_{xn}), K)$$

(4.5.3.3)

where N is a normal distribution and K is

$$\begin{pmatrix} k(x_1, x_1) & \cdots & k(x_1, x_n) \\ \vdots & \ddots & \vdots \\ k(x_n, x_1) & \cdots & k(x_n, x_n) \end{pmatrix}$$

(4.5.3.4)

The goal is to generate a Gaussian process with covariance K and mean μ. First, we calculate the Cholesky decomposition of K by:

$$K = LL^T$$

(4.5.3.5)

where L is lower triangular. We generate

$$\mathbf{u} = \mathrm{N}(0, \mathbf{I})$$

(4.5.3.6)

where N is a Gaussian with 0 mean and standard deviation I which is the identity matrix. Then we calculate

$$x = \mathrm{\mu} + L\mathbf{u}$$

(4.5.3.7)

Finally, we calculate

$$\mathbf{E}(\mathrm{x}-\mathrm{u})(x-u)^{T}=\mathbf{LE}[\mathbf{uu^{T}}]\mathbf{L^{T}}=\mathrm{K}$$

(4.5.3.8)

A grid search was applied for RBF parameter. Grid was [1.0, 1.1, 1.2, 1.3, 1.4, 1.5] with 10-fold cross-validation. RBF of 1.0 gave the best result.

4.5.4 Support Vector Machine - Classification (SVM)

A Support Vector Machine (SVM) implements classification by discovering the hyperplane that optimizes the margin intermediate the two classes. The vectors that outline the hyperplane are the support vectors. To describe an optimum hyperplane, we need to optimize the thickness of the margin (w) where x is observations and b the bias term.

Hyperplane are defined by

$$(w.x+b) \geq \forall x \ of \ class \ 1 \qquad\qquad (w.x+b) \leq -1, \forall x \ of \ class \ 2$$

(4.5.4.1)

$$\frac{w}{||w||}.(x_2-x_1)=width=\frac{2}{||w||}$$

$$w.x_2+b=1$$

$$w.x_1+b=-1$$

$$w.x_2+b-w.x_1-b=1-(-1)$$

$$w.x_2-w.x_1=2$$

$$\frac{w}{||w||}(x_2-x_1)=\frac{2}{||w||}$$

(4.5.4.2)

We calculate w and b by the solution of the following objective function using Quadratic Programming.

$$min\frac{1}{2}||w||^2$$

$$s.t. y_i(w.x_i + b) \geq 1, \forall x_i$$

(4.5.4.3)

If the data is linearly separable, in that case, SVM finds a unique global minimum. In the ideal case, SVM will discover hyperplanes that separate the data into two classes and maximizes the width of the margin. In real situation cases, some of the points are misclassified, and SVM discovers the hyperplane, which maximizes the margin and minimizes the incorrect misclassifications.

The algorithm attempts to uphold the slack variable to zero while optimizing margin. Though, it does not diminish the number of misclassifications instead the sum of distances from the margin hyperplanes are minimized. The constraint for the equation becomes:

$$y_i(w.x_i + b) \geq 1 - \varepsilon_i, \forall x_i \qquad \varepsilon_i \geq 0$$

Objective function penalizes for misclassified instances and those within the margin:

$$min\frac{1}{2}||w||^2 + C\sum_i \varepsilon_i$$

(4.5.4.4)

The easiest method to distinguish the two clusters of data is with a straight line (one dimension), flat plane (two dimensions), or an N-dimensional hyperplane. There are circumstances where a nonlinear area can separate the clusters more ably. SVM computes this by means of a nonlinear kernel function to map the data into a different space.

- If such a linear decision surface does not occur, the data is transformed into a much higher dimensional space (i.e., the feature space) where the splitting decision surface is found,
- The feature space is built through a mathematical projection called kernel trick.

4.5.4.1 Kernel Trick

The linear classifier relies on the inner product between vectors

$$K(x_i, x_j) = x_i^T x_j$$

(4.5.4.5)

When mapping every data point into high-dimensional space through some transformation function φ, the inner product turns out to be:

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

(4.5.4.6)

A kernel is a function that is the same as the inner product in some feature spaces. A kernel function subliminally maps to space lacking the need to compute $\varphi(x)$.

C and γ are the parameters for a nonlinear support vector machine (SVM) with a Gaussian radial basis function kernel. Intuitively, the gamma parameter describes how far the effect of a single training example reaches, with low values implies ‘far’ and high values implies ‘close.’ The gamma parameters can be considered as the inverse of the radius of influence of examples selected by the model as support vectors.

The *C* parameter trades off misclassification of training examples against the simplicity of the decision surface. A low *C* makes the decision surface smooth, while a high *C* aims at classifying all training examples correctly by giving the model freedom to select more samples as support vectors.

Grid Search and 10 fold cross-validation was applied with the following parameter

C = [0.001, 0.01, 0.1, 1, 10]

γ(gamma) = [0.001, 0.01, 0.1, 1, 2]

Best value for Support Vector Classifier was found to be $\gamma = 2$ and $C = 1$.

4.5.5 Quadratic Discriminant Analysis (QDA)

In the quadratic discriminant analysis, two or more classes are separated by quadric surfaces. It is considered as the generalization of linear discriminant analysis by a degree of two. If we have a training set of *x* observation with the attached type of observation y, the problem is to classify any new observation into which y type it belongs. In Quadratic Discriminant Analysis, the correct solution is assumed to be of quadratic. It should be of the following kind.

$$X^T AX + b^T X + c$$

(4.5.5.1)

If we consider the particular case where each observation is of two measurements, then the resulting surfaces will be conic. That is either they are hyperbola, parabola, ellipse, circle or line. In this perspective, we say that quadratic discriminant analysis is the generalization of its linear counterpart.

There are similarities and differences between quadratic discriminant analysis and linear discriminant analysis. As in linear discriminant analysis, it is supposed that classes are typically distributed. The difference between QDA and LDA is that there is no assumption on the covariance of every class as the normality assumptions consequences are that the best possible hypothesis test for a given measurement classification is the likelihood ratio test.

If we suppose that there are only two classes that is y belongs to {0, 1}, and the means of two categories are $\mu_{y=0}$ and $\mu_{y=1}$ and the covariance are $\Sigma_{y=0}$ and $\Sigma_{y=1}$ We can, in this case, calculate the likelihood ratio as follows:

$$Likelihood\ ratio = \frac{\sqrt{|2\pi\Sigma_{y=1}}^{-1} \exp(-\frac{1}{2}(x-\mu_{y=1})^T \Sigma_{y=1}^{-1}\left(x-\mu_{y=1}\right))}{\sqrt{|2\pi\Sigma_{y=0}}^{-1} \exp(-\frac{1}{2}(x-\mu_{y=0})^T \Sigma_{y=0}^{-1}\left(x-\mu_{y=0}\right))} < t$$

(4.5.5.2)

The resulting surfaces from the above formula are quadratic or conic. Population mean and variance-covariance are substituted in the above equation. In this thesis, default parameters of Sci-kit learn package for Quadratic Discriminant Analysis were used with tenfold cross-validation.

4.5.6 Decision Tree and Ada Boost

A decision tree entails a set of correct/false decision rules. This can be viewed as a game of various questions where we ask different questions centred on the answers to earlier questions, and then at the termination, we make an estimate grounded on all the answers. We can envisage a decision tree as a set of nodes (conforming to correct/false questions), each of which has two subdivisions reliant on the answer to the problem. Contrasting to real trees, we typically draw them with their "root" at the TOP, and the "leaves" at the bottom:

A. How to pick nodes

A selected attribute *A*, with *K* different values, splits the training set E into subsets E_1, E_2, …, E_k. The estimated entropy (EH) left over after trying to attribute *A* with branches *i*=1, 2, …, *k* is

$$EH(A) = \sum_{i=0}^{K} \frac{p_i + n_i}{p + n} H(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i})$$

(4.5.6.1)

The following formula calculates the entropy

$$H\left(\frac{p}{p+n}, \frac{n}{p+n}\right) = -\frac{p}{p+n} log_2\left(\frac{p}{p+n}\right) - \frac{n}{n+p} log_2(\frac{n}{p+n})$$

(4.5.6.2)

Information gain (I) or a reduction in entropy for an attribute is calculated

$$I(A) = H\left(\frac{p}{p+n}, \frac{n}{p+n}\right) - EH(A)$$

(4.5.6.3)

The depth of a decision tree is the length of the largest path from a root to a leaf. The leaf is the endpoint of a tree.

We create a node for an attribute that has the largest I(A). A grid search was used with tenfold cross-validation with the following parameters

Depth = [2, 3, 4, 5, 6, 7, 8, 9, 10], Leaf = [1, 5, 10, 15, 20]

The best results were obtained at depth 5 and leaf 10.

B. Random Forest

A Random Forest entails an assembly or ensemble of simple tree predictors, each adept at making a response when offered with a set of predictor values. Aimed at classification problems, this response takes the arrangement of a class association, which links, or classifies, a set of independent predictor values with one of the classes present in the dependent variable. Otherwise, for regression problems, the tree response is an approximation of the dependent variable assumed the predictors.

A Random Forest entails a random number of simple trees, which are used to decide the concluding outcome aimed at classification problems, the ensemble of simple trees votes for the entire prevalent class. In the regression problem, their responses are an averaged (taken the mean of) to obtain an approximation of the dependent variable.

C. Bagging

In ensemble algorithms, bagging procedures form a class of algorithms, which construct numerous occurrences of a black-box forecaster on random subsets of the original training set and then combined their specific predictions to create a final prediction. These procedures are used as a way to decrease the variance of a base estimator (e.g., a decision tree).

D. Boosting

Boosting is an alternative method for refining the prediction power from a decision tree. Boosting works likewise to *bagging* except that the trees are grown consecutively consuming information from formerly grown trees. Boosting also does not include bootstrapping, in its place each tree is fitted on an adapted version of the original data.

Instead of fitting a lone big decision tree which results in hard fitting the data, and possibly overfitting the boosting method learns slowly. Given the existing model, we fit a decision tree to the residuals from the model, rather than the result Y.

Adaboost was grid searched with the following number of estimators [5, 10, 15, 20, 25, 30, 35, 40] and decision tree with above-mentioned grid search was used as a base estimator. A grid search was performed with tenfold cross-validation. The best result was found with 30 number of estimators with a depth of 10 and leaf of 15.

Chapter V - Results

In this chapter, a Table providing precision, recall, accuracy, and F1 score (section 5.2) for each algorithm is presented. The experiments has been done using he following dataset:

(i) The original dataset,
(ii) The original dataset augmented by Markov transition fields and Gramian Angular Field,
(iii) The original dataset passed through Smote ENN, and
(iv) The original dataset augmented by Markov transition fields, Gramian Angular Field, and passed through SMOTE ENN Algorithm. After this, a description of the result of the algorithm is presented.

The four metrics -- precision, recall, accuracy, and F1 -- are selected to measure the performances of all algorithms. Precision is selected because it is a measure of the fraction of correct positive selected among all selected positive. We consider recall because it is a measure of a collection of all true positive selected among all positives identified. Accuracy gives us an indication of how accurately a classifier is classifying. The most critical measure is F1 score not only because it is a measure that combines the measures of recall and precision but also because we are comparing datasets that are imbalanced with datasets that are balanced.

5.1 Methodology

Experiments were conducted on a machine with Intel processor core i7-7700HQ CPU @ 2.80Ghz 2.80 GHz having a RAM of 16 GB and Nvidia GTX 1070 Ti with 8 GB of memory. Sci-kit learn was used for all the algorithms except MCNN:

- **Nearest Neighbors -** A grid search was applied on K Neighbors Classifier with 10-fold cross-validation. k was searched from 2 to 7, and the best results were obtained for $k = 3$,
- **Naïve Bayes -** No grid search was applied on naïve Bayes algorithm only tenfold cross-validation was applied,

- **Gaussian Process -** A grid search was applied for RBF parameter. Grid was [1.0, 1.1, 1.2, 1.3, 1.4, 1.5] with ten-fold cross-validation. Best results were found for RBF of 1.0,
- **Linear Support Vector Machine - Classification (SVM) and RBF Support Vector Machine - Classification (SVM) -** Grid Search, and 10 fold cross-validation was applied with the following parameter

  C = [0.001, 0.01, 0.1, 1, 10], γ = [0.001, 0.01, 0.1, 1, 2]

  Best value for Support Vector Classifier was found to be $\gamma = 2$ and $C = 1$,
- **Quadratic Discriminant Analysis (QDA) -** In this thesis, default parameters of sci-kit learn package for Quadratic Discriminant Analysis were used with tenfold cross-validation.
- **Decision Trees -** A grid search was used with tenfold cross-validation with the following parameters

  Depth = [2, 3, 4, 5, 6, 7, 8, 9, 10], Leaf = [1, 5, 10, 15, 20]

  Best Results were obtained for depth 5 and leaf 10,
- **Ada Boost –** Ada boost was grid searched with the following number of estimators [5, 10, 15, 20, 25, 30, 35, 40] and decision tree with above-mentioned grid search was used as a base estimator. A grid search was performed with tenfold cross-validation. The best result was found with 30 number of estimators with a depth of 10 and leaf of 15,
- **Multi-Scale Convolutional Neural Networks (MCNN) -** For window slicing, the length of slices was set to 0.9b where b is the initial length of the series. Number of filters was 256 for CNN. The space of hyper-parameters search was filter size, batch size and pooling factor. We used batch size of {4, 8, 16, 32, 64}. Filter sizes were searched in grid space of {3, 5, 7, 9}. Grid space for pooling was {2, 3, 5}. 256 neurons were used for fully connected layers, and 128 neurons were used for branches. We used a mini-batch stochastic gradient with momentum as the optimizing algorithm. Early stopping was applied to validation error to avoid overfitting.

5.2 Precision, Recall, Accuracy, and F1 score

Normal signals are considered as negative examples, and hypertensive signals are considered as positive examples. Following metrics were used to evaluate the algorithms.

Precision is defined by the following

$$\boldsymbol{Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}}$$

It is an indicator of how many true positives were selected from all positive selected. It is an indicator of how precise is an algorithm is.

The recall is defined by

$$\boldsymbol{Recall = \frac{True\ Positive}{True\ Positive + False\ Negative}}$$

It is an indicator of an algorithm's ability to pick positives among all the positive in the dataset.

Accuracy is defined by

$$\boldsymbol{Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative}}$$

Accuracy is an indicator of how accurately an algorithm predicts.

F1 score is defined by

$$\boldsymbol{F1\ score = \frac{Precision * Recall}{Precision + Recall} * 2}$$

F1 score is a combination of precision and recall. If we want to see the combined effect of precision and recall, we use the F1 score.

The confusion matrix is a matrix containing the horizontal axis as predicted results and the vertical axis as true results. With the help of the confusion matrix, we can see how many positive were classified accurately and how many were classified inaccurately. Similarly, from the confusion matrix, we can see how many negative were classified accurately and how many were classified inaccurately. Confusion matrices are presented in appendix B.

Results of algorithms are presented below. Discussion and insights as to why we were obtaining these results are presented in chapter 6.

5.3 Nearest Neighbors

Results of Nearest Neighbors are presented in Table 5.1. Nearest Neighbors performed the best with SMOTE ENN but without images augmentation. In fact augmentation of data with Markov transition fields and Gramian Angular Field images (MTF+GAFI) deteriorated its performance. SMOTE ENN algorithm for balancing the dataset enhanced its performance. Precision also decreased with augmenting data with MTF+GAFI**.** Robust precision increased to 100% by balancing the dataset with SMOTE ENN and augmenting the data with Markov transition fields, and Gramian Angular Field images.

Table 5.1. Nearest Neighbors Accuracy, Precision, Recall, and F1 Score
(i) original dataset, (ii) original dataset augmented by Markov transition fields and Gramian Angular Field,
(iii) original dataset passed through Smote ENN, and (iv) original dataset augmented by Markov transition fields, Gramian Angular Field, and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Nearest Neighbors				
original dataset	0.9291	0.7500	0.0952	0.1690
original dataset augmented by Markov transition fields and Gramian Angular	0.9083	0.3281	0.4118	0.3652
original dataset passed through Smote EN	**0.9836**	**1.0000**	**0.9713**	**0.9854**
original dataset augmented by Markov transition fields, Gramian Angular Field, and Passed Through Smote ENN	0.8326	**1.0000**	0.7755	0.8736

5.4 Naïve Bayes

Results of Naïve Bayes are presented in Table 5.2. Both images augmentation and SMOTE ENN algorithm enhanced Naïve Bayes accuracy. Though the overall performance of Naïve Bayes was relatively low as compared to other algorithms. Precision was excellent without enhancement by MTF+GAFI. In fact it deteriorated slightly with augmentation of MTF+GAFI. SMOTE ENN algorithm for balancing the dataset also slightly decreased its precision. But the combined effect of both augmentation by MTF+GAFI and SMOTE ENN enhanced its precision to 99%. Naïve Bayes recall on its own was not good, just over 9% on original dataset. MTF+GAFI augmentation increased it to 12%. A significant increase in recall came with balancing the dataset with SMOTE ENN. F1 Score of Naïve Bayes algorithm without balancing the data with SMOTE ENN is not very impressive.

Table 5.2. Naïve Bayes Accuracy, Precision, Recall, and F1 Score
(i) original dataset, (ii) original dataset augmented by Markov transition fields and Gramian Angular Field, (iii) original dataset passed through Smote ENN, and (iv) original dataset augmented by Markov transition fields, Gramian Angular Field, and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Naïve Bayes				
original dataset	0.2837	0.9841	0.0944	0.1722
original dataset augmented by Markov transition fields and Gramian Angular Field	0.6112	0.9412	0.1221	0.2162
original dataset passed through Smote ENN	0.6565	0.9768	0.6206	0.7590
original dataset augmented by Markov transition fields, Gramian Angular Field, and Passed Through Smote ENN Algorithm	**0.8281**	**0.9961**	**0.7725**	**0.8702**

5.5 Gaussian process

The accuracy of the Gaussian Process (Table 5.3) decreased with the enhancement of data with images, but its accuracy was increased to nearly 99% by balancing the data with the SMOTE ENN algorithm. Precision also decreased by augmenting the dataset with MTF+GAFI. But precision becomes 100% by balancing the data with the SMOTE ENN algorithm. The combined effect of MTF+GAFI augmentation and SMOTE ENN algorithm produced a precision of 99%. Similarly, the recall was decreased by augmenting the dataset set with MTF+GAFI, While SMOTE ENN raised the recall to 98%. The combined effect of MTF+GAFI and SMOTE ENN were nearly the same as SMOTE ENN alone. F1 score decreased with enhancement by MTF+GAFI.

Table 5.3 Gaussian Process' Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Gaussian process				
Original Dataset	0.9531	0.4762	0.8333	0.6061
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9347	0.3725	0.4872	0.4222
Original Dataset Passed Through Smote ENN	0.9900	**1.0000**	0.9823	0.9911
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	**0.9910**	0.9987	**0.9858**	**0.9922**

5.6 Linear SVM

Results of Linear SVM are presented in Table 5.4. The accuracy of Linear SVM was enhanced by augmenting the data with images. Accuracy was also improved by balancing the data with the SMOTE ENN algorithm. The precision of Linear-SVM increased from 7% to 84% with augmentation of MTF+GAFI. While precision went from 7% to 99% with the balancing of data with SMOTE ENN, the combined effect of enhancement of the dataset with MTF+GAFI and SMOTE ENN raised the precision to nearly 100%. The recall was 100% without augmentation of the dataset with MTF+GAFI and without balancing the dataset with SMOTE ENN. It decreased to 79% when we augmented the dataset with MTF+GAFI. Recall also dropped 3% when we balanced the dataset with SMOTE ENN. The combined effect of both augmentation with MTF+GAFI and the SMOTE ENN algorithm was to decreased the recall 2% from 100% that linear-SVM was able to achieve. F1score increased drastically with augmentation of MTF+GAFI.

Table 5.4 Linear SVM's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Linear SVM				
Original Dataset	0.9303	0.0794	**1.0000**	0.1471
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9761	0.8431	0.7963	0.8190
Original Dataset Passed Through Smote ENN	0.9822	0.9949	0.9736	0.9841
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	**0.9887**	**0.9987**	0.9821	**0.9903**

5.7 SVM-RBF

SVM-RBF best accuracy (Table 5.5) was without augmentation with images or SMOTE ENN algorithm for balancing the data. This was 93% and decreased to 57% with expanding with images and SMOTE ENN algorithm. Precision was 20% without augmenting the dataset with MTF+GAFI. It was raised to 56% when the dataset was supplemented with MTF+GAFI. Though SMOTE ENN alone increased the precision to 77%. The combined effect of SMOTE ENN and augmentation by MTF+GAFI raised the precision to 100%. The recall was 100% without augmentation with MTF+GAFI. The recall was 84% with images. The combined effect of enlarging the dataset with MTF+GAFI and SMOTE ENN algorithm the recall came out to be 57%.

Table 5.5. SVM-RBF's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
RBF				
Original Dataset	**0.9399**	0.2063	**1.0000**	0.3421
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9359	0.5632	0.8473	0.6766
Original Dataset Passed Through Smote ENN	0.8760	0.7761	**1.0000**	**0.8739**
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.5784	**1.0000**	0.5784	0.7329

5.8 QDA

QDA accuracy (Table 5.6) was 27% but augmenting the data with images its accuracy jumped to 92%. Its accuracy also raised from 27% to 99% when data was balanced with SMOTE ENN. By combining the augmentation of data with images and SMOTE ENN algorithm, its accuracy jumped to 100%. Precision decreased when augmenting the dataset with MTF+GAFI from 96% to 35% but SMOTE ENN alone increased the precision to 100%. The recall was also decreased with the enhancement of a dataset with MTF+GAFI. But it increased to 99% by balancing the dataset with SMOTE ENN. The F1 score also decreased with MTF+GAFI augmentation.

Table 5.6 QDA's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
QDA				
Original Dataset	0.2704	0.9683	0.0916	0.1674
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9221	0.3529	0.3830	0.3673
Original Dataset Passed Through Smote ENN	0.9950	**1.0000**	0.9911	0.9955
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	**1.0000**	**1.0000**	**1.0000**	**1.0000**

5.9 Decision Tree

The decision tree approach (Table 5.7) performed very well on its own. There was a slight increased in performance with the SMOTE ENN algorithm and data augmentation with images. Precision increased from 88% to 92% when we augmented the dataset with MTF+GAFI. SMOTE ENN alone increased the precision to approximately 99%. The combined effect of MTF+GAFI augmentation and the SMOTE ENN algorithm was to raise the precision to 100%.The recall was decreased when we augmented the dataset with MTF+GAFI from around 91% to 81%. Smote ENN alone increased the recall from 91% to 98%. The combined effect of both augmentations by MTF+GAFI and Smote ENN was to raise the recall to 99%. F1 score slightly decreased with enlargement of the dataset with MTF+GAFI. Decision tree diagrams are presented in appendix C.

Table 5.7 Decision Tree's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Decision Tree				
Original Dataset	0.9856	0.8889	0.9180	0.9032
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9812	0.9216	0.8103	0.8624
Original Dataset Passed Through Smote ENN	0.9936	0.9987	0.9898	0.9942
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	**0.9962**	**1.0000**	**0.9935**	**0.9968**

5.10 Ada Boost

Due to its similarity to the decision tree algorithm, the Ada Boost performed quite similar to a decision tree as far as accuracy is concerned. Precision (Table 5.8) increased from 90% to 94% when augmenting the dataset with MTF+GAFI. Smote ENN raised the precision to 100%, and the combined effect of both was also 100% of precision. The recall was 100% before it decreased to around 87% when we augmented the dataset with MTF+GAFI. Smote ENN increased the precision to about 99%, and the combined effect of both was also approximately 99% of precision. F1 Score decreased from 0.95 to 0.90 after augmenting the dataset with MTF+GAFI.

Table 5.8 Ada Boost's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Ada Boost				
Original Dataset	0.9928	0.9048	**1.0000**	0.9500
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9874	0.9412	0.8727	0.9057
Original Dataset Passed Through Smote ENN	**0.9964**	**1.0000**	0.9936	**0.9968**
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.9955	**1.0000**	0.9922	0.9961

5.11 MCNN

MCNN (Table 5.9) performed well on its own, but its accuracy slightly increased with the Smote ENN algorithm and data augmentation with images. Precision was 47% and it decreased to 26% with augmentation of the dataset with MTF+GAFI. But it increased to 91% with Smote ENN. The combined effect of both was to increase the precision to 93%. Recall increased from around 40% to approximately 80% when we augmented the dataset with MTF+GAFI. Smote ENN alone raised the recall to 95%, and the combined effect of both was also 95%. The F1 score was slightly decreased with data augmentation by MTF+GAFI.

Table 5.9 MCNN's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
MCNN				
Original Dataset	0.9221	0.4706	0.4068	0.4364
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9399	0.2698	0.8095	0.4048
Original Dataset Passed Through Smote ENN	0.9414	0.9164	0.9500	0.9329
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	**0.9515**	**0.9327**	**0.9572**	**0.9448**

5.12 Summary

The results obtained (Table 5.10) show the algorithms perform better when the dataset is first processed through Smote ENN algorithm. We can also conclude that balancing provide better results than augmentation.

Table 5.10 All Algorithm's Accuracy, Precision, Recall, and F1 Score of (i) Original Dataset, (ii) Original Dataset Augmented by Markov transition fields and Gramian Angular Field, (iii) Original Dataset Passed Through Smote ENN, and (iv) Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Algorithm	Accuracy	Precision	Recall	F1
Nearest Neighbors				
Original Dataset	0.9291	0.7500	0.0952	0.1690
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9083	0.3281	0.4118	0.3652
Original Dataset Passed Through Smote ENN	0.9836	1.0000	0.9713	0.9854
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.8326	1.0000	0.7755	0.8736
Naïve Bayes				
Original Dataset	0.2837	**0.9841**	0.0944	0.1722
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.6112	0.9412	0.1221	0.2162
Original Dataset Passed Through Smote ENN	0.6565	0.9768	0.6206	0.7590
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.8281	0.9961	0.7725	0.8702
Gaussian process				
Original Dataset	0.9531	0.4762	0.8333	0.6061
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9347	0.3725	0.4872	0.4222
Original Dataset Passed Through Smote ENN	0.9900	1.0000	0.9823	0.9911
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.9910	0.9987	0.9858	0.9922
Linear SVM				
Original Dataset	0.9303	0.0794	1.0000	0.1471
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9761	0.8431	0.7963	0.8190
Original Dataset Passed Through Smote ENN	0.9822	0.9949	0.9736	0.9841
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.9887	0.9987	0.9821	0.9903
RBF				
Original Dataset	0.9399	0.2063	1.0000	0.3421
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9359	0.5632	0.8473	0.6766
Original Dataset Passed Through Smote ENN	0.8760	0.7761	**1.0000**	0.8739
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.5784	**1.0000**	0.5784	0.7329
QDA				
Original Dataset	0.2704	0.9683	0.0916	0.1674
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9221	0.3529	0.3830	0.3673
Original Dataset Passed Through Smote ENN	0.9950	**1.0000**	0.9911	0.9955
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	**1.0000**	**1.0000**	**1.0000**	**1.0000**
Decision Tree				
Original Dataset	0.9856	0.8889	0.9180	0.9032
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9812	0.9216	0.8103	0.8624
Original Dataset Passed Through Smote ENN	0.9936	0.9987	0.9898	0.9942
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.9962	**1.0000**	0.9935	0.9968
Ada Boost				
Original Dataset	**0.9928**	0.9048	**1.0000**	**0.9500**
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	**0.9874**	**0.9412**	**0.8727**	**0.9057**
Original Dataset Passed Through Smote ENN	**0.9964**	**1.0000**	0.9936	**0.9968**
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.9955	**1.0000**	0.9922	0.9961
MCNN				
Original Dataset	0.9221	0.4706	0.4068	0.4364
Original Dataset Augmented by Markov transition fields and Gramian Angular Field	0.9399	0.2698	0.8095	0.4048
Original Dataset Passed Through Smote ENN	0.9414	0.9164	0.9500	0.9329
Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm	0.9515	0.9327	0.9572	0.9448

Chapter VI - Discussion

In this thesis, we explored the intracranial pressure (ICP) classification. We introduced a SMOTE ENN algorithm for balancing the data, and its effect was examined on the classification of ICP. The data was further augmented with Markov transition field, and Gramian angular field images and implications of this augmentation were also investigated. The information is divided into four sets

1. Data without Smote ENN algorithm and without augmenting it with Gramian angular field images and Markov transition matrix.
2. Data with Smote ENN algorithm but without augmenting it with Gramian angular field images and Markov transition matrix.
3. Data without Smote ENN algorithm but augmenting it with Gramian angular field images and Markov transition matrix.
4. Data with Smote ENN algorithm and augmenting it with Gramian angular field images and Markov transition matrix.

These datasets were passed to following classification algorithms.

- Nearest Neighbors
- Naïve Bayes
- Gaussian Process
- Linear Support Vector Machine - Classification (SVM)
- RBF Support Vector Machine - Classification (SVM)
- Quadratic Discriminant Analysis (QDA)
- Decision Trees
- Ada Boost
- Multi-Scale Convolutional Neural Networks (MCNN)

The effect of Smote ENN and augmentation of data with Gramian angular field images and Markov transition field were compared to original ICP data for classification normal and hypertensive ICP signals.

6.1 Images Augmentation

Data augmentation with MTF+GAFI correspondence to the expansion of data space. If data space is of dimension d, Markov transition fields image of dimension m, and Gramian Angular Field image of dimension g than by augmenting data with MTF+GAFI the dimension of data space becomes $d*m*g$ where d, m, and g are spaces representing data in three different representation. This three representation of data are concatenated, and the data space of dimension $d*m*g$ is created.

6.2 Nearest Neighbors

Nearest Neighbor algorithm accuracy decreases as data is augmented with MTF+GAFI. This is because of the expansion of space as it becomes difficult for Nearest Neighbors to be accurate in the expanded space. But as the Smote ENN connects the dots for minority class by linearity, it becomes easier for the Nearest Neighbor to be correct. The combined effects decrease the accuracy from which Smote ENN found.

Recall shows the same trend. Precision decreases as it is picking false positive in the expanded space due to its near neighbour nature. While Smote ENN increases the precision as data is more connected linearly for the minority class, the combined effect of image augmentation and Smote ENN is also very good. F1 score increases with data augmentation by images, but Smote ENN increases F1score significantly. The combined effect is decreasing in the F1 rating.

6.3 Naïve Bayes

The accuracy of the Naïve Bayes algorithm increases due to augmentation of data with images. That is because $P(x|C)$ that is the likelihood term increases due to augmentation and balancing of data. The combined effects of both superimpose each other in a constructive sense. Increase in likelihood term also increases the probability of picking false positive that is the reason we see a decrease in precision by augmenting the dataset with images. We also see the same effect of Smote ENN due to the same reason.

Recall increases slightly because of the better estimate of the likelihood term with data augmentation with images. It increases significantly with Smote ENN algorithm. That is because

Smote ENN affects both the likelihood term as well as the probability of a class. The combined effect also superimposed in a constructive sense. The F1 score shows the same trend.

6.4 Gaussian Process

Accuracy, precision, recall, and F1 all decrease with data augmentation with images. This is because in expanded space we have three kinds of representation of the same data and covariance matrix among these three representations is a weaker covariance matrix as compared to the covariance matrix of single representation matrix. As Smote ENN balances the dataset, it has in a single representation the covariance matrix of single representation increases, which results in an increase in accuracy, precision, recall, and F1 score.

If dataset augmentation with images occurs after the Smote ENN algorithm than the balancing of the dataset also increases the covariance matrix of three representation which results in an increase in accuracy, precision, recall, and F1 score.

6.5 Linear Support Vector Machine - Classification (SVM)

We see that augmenting the dataset with MTF+GAFI accuracy and precision increases as segmentation of space in d*m*g dimensional space is more accessible for support vector's planes while recall decreases because in expanded space it becomes difficult to pick a more significant number of positive. As the effect of Smote ENN is to introduce a particular kind of linearity among the minority class, therefore, it becomes easier for the linear support vector to find planes for segmentation. This is the reason that we see an increase in accuracy, precision as well as recall.

If we see the F1 score, it increases with data augmentation with MTF+GAFI but increases significantly with Smote ENN algorithm and the combined effect of both makes an F1 score of around 0.99.

6.6 RBF Support Vector Machine - Classification (SVM)

Accuracy slightly increases with data augmentation with MTF+GAFI this is because Markov transition fields images are making it difficult for RBF to classify. We see accuracy decreasing with Smote ENN. This is because of linearity introduced among minority class by Smote ENN, which makes RBF challenging to classify. We also observe that accuracy decrease significantly if we combine the effect of Smote ENN and data augmentation by MTF+GAFI. This is because of linearity introduce by Smote ENN and data augmentation by Markov transition fields Images.

6.7 Quadratic Discriminant Analysis (QDA)

Accuracy jumps significantly for QDA when data is augmented with images that may be because quadratic nature introduced in the dataset by polar coordinates. Accuracy also significantly jumps when data is passed to Smote ENN algorithm that is because QDA is a generalization of LDA and it can handle the linearity introduced by SMOTE ENN. The combined effect makes accuracy 100%.

The precision, on the other hand, decreases with image augmentation as Markov transition fields, and Gramian Angular Field images pick more false positives. But Smote ENN makes precision 100%, and the combined effect is also 100%.

The recall is also reduced with image augmentation as the expansion of space makes it difficult for QDA to pick positive examples from all the positive cases. But Smote ENN compensates this as the linearity nature of SMOTE ENN is selected very well by QDA. The F1 score shows a similar trend.

6.8 Decision Trees

The accuracy of the decision tree was excellent even without augmentation by images and without Smote ENN algorithm. Its accuracy decreased very slightly when images were augmented. As the data space expanded, it became somewhat tricky for the decision tree to classify. Smote ENN increased its accuracy somewhat because of linearity introduced by Smote ENN. The combined effect of space expansion and Smote ENN has also improved accuracy. Precision increased with the expansion of data space, and it increased significantly with the Smote ENN algorithm. As an expansion of data space and linearity introduced in minority class due to Smote ENN aided decision tree to be more precise.

Recall reduced with the expansion of data space as it becomes more challenging to pick the positive examples in an expanded space. But recall increased with Smote ENN as by balancing the data it becomes more comforTable for the decision tree to select the positive examples. The F1 score showed the same trend as mentioned above due to the same reasons.

6.9 Ada Boost

As Ada boost is just an ensemble method of the decision tree, it shows the same trend as a decision tree. The reasons for these trends are also the same, which are described in decision tree discussion. The results of the Ada boost are slightly better than the decision tree is because it is an ensemble method.

6.10 Multi-Scale Convolutional Neural Networks (MCNN)

The accuracy of the multi-scale convolutional neural network increases somewhat with the augmentation of images due to the expansion of space by polar coordinates and Markov transition field. It becomes slightly more accessible for moving averages of MCNN and its frequency component to classify the data, but the increase is minimal. Smote ENN increase the ability for moving averages to find smother representations of data and thus its accuracy increase slightly. The Combined effect is also an increase in accuracy.

Precision decreases with enhancing the dataset with images as it is finding the more false positive. The expansion of space with polar component and Markov transition field combined with moving averages and frequency components picks more false positives. While Smote ENN as it created linearity in positive examples; therefore, it becomes easier for MCNN to pick true positives, this is the reason that precision increases with Smote ENN. The combined effect is also an enhancement of precision.

Recall increases with the enhancement of data with images as it is picking more positives from all positives. Expansion of space with moving averages and frequency components increases the ability to pull more and more positives from all positives. Same is the case with Smote ENN algorithm and combined constructive effect superposition of the previous to effects.

F1 score decreases with images augmentation but increases with Smote ENN. MCNN did not show best results as compared to other algorithm. This is because number of training examples are just 1990 for unbalanced datasets and 3656 for balanced dataset.

Chapter VII - Conclusion and Future Direction of Work

7.1 **Conclusion**

This thesis was an investigation in comparing various algorithm mentioned above and selecting the best algorithm for each dataset used in this thesis. The conclusion is presented below.

Table 7.1 illustrates the best performance in each category. We clearly see that Ada boost is performing best in the majority of the categories, and QDA performance is best for the category where images are augmented, and data is passed through Smote ENN.

- ***Accuracy***

Ada Boost is the best algorithm accuracy wise on a dataset without images (original dataset), a dataset with images, and dataset without images but with Smote ENN. While QDA has the best accuracy on the dataset without images but with Smote ENN.

- ***Precision***

Naïve Bayes performance is best as far as precision is concerned on the dataset without images and dataset with images. Its precision decreases if the dataset is passed through Smote ENN algorithm. If the dataset is passed through Smote ENN algorithm, and the dataset is not augmented with images than nearest neighbours, and QDA has the best precision. On the other hand, if the dataset is passed through Smote ENN algorithm, and the dataset is also augmented by images than Ada Boost, Decision Tree, QDA, and Nearest Neighbors shows the best precision.

- ***Recall***

On the original dataset, Ada Boost and Linear SVM has the best recall. Ada Boost also has the best recall on the dataset with images and dataset that has been passed by Smote ENN but not augmented by images. QDA has the best recall on the dataset that is passed through Smote ENN and augmented with images.

- ***F1 Score***

Ada Boost has the best F1 score on the original dataset and dataset augmented with images. The Gaussian process has the best F1 score on the dataset that has been passed through Smote ENN but is not augmented with images. While QDA has best F1 score on the dataset passed through Smote ENN algorithm, and when the dataset is augmented with images.

From this study, we can conclude that Ada Boost is the most invariant to the data representation and distribution among the examined algorithms.

Table 7.1 Best Performance for Accuracy, Precision, Recall, and F1 Score of Original Dataset, Original Dataset Augmented by Markov transition fields and Gramian Angular Field, Original Dataset Passed Through Smote ENN, and Original Dataset Augmented by Markov transition fields and Gramian Angular Field and Passed Through Smote ENN Algorithm

Best Performance	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Original dataset	Ada Boost	Naïve Bayes	Ada Boost, Linear SVM	Ada Boost
Original dataset augmented by Markov transition field and Gramian angular field	Ada Boost	Naïve Bayes	Ada Boost	Ada Boost
Original dataset balanced by Smote-ENN	Ada Boost	Nearest Neighbors, QDA	Ada Boost	Gaussian process
Original dataset augmented by Markov transition field and Gramian angular field and balanced by Smote ENN	QDA	Ada Boost, Decision Tree, QDA, Nearest Neighbors	QDA	QDA

7.2 Thesis contribution

This thesis contribution is in exploring the effects of augmentation of datasets with different representations of the original dataset and investigation on the ability of classification of different classifying algorithms. In particular, Gramian Angular Plots representation and Markov Transition Fields representation were explored.

This thesis also investigates the effects of balancing the dataset with Smote ENN algorithm and its effects on the ability to classify different algorithms. Finally, the combined effect of augmenting the dataset with Gramian Angular Plots representation and Markov Transition Fields representation and SMOTE ENN algorithm were studied.

The contribution of this thesis is converting time series into images by using Markov transition fields and Gramian angular field and using MCNN as the convnet.

7.3 Direction of future work

The future direction of this work is a further investigation in many different forms of representing the same dataset and exploring the effects in classifying ability of various classifying algorithms. The possible different representations can be Cylindrical and spherical coordinate systems representation, Homogeneous coordinate system representation, Curvilinear coordinates representation, Orthogonal coordinates representation, Skew coordinates representation, Log-polar coordinate system representation, Plücker coordinates representation, Generalized coordinates representation, Canonical coordinates representation, Barycentric coordinates representation, Trilinear coordinates representation, and many other representations.

The intuition behind exploring the different representation of the dataset for various algorithms is that the calculus of algorithms may match easily with the dataset representation. This may result in better classifying metrics.

`

Appendix A

From the MIMIC II database following segments were selected to extract ICP signals

3142868,

3148126,

3160820,

3169632,

3189000,

3270954,

3270980,

3309132,

3319401,
3365681,

3453290,

3487247,

3543187,

3562822,

3624651,

3629298,

3642023,

3655233,

3656395,

3668415,

3688532,

3693937,

3700665,

3774557,

3938777

Appendix B Confusion matrices

B.1 Nearest Neighbors

ML Classification without images

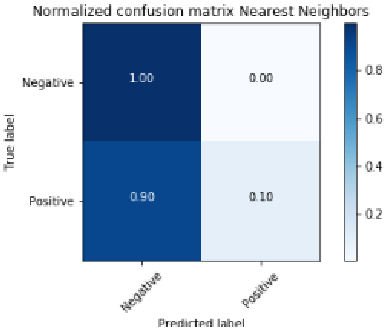


Figure B.1.1 Nearest Neighbor Normalized Confusion Matrix Of Original Dataset

ML classification with images

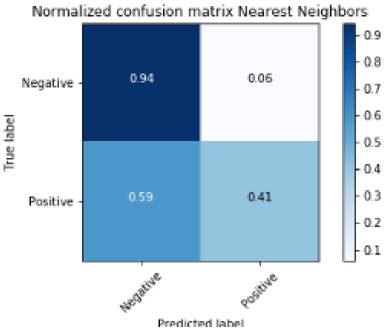


Figure B.1.2 Nearest Neighbor Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

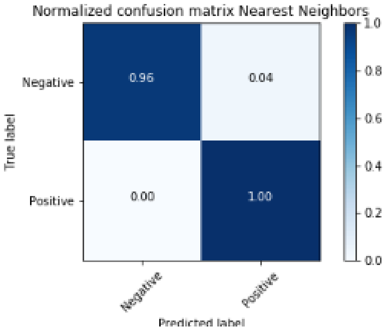


Figure B.1.3 Nearest Neighbor Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

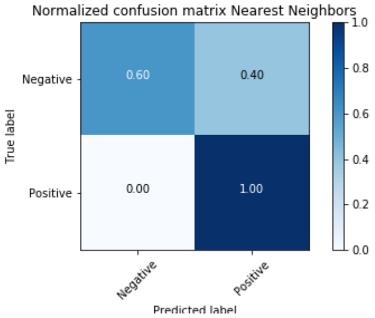


Figure B.1.4 Nearest Neighbor Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.2 Naïve Bayes

ML Classification without images

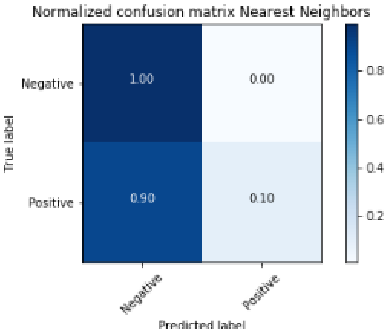


Figure B.2.1 Naïve Bayes Normalized Confusion Matrix Of Original Dataset

ML classification with images

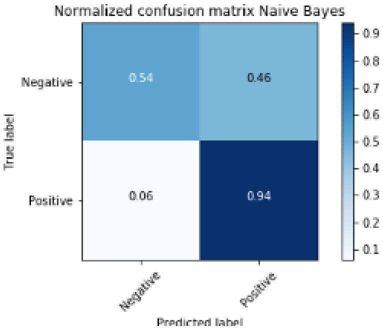


Figure B.2.2 Naïve Bayes Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

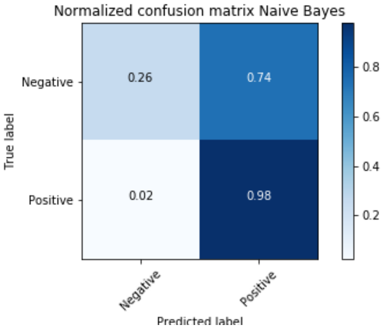


Figure B.2.3 Naïve Bayes Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

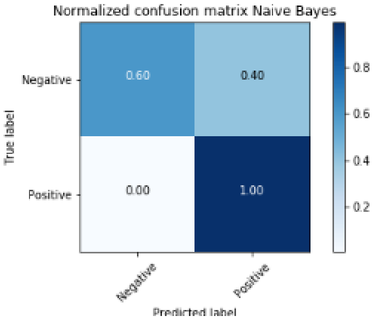


Figure B.2.4 Naïve Bayes Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.3 Gaussian process

ML Classification without images

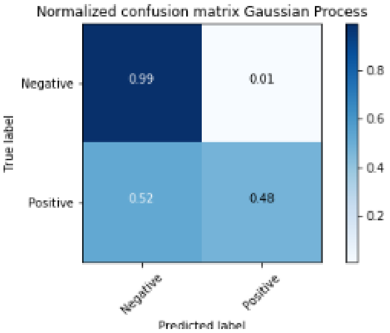


Figure B.3.1 Gaussian Process Normalized Confusion Matrix Of Original Dataset

ML classification with images

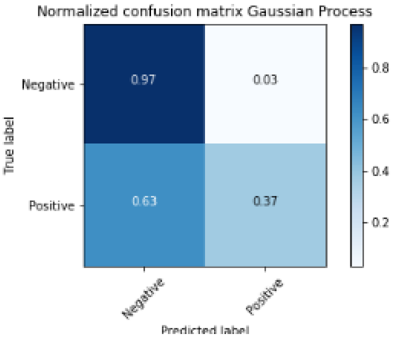


Figure B.3.2 Gaussian Process Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

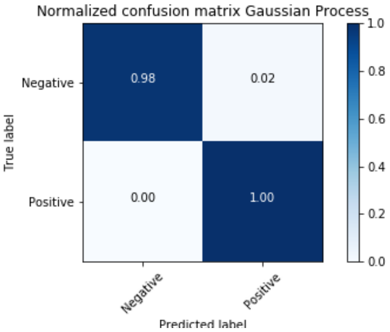


Figure B.3.3 Gaussian Process Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

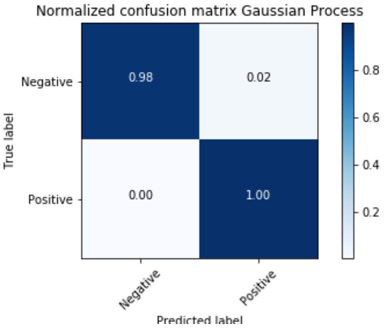


Figure B.3.4 Gaussian Process Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.4 Linear SVM

ML Classification without images

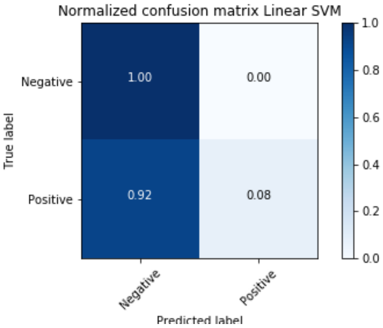


Figure B.4.1 Linear SVM Normalized Confusion Matrix Of Original Dataset

ML classification with images

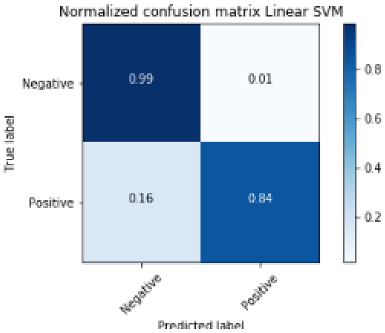


Figure B.4.2 Linear-SVM Normalized Confusion Matrix Of Original Dataset
Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

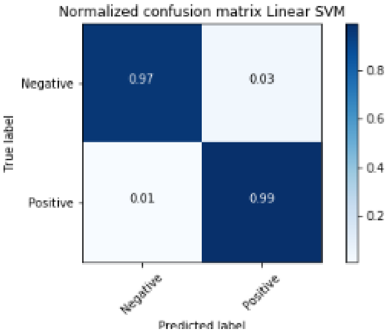


Figure B.4.3 Linear-SVM Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

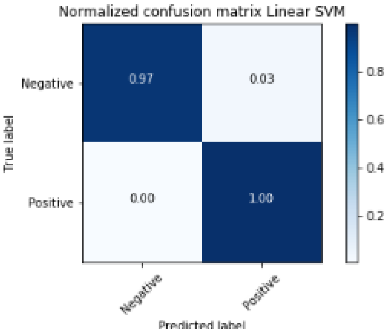


Figure B.4.4 Linear-SVM Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.5 SVM-RBF

ML Classification without images

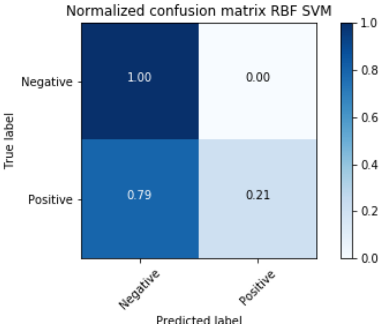


Figure B.5.1 SVM-RBF Normalized Confusion Matrix Of Original Dataset

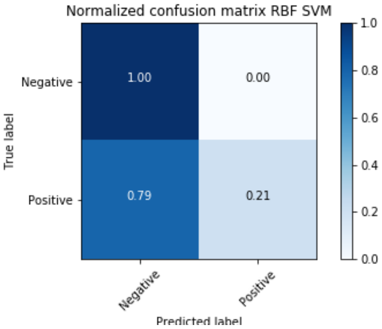


Figure B.5.2 SVM-RBF Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

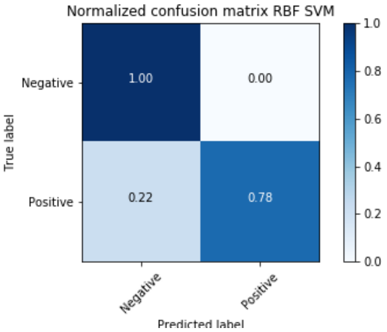


Figure B.5.3 SVM-RBF Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

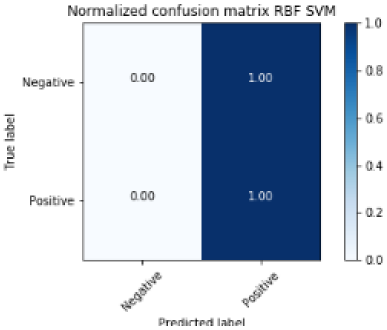


Figure B.5.4 SVM-RBF Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.6 QDA

ML Classification without images

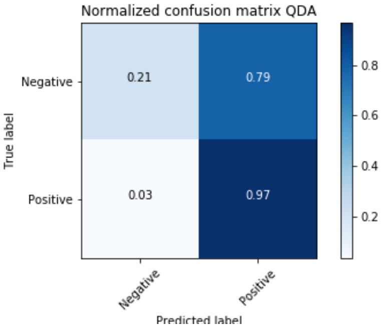


Figure B.6.1 QDA Normalized Confusion Matrix Of Original Dataset

ML classification with images

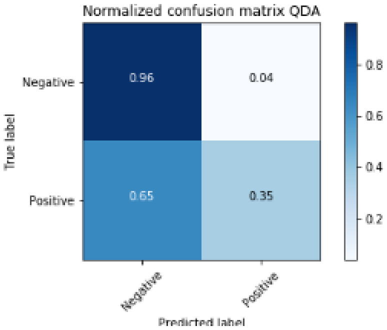


Figure B.6.2 QDA Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

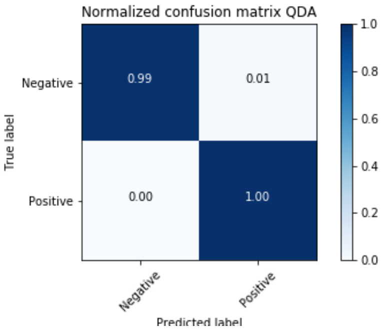


Figure B.6.3 QDA Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

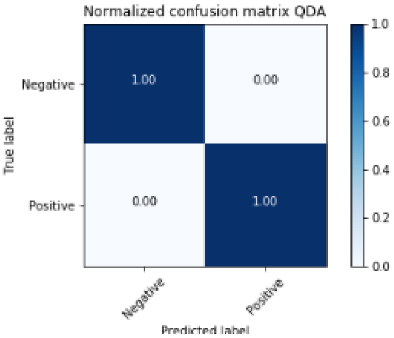


Figure B.6.4 QDA Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.7 Decision Tree

ML Classification without images

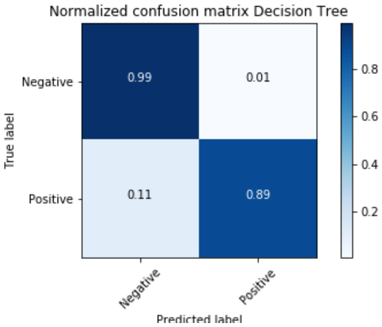


Figure B.7.1 Decision Tree Normalized Confusion Matrix Of Original Dataset

ML classification with images

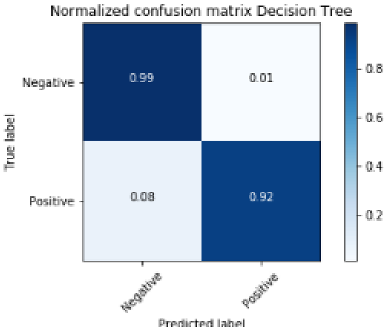


Figure B.7.2 Decision Tree Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

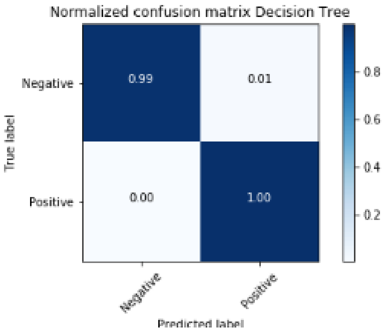


Figure B.7.3 Decision Tree Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

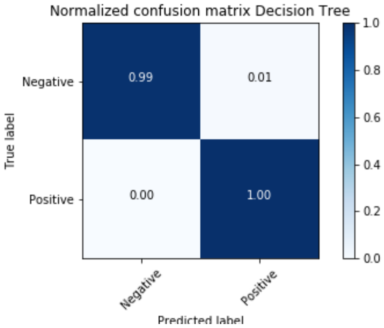


Figure B.7.4 Decision Tree Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.8 Ada Boost

ML Classification without images

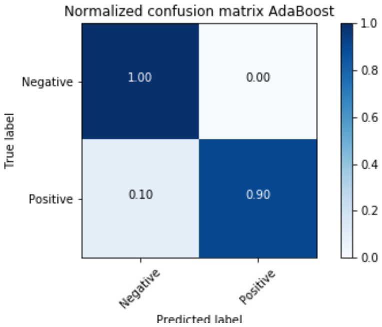


Figure B.8.1 Ada Boost Normalized Confusion Matrix Of Original Dataset

ML classification with images

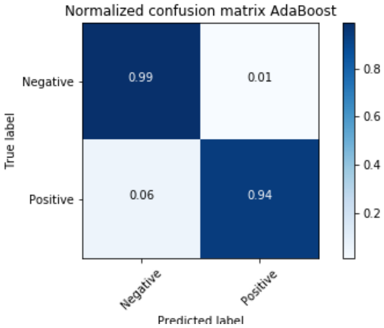


Figure B.8.2 Ada Boost Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

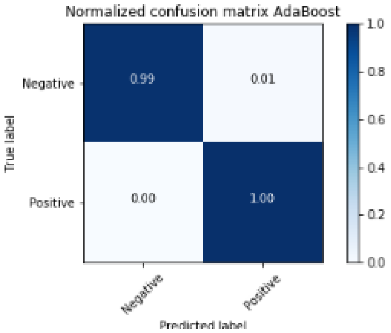


Figure B.8.3 Ada Boost Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

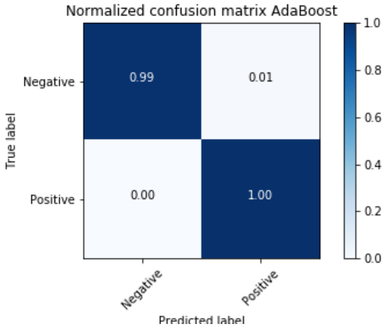


Figure B.8.4 Ada Boost Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

B.9 MCNN

ML Classification without images

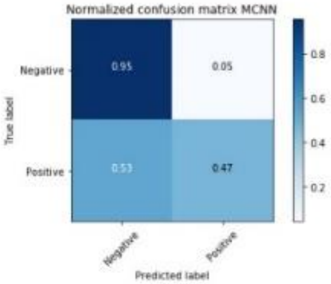


Figure B.9.1 MCNN Normalized Confusion Matrix Of Original Dataset

ML classification with images

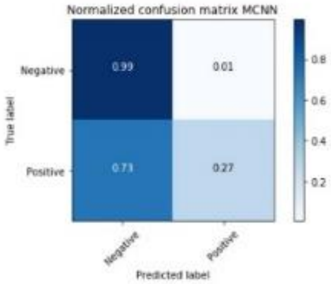


Figure B.9.2 MCNN Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images

ML Classification without images with Smote ENN

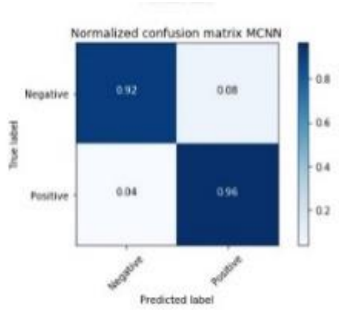


Figure B.9.3 MCNN Normalized Confusion Matrix Of Original Dataset Passed Through Smote ENN algorithm

ML Classifications with images after Smote ENN

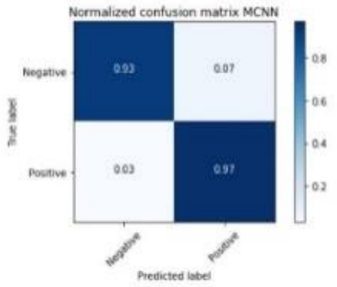


Figure B.9.4 MCNN Normalized Confusion Matrix Of Original Dataset Augmented with Markov transition fields and Gramian Angular Field images and Passed Through Smote ENN algorithm

Appendix C

Following figures shows the decision tree of the four datasets. We can observe the branching points that decision tree took for the bifurcation for each category. It seems the first bifurcation for all category begins with data point 5999, which corresponds to starting of hypertensive signal in positive examples.

- Decision tree of the dataset without images

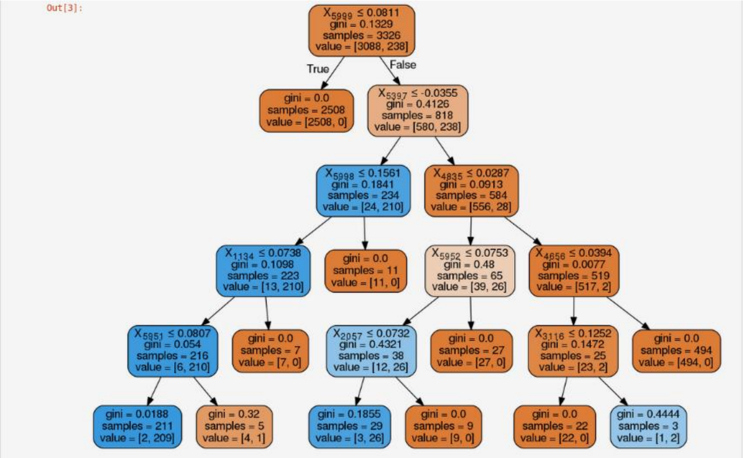

Out[3]:
$X_{5999} \le 0.0811$ gini = 0.1329 samples = 3326 value = [3088, 238]
True
False
gini = 0.0 samples = 2508 value = [2508, 0]
$X_{5397} \le -0.0355$ gini = 0.4126 samples = 818 value = [580, 238]
$X_{5998} \le 0.1561$ gini = 0.1841 samples = 234 value = [24, 210]
$X_{4835} \le 0.0287$ gini = 0.0913 samples = 584 value = [556, 28]
$X_{1134} \le 0.0738$ gini = 0.1098 samples = 223 value = [13, 210]
gini = 0.0 samples = 11 value = [11, 0]
$X_{5952} \le 0.0753$ gini = 0.48 samples = 65 value = [39, 26]
$X_{4656} \le 0.0394$ gini = 0.0077 samples = 519 value = [517, 2]
$X_{5951} \le 0.0807$ gini = 0.054 samples = 216 value = [6, 210]
gini = 0.0 samples = 7 value = [7, 0]
$X_{2057} \le 0.0732$ gini = 0.4321 samples = 38 value = [12, 26]
gini = 0.0 samples = 27 value = [27, 0]
$X_{3116} \le 0.1252$ gini = 0.1472 samples = 25 value = [23, 2]
gini = 0.0 samples = 494 value = [494, 0]
gini = 0.0188 samples = 211 value = [2, 209]
gini = 0.32 samples = 5 value = [4, 1]
gini = 0.1855 samples = 29 value = [3, 26]
gini = 0.0 samples = 9 value = [9, 0]
gini = 0.0 samples = 22 value = [22, 0]
gini = 0.4444 samples = 3 value = [1, 2]

Figure C.1 Decision tree of the dataset without images

- Decision tree of the dataset with images

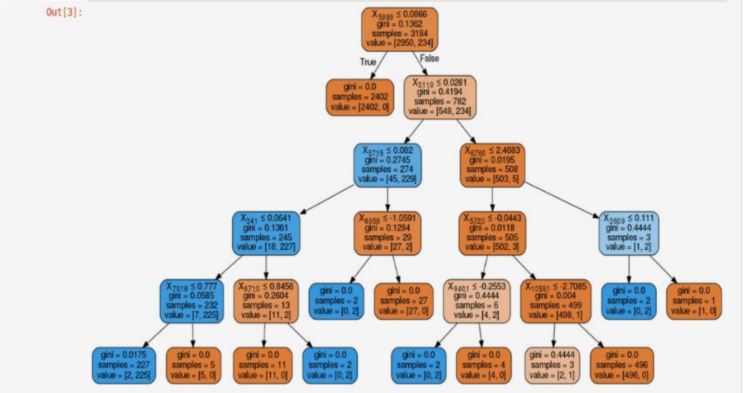

Out[3]:
$X_{5999} \le 0.0866$ gini = 0.1362 samples = 3184 value = [2950, 234]
True
False
gini = 0.0 samples = 2402 value = [2402, 0]
$X_{3119} \le 0.0281$ gini = 0.4194 samples = 782 value = [548, 234]
$X_{5715} \le 0.082$ gini = 0.2745 samples = 274 value = [45, 229]
$X_{6760} \le 2.4683$ gini = 0.0195 samples = 508 value = [503, 5]
$X_{341} \le 0.0641$ gini = 0.1361 samples = 245 value = [18, 227]
$X_{8958} \le -1.0591$ gini = 0.1284 samples = 29 value = [27, 2]
$X_{5720} \le -0.0443$ gini = 0.0118 samples = 505 value = [502, 3]
$X_{3609} \le 0.111$ gini = 0.4444 samples = 3 value = [1, 2]
$X_{7018} \le 0.777$ gini = 0.0585 samples = 232 value = [7, 225]
$X_{9710} \le 0.8456$ gini = 0.2604 samples = 13 value = [11, 2]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.0 samples = 27 value = [27, 0]
$X_{9401} \le -0.2553$ gini = 0.4444 samples = 6 value = [4, 2]
$X_{10591} \le -2.7085$ gini = 0.004 samples = 499 value = [498, 1]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0175 samples = 227 value = [2, 225]
gini = 0.0 samples = 5 value = [5, 0]
gini = 0.0 samples = 11 value = [11, 0]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.0 samples = 2 value = [0, 2]
gini = 0.0 samples = 4 value = [4, 0]
gini = 0.4444 samples = 3 value = [2, 1]
gini = 0.0 samples = 496 value = [496, 0]

Figure C.2 Decision tree of the dataset with images

- Decision tree of the dataset without images and Smote ENN

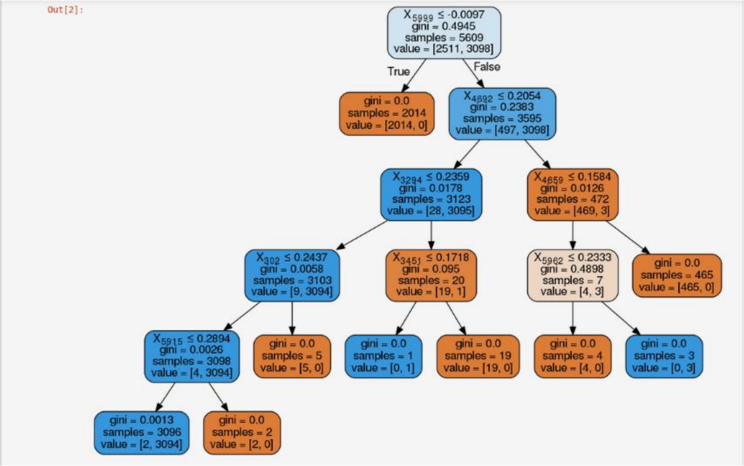


Figure C.3 Decision tree of the dataset without images and Smote-ENN

- Decision tree of the dataset with images and Smote ENN

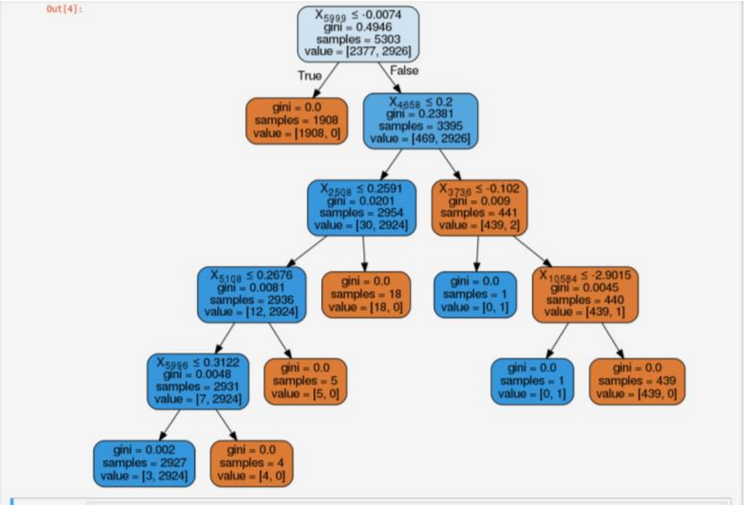


Figure C.4 Decision tree of the dataset with images and Smote-ENN

REFERENCES

[1] K. Morris, R. Forsyth, R. Parslow, R. Tasker, and C. Hawley, “UK Paediatric traumatic brain injury study group. Intracranial pressure complicating severe traumatic brain injury in children: monitoring and management,” Intensive Care Medicine, vol. 32, pp. 1606-1612, October 2006.

[2] N. Juul, G. Morris , S. Marshall, and L. Marshall, “Intracranial hypertension and cerebral perfusion pressure: influence on neurological deterioration and outcome in severe head injury,” The Executive Committee of the International Selfotel Trial. J Neurosurg, vol. 92, pp. 1-6, Jan. 2000.

[3] Z. Cui, W. Chen, and Y. Chen, “Multi-Scale convolutional neural networks for time series classification,” Computer vision and pattern recognition, vol. 4, pp. 1-10, May 2016.

[4] Z. Wang and T. Oates, “Encoding time series as images for visual inspection and classification using tiled convolutional neural networks,” in AAAI Workshops (2015), pp. 40–45.

[5] A. Marmarou, R. Anderson, J. Ward, S. Choi, and H. Young, “Impact of ICP instability and hypotension on outcome in patients with severe head trauma,” Journal of Neurosurgery, vol. 75, pp. S1-S66, Nov. 1991.

[6] S. Mayer, and J. Chong, “Critical care management of increased intracranial pressure,” SAGE Journals, vol. 17, pp. 55-67, March 2002.

[7] M. Balestreri , M. Czosnyka, D. Chatfield, L. Steiner, E. Schmidt, P. Smielewski, B. Matta, and J. Pickard, “Predictive value of Glasgow coma Scale after brain trauma: change in trend over the past ten years,” J Neurol Neurosurg Psychiatry, vol. 75, pp. 161-162, Jan 2004.

[8] M. Czosnyka and J. Pickard, “Monitoring and interpretation of intracranial pressure,” J Neurol Neurosurg Psychiatry, vol. 75, pp. 813-821, June 2004.

[9] S. Holm, and P. Eide, “The frequency domain versus time domain methods for processing of intracranial pressure (ICP) signals,” Med Eng Phys, vol. 30, pp. 164-170, March 2008.

[10] K. O'Phelan, P. Efird, K. Johnson, M. Albano, J. Beniga, D. Green, and C. Chang, “Patterns of increased intracranial pressure after severe traumatic brain injury,” Neurocrit Care, no.12028, pp. 280-286, Jan. 2009.

[11] G. Castellani, C. Zweifel, D. Kim, E. Carrera, D. Radolovich, P. Smielewski, P. Hutchinson, J. Pickard, and M. Czosnyka, “Plateau waves in head-injured patients requiring Neurocritical care,” Neurocritical Care, vol. 11, pp. 143-150, Oct. 2009.

[12] A. Seely, and P. Macklem, “Complex systems and the technology of variability analysis,” Critical Care, vol. 8, pp. R367-384. Sep. 2004.

[13] S. Mayer, J. Chong, “Critical care management of increased intracranial pressure,” Sage Journals, vol. 17, pp. 55-67, March 2002.

[14] P. Eide, “Intracranial pressure parameters in idiopathic normal pressure hydrocephalus patients treated with ventriculoperitoneal shunts,” Acta Neurochirurgica, vol. 148, pp. 21-29, Jan. 2006.

[15] P. Eide, E. Park, and J. Madsen, “Arterial blood pressure vs. intracranial pressure in normal pressure hydrocephalus,” Acta Neurologica Scandinavica, vol. 122, pp. 262-269, Sep. 2010.

[16] E. Cardoso, J. Rowan, and S. Galbraith, “Analysis of the cerebrospinal fluid pulse wave in intracranial pressure,” Journal of Neurosurgery, vol. 59, pp. 817-821, Nov. 1983.

[17] C. Shields, P. McGraw, and H. Garretson, “Accurate intracranial pressure monitoring: technical note,” Neurosurgery, vol. 14, pp. 592–593, May 1984.

[18] R. Hamilton, P. Xu, S. Asgari, M. Kasprowicz, P. Vespa, M. Bergsneider, and X. Hu, "Forecasting intracranial pressure elevation using pulse waveform morphology," 2009 Annual international conference of the IEEE engineering in medicine and biology society, vol. IEMBS.2009.5332749 , Sept. 2009, pp. 4331-4334.

[19] F. Scalzo, D. Liebeskind, and X. Hu, "Reducing false intracranial pressure alarms using morphological waveform features," IEEE Transactions on biomedical engineering, vol. 60, pp. 235-239, Jan 2013.

[20] J. Szewczykowski, J. Szukalski, K. Sliwka, G. Taracha, S. Pawlowski, and S. Sliwka, "Does "Cortisol Ejection" Disturb PSDICP During CPERT Test?," Intracranial Pressure V, pp. 269-275, 1983.

[21] J. Fan, "An Approach to Determining Intracranial Pressure Variability Capable of Predicting Decreased Intracranial Adaptive Capacity in Patient with Traumatic Brain Injury," vol. 11, pp.317-324, April 2010.

[22] B. Quachtran, R. Hamilton, and F. Scalzo, "Detection of Intracranial Hypertension using Deep Learning," 23rd International Conference on Pattern Recognition (ICPR 2016), April 2017, pp. 2491-2496.

[23] Y. Son, S. Lee, H. Kima, E. Song, H. Huh, M. Czosnyka, and D. Kima, "Automated artifact elimination of physiological signals using a deep belief network: An application for continuously measured arterial blood pressure waveforms," Information Sciences, vol. 456, pp. 145-158, August 2018.

[24] C. Nucci, P. Bonis, A. Mangiola, P. Santini, M. Sciandrone, A. Risi, and C. Anile, "Intracranial pressure wave morphological classification: automated analysis and clinical validation," Acta Neurochirurgica, Volume 158, pp. 581–588, March 2016.

[25] M. Shaw, I. Piper, and C. Hawthorne, "Multi-resolution Convolution Methodology for ICP Waveform Morphology Analysis," Intracranial Pressure and Brain Monitoring XV, pp. 41-44. 2016.

[26] Soo Y. Ji, K. Najarian, T. Huynh, and D. Jeong, "An Integration of Decision Tree and Visual Analysis to Analyze Intracranial Pressure," Neuroproteomics-Methods in Molecular Biology, vol. 1598, pp. 405-419, May 2017.

[27] S. M. Pincus, "Approximate entropy as a measure of system complexity," PNAS, vol. 88, pp. 2297-2301, March 1991.

[28] J. ZIV, and A. Abraham, "Universal Algorithm for Sequential Data Compression Lempel," IEEE Transactions on Information Theory, vol. 23, pp. 337-343, May 1977.

[29] A. Bravi, A. Longtin, and A. Seely, "Review and classification of variability analysis techniques with clinical applications," BioMedical Engineering, vol. 10, Oct. 2011.

[30] P. Naraei, N. Mohsen and A. Sadeghian, "Toward learning intracranial hypertension through physiological features: A statistical and machine learning approach," Intelligent Systems Conference (IntelliSys 2017), Sept. 2017, pp. 395-399.

[31] P. Naraei, and A. Sadeghian, "A PCA based feature reduction in intracranial hypertension analysis," IEEE 30th Canadian Conference on Electrical and Computer Engineering (IEEE CCECE 2017), pp. 1-6, May 2017.

[32] P. Naraei, M. Kenez, and A. Sadeghian, "A hybrid wavelet-based K-means clustering approach to detect intracranial hypertension," 3rd IEEE Canada International Humanitarian Technology Conference (IHTC), pp. 21-25. IEEE, 2017.

[33] E. Keogh, and M. Pazzani, “Scaling up dynamic time warping for datamining applications,” Sixth ACM SIGKDD International conference on knowledge discovery and data mining, 2000, pp. 285-289.

[34] S. Andriana, O. Campanharo, M. Sirer, R. Malmgren, F. Ramos, and L. Amaral, “Duality between time series and networks,” PLOS ONE, vol.6-e23378, August 2011.

[35] J. Lines, L. M. Davis, J. Hills, and A. Bagnall, “A shapelet transforms for time series classification,” 18th ACM SIGKDD international conference on Knowledge discovery and data mining (ACM 2012), August 2012, pp. 289–297.

[36] Y. Zheng, Q. Liu, E. Chen, J. L. Zhao, L. He, and G. Lv, “Convolutional nonlinear neighbourhood components analysis for time series classification,” Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2015):Advances in Knowledge Discovery and Data Mining, vol. LNCS 9078, May 2015, pp. 534–546.

[37] J. Claude Hemphill, Peter Andrews and Michael De Georgia, “Multimodal monitoring and neuro-critical care bioinformatics,” Nature Reviews Neurology, vol. 7, pp. 451–460 July 2011.

[38] J. Grabocka, N. Schilling, M. Wistuba, and L. Schmidt-Thieme, “Learning time-series shapelets,” Kdd ’14 20th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (ACF 2014), August 2014, pp. 392–401.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” Advances in Neural Information Processing systems, NIPS, 2012, pp. 1097-1105.

[40] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet, “A unified embedding for face recognition and clustering,” IEEE Conference on Computer Vision and Pattern Recognition, vol 1, pp. 815-823, March 2015.

[41] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, “Unsupervised feature learning for audio classification using convolutional deep belief networks,” Advances in Neural Information Processing Systems, NIPS, 2009, pp. 1096-1104.