THESIS
3
4126722

LIBRARY
Michigan State
University

This is to certify that the
dissertation entitled

ON PREDICTION AND DETECTION OF EPILEPTIC SEIZURES BY MEANS OF GENETIC PROGRAMMING ARTIFICIAL FEATURES

presented by

HIRAM ALEXER FIRPI

has been accepted towards fulfillment
of the requirements for the

Ph.D. degree in Electrical Engineering

Erik D Goodman
Major Professor's Signature

May 2, 2005
Date

MSU is an Affirmative Action/Equal Opportunity Institution

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE

2/05 c:/CIRC/DateDue.indd-p.15

ON PREDICTION AND DETECTION OF EPILEPTIC SEIZURES BY MEANS OF GENETIC PROGRAMMING ARTIFICIAL FEATURES

By

Hiram Alexer Firpi

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Department of Electrical and Computer Engineering

2005

ABSTRACT

ON PREDICTION AND DETECTION OF EPILEPTIC SEIZURES BY MEANS OF GENETIC PROGRAMMING ARTIFICIAL FEATURES

By

Hiram Alexer Firpi

This work presents a novel, general-purpose algorithm called Genetic Programming Artificial Features (GPAF), which consists of a genetic programming (GP) algorithm and a *k*-nearest neighbor classifier, and which surpasses the performance of another recently published method called Genetically Found, Neurally Computed Artificial Features for addressing similar classes of problems. Unlike conventional features, which are designed based on human knowledge, experience, and/or intuition, the artificial features (*i.e.*, features that are computer-crafted and may not have a known physical meaning) are systematically and automatically designed by a computer from data provided. In this dissertation, we apply the GPAF algorithm to one of the most puzzling brain-disorder problems: the prediction and detection of epileptic seizures. Epilepsy is a neurological condition that makes people susceptible to brief electrical disturbance in the brain thus producing a change in sensation, awareness, and/or behavior; and is characterized by recurrent seizures. It affects up to 1% of the worldwide population, or sixty million people, and 25% cannot be fully controlled by current pharmacological or surgical treatment. The possibility that an implantable device might eventually warn patients of an impending seizure is of utmost importance, allowing on-the-spot medication or safety measures.

Epileptic electroencephalographic (EEG) signals were treated from a chaos theory perspective. First, we reconstructed the EEG state-space trajectories via a delay-embedding scheme. Then these pseudo-state-space vectors were input to a genetic programming algorithm, which designed one or more (non)linear features providing an artificial space where the baseline (nonseizure data) and preictal (preseizure data, or ictal data in case of detection) classes are sufficiently separated for a classifier to achieve better accuracy than using principal components analysis, our benchmark feature extractor. The GPAF algorithm was applied to data segments extracted from 730 hours of EEG recording obtained from seven patients. The machine automatically discovered one or more patient-specific features that predicted epileptic seizures with a time horizon from one to five minutes before the unequivocal electrographic onset of each seizure. Results showed that 43 of 55 seizures were correctly predicted, for a 78.19% correct classification rate, while 55 epochs out of 59 representative of baseline conditions were classified correctly, for a low false positive rate per hour of 0.0508. In the case of detection, a low false-positive-per-hour-rate and a high detection rate were also achieved. A generic (cross-patient) model for prediction of epileptic seizures was also found, at the expense of decreased performance with an average of 69.09% sensitivity. The GPAF algorithm was additionally investigated to design seizure detectors. Evaluating 730 hours of EEG recording showed that with customized, artificially designed detectors, 83 of 86 seizures were detected. Seven previously unreported seizures were also detected in this work.

To Hiram, Elvira, Anick, Jennifer, and my wife Linda:
people with whom I have shared most of my life

Copyright by
HIRAM A. FIRPI
2005

To Hiram, Elsira, Amed, Jennette, and my wife Linés,
people with whom I have shared most of my life.

"Discovery consists of seeing what everybody
has seen and thinking what nobody has thought"

–Albert von Szent-Gyorgyi

A good feature extractor consists of "seeing" what every
method has seen and computing what no method has computed

ACKNOWLEDGMENTS

First, I want to thank Dr. Erik Goodman for his advice, dedication, and time as my advisor. Second, to Dr. Javier Echauz, who has been not just a member of the Ph.D. committee but also a friend. I want to thank him because of his dedication, motivation, and time to teach me, with lessons even delivered by phone. Also, I want to give my gratitude to Barbara O'Kelly for her help and time.

Additionally, I want to give many thanks to the Harriet G. Jenkins Predoctoral Fellowship for their financial support throughout the years to complete this work.

I want to thank my father Hiram Firpi-Valencia and my mother Elsira Cruz-Castro because of everything that they have done for me, for their unconditional support, their motivation, encouragement, and advice throughout all my life. They are my heroes and role models to follow because they have been by far the most persistent people I have ever known. Without their guidance and inspiration nothing of what I have done could have been possible. Additionally, I want to thank my brother Hiram Amed Firpi-Cruz and my sister Jennette E. Firpi-Cruz for their support and all the moments that we have shared together. Also, I want to thank my wife Linés Ortiz-Molina for her help and motivation. She has been with me even in the most difficult moments that a person can experience. I also want to thank my uncle Francisco Cruz-Castro for the adventure and exploration senses that he gave me, which in part are the spark of my life.

Finally, I want to thank Miguel A. Figueroa-Villanueva because of his time and programming contribution to this work. Also, to all those people that in same way or another contributed to make this work possible.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

ANN:	Artificial Neural Network
EEG(s):	Electroencephalograph(s)
EEGer(s):	Electroencephalographer(s)
FN:	False Negative
FP:	False Positive
FPH:	False Positives per Hour
GA:	Genetic Algorithm
GFNCAF:	Genetically Found, Neurally Computed Artificial Feature
GP:	Genetic Programming
GPAF:	Genetic Programming Artificial Feature
k-NN:	*k* Nearest Neighbor Classifier
MTLE:	Mesial Temporal Lobe Epilepsy
PCA:	Principal Components Analysis
PDF:	Probability Density Function
P_{FA}:	Probability of False Alarms
P_{FN}:	Probability of False Negatives
UCO:	Unequivocal Clinical Onset
UEO:	Unequivocal EEG (Electrographic) Onset

Chapter 1

Introduction

> *A woman wakes up and brushes her teeth. Then, she goes to the kitchen and prepares her breakfast. Later, the woman gets dressed and takes her belongings and she is ready to go to work. On her way to work, suddenly a small device starts to beep and she stops the vehicle and takes a pill: the beeping indicates to her that soon she will suffer an epileptic seizure.*

In the near future, the above hypothetical scenario is likely to be possible with the help of a new device that will be marketed to detect or predict epileptic seizures. Although epileptic seizures have been studied extensively in recent decades, many of the circumstances surrounding epileptic seizures are unknown to this day. Researchers have used several techniques to analyze electroencephalographic signals (EEGs) to develop a methodology to predict and detect such seizures. In this work, we shall present an algorithm that will create *artificial features* directly from EEG signals. In theory, these features should provide a classifier with equal or better discrimination power to predict or detect a seizure than the classical features used for such tasks.

1.1 Motivation

Since the invention of the electroencephalograph, great progress has been made in studying brain disorders. One of the most studied brain disorders is epilepsy, a neurological condition that makes people susceptible to brief electrical disturbance in the brain and that is described by a change in sensation, awareness, and/or behavior. Epilepsy is characterized by recurrent seizures. It affects up to one percent of the population of the world, or sixty million people, and 25% cannot be fully controlled by current medical or surgical treatment.

Many approaches have been proposed to extract information from EEG signals in seeking to develop algorithms to predict or detect epileptic seizures. To extract the relevant information that can facilitate such prediction or detection, features are calculated using conventional techniques and methodologies, which are time-consuming, trial-and-error processes requiring a great deal of effort from researchers.

In this manuscript, we propose a systematic methodology to automatically create *artificial features*—features that are *computer-generated* and may not have a known physical meaning—directly from EEG signals, and which we project to have similar or better discrimination power to categorize EEG signals between *preictal* or *baseline*, in the case of prediction, and between *ictal* and *baseline*, in the case of detection.

1.2 Objectives

The main objectives of this work are as follows:

1. Extend the framework of Firpi and Echauz [31] by creating artificial features directly from EEG signals. In [31], features were created using a *features-from-features* basis. Here, a *features-from-data* basis will be used.

2. Prescribe a general-purpose systematic methodology that can automatically create features that capture generalizations without the intervention of researchers performing exhaustive analysis to select features.
3. Generalize the capabilities of the Genetically Found Neurally Computed Artificial Features algorithm (GFNCAF), proposed by the author in [31], by using Genetic Programming (GP), which allows for the evolution of more complex networks, unlike GFNCAF, which used an arbitrary, fixed network.
4. Apply the GFNCAF extension, called Genetic Programming Artificial Features (GPAF) algorithm, to create artificial features directly from raw EEG data, to predict and detect epileptic seizures.

1.3 Methods

A large archive (730 hours) of digital intracranial EEG recordings obtained from patients undergoing presurgical evaluation at the Emory Epilepsy Center of Emory University in Atlanta, GA, was available for this purpose. Data were obtained as a routine procedure independent of our work. No human subject was placed at risk by or at Michigan State University. The prediction and detection problems will be cast as a dichotomous classification, in which EEG signals are fed to the GP algorithm in order to produce artificial features that can grant maximal separability between ictal and baseline EEG signals, or between preictal (before seizure) and baseline EEG signals, as possible. A library of mathematical operators (or functions) will be provided to the GP learning/classification system in order for it to develop complex artificial features.

1.4 Contributions

The contributions of this work include:

- An algorithm that can automatically generate artificial features from raw EEG signals to *predict* epileptic seizures with similar or better classification rates than those obtained through previously published methods.
- An algorithm that can automatically generate artificial features directly from raw EEG signals to *detect* epileptic seizures, similarly to or better than through previously published methods.
- The first step toward a generic model of artificial features for prediction of epileptic seizures.
- A generic model for detection of epileptic seizures that has similar or better performance to that of current methods/features.

1.5 Outline

Chapter 2 introduces the epilepsy disorder and defines some of the terminology used in the medical community. A brief overview is presented on the literature about the importance of relevant measurements to apply to the prediction and detection of epileptic seizures. A review is provided of the classical and non-classical techniques used today to extract features from EEG signals for prediction and/or detection applications. A brief overview of a related algorithm, called the Genetically Found, Neurally Computed Artificial Features algorithm, is presented. Finally, a brief explanation about genetic algorithms and genetic programming is provided.

In Chapter 3, we introduce the genetic programming artificial feature algorithm (GPAF). Later, the GPAF is tested on a problem where randomly generated vectors are classified as either parallel or nonparallel.

In Chapter 4, we present the implementation of the GPAF algorithm to design artificial features for prediction of epileptic seizures. We explain a method to reconstruct space-state trajectories, called delay embedding. Later, these reconstructed states are input to the GPAF algorithm, which constructs the predictors. Many experiments are run on the EEG data of seven patients.

In Chapter 5, we implement the algorithm to design artificial features for detection of epileptic seizures. Details about detection are explained. Detectors are tested on EEG data of seven patients. Finally, Chapter 6 concludes with benefits, limitations, and recommendations regarding the GPAF algorithm.

Chapter 2

Literature Review

This chapter first defines what epilepsy is. Also provided is a review of previous work, methodologies and techniques that have been used to predict and detect epileptic seizures. Classical signal analysis methods used in recent decades to analyze electroencephalographic signals (EEGs), such as Fourier transforms and wavelets, as well as others, will be discussed. Furthermore, an overview will be presented on related work presented in [30] and [31]. Finally, the foundation of genetic programming is laid out.

2.1 Epilepsy Overview

According to the Epilepsy Foundation of America [22], epilepsy is a neurological condition that makes people susceptible to seizures. A *seizure* is a term used to describe change in sensation, awareness, or behavior brought about by a brief electrical disturbance in the brain. Seizures vary from a momentary disruption of the senses, to short periods of unconsciousness or staring spells, to convulsions. There are many types of seizures, and some people have only one type of seizure, while others may have more than one type. Even though they look different, all seizures are caused by a sudden change in how the neurons of the brain send electrical signals to each other [22]. Epilepsy can be caused by anything that affects the brain, such as tumors or strokes. Sometimes epilepsy is inherited; sometimes it is caused by trauma, like from head injury in a car accident; however, in many cases no cause can be found for such disorder.

Today, one way to treat epilepsy is through the use of anticonvulsant medicines. These medications control seizures for many people with this disorder; however, they do not cure it. Other alternatives treat the condition by means of surgery, by a demanding ketogenic (high-fat) diet, which is principally administered to children, or by electrical stimulation of the vagus nerve, a large nerve leading into the brain, through implantation of a vagus nerve stimulator that offers partial relief [1]. From a control systems perspective, all of these treatment options are still a form of sensorless, open loop control. Regardless, the goal of any treatment is to stop seizures with as few side effects as possible.

There are some controversial findings concerning seizure prediction by animals. Some studies have suggested that dogs can predict seizures in children. In [29] and [54], Kirton *et al.* have reported that dogs could predict epileptic seizures from minutes to hours before the seizure occurred. The researchers collected the data by means of questionnaires sent to the owners who claimed their dogs could predict epileptic seizures in their children. The study reported that among 60 dogs, nine dogs were judged to be able to predict epileptic seizures with 80% accuracy (of course, this measure is difficult to state with high reliability, as it clearly depends on the interpretation of the dog's owner). The dogs were reported to let the children know that they were about to suffer a seizure by licking, whimpering, or standing next to them. (Of course, these behaviors are not unusual, and it is easy to see how the observer's biases might influence the interpretation of the data.) In previous studies, the dogs were reported to bark for no apparent reason and behaved erratically (much like other animals foreshow tsunamis, etc.). One interesting point is that those dogs were not trained to do such task. Therefore, such behavior, if present, was somehow learned by the dogs without human training. The study does not indicate how the dogs learned such "ability" or what the dogs are measuring or seeing in the child to be able to predict the seizure. However, some researchers have conjectured that these dogs can "see" or "smell" small electrical or chemical changes, using visual or olfactory or increased sweating information to predict seizures. Critics in the epilepsy medical community have pointed to the fact that the evidence collected for such claims is weak, that the dogs' bizarre behavior may in fact be what triggers some of the "predicted" seizures, and that in a large number of cases, the seizures are psychogenic (clinical fits without corresponding EEG evidence), however such dogs seem to be of great psychological value to the patients.

Since the invention of the electroencephalogram, most of the studies on epileptic seizures have been carried out using electroencephalographic (EEG) signals. Many approaches have been proposed to extract information from EEG signals that can be used to develop algorithms to predict or detect epileptic seizures (many of them are discussed in Section 2.4). Features are highly informative measures or attributes eluded from raw data can facilitate tasks such as prediction or detection. These are calculated using conventional methodologies that are time-consuming, trial-and-error processes requiring a great deal of effort from researchers. All of these techniques rely on knowledge of a feature formula or algorithm that may have been obtained from intuition, tradition, the physics of the problem, analogies to problems in other fields, etc. There is no guarantee that any of these conventional features extracts maximally relevant information from the raw data. Nevertheless, seizure prediction studies using conventional features increasingly hint at the fact that the information is there, waiting to be fully extracted. Litt *et al.*, in [60], presented evidence that mesial temporal lobe seizures are generated in a series of events that evolve over hours, leading to the clinical seizure onset. This series of events can be recorded by depth intracranial EEG. However, this work was conducted by scoring, manually, many hours of EEGs and the predictability was largely limited to what a basic energy feature could reveal about preictal changes. More recently, it has been shown that complicated feature calculations can be realized in miniaturized hardware using a cellular neural network approximation of the feature on a chip [70], [90].

However, before beginning a full discussion of previous work in the literature review, we note from the preceding paragraph the importance of selecting good measures to detect and predict epileptic seizures. As noted, the prediction/detection of epileptic seizures can be viewed from a pattern recognition point of view. The stages that involve pattern recognition are detailed in [18]. The system is summarized in three stages: preprocessing, feature extraction, and classification. Preprocessing refers to those processes or operations on the data that are performed before other stages are started. The main task of preprocessing is to improve the quality of the data/images. In an EEG analysis framework, the preprocessing stage is used to maintain the integrity of the signal and to filter out line noise using a 60 Hz notch filter. Also, a low-pass filter is used prior to digital sampling to avoid aliasing of signals.

Feature extraction is the process by which quantitative or qualitative measures are obtained that distill relevant information from raw data. A dimensionality reduction is obtained through this process by mapping the high-dimensional space of the raw data to a relatively low-dimensional space of features. However, other processes, like the feature creation process, in which new features are created from other features or directly from raw data, can also be performed in this stage. By means of feature extraction techniques, we can also reduce the number of features to be used and then select those features that improve the discrimination task. Because this stage is the main issue in this work, in Section 2.2 we cite, discuss, and extend the theory on this subject.

Finally, classification is the process by which a system categorizes an object, making use of its features. There are two types of classifiers: *supervised classifiers* and *unsupervised classifiers* (although there already are classifiers proposed that combine supervised and unsupervised techniques). Supervised classifiers are those that must be trained using labeled samples, *i.e.*, samples whose classes are known *a priori*. There are two kinds of supervised classifiers: parametric and nonparametric. Parametric classifiers are those classifiers that assume a probability density function (PDF) for the data. Among those are Gaussian and mixture model [18] classifiers. On the other hand, nonparametric classifiers do not assume a probability density function. Among those classifiers are *k*-Nearest Neighbors [76], its fuzzy version, Fuzzy *k*-Nearest Neighbor [51], Parzen-windows [18], and Genetic Classifiers [73]. Artificial Neural Networks [41] are also nonparametric classifiers, given that they do not assume a PDF, though they are considered to be hybrid methods because the parameters, called weights, must be calculated. More details about classifiers are beyond the scope of this work, however, for in-depth information, readers may refer to the references cited above.

Nevertheless, before we continue with the next section, we need to define and state clearly some terminology about epilepsy that is used in the medical community. A *preictal signal* refers to an EEG signal from a time period preceding the known occurrence of an epileptic seizure. The term *preseizure* will be used interchangeably with preictal throughout the text. An *ictal signal* is the EEG recorded during a seizure. A *baseline signal* is referred to as a normal signal (nonseizure signal), specifically excluding preictal signals. It will also be called *seizure-free* or *nonseizure* data throughout the text. *Unequivocal clinical onset* (UCO) of the seizure is when an observer can see the consequences of the seizure in the person, such as a convulsion. *Unequivocal electrographic onset* (UEO) is when the seizure onset is identified in EEG by an expert electroencephalographer.

2.2 Relevance of Features

As was stated previously, *features* are highly informative measures or attributes eluded from raw data for tasks such as classification, regression, and probability density estimation. Feature extraction have been pinpointed as the cornerstone of most proposed works in order to advance the seizure prediction and detection objectives. For instance, Figure 2.1 depicts the waveform of a signal from which we can obtain several measures such as the spectral components derived from the Fast Fourier Transform, and statistics, such as mean, variance, skewness, etc. These measurements, which make this signal

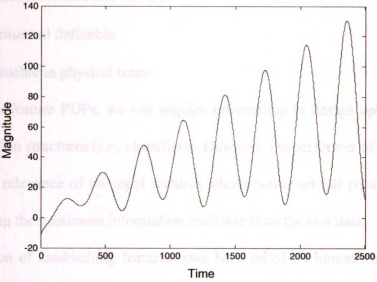


Figure 2.1 Waveform of an arbitrary signal.

different from others, are named features. Many authors have agreed that the specification of useful features (sensitive, specific, robust, computationally efficient, among others) is the most important key to machine learning with statements like: "*In general, feature extraction is the most crucial aspect of classification*" [52], *"An ideal feature extractor would yield a representation that makes the job of the classifier trivial"* [18], *"The precise choice of features is perhaps the most difficult task in pattern processing"* [65], *"the selection of variables is a key problem in pattern recognition"* [36]. In fact, according to Kil and Shin [52], a good set of features should have the following attributes:

- A large interclass mean and small intraclass variance
- Discard irrelevant variables
- Computationally inexpensive to measure

- Uncorrelated with other features
- Mathematical definable
- Explainable in physical terms

From the feature PDFs, we can acquire information to design optimal or close-to-optimal decision structures (*i.e.*, classifiers). However, the performer of the classifiers are linked to the relevance of the input features whose power set (all possible subsets) may not be carrying the maximum information available from the raw data.

The action of establishing features have been relied on humans by means of trail-and-error or previous knowledge or intuition of the physics as approaches to create them. With the exponential increasing of the computing power of processors, a new realm has been opened to explore new methods to systematize the prescription of features in a given problem. In works [8], [31], and [89], the authors took advantage of the increasing power of computers and proposed algorithms whose results defy the classical subset selection results. In these manuscripts the creation of artificial features was proposed. Artificial features are defined as features computer-generated and do not necessary have a physical meaning. By definition, artificial features possess the first five attributes mentioned above; however, the sixth attribute not necessarily has to be true. In fact, the creation artificial features is motivated by the fact that conventional features (which possesses the six attributes), when chosen arbitrarily or simply because of tradition, in situations where domain knowledge is insufficient, often fall far short of performance requirements. It is the relaxation of this last attribute that may give to the artificial features the capability to "capture" relevant information from the raw data. Soft-computing methods like evolutionary algorithms (discussed further) can be used to construct and optimize artificial features for a given problem. These artificial features are expected to resemble or surpass the classification accuracy of conventionally chosen features. The contrasting characteristics between conventional and the desired characteristics for the artificial features are provided in Table 2.1.

Table 2.1 Conventional vs. artificial features.

Conventional Features	Artificial Features
Serial	Parallel
Von Neumann architecture	Neural architecture
Programmed	Learned
Combinatorial	Inductive
Ad-hoc	Optimized
Slow computation	Fast computation
Based on intuition or sufficient domain knowledge	Data-driven

Generally, when computing features for a given problem, many of them are calculated or collected. Later, it is in the feature extraction stage where commonly the dimensionality of the feature space is reduced or the informative ones are selected by means of classical algorithms like principal components analysis, making them "top-down" approaches. In contrast, artificial features algorithms are "bottom-up" approaches, creating just the right set of features. That is, the artificial features algorithm decides the nature and the number of features that are necessary, from few to many features.

In the literature, feature extraction methods (which are also called feature creation, feature optimization, feature learning, feature discovery, feature mapping, feature augmentation, feature transformation [104], and signal or data projection [52]) have been proposed in two ways: linear feature extractors and nonlinear feature extractors. Linear feature extractors create features as a linear combination of input features (*e.g.*, principal components analysis) or create features as a linear combination of the raw inputs by means of such methods as adaptive noise filtering and time-frequency transform [52] [32] [79]. On the other hand, nonlinear feature extractors create features as a nonlinear combination of input features by means of a hidden layer in a neural network [10] or by combining them using mathematical operators [8] [30] [89] [55].

In the next section, we mention the classical measures or features used to analyze EEG signals, and related work that proposes different approaches to obtain better and more relevant (informative) features.

2.3 Conventional Measures Used in Signal Analysis

Although there exist many measures from various research fields, in this work, we recall some of the measures most used for prediction and detection of signals, not just in epilepsy applications, but also in different areas like signal processing, system identification, stochastic systems, and so on.

Arithmetic Mean: statistical measure that yields the expected value (average or first moment statistic) of a given set of data *X*. This mean is the most commonly used.

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i \,. \tag{2.1}$$

Geometric Mean: another kind of mean used to determine average factors.

$$\mu_G = \left(\prod_{i=1}^{N} x_i\right)^{1/N} . \tag{2.2}$$

Harmonic Mean: another of the various ways to calculate the average of a set of data.

$$\mu_H = \frac{N}{\sum_{i=1}^{N} \frac{1}{x_i}} . \tag{2.3}$$

Variance: measures how widely spread or distributed the data are from the mean μ.

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu)^2 . \tag{2.4}$$

Standard Deviation: the square root of the variance.

$$\sigma = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N} (x_i - \mu)^2} \,. \tag{2.5}$$

Skewness: measures the degree of asymmetry of the distribution of a set of data.

$$\gamma_1 = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{x_i - \mu}{\sigma}\right)^3 . \tag{2.6}$$

Kurtosis: measures the degree of peakedness of the distribution of a set data X (the 3 is subtracted to make it 0 for Gaussian distribution).

$$\gamma_2 = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{x_i - \mu}{\sigma}\right)^4 - 3 \,. \tag{2.7}$$

Signal Energy: measures the energy of a given signal $x(t)$.

$$E = \sum_{i=1}^{N} x_i^2 \ . \tag{2.8}$$

Energy of Wavelet Packets: the energy of wavelet-packet coefficients is derived from a five-level decomposition of signal within a sliding observation window into wavelet-packet coefficients based, *e.g.*, on the Daubechies 4 mother wavelet. The sum of squared coefficients at each terminal node of the decomposition tree gives spectral energies.

$$\Psi = \sum_{i=1}^{N} a_i^2 \ . \tag{2.9}$$

Spectral Entropy: the normalized spectral entropy is the entropy of the spectrum of the signal *x* normalized by $\log_2(\text{length}(S))$, where $S = f(x)/\Sigma(f(x))$ and $f(\cdot)$ is the power spectral density function. The normalized spectral entropy is an index from 0 to 1 almost insensitive to the signal length.

$$H = -\sum_{i=1}^{N} \frac{S_i \log_2(S_i)}{\log_2(N)} . \tag{2.10}$$

Signal Curve length: it is calculated as the sum of the lengths of the vertical line segments between successive samples of a signal. The signal curve length is simultaneously sensitive to amplitude (AM) and frequency modulation (FM).

$$L = \sum_{i=1}^{N} \left| x_i - x_{i-1} \right| . \tag{2.11}$$

2.4 Signal Analysis and Feature Extraction in Epilepsy

Various methodologies have been used for decades to analyze EEG signals to extract relevant features that give insight as to when an epileptic seizure will occur. Among the most classical methods are Fourier transform, wavelets, and chaos analyses. A brief overview of these methodologies is provided subsequently.

The brain disorder we deal with here is *mesial temporal lobe epilepsy* (MTLE). MTLE is strongly associated with complex partial seizures, the most common type of seizure, being present in 40% of all epilepsy cases reported [14]. In this section, we cite papers and manuscripts that present approaches to predict and detect MTLE seizures. We mention for each citation what problems the author(s) are dealing with.

2.4.1 Fourier Transform

Early on, when EEG signals began to be studied, Fourier transform was the most logical method applied to study such signals. Fourier transform is a technique used to represent aperiodic signals for all values of time in the frequency domain, unlike the Fourier series, which is a technique to represent periodic signals over specific values of time. This transformation gives us the frequency components, better known as the frequency spectrum, of the signal at hand. The Fourier Transform (FT) is

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-j\omega t}\, dt \,, \tag{2.12}$$

where ω is the frequency ($\omega = 2\pi f$) and $f(t)$ is the signal in the time domain. As mentioned, FT shows the spectral components of the signal at hand; and in EEG signals the frequency bands most widely studied to track the seizures are the alpha band, α, (8-13 Hz), theta band, θ, (4-8 Hz), delta band, δ, (.5-3.5 Hz) and beta band, β, (14-30 Hz, although it can reach up to a maximum of 40 Hz). Authors [103] have applied the FT to see if there are indications that can shed a clue to predict seizures. However, one of the main problems of FT is that it can show the frequency components of the signal but it cannot indicate in what instant of time these occur because time is integrated out in Equation (2.12). However, EEGs are non-stationary signals time localization is of interest.

2.4.2 Wavelet Transform

In [94], the wavelet transform has been used to detect epileptic seizures. Wavelet transform [39] is a mathematical transformation that uses scale and translation properties to decompose the input signal into its components of frequency that appear at different resolutions. One of the main properties of wavelets is that they decompose the frequency components so as to indicate in what interval of time a given component occurs. The wavelet transform is defined by the following mathematical expression:

$$\Psi_x^{\psi}(\tau,s)=\frac{1}{\sqrt{|s|}}\int x(t)\psi^{*}\left(\frac{t-\tau}{s}\right)dt\,, \tag{2.13}$$

where s and τ are the scale and translation parameters respectively, $\psi(t)$ is the transformation function, often called the mother wavelet, and $x(t)$ is the input signal. In [94], Szilágyi, Benyó, and Szilágyi used wavelets to decompose the EEG signals into their spectral components α, θ, δ, and β bands. Later, these spectral measurements were input to an artificial neural network (discussed in detail below) that was trained to output whether an EEG signal is an epileptic or a baseline (normal) wave.

In [9], Chen, Zhong, and Yao used wavelet-transformed features to extract or detect the characteristic values (the authors refer to these values as singular values) of the sharp waves and spikes, which indicate a clinical onset, and which are embedded in EEG signals. The authors defined a wavelet that permits reconstruction of the high frequency components of the EEG and detection of the spikes by taking those as singular values and using local maxima of their wavelet transform modulus.

In [63], where Mehta, Koser, and Venziale are also interested in detecting ictal events, they proposed an automated method to detect ictal processes based on wavelet decomposition. In their publication, the authors claimed that the average power spectrum of the EEG of a resting, healthy person follows power law attenuation with a scale-invariant property over a range of clinically interesting frequencies, suggesting a hidden scale-invariant property. This statistical scale-invariant property was monitored, and when appreciable changes in the statistical measurement were detected, an onset of seizure could be determined. These statistical changes were represented using wavelets with which coefficients measured the variance of progression over different scales, thus, reflecting changes of scale-invariant signals.

2.4.3 Fractal Dimension

Fractal dimension [28] is a concept that represents the space-filling ability of an irregular geometric object of interest with an infinite nesting of structure (*e.g.* line segment, square, etc.). While the familiar dimensions, such as topological dimension, are specified as integer values, a fractal dimension may be a fractional value. The fractal dimension is defined by the following equation in the limit as *r* vanishes:

$$FD = \frac{\log(N)}{\log(r)}, \qquad (2.14)$$

where N is the number or bulk measure of self-similar pieces and r is the magnification factor.

One area where fractal dimension is usefully applied is analysis of time series. In [26] and [27] the application of FD to predict and detect the clinical onset of seizures was proposed as a real-time tool. In their work, the authors used the Katz method to calculate the FD of the EEG waveforms. They used baseline signals as well as ictal data to test the techniques. Esteller *et al.* [25], determined that using the Katz approach to FD was a good way to detect onset of seizures. On the other hand, in [25], Esteller *et al.* proposed an even more efficient measure to detect seizure onsets. The feature, called *curve length* or *line length*, is less computationally expensive. In their work, the authors showed that, by processing the EEG signals with the curve length feature, they were able to detect seizure onsets with a delay of 4.1 seconds, with 0.051 false positives per hour and one false negative out of 111 seizures analyzed in over 1,215 hours of intracranial EEG data.

In [53], Kirlangic *et al.* also proposed the use of FD as a feature for an adaptive EEG segmentation algorithm (AEEGS) in epilepsy, instead of using Fourier transforms (FT) and Fast Fourier Transform used previously in the AEEGS algorithm.

2.4.4 Artificial Neural Networks

Artificial neural networks (ANNs) [41] are networks (*i.e.* equations) that can do a mapping of input to output with high levels of complexity. This complexity depends on the number of layers and neurons that the network contains. ANNs can perform many of the complex tasks that are present in real-world application problems, like nonlinear classification, function approximation, and control of dynamic systems, among others.

ANNs have also been used to classify or detect epileptic EEG signals; however, they have been used more to classify EEG states rather than to create features. In [67], the authors used ANNs to classify EEG signals between normal (baseline) and ictal signals. The authors used, as feature inputs, classical statistics of signals, such as zero-crossing, variance, integral of absolute value, parameters extracted from amplitude histogram, and power spectrum values from the four different spectral components of EEGs.

In [72], Özdamar *et al.* used a variant of neural networks, called adaptive resonance theory neural network (ARTNN), for floating point values. This network is an unsupervised self-organizing system that can cluster data into different classes; such networks are suitable for input patterns that cannot be defined by a static input. In this work, the ARTNN should be able to detect the transient patterns of EEG signals and classify them into their respective classes. When a signal that cannot be classified is input, the algorithm stores the signal in a database for future classification. Since the network is an unsupervised technique, an external trainer was used to label output from the network that classifies the output between spikes or non-spikes. EEG raw data were the input to the network.

Another algorithm to detect epileptic waveforms was proposed in [74], in which Park *et al.* used a wavelet transform, a neural network, and an expert system to design a multiple channel system to detect epileptiform signals. The EEG signals (raw data) were preprocessed by the wavelet transform (WT) to reduce the number of inputs to the neural network, thus the WT works as the feature extractor. Later, once the preprocessed data are input to the neural network, the ANN performs signal processing and accentuates the epileptic spikes. The network was trained, using the backpropagation method, to output a threshold value that is input to an expert system, a database of 16 *if-then* rules that captures the knowledge and experience of EEGers, and takes this input along with the inputs of neighboring channels (surrounding electrodes) to make a decision whether the spikes at hand are epileptic or not. The authors reported up to an average of 5.5 false detections per hour.

Bigan [4] presented an automatic system to identify EEG waveforms that use an ANN, whose inputs (features) are measurements of frequency changes in the EEG signals during an interval of time. Additionally, classical measurements such as magnitude and power spectrum analysis were also calculated from EEGs and those were input, along with the ANN output, to create a final decision regarding seizure onset. Other works using variants of artificial neural networks to predict or model epileptic seizures have been proposed in [6], [7], and [19].

2.4.5 Lyapunov Exponents

Lyapunov exponents [68] (LE) are standard measures to determine whether a system is chaotic. LE measures the average rate of convergence or divergence of nearby trajectories. The LE is defined as

$$\lambda = \lim_{\substack{t \to \infty \\ |\Delta x_0| \to 0}} \frac{1}{t} \ln \frac{|\Delta x(X_0, t)|}{|\Delta x_0|}, \tag{2.15}$$

where $\Delta x(x_0, t)$ is the radius of an ellipsoid evolving in time. LE can also be viewed as a measure of how easy it is to perform prediction in a given system: positive exponents indicate chaoticity and thus unpredictability.

Iasemidis *et al.* [45] proposed the design of a mesial temporal lobe epileptic seizure prediction implantable system using a version of LE, which they called short-term maximum LE. Using LE to measure the rate of divergence in the brain is just an approximation of the LE, given that the brain is a highly nonstationary system. The authors claimed to have obtained good results, predicting 82% of all cases of seizures using EEG data from different patients. The authors also claimed to have predicted seizures with an average of 71.1 min. before the ictal onset. Controversy over these results remains, since at the reported high FPH levels, random chance cannot be ruled out.

2.4.6 Fuzzy Logic

Fuzzy logic [102] is a technique that offers some of the linguistic ambiguity that exists in human language (*e.g.*, low, medium, high). This property makes fuzzy logic suitable to deal with ambiguous and noisy data. In fuzzy logic, rules are stated using multivalent *if-then* statements, *i.e.*, if an event occurs to some degree, then its respective output is fired to some other degree. For this reason, Harikumar and Sabarish [42] proposed the use of fuzzy logic to classify the risk level of epileptic seizures. In this work, the authors calculated five features: signal energy, number of positive and negative spikes exceeding a threshold, number of sharp waves, total number of spikes and sharp waves in neighboring channels, and variance. These were then input to the fuzzy logic system and three outputs per channel were calculated indicating the seizure risk. The output was expressed with five fuzzy sets: normal, low risk, medium risk, high risk, and very high risk. Inputs were also represented with similar statements. With these inputs, rules could be constructed to design the fuzzy logic system. A post-processing of the fuzzy logic output was carried out to obtain better classification. The authors obtained up to 80% correct classification of the onset of seizures.

2.4.7 Cellular Neural Networks

A cellular neural network (CNN) [11] [12] is a recurrent nonlinear network in which neurons are locally connected and the dynamics are identical for each node. Commonly, these neurons are called cells. CNNs of one or two dimensions are the most commonly used.

Each cell in the CNN contains an input, an internal state, and an output. The state equation for cell C_{ij} is defined as follows:

$$\dot{x}_{ij} = -x_{ij} + \sum_{mn\in N_{ij}} A_{mn} y_{mn} + \sum_{mn\in N_{ij}} B_{mn} u_{mn} + z_{ij}\,, \tag{2.16}$$

where $x_{ij} \in \mathbf{R}$ is the state, $y_{ij} \in \mathbf{R}$ is the output, $u_{ij} \in \mathbf{R}$ is the input, and $z_{ij} \in \mathbf{R}$ is the bias. A_{mn}, the feedback weights, and B_{mn}, the feedforward weights, are connected from cell C_{mn} to C_{ij}. N_{ij} is the 1-neighborhood of C_{ij}, where C_{mn} is also a member ($C_{mn} \in N_{ij}$).

The output equation is described by the following piecewise linear equation:

$$y_{ij} \equiv f\left(x_{ij}\right) = \frac{1}{2}\left(\left|x_{ij}+1\right| - \left|x_{ij}-1\right|\right). \tag{2.17}$$

Tezlaff *et al.*, in [97], proposed the use of CNN to anticipate or predict epileptic seizures, first studying the patterns that the CNN outputs when the onset of a seizure was occurring vs. not occurring, in a software simulation environment. Then the authors implemented the CNN parameters (*i.e.*, control and feedback weights) in a Universal Machine CNN (UM-CNN), a CNN computer. The authors used this to state the viability of a portable device to predict epileptic seizures. We note that the feature extraction here is performed by the CNN. Similar work was presented in [96], except that this time they used a discrete time-delay CNN, a variant in the universe of CNNs.

2.4.8 Genetic Algorithm

The genetic algorithm (GA) is a global search, general-purpose optimization algorithm in which, by means of so-called genetic operators, many possible solutions are evolved in an attempt to locate the best solution. Because our approach involves evolutionary algorithms (*i.e.*, GA or genetic programming), we leave further details for the next section to explain those algorithms. However, the definition provided for GA gives an intuitive idea to understand the papers that will be discussed next.

In [15], [16], and [17], unlike in previous work mentioned where a few features were used as inputs, the EEG signals were processed by multiple levels of feature extraction, starting with six features: curve length, signal energy, nonlinear energy, spectral entropy, and energy of wavelet packets. Later, signals processed by those features were analyzed visually and processed into a second stage of features, such as maximum spikes, minimum spikes, kurtosis, variance, and skewness, among others. These processed EEG signals were again processed in a third stage of feature extraction, where similar features to those calculated in the second stage were used. After this stage, a GA was used to look for the optimal set of features that would be input to a classifier, which would decide if the EEG signal is a baseline (normal) or a preictal signal. Also, the GA would select which channels (*i.e.*, electrodes) would be used for the study. A probabilistic neural network (PNN) was used as a classifier. Experiments were run using data from six different patients. For each patient, a different set of features was the optimal set to perform the best classification. The results suggested that a specific predictor is necessary for every patient.

2.4.9 Additional Work

Although several approaches have been proposed to address epilepsy prediction and detection problems, in this work we do not discuss all of them in detail. Briefly, other approaches have been proposed in [101], where nonlinear dynamic systems theory was used to detect seizures. Another scheme was proposed in [99], a method for feature extraction using an extension of singular value decomposition (SVD) to detect epileptic seizures. Other researchers have tried to establish an analytical model for epilepsy, as was the case in [69].

2.5 Genetically Found, Neurally Computed Artificial Features

Regarding the approaches discussed in the previous section, classical features are selected manually, *i.e.*, by the expert or researcher who is studying the problem. Those procedures are what we term *conventional features-from-data-based*. On the other hand, in the work of D'Alessandro *et al.*, a higher level of intelligence was added to select the optimal set by using a GA; however, the set of features used were still conventional features, and so analysis and study by the researcher were required.

In [31], Firpi and Echauz proposed an approach called *genetically found, neurally computed artificial features* (GFNCAF). In their work, the GFNCAF algorithm was used to create an artificial feature from EEG signals that were processed using conventional features: wavelet packets, skewness, and curve length. This procedure is termed *artificial-features-from-conventional-feature-based.* The GFNCAF algorithm is composed of a genetic algorithm, a neural network and a classifier, all connected in cascade. Figure 2.2 depicts the GFNCAF algorithm. Briefly, it works as follows: the conventional features (*i.e.*, processed EEG signals), denoted as $\boldsymbol{x}_i$ in Figure 2.2, are input to the neural network, which in [31], for convenience, is implemented as an algebraic network (called so because it uses only elementary mathematical operators). The operators, denoted in Figure 2.2 as o_i, on each node of the network are selected by the GA to create the artificial feature. Later, the classifier receives the output from the network, denoted as y_i, and evaluates the training or testing data to decide whether the EEG signal was preictal vs. baseline, in the case of prediction, or baseline vs. ictal, in the case of detection. For this work, the authors implemented a *k*-nearest neighbor classifier.

The GFNCAF algorithm was used to create artificial features that can help to predict a seizure 10 minutes before it occurs. In fact, the algorithm was able to find features that have given the classifier more discrimination power than using the three original classical features. Similar results were also obtained when GFNCAF was used to find artificial

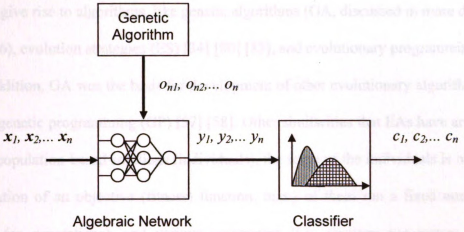


Figure 2.2 Genetically found, neurally computed artificial feature algorithm block diagram.

features that could *detect* seizures with equal or better performance than by using the original three features.

In the conclusion, the authors touched upon some points that could improve the GFNCAF algorithm. Among them, they indicated that more flexibility needed to be allowed to the neural network, and not just to the inputs, in order for the GFNCAF algorithm to advance from creating simple features to creating complex artificial features that may give more discrimination power among the classes.

2.6 Evolutionary Algorithms

Evolutionary Algorithms (EAs) are a family of algorithms that draw analogies from natural evolution. Some of the evolutionary processes mimicked by such algorithms include: reproduction, selection (or survival of the fittest), mutation, and genetic crossover. In conjunction with a fitness function evaluation, combinations of any of these processes give rise to algorithms, like genetic algorithms (GA, discussed in more detail in Section 2.6), evolution strategies (ES) [34] [80] [85], and evolutionary programming (EP) [35]. In addition, GA was the basis for development of other evolutionary algorithms, for example, genetic programming (GP) [57] [58]. Other similarities that EAs have are: most maintain population-based solutions (individuals), the score of the individuals is obtained via evaluation of an objective (fitness) function, many of them run a fixed number of iterations (or generations), and various parameters (*i.e.*, *strategy parameters*, which define the behavior of the EA—for example, mutation rate, crossover probability, selection probability, size of the population, and number of generations) must be set.

2.7 Genetic Algorithms

The *genetic algorithm* (GA) [38] [40] [43] [66], introduced by Holland in the late 1960s [44], is a general-purpose global search optimization method that imitates a small subset of the processes of natural evolution. The main properties of GA are its operators: *selection*, *crossover*, and *mutation*. There are two primary types of GA: binary and real-valued. In a binary GA, solutions, also called *individuals or chromosomes*, are represented as strings or vectors of binary values.

Because most real-world problems deal with numbers (integer or real), a mapping function is often needed to map a binary string to a set of numbers, formally called a *phenotype*. These phenotypes are evaluated using an objective or *fitness function*, which is an equation or algorithm used by the GA to assign a cost or value to each evaluated solution. This score is called the *fitness* of the solution. Fitness functions are either to be maximized or minimized, depending on the formulation of the problem. Another kind of GA, a real-valued GA, uses a continuum of real values (or, in fact, finite-length, floating-point representations of such numbers) to represent the parameters of the problem under evaluation. A brief overview is shown about the GA operators mentioned previously.

Selection

The selection operator is the process by which a set of individuals, also known as an *intermediate population*, is picked from a population using a criterion based on the fitness of the individual. There are several selection types. One of the most popular types is fitness proportional selection, often called roulette wheel selection. This selection type relates the probability of selection of an individual in a linear fashion to its relative fitness in the population. This means that individuals with a better fitness have a proportionally higher probability to be selected. Another commonly used selection method is *tournament selection*, which closely emulates the mating competition in natural selection. In this approach, a small set of chromosomes is randomly selected, and then the chromosome among them with the best fitness is selected. These tournaments are repeated as many times as parents are needed.

Crossover

Crossover or recombination is the operator by which the GA creates new individuals, called *offspring*, which are combinations of (usually pairs of) the parents selected using the previous operator, (*i.e.*, the intermediate population) and that will belong to the new population. *One-point crossover* randomly picks a crossover point between two loci in the bit strings and exchanges the bit strings after this point. *Two-point crossover* is similar to the former, but instead picks two positions on the string and exchanges the values at the loci between them. In a real-valued GA, we can use recombination similarly to its use in a binary GA. However, other approaches have also been studied for recombination, such as a blending method, blend crossover (BLX-α) and quadratic crossover, among others [2] [24].

Mutation

The main idea behind a mutation operator is to introduce diversity into the population. In a binary GA, mutation is quite simple: invert the bit value (negating, *i.e.*, 0 to 1 or vice-versa) at each locus, based on a probability called the mutation rate. For a real-valued GA, although there are different types of mutation [64], one of the most simple is *uniform mutation*. The number of genes that will be mutated is derived by multiplying the dimension (*i.e.*, length) of the individual by the mutation rate and the population size. This determines the number of genes randomly selected on the chromosome. Finally, those genes that were selected for mutation are replaced with new uniform-randomly generated values.

The canonical GA is implemented as follows (Figure 2.3 shows a flowchart of the GA):

1. The problem to be solved is cast as the maximization/minimization of a fitness function that indicates the desirability of a potential solution.
2. A population of candidate solutions is initialized randomly, and is subject to certain constraints.

3. Each chromosome is evaluated using the fitness or cost function and assigned a fitness score.
4. Each chromosome is assigned a probability of reproduction in each iteration, better called a *generation*. This probability is assigned using a selection operator that generally depends upon the fitness distribution of individuals.
5. During each generation, a new population is actively evolved by applying the operations of selection, crossover, and mutation. Typically, the algorithm is terminated upon reaching a predefined maximum number of generations or a level of fitness desired.

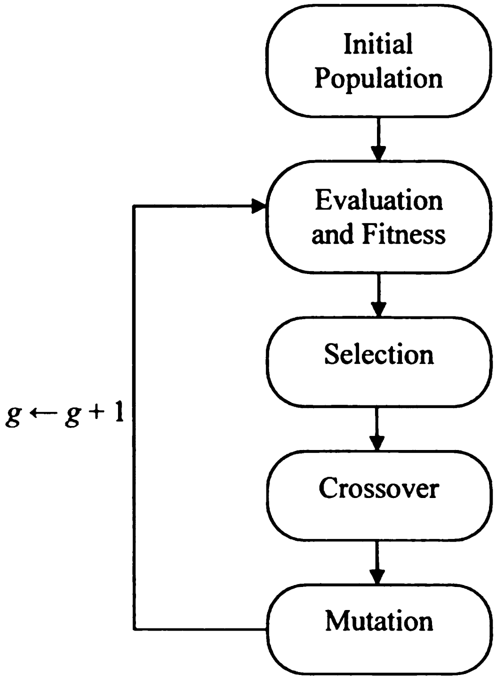


Figure 2.3 Steps of a canonical genetic algorithm (GA). The variable g denotes the number of generations.

2.8 Genetic Programming

Genetic programming (GP) [57], established formally by Koza [56], is closely related to GA, although it has a few critical differences. In GP, the length of the chromosome is variable and the representation is formulated using trees instead of the strings used in GA. Another important aspect of GP, unlike GA, in which chromosomes often directly encode the solution to the problem, is that the tree provides a program or solver that is used to solve the problem, or instructions for how to construct a solution. In GP, there are two types of nodes: *functions*, which have some number of arguments that they operate on, and *terminals*. Figure 2.4 depicts an example of a GP program (function), which illustrates the functions {cos, log, $(\ )^2$, +, ×} and terminals $\{v_1,v_2\}$ and encodes the function $\cos(v_1)(\log(v_2) + (v_1)^2)$.

Given that in the GP algorithm, the result is a program rather than a solution, the evaluation to determine the performance of the resultant program is somewhat different than for a GA. In GP, a set of input data is provided to the program (*i.e.*, the program is executed in some environment) and the output data of the program are evaluated in order to know the performance of the program. For GP problems in which the performance is partially determined by the environment, each program should be evaluated using many samples as input to determine the fitness of the tree.

As in GA, in GP, genetic operators: selection, crossover, and mutation, are used as the engine to create new programs and add diversity in the population. Selection in GP is typically implemented using the same operators as in GA, such as tournament selection

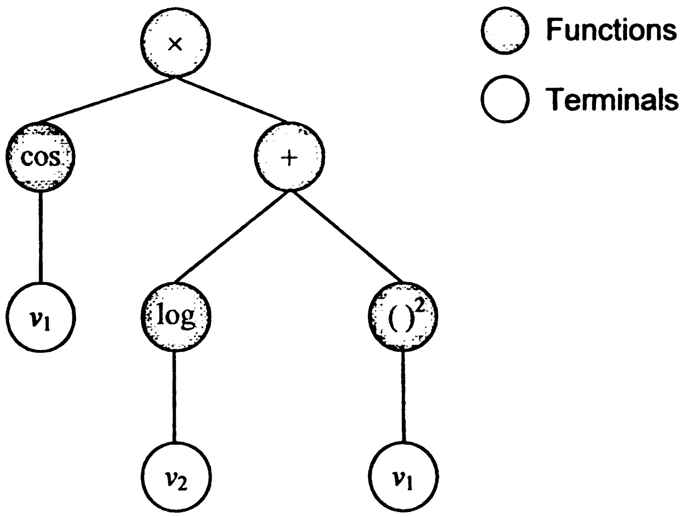


Figure 2.4 A tree diagram of a GP chromosome. The dark circles indicate the function nodes and the light circles indicate the terminal nodes.

and fitness proportional selection (see Section 2.6), although other selection routines have also been proposed, such as fitness-overselection.

Among crossover routines for GP are subtree exchange crossover, self-crossover, module crossover, and context-preserving crossover. Crossover, in GP, typically creates two different trees from two parent trees by selecting random points in each tree and swapping the subtrees below them. Figure 2.5 shows an illustration of crossover in a GP context. There are many mutation routines similar to those used in GA, and others specifically designed for GP. Some of these are: point mutation, permutation, hoist, collapse subtree mutation, and gene duplication [3], among others. Figure 2.6 illustrates a typical form of mutation in a GP context. The iteration of the GP algorithm is similar to those steps presented for GA in Figure 2.3.

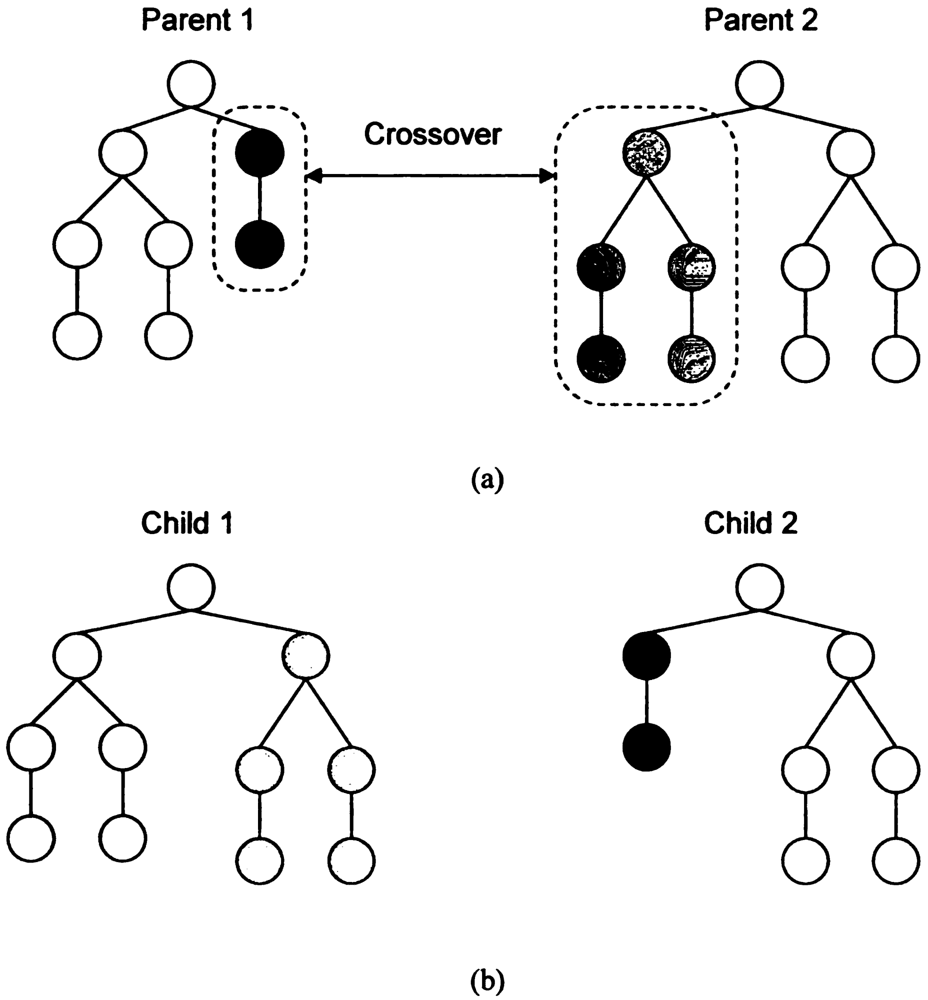


Figure 2.5 Illustration of a typical crossover operator in GP. (a) The parents, indicating the subtrees that will be swapped. (b) The results (children) after crossover is performed.

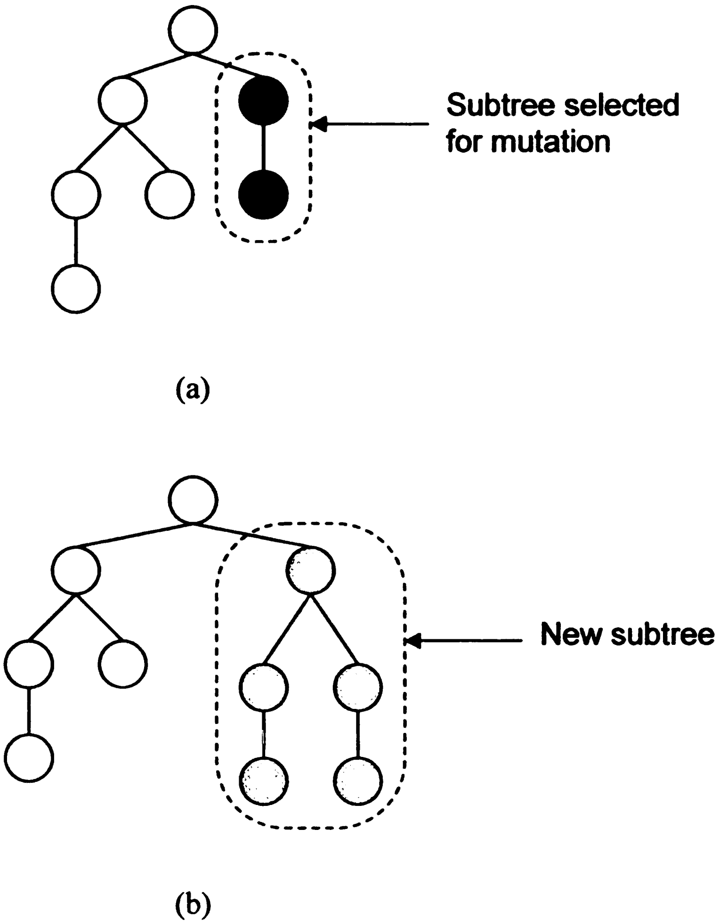


Figure 2.6 Illustration of the mutation operator in a GP framework. (a) The individual, indicating the subtree that will be mutated. (b) The result after mutation is performed.

Chapter 3

Genetic Programming to Create Artificial Features

In this chapter we discuss the limitations of the GFNCAF algorithm and introduce and discuss the genetic programming-derived artificial feature (GPAF) algorithm, which is the proposed solution to overcome the limitations encountered in the GFNCAF approach. Later, we apply the GPAF algorithm in a demonstration problem where the algorithm should learn the concept of parallelism, creating an equation able to determine whether or not two randomly generated vectors are parallel. A comparison with GFNCAF results on this problem is provided.

3.1 GFNCAF Algorithm Limitations

The previous chapter discussed the genetically found, neurally computed artificial feature algorithm. This algorithm was intended to create artificial features—that is, computer-crafted features—that have the same or greater relevance as the classical features used to create the new one, and therefore, to give the classifier more power to distinguish among classes. However, the GFNCAF algorithm still has various limitations that were noted by Firpi in [30]. The limitations are summarized as follows:

- The algorithm starts creating features from a limited library of features, rather than creating the features directly from the data under study

- The unary and *n*-ary operator set was limited to a set of only a few mathematical operations
- The neural network used in their work was based on an arbitrarily chosen, fixed-architecture network (number of nodes determined *a priori*), in which the only topological connections that could be connected or disconnected were the inputs
- The topological structure of the network does not evolve, limiting the space of possible artificial features (solutions)

The authors suggested how to extend the capability and performance of the algorithm to give more relevance and power to the new features. Those points are:

- Allow arbitrary topological connections in the neural network to be active or inactive when creating new artificial features
- Let the evolutionary part of the algorithm develop its own neural network structure and control the number of unary/*n*-ary layers, nodes, and connections
- Allow more complex unary and *n*-ary operators in the neural network nodes (*e.g.*, trigonometric and transcendental operations)
- Allow the algorithm to start creating features directly from data under study, rather than using a limited library of pre-defined features

In the next section, we discuss an approach to implementing some of the above ideas.

3.2 Genetic Programming Artificial Features

The limitations described in the first and second bullets above regarding the GFNCAF algorithm can be overcome by using GP. The typical tree representation of GP programs can be seen as an algebraic network, or vice versa, from a GFNCAF point of view. Similarities and differences between the GFNCAF and the GP algorithms were noted in [30]. After depth and size parameters for the tree are set up, the GP can evolve any tree structure allowed by its syntax rules, to create simple features (functions) as well as complex features, unlike in the GFNCAF algorithm, in which the algebraic network (replaced by a tree diagram in the GP framework) is a fixed (although previous knowledge could also be added to a network when it is being designed). Its topological connections are fixed, and the only topological connections able to be on or off are the inputs. GPs have been used to create artificial features previously. Similar approaches appeared in [55] [89], where the authors used GP to create features (although they refer to this process as feature extraction) that provide the classifier better performance than using the original features.

Because GP can use any type of function with verified closure (*i.e.*, the function should be defined for any argument the system can present it), there are many functions available, such as trigonometric, polynomial, and transcendental functions. This richness helps to address the limitation stated in the third bullet of the previous section. Figure 3.1 shows the block diagram of the *genetic programming artificial features algorithm* (GPAF). The genetic programming produces the program (function) that creates the artificial feature. The program receives as input (*i.e.*, as a terminal in the GP tree) the data of interest (input signal) and the output is the result of this artificial feature. The signals input to the GP are not limited to only one; many inputs can be provided, which would result in a GP program with multiple terminals. Data processed by the artificial feature is

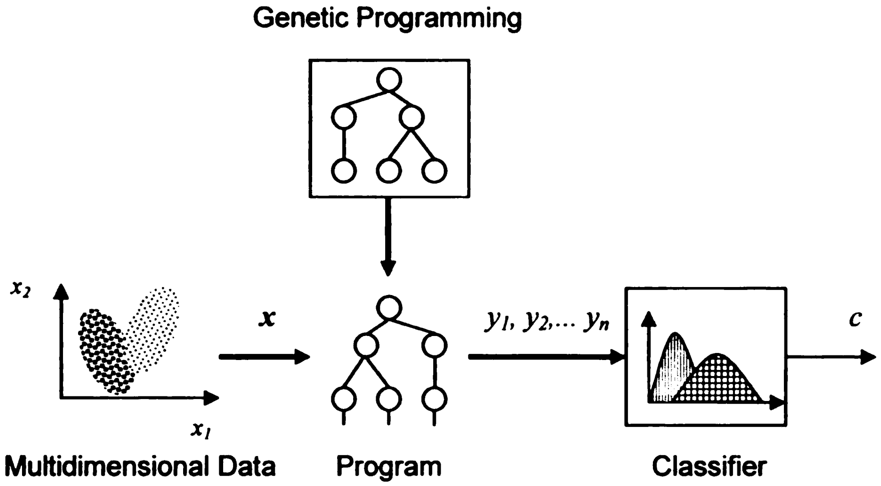


Figure 3.1 Diagram of the GP-based artificial features algorithm. The input data (signal) are processed by the program generated by GP. Later, the output is used to train the classifier.

used to train a classifier, which is being used as a supportive component to evaluate the performance of the GP program or artificial feature. In addition, testing data must be processed by the artificial feature, in order to be evaluated by the classifier. The classifier can be a parametric or nonparametric classifier, such as maximum likelihood or *k*-nearest neighbor classifiers, among others. Because one of the contributions of this work is to prescribe a general-purpose algorithm that automatically creates artificial features from conventional features or raw data, we selected the *k*-nearest neighbor classifier because is a nonparametric, nonlinear classifier capable to produce multiple thresholds and an easy trainable procedure. The error or misclassification of samples in the classifier is the measure used by the GP for the fitness or cost function.

3.3 Comparing GPAF and GFNCAF Algorithms

In [30], the feasibility of the GFNCAF algorithm was tested with a problem in trying to learn the concept of parallelism from a set of randomly generated vectors. In the present work, we use this same problem to show clearly the implementation of the GP-based artificial features. Also, the results obtained are compared with those obtained in [30].

In the vector parallelism learning problem, it is desired to find an efficient method of testing whether two given vectors are parallel (to within some error tolerance) or not. A simple analytic geometric equation that precisely measures whether two vectors are parallel is the absolute cosine formula

$$|\cos(\theta)| = \frac{|(\Delta x_1, \Delta y_1)\cdot(\Delta x_2, \Delta y_2)|}{\|(\Delta x_1, \Delta y_1)\|\|(\Delta x_2, \Delta y_2)\|} = \frac{|\Delta x_1 \Delta x_2 + \Delta y_1 \Delta y_2|}{\sqrt{(\Delta x_1{}^2 + \Delta y_1{}^2)(\Delta x_2{}^2 + \Delta y_2{}^2)}}, \quad (3.1)$$

where (Δx_1, Δy_1) and (Δx_2, Δy_2) are increments of each component for two given vectors (the vectors' tails need not originate from the point (0,0) on the plane) . These data are relevant features. If the result of $|\cos(\theta)|$ is equal to one, the vectors are parallel; otherwise the vectors are nonparallel. This equation can be seen as a nonlinear, reduction transformation given that the equation is reducing the feature space (dimensionality) from four features to one feature.

In a first experiment, we use the GPAF algorithm to find an artificial feature that has similar or equal performance to the absolute cosine equation, using as possible terminals any of the increment features, *i.e.*, Δx_1, $\Delta y_{1,}$ Δx_2, and/or Δy_2. Later, in a second experiment, we input to the GPAF algorithm the terminal sets (Δx_1, Δy_1, x_1, y_1) and (Δx_2,

Δy_2, x_2, y_2), where (x_1, y_1) and (x_2, y_2) are the tail coordinates of two vectors at hand. It should be noted that these initial points of a vector do not in themselves provide relevant information to determine whether two vectors are parallel. Therefore, it is expected that the GPAF algorithm will realize which features are irrelevant and disregard them, *i.e.*, it will not use them in designing the artificial feature. The functions available for the GP are $\{+, -, \times, \div, \mathrm{I}, |\ |, (\ )^2, \sqrt{\ }\}$, the same set of mathematical operators used in experiments with the GFNCAF algorithm. Because we are providing conventional (original) features to the algorithm to create an artificial one, this problem is what we term the artificial-feature-from-classical-feature-based approach.

To measure the performance of a GP program (artificial feature), we use Equation (3.2):

$$P_{error} = \frac{\sum_{i=1}^{N} |y_i - \hat{y}_i|}{N}, \tag{3.2}$$

where P_{error} is the percent of misclassification of samples, y_i is the output of the classifier that value of 0 or 1, $\hat{y}_i$ is the desired output, i is the sample index, and N is the total number of samples evaluated..

In these experiments, for the classifier stage (see Figure 3.1), we use the same classifier used in [30], *k*-nearest neighbor with Euclidean distance as metric, to be able to compare the results of the GPAF algorithm, presented here, with those reported by Firpi and Echauz using the GFNCAF algorithm. The constant *k* is set to three. Again, the output of the classifier is one if the vectors are parallel, and zero if they are not.

3.4 Results and Comparison

Similar to the data generated by Firpi and Echauz, in these experiments, 128 parallel and 128 nonparallel vector pairs were generated for training while another set of equal size was generated for testing. The vectors had randomly generated increments in a range of (–5, 5) and initial points in a range between (–10, 10). Table 3.1 shows the classification accuracies of all sets of test vectors. The number of neighbors, *k*, was set 5.

From the table, it can be observed that by using the *k*-NN classifier with 4 increment features as inputs, we can obtain up to 75% of classifications correct. Also, as was expected, when four features are processed by Equation (3.1) (*i.e.*, absolute cosine formula), the classifier categorizes with 100% accuracy.

Table 3.1 Results of the classifier using the indicated feature inputs.

Feature Inputs	Input to Classifier	Correct percent
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2$	$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2$	75%
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, x_1, y_1, x_2, y_2$	$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, x_1, y_1, x_2, y_2$	55%
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2$	1 absolute cosine feature	100%
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2$	1 GP feature (Eq. (3.3))	100%
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2$	1 GFNC feature (Eq. (3.4))	100%
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, x_1, y_1, x_2, y_2$	1 GP feature (Eq. (3.5))	100%
$\Delta x_1, \Delta y_1, \Delta x_2, \Delta y_2, x_1, y_1, x_2, y_2$	1 GFNC feature (Eq. (3.6))	84.2%

When four relevant features were processed by the GP feature, a 100% correct classification was obtained. Figure 3.2 depicts the tree diagram for the GP artificial feature. From this tree, a GP artificial feature designed by the GPAF algorithm, Equation (3.3) can be obtained. It can be noted that the GP algorithm examines every input feature even when it is able to disregard any feature input.

$$\left|\frac{\Delta x_2|\Delta x_1||\Delta y_1|\sqrt{|\Delta x_1|}}{||\Delta x_1|\Delta x_2\Delta y_2|-|\Delta y_1|\Delta x_2{}^2}\right|-\Delta x_2{}^2-\Delta x_2{}^2 . \tag{3.3}$$

For comparison, Equation (3.4) shows the formula that was found by the GFNCAF algorithm with four relevant features in [30]. The interested reader is referred to the indicated source.

$$\sqrt{\left(\sqrt{\frac{\Delta x_1}{\Delta x_2}}-\left(\sqrt{\Delta y_1}-\Delta y_2\right)^2\right)^2-\sqrt{\left(\Delta x_1-\Delta x_2{}^2\right)^2\cdot\left(\Delta y_1{}^2-\Delta y_2{}^2\right)}}. \tag{3.4}$$

It can be noted that the equations are different; however, both of them still meet the requirement to classify random vectors correctly.

The evolution of fitness, number of nodes, and number of levels (depth) through generations are shown in Figure 3.3. This figure portrays the accuracy of the program (artificial feature) versus its complexity.

In the second experiment, running the GPAF algorithm with eight possible terminal values, a classification of 100% was obtained. The tree diagram is shown in Figure 3.5. From the tree diagram, the following analytical expression can be obtained:

$$\sqrt{||y_1+\Delta x_2|\Delta x_2|}\left(\Delta y_1\Delta x_2-\Delta x_1\Delta y_2\right). \tag{3.5}$$

All four relevant features appear in the equation. Also, an irrelevant feature, y_1 (initial point), was included by the algorithm. This variable can be considered noise; nonetheless, the artificial feature achieves perfect classification at the expense of added computational effort. It is likely that a bigger sampling of random vectors where y_1 would have been represented throughout the whole plane would have made this variable's irrelevance more obvious to the algorithm. Figure 3.4 presents the accuracy versus complexity graph of this experiment.

Again, for comparison purposes, the expression found by the GFNCAF algorithm for the same problem (*i.e.*, with eight feature inputs) is stated in Equation (3.6). We note GFNCAF was also confused by a non-essential feature (y_2) in this case.

$$\left(\Delta x_1^2 - \Delta x_2^2\right)\Delta y_2^4 y_2^2 . \tag{3.6}$$

Table 3.1 shows that this equation achieved an 84.2% classification of correct vectors. Although GFNCAF algorithm could not find an artificial feature that performed perfect classification, the tendency of the algorithm to pay attention to the relevant features and disregard those that are irrelevant—the same tendency noticed in the GPAF algorithm—is apparent.

The GP used for these experiments included simple crossover and mutation operators with variable probability of occurrence [87]. To avoid bloat (*i.e.*, unwarranted growth of tree size until reaching the maximum depth), a dynamic depth technique was used [88]. The number of generations was set to 100 and the population size was 50 individuals. Lexicographic parsimony pressure [61] was the selection method used. The initial population was created using the Ramped Half-and-Half method [57] with size variation.

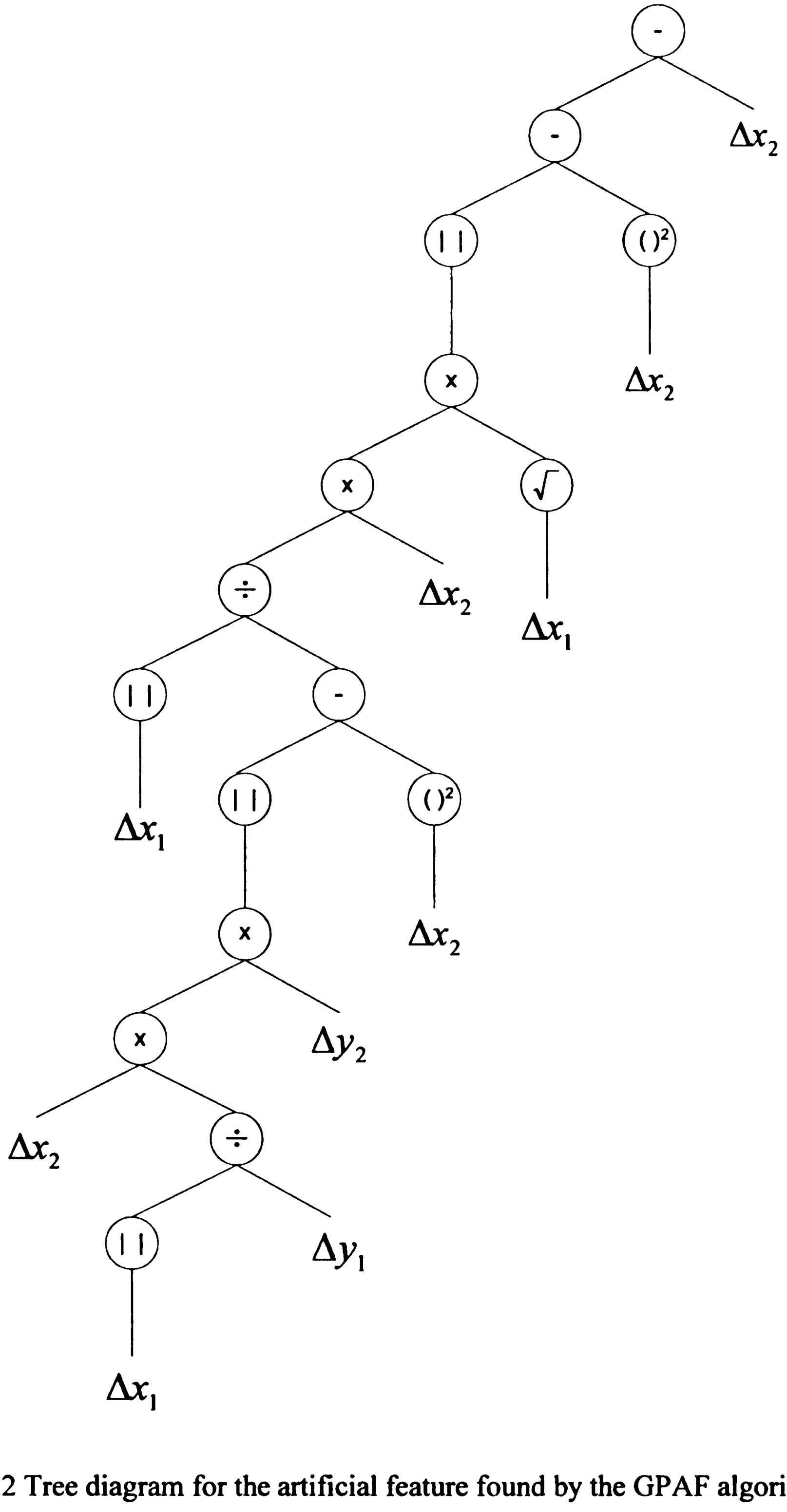


Figure 3.2 Tree diagram for the artificial feature found by the GPAF algorithm.

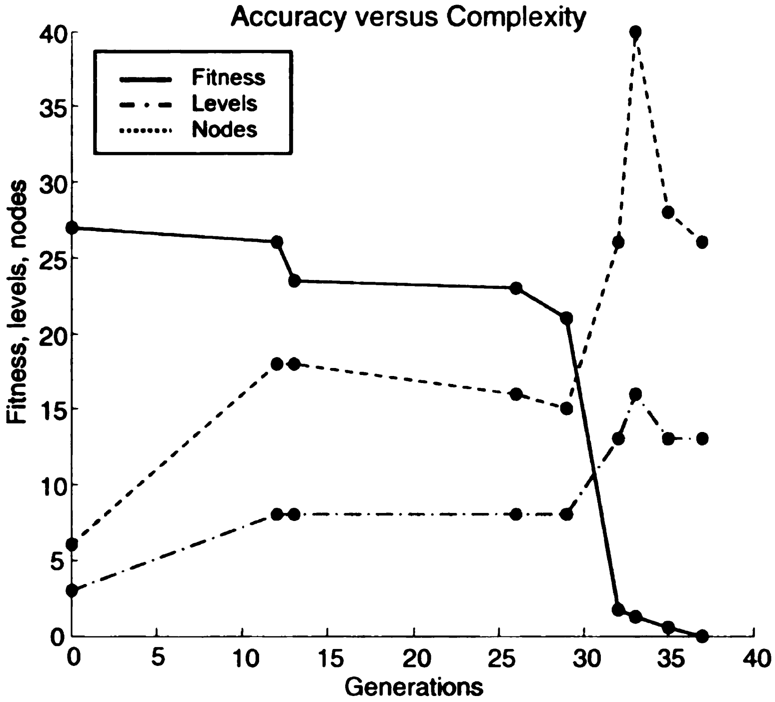


Figure 3.3 Evolution of fitness, levels (depth), and nodes for GP feature with 4 inputs.

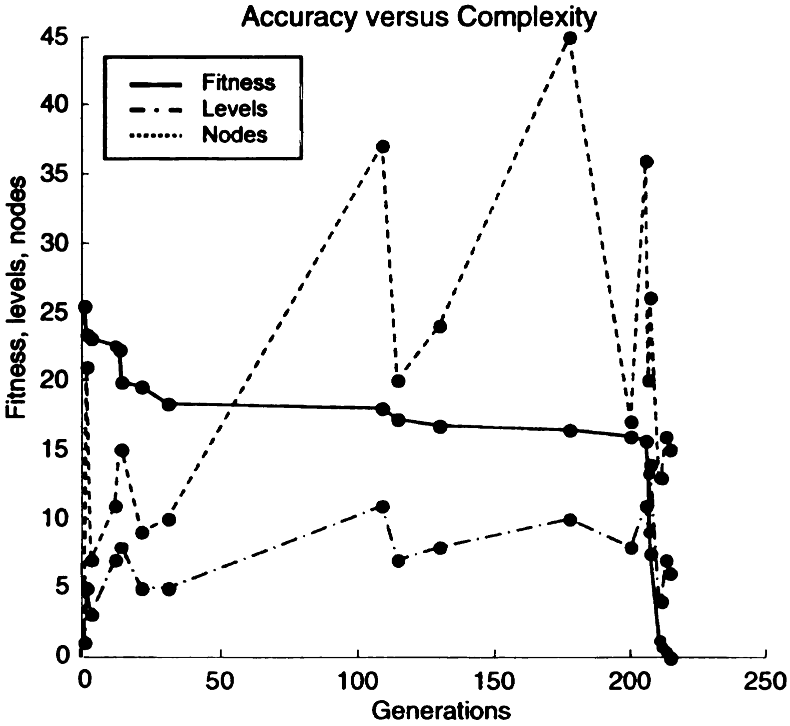


Figure 3.4 Evolution of fitness, levels (depth), and nodes for GP feature with 8 inputs.

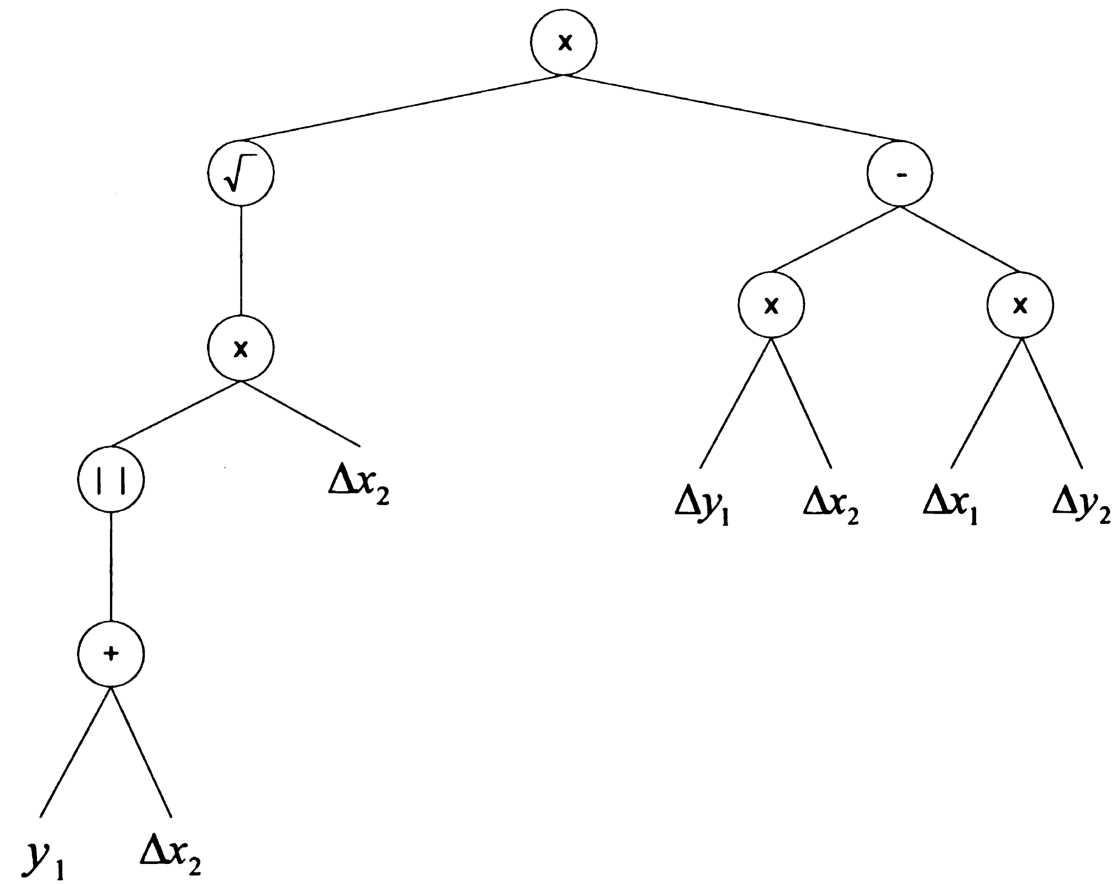


Figure 3.5 Tree diagram for the artificial feature with 8 feature inputs.

3.5 Summary

In this chapter, we introduced, discussed, and tested the feasibility of the GP-based artificial features (GPAF) algorithm to create artificial features for a test problem in which a function was sought that could help categorize a set of random vectors as parallel vs. nonparallel. The problem used an artificial-feature-from-classical-feature-based approach. The algorithm found artificial features that process original features, giving a resultant feature that later was input to the classifier. Then a *k*-NN classifier was used, classifying the inputs into their respective classes. A comparison of results with another work, the genetically found, neurally computed artificial feature (GFNCAF) algorithm [30], was presented. From results, it can be seen that both GFNCAF and GPAF are able to create highly complex, nontrivial, nonlinear, transformations that could increase the separability among the classes—in this case, between parallel and nonparallel vectors—helping the classifier to make a better decision when classifying the data. However, the properties of the GP algorithm (*i.e.*, just indicating to the algorithm the maximum depth and the number of nodes, and the algorithm takes care of the feature's topology) make the GPAF algorithm more flexible to create simpler as well as more complex features. For this reason, sometimes the parsimony of the GPAF algorithm's results may be smaller. For instance, in Equation (3.5), the GPAF algorithm created an equation that contains more elements (*i.e.*, inputs and operators) than that created by the GFNCAF algorithm (Equation (3.6)) for the same experiment, but the GPAF-designed feature achieved perfect classification. The GP algorithm cannot ensure that it will find results that achieve the intended objectives with the maximum parsimony possible in a limited-time framework (in fact, no evolutionary algorithm can), but only that at least a suboptimal solution will be found. Because the GP algorithm is the search engine of the GPAF algorithm, the previous statement also applies to it.

On the other hand, with the inclusion of the GP algorithm to replace the evolutionary component and the algebraic network, the first and second recommendations bulleted in Section 3.1 to improve the GFNCAF algorithm were addressed and implemented. The third and fourth bullets were not implemented because the degree of complexity of the problem in this chapter (*i.e.*, categorization of parallel and nonparallel vectors) did not require improvements for such limitations. However, in the next chapter, such limitations will be addressed for the prediction and detection of epileptic seizures.

Chapter 4

GP Artificial Features Applied to Seizure Prediction

In this chapter, we explain the application of the GPAF algorithm to epileptic seizure prediction. The next section shows describes how the electroencephalographic signals (EEGs) used in this research were obtained. The EEGs were intracranial voltage signals measured in patients with mesial temporal (middle of lateral brain lobes) epilepsy. We also present some concepts from nonlinear dynamic systems theory, concerning state-reconstruction via a delay-embedding method, and provide an example of using such a method. Later, we merge nonlinear dynamic systems concepts with the GPAF algorithm. Additionally, we define the benchmark classifier that will be compared in performance with those designed by the GPAF algorithm. The following sections contain the mathematical development of the performance metrics for the GP version. Also, we present the function and terminal sets chosen for the GP algorithm. Results for various patients are presented in the last section.

4.1 Clinical Context

The anonymized intracranial EEG data archive used for these experiments was obtained from a tripartite database from Georgia Institute of Technology, Emory University, and University of Pennsylvania. The EEG data were recorded from epileptic patients undergoing a pre-surgical evaluation. According to the Epilepsy Foundation of America [23], of every 100 new cases of epilepsy reported, one patient is a candidate for surgery and 70% percent of those are cured by removing a part of the affected region in the brain. To perform these evaluations, recordings were obtained by surgically implanting scalp electrodes or intracranial electrodes deep inside the brain. Intracranial electrodes are preferred over scalp electrodes because EEG signals are recorded practically clear of undesired artifacts created by physiological activities like heartbeat, chewing, moving, blinking, etc. The patients suffered complex partial seizures, seizures that arose focally and caused of loss of consciousness. Intracranial recordings allow neurologists to identify more accurately the foci, where the seizures originated. Identification of the correct foci was not possible using scalp electrodes. For each patient, a six-contact depth electrode was placed in each temporal lobe throughout a hole drilled in the back of the skull (occipital region) as depicted in the picture in Figure 4.1. To know the coordinates at which the electrodes should be placed, high-resolution MRI images from the patient were used.

Epileptologists visually reviewed all the EEG recordings and marked each seizure by first identifying seizure activity on the EEG and later moving backward in time to determine where the earliest EEG changes from baseline to epileptic seizure activity occurred in the patient. Second, the unequivocal electrographic onset (UEO) was identified. This is the point in time that marks the onset of seizures [82] without any doubt that a seizure is occurring. Additionally, for each patient, intracranial EEG archive, synchronized video, and medical records were revised. Other relevant clinical aspects

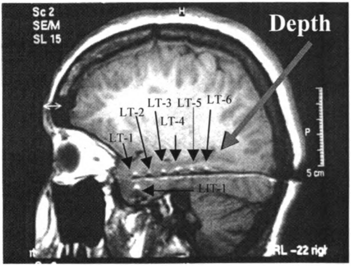


Figure 4.1 Electrodes placed in the hole drilled in the back area of the skull.

like clinical activity, sleep-cycle times, and times and doses of medication administered during the hospital stay were also recorded.

4.1.1 Data Acquisition Equipment and Process

Continuous IEEG and video of patients were collected on a digital, 64-channel, 12-bit Nicolet BMS-5000 epilepsy monitoring system. The EEG signal was downloaded from tape to CD-ROM. EEGs were digitized at 200 Hz, and recorded after filtering through a bandpass of 0.1–100 Hz. A digital 60-Hz notch filter was also employed to suppress line noise. Hospital stays varied from 3 to 14 days. For this work, 66 CDs totaling several gigabytes of raw data were available. Figure 4.2 shows the electroencephalographic signal for a case in which an epileptic seizure will not occur immediately thereafter and Figure 4.3 shows the electroencephalographic signal shortly before an epileptic seizure. The durations of these signals are 10 minutes.

4.1.2 EEG Data Description

As was mentioned previously, the anonymized intracranial EEG data archive used for these experiments was obtained mostly from Emory University School of Medicine. From this archive, we have available EEG data for seven patients. There are baseline signals as well as (pre)ictal signals. Table 4.1 shows the number of tapes that are seizure-free signals and the number of tapes that contain one or more seizures available for each patient.

Table 4.1 Number of baseline and seizure tapes available.

Patient	No. of Baselines	No. of Seizures
A	2	4
B	2	4
C	1	8
D	14	6
E	20	5
F	3	6
G	11	8

The durations of the EEG recordings vary from 38 hours of recording for patient D up to 275 hours of recording available for patient G. Each EEG recording tape lasts approximately eight hours (although a few are shorter).

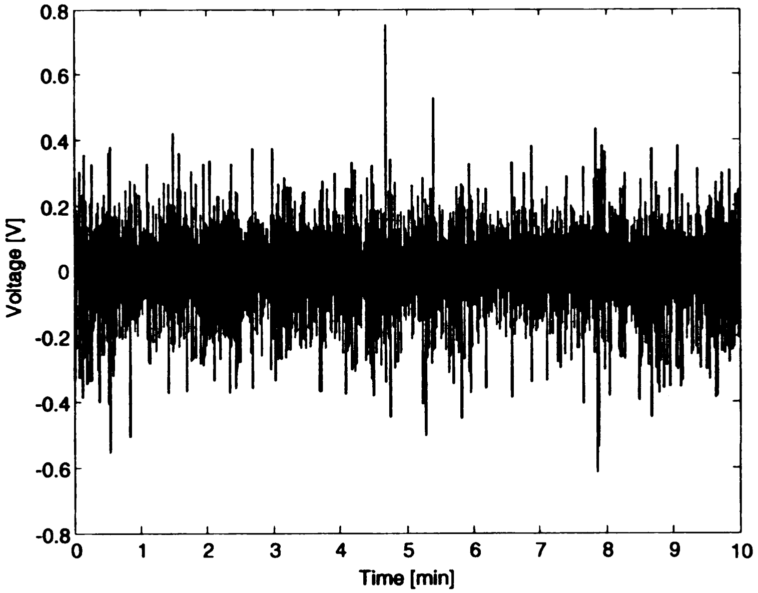
Because during the EEG signal recording procedure much interference, such as 60 Hz line voltage, blackouts, and other forms of noise can be introduced, the EEG recordings are not pure or clean. To remove some of these artifacts, a preprocessing step is carried out. Even after that, many of those artifacts pass through the filters. For this reason, if we design an algorithm to predict seizures, it may happen that the algorithm triggers (indicates a seizure) because of one of those artifacts and not because a seizure will actually occur. Given this situation, if all the available EEG hours were evaluated, we would need to analyze and study all the recordings and verify one-by-one all places throughout the EEG signal where the algorithm triggered to indicate a seizure in order to see whether the algorithm is responding to valid EEG data or, instead, to a recognizable artifact. This verification would be tedious beyond practical limits.

For this reason, baseline and preictal data segments of ten-minute duration were extracted from the whole EEG data set for each patient for training and validation purposes. The extracted EEG data segments are a statistical sample that represents the whole EEG archive of a patient. These segments were analyzed "by eye" to be sure that they were clean and that no recognizable artifacts were present. As a general rule, we extracted as many baseline epochs as there were epileptic seizures in the patient's EEG record; however, in some cases, a few more baseline epochs were extracted.

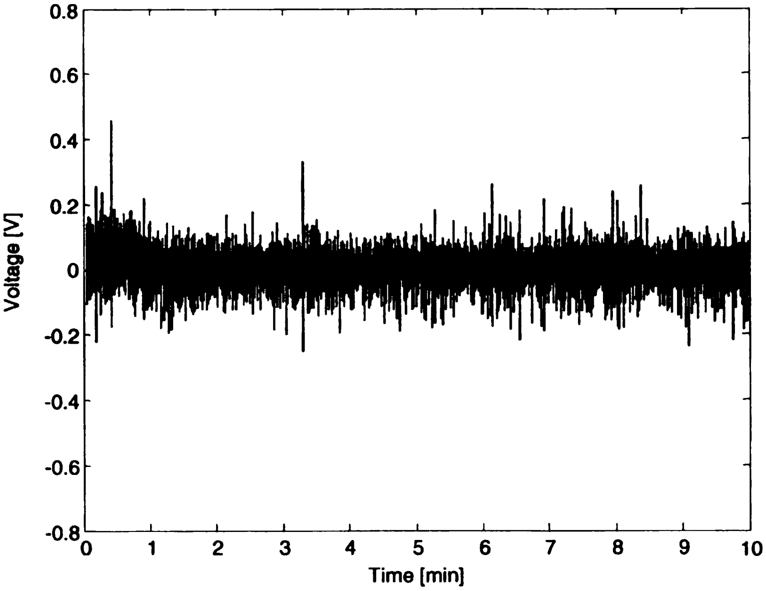
4.2 EEG Signals at a Glance

In this work our interest is to apply the GPAF algorithm to extract or create artificial features that increase the performance of the classifier. In this chapter, we apply the GPAF algorithm to create a feature that could perform an acceptable rate of classification in categorizing EEG data as either baseline or preictal waveforms.

In this work, we deal with mesial temporal lobe epilepsy (MTLE). First, let us look at the EEG data for both baseline and preictal classes. Figure 4.2 shows two *epochs* (*i.e.*, data segments), where one is a baseline segment and the second is a preictal segment; both of them of ten minutes duration (as mentioned before, EEG signals were sampled at 200 Hz; therefore, a ten-minute epoch contains 120,000 points). The preictal epoch initial point was taken to begin fifteen minutes before the UEO, and it ended five minutes before the UEO. That is, in order to obtain a prediction a sufficient time in advance of the seizure onset, we wanted to base the prediction on a signal epoch ending at least five minutes before UEO. From the illustration, it can be noted that there is no obvious visual pattern that helps us to distinguish one signal from another. Figure 4.3 presents the (empirically determined) probability density function of both EEG data epochs. It can be seen in the picture that the baseline data and preictal data are almost completely overlapping. In fact, if we calculate the first- and second-order statistics for both signals, we obtain that the means for both classes are approximately zero (*i.e.*, $\mu_{\text{baseline}} \approx 0$ and $\mu_{\text{preictal}} \approx 0$) and the variances are $\sigma^2_{\text{baseline}} = 2550.0$ and $\sigma^2_{\text{preictal}} = 707.02$, making virtually impossible a high classification rate without any preprocessing of the EEG signals.



(a)



(b)

Figure 4.2 Illustration of ten-minute-duration baseline signal (a) and ten-minute-duration preictal signal, extracted until five minutes before UEO (b).

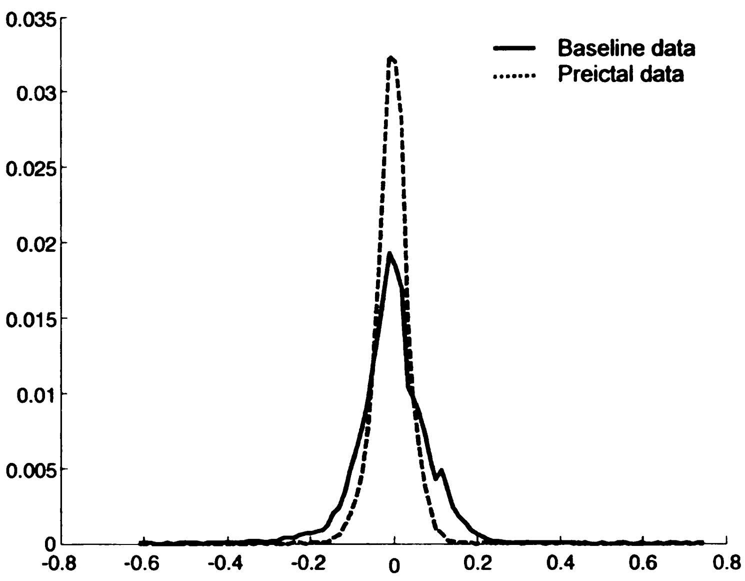


Figure 4.3 Probability density functions for two-ten minutes epoches: one baseline (solid-line) and one preictal (dashed-line).

4.3 Nonlinear Dynamical Systems and State-Space Reconstruction

It has been proposed that EEG signals are nonlinear [77], nonstationary [103], high-dimensional (because of the number of neurons in the brain), and chaotic (complicated, but of relatively low dimension) [47], [78]. Whether or not EEG signals are nonlinear is still being argued by other researchers, some of whom propose that EEG signals are linearly generated, *i.e.*, that one should treat EEG signals like output of a linear system to which the input is noise [86], [98]. Nonetheless, in this work, we treat EEG signals as nonlinear and chaotic signals. EEG signals have been analyzed from many perspectives and using different methodologies, most of which were cited and briefly discussed in Chapter 2.

Many authors have applied nonlinear dynamics tools to analyze EEG data. Chaos theory [91] states that in a system with apparently disordered dynamics, an underlying order resides in the data. Chaotic systems have various characteristics [91]:

- Chaotic systems are aperiodic
- They are sensitive to the initial conditions. A slight change in them can cause a completely different long-term output response (therefore, unpredictability in the long term)
- The equations that dictate the dynamics of the system are nonlinear.

Point two (above) states that long-term prediction is impossible; however, it has also been stated that prediction in the short term, in chaotic systems, is possible.

Let us consider a system with dynamics described by

$$\dot{\mathbf{x}} = \mathbf{F}(x_1(t), x_2(t), \cdots, x_n(t)) \tag{4.1}$$

with a unique observable output described by

$$y(t) = f(x_1(t), x_2(t), \cdots, x_n(t)), \tag{4.2}$$

where n is the number of equations that describe the dynamic behavior of the system. Evaluating these equations from an initial state $\mathbf{x}_0$, a trace is left afterward by the state vector $\mathbf{x}$ while traversing the state-space $\mathbf{R}^n$. This state trajectory is denoted $\beta_t(\mathbf{x}_0)$.

Nonlinear dynamic theory asserts that the state trajectory $\beta_t(\mathbf{x}_0)$ can be reconstructed by uniformly sampling Equation (4.2) to obtain a time-series, which we denote y_k, where k = 0, ..., $n_s - 1$, where n_s is the number of samples and

$$y_k \equiv y(k\tau), \tag{4.3}$$

where τ is the delay or sampling interval. This strategy is called *time-delay embedding* [95]. Figure 4.4 illustrates a state trajectory being reconstructed by delay-embedding. This method involves building a pseudo-state vector λ_k (also known as the embedding vector) of n_e delayed samples (using Equation (4.3)) from the observable measurements $y(t)$:

$$\lambda_k = \begin{bmatrix} y_{k-(n_e-1)} & y_{k-(n_e-2)} & \cdots & y_k \end{bmatrix}. \tag{4.4}$$

The parameter n_e is also known as the *embedding dimension*. Equation (4.4) is a pseudo-state vector at time k. To perform the state reconstruction completely, we need to slide the window through all the samples taken, n_s. This last statement can be formulated as a matrix: if we stack all the embedding vectors in a vector column, we obtain the following matrix:

$$\Gamma = \begin{bmatrix} y_0 & y_1 & y_2 & \cdots & y_{n_e-1} \\ y_1 & y_2 & y_3 & \cdots & y_{n_e} \\ y_2 & y_3 & y_4 & \cdots & y_{n_e+1} \\ \vdots & & & \ddots & \vdots \\ y_{N_e-1} & y_{N_e} & y_{N_e+1} & \cdots & y_{n_s-1} \end{bmatrix}, \tag{4.5}$$

where N_e denotes the number of embedding vectors and equals $n_s - (n_e - 1)$. This matrix is called the *embedding matrix*. Such a matrix contains or samples the trajectory of $\beta_t(\mathbf{x}_0)$. However, the embedding matrix Γ does not contain an exact copy of the trajectory but a distorted or copy of the "genuine" trajectory of $\beta_t(\mathbf{x}_0)$. To be more consistent with topological concepts, the embedding matrix is a diffeomorphism of the original trajectory. A *diffeomorphism* [37] is a differentiable homeomorphism (smoothly distorted

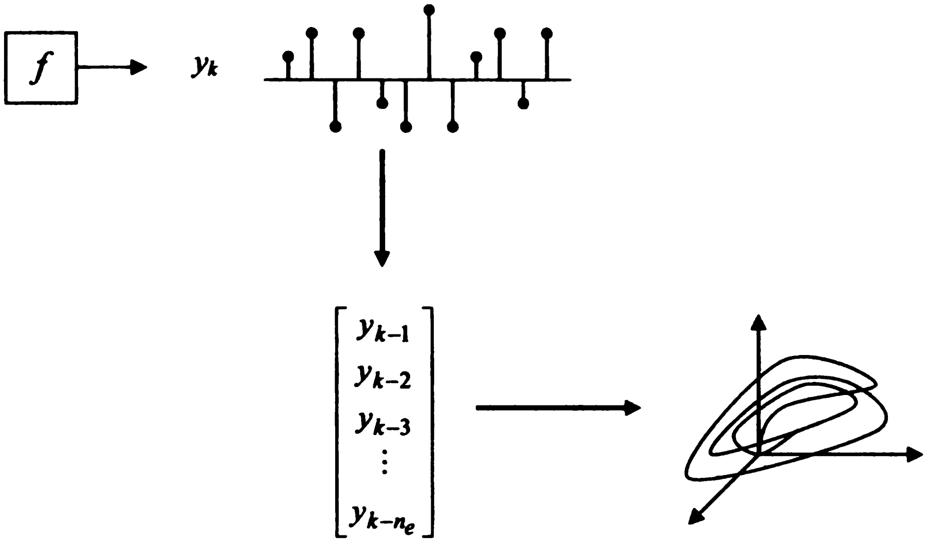


Figure 4.4 Reconstruction of the delay-embedding trajectory from a time-series.

copy) of the original trajectory that preserves the topological properties of the trajectory, such as number of holes, sharp corners, and thus properties like fractal dimension and Lyapunov exponents. Although more details are beyond the scope of this work, interested readers are referred to a full explanation in [20].

One valid question that comes up is what value should n_e be set to? If the embedding dimension is set such that $n_e \geq n$ (recalling n is the order or the number of equations that describe the system in Equation (4.1)), a state reconstruction can be generated though not guaranteed (*e.g.*, a Klein bottle needs a four-dimension Cartesian space to describe its trajectory even though it is a two-dimension manifold) not, because only n equations are needed to solve for n unknowns. On the other hand, if $n_e < n$, it is still possible to "recover" the state-space if the trajectory settles in a lower-dimensional region. Takens [95] proposed the embedding theorem, which gives us an upper bound on the embedding dimension n_e. Takens' bound guarantees that the state-space can be reconstructed by setting $n_e = 2n + 1$. However, in practice, the value of n is not known; but given such an upper bound for n_e, various strategies have been conceived. One of the simplest methods used is to start evaluating the model with $\{n_e = 1, 2, \ldots, n_s\}$ and increase it until an acceptable prediction error is obtained, *i.e.*, the difference between the predicted value and the real value is low [75]. Additionally, some researchers have said that what the embedding theorem states is that even if a system is of infinite order, a finite number of delays of the observable measurement (observable output) can be used to realize the whole dynamics, given that the state confines its trajectory to a space of finite dimension. Such cases have been reported in studies of turbulent fluid flow [84].

For the sake of clarity and comprehension, consider a well-known chaotic system: the Lorenz attractor. The equations that govern the dynamics, although simple, produce chaotic behavior. When a three-dimensional system is plotted in a three-dimensional space, the trajectory of the system can be observed. Figure 4.5(a) illustrates the Lorenz attractor in a three-dimensional plot. Equation (4.4) states that we can reconstruct a diffeomorphism of the original trajectory using the previous values of an observable measure from the system under study. In this case, we know that the order of the system is $n = 3$. Therefore, using a pseudo-state vector formed by one state variable and two delayed samples of it, we reconstruct the trajectory. Figure 4.5(b) shows the state reconstruction of the Lorenz attractor.

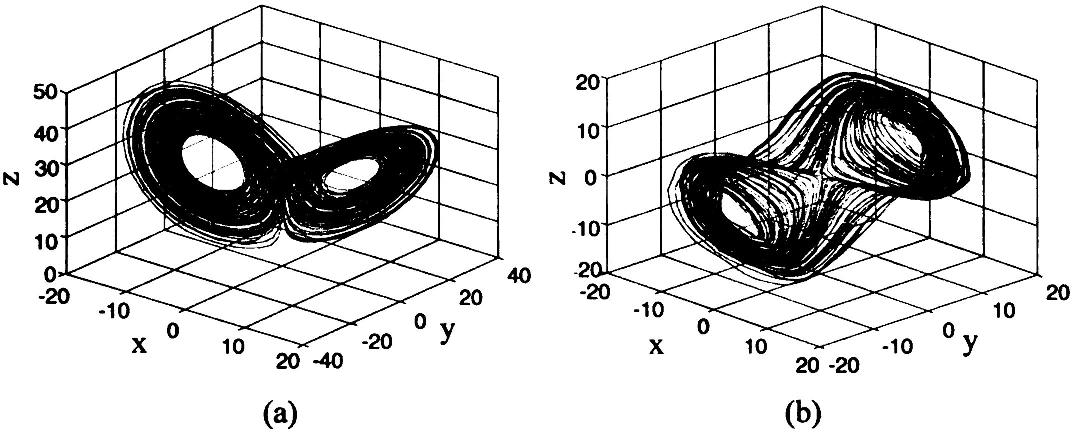


Figure 4.5 (a) Lorenz attractor and (b) its reconstruction using the previous values of a state variable.

4.4 State-Space Reconstruction and the GPAF Algorithm

Early in the previous section, it was mentioned that the dynamics of the EEG signals are considered to be of high-dimensionality. This fact is discernible because the human brain has approximately 10^{11} neurons and each of these neurons can be described with an equation that dictates its dynamics, making the order of the brain practically infinite or at least very high-dimensional. However, it was also mentioned that researchers have stated that the embedding theorem states that a finite number of delays of the observable measurement (observable output) can be used to reproduce the whole dynamics when the state confines its trajectory to a space of finite dimension. Thus, we can hope to capture a "pocket" of deterministic components of the state trajectory of the EEG signals by taking previous samples of the observable output and creating an artificial state vector with n_e

elements (denoted λ_k in Section 4.3). The idea of modeling EEG signals by a delay-embedding method has been proposed before in [20].

Given that we can reconstruct the state trajectory from previous values, our approach is to input such deterministic components, represented as pseudo-state vectors (*i.e.*, λ_k) evolving in time (*i.e.*, embedding matrix Γ), to a genetic programming module and by means of the algorithm, to find a pattern or transformation, probably nonlinear (although a linear transformation is possible), that yields a subspace where the baseline and preictal classes exhibit the maximum separability possible, making viable the prediction of epileptic seizures. In other words, the GP algorithm should combine the inputs (states) in a (non)linear way and output a function that separate the classes such that the performance of the classifier is better than without a transformation or a benchmark previously defined. Let us define an expression that captures mathematically the essence of what was stated above. Let

$$y_i[k] = \phi_i(\lambda), \tag{4.6}$$

where ϕ_i is the transformation function (or artificial feature) designed by the GP algorithm; i is the artificial feature index (equal to 1 if the GP module is designing just one artificial feature), and

$$\lambda = \{x[n], x[n-\tau], x[n-2\tau], \cdots, x[n-(n_e-1)\tau]\} \tag{4.7}$$

is the set of delayed samples (inputs) that the GP will use in any combination (with replacement) to construct the artificial features. Therefore, the artificial feature is a function defined ϕ_i: $\mathbf{R}^{n_e} \rightarrow \mathbf{R}$. The parameter τ is the delay time, which will be determined using the autocorrelation function (discussed in Section 4.5), and n_e is the embedding dimension. There are various methodologies to determine an adequate embedding dimension, the simplest being trial-and-error in increasing dimension, as mentioned before.

One work of particular interest in this area, due to its relationship with our work, was presented in [62]. McConaghy *et al.* proposed the use of a GP algorithm to reconstruct a chaotic dynamic state of a given system. The GP system designed polynomials that reconstructed the exact dynamics of the chaotic system. Also, the GP determined the embedding dimension given that the authors added the delay operator z^{-1} to the function set, which applies a unit delay to all terminals in its argument. Therefore, before evaluating the program (individual or chromosome) to score its fitness, a preprocessing is performed on each individual to apply the delay operators such that the terminals in the argument receive their respective delays.

As is done with many conventional features—for example, the Fourier coefficients, or the energy of a signal, etc., in which the feature is calculated over data viewed through an observation window—in the genetic programming artificial feature (GPAF) algorithm, we will also use an observation window, also known as a *sliding window*, which will be slid through the entire signal being processed. The sliding window is defined by two parameters: length of the window, denoted L, and the displacement factor, denoted D, alternatively described as the percentage of overlap between adjacent sampling windows. Figure 4.6 depicts the sliding window concept over an arbitrarily selected fifteen-second-duration EEG signal. The length L defines the number of points that will be analyzed at

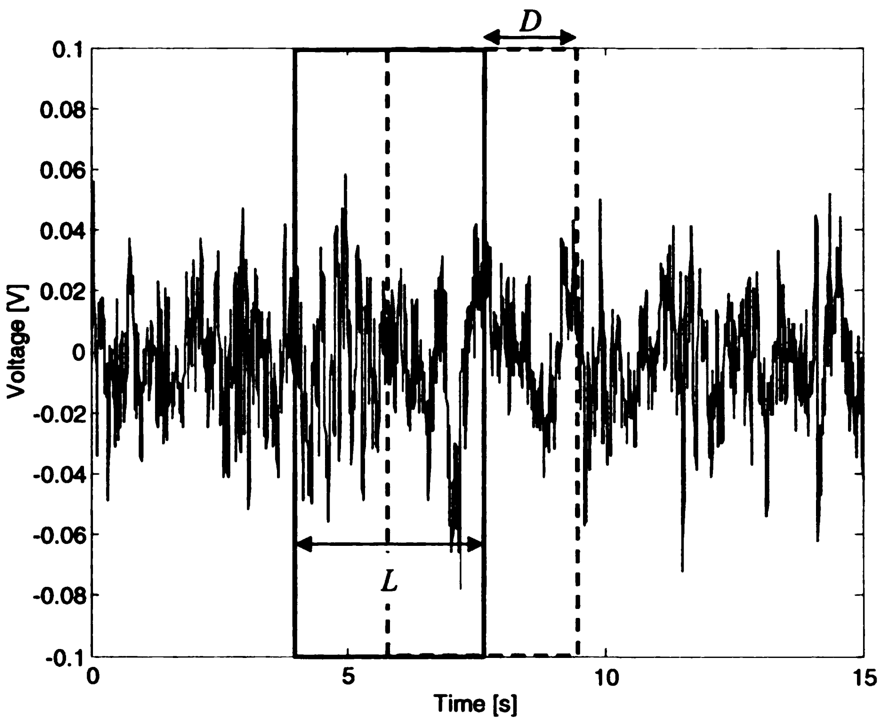


Figure 4.6 Sliding window slides (solid rectangle) over an EEG signal. The dashed rectangle indicates the position where the window will be displaced in the next comoutation.

the same time. On the other hand, D is defined as the number of new points that will be used in the next evaluation, such that $D = L - O$, where O is the overlap, or the number that will be evaluated again from the previous position of the window. To compress and smooth out the variability in the data, a summation operator will be applied at each window position. Recalling Equation (4.6) and adding to it the sliding window, *i.e.*, a summation operator, the resultant equation is stated as

$$y_i[k] = \sum_{n=1+(k-1)D}^{D(k-1)+L} \phi_i(x[n], x[n-\tau], \ldots, x[n-(n_e-1)\tau]). \tag{4.8}$$

The $y_i[k]$ is the GP artificial feature of the EEG signal $x[n]$, termed here the *artificial feature time series*; the subscript i denotes the number of the feature (*i.e.*, we can have a single artificial feature or a vector of such features), n is the index that controls the displacement inside the sliding window, and k is the discrete time unit of this time-series (or the index that indicates the next sliding-window position). The sliding window observes all L points in window-position k, whereas index n is delimited by $1 + D(k-1) \leq n \leq D(k-1) + L$ (superscript and subscript on the summation symbol). In this work, we selected n to advance by steps of one, thus taking advantage of all of the EEG data available and scrutinizing the data in maximum detail looking for patterns.

A point $y_i[k]$ in Equation (4.8) is proportional to an average of the GPAF-processed points observed through the sliding window, in which the argument of the summation ϕ_i (*i.e.*, the GPAF function) is found by the GP algorithm, and in which at each window position k, the number L of points will be reduced, by means of Equation (4.8), to a single point at the output. Thus, if we have a signal that contains P_{Tot} points and we set displacement to D, the number of output points after the entire signal is processed by the GP artificial feature is

$$\kappa = \frac{P_{Tot}}{D} - \frac{L}{D} + 1, \qquad (4.9)$$

where the $1 - L/D$ adjusts for the beginning and end of the signal where the window of length L is not complete; thus, some of those positions (or points) are not calculated.

To summarize all the components of the GPAF algorithm, a block diagram is presented in Figure 4.7. Figure 4.8 illustrates the transformation performed by the GPAF.

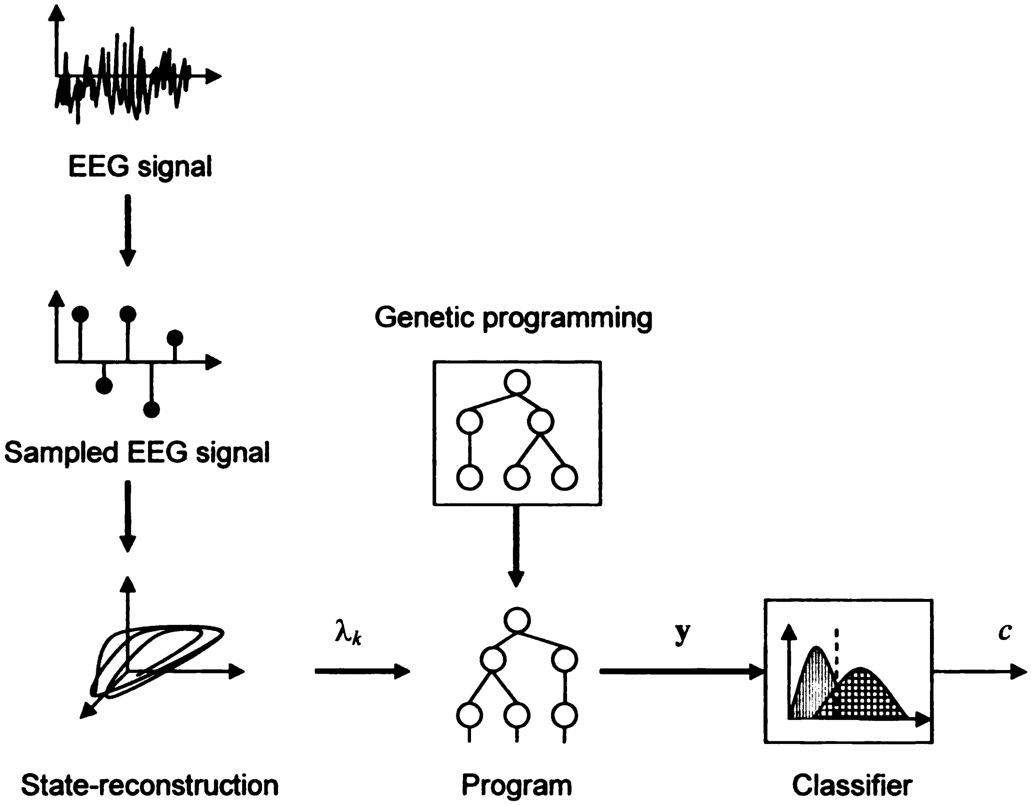


Figure 4.7 Diagram of the genetic programmed artificial features algorithm. The input signal is processed by the program generated by the GP. Later, the output is used to train the classifier.

The function ϕ is the artificial feature, *i.e.*, the program generated by the GP that increases the separation between classes. Once the EEG data are processed by the artificial feature, **y** (the artificial feature space) is obtained and input to the classifier (denoted as μ). The classifier is used as an auxiliary component to calculate the fitness of the program (artificial feature); however, a researcher may decide not to use a classifier and to let the GP algorithm designs the decision structure (*i.e.*, classifier), but that will carry an enormous computational cost. Additionally, if the researcher decides to use a classifier to calculate the fitness, he/she must be aware that the artificial features are optimized for the classifier that was selected. Although it may be desired that the artificial features work well with any classifier, such performance cannot be guaranteed. The GPAF algorithm will continue evaluating new programs (artificial features or individuals) until the GP reaches a pre-set maximum number of generations or the fitness goal is met, according to a fitness function explained in Section 4.6.

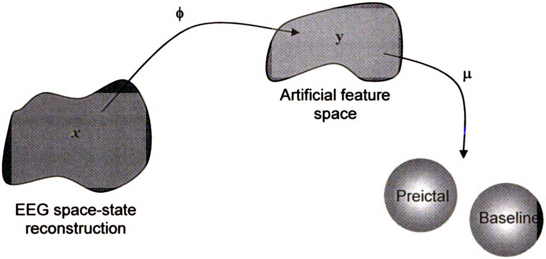


Figure 4.8 Schematic of the nonlinear transformation performed by the GPAF algorithm, denoted ϕ, in cascade with a classifier, denoted μ, which categorizes the data into one of two possible classes: baseline or preictal. The variable $\boldsymbol{x}$ is the EEG signal at hand and variable $\mathbf{y}$ represents the data after processing by the GP artificial feature.

4.4.1 Classifier

Although in the GPAF algorithm any classifier may be used, in this work we selected the classifier with two aspects in mind. Recalling from Chapter 1, one of the contributions of this work is to prescribe a general-purpose algorithm that automatically creates artificial features from conventional features or raw data. Therefore, we selected the *k-nearest neighbor* (*k*-NN) classifier as the classifier component for the GPAF algorithm. This classifier is a nonparametric, nonlinear, and thus a multi-threshold one, making it suitable even for multi-dimensional, multi-modal problems. In addition, the training process is relatively easy, by just giving to the classifier the whole training dataset. Again, as was previously mentioned, because we selected the *k*-NN classifier, the artificial features are going to be optimized for such classifier. We do not guarantee that such artificial features will have a good performer if any classifier such statement cannot be guaranteed.

There are many metrics or rules used in *k*-nearest neighbor classifiers. Among them are the Minkowski metric, Manhattan distance (also known as city block distance), and the Tanimoto metric. In this work, we will use the Euclidean distance, which is the most common metric used and a special case of the Minkowski metric, and is defined as

$$D(\mathbf{a},\mathbf{b})=\sqrt{\sum_{i=1}^{d}(a_i-b_i)^2}\ , \tag{4.10}$$

where **a** and **b** are the vectors between which the metric is to be calculated.

4.4.2 Detailed Example

Figure 4.9 shows all the components that comprise the GPAF algorithm and the steps involved in the evaluation of one individual. In the figure, we provide an example where two artificial features are being sought (thus, an individual corresponds to a two-tree "forest"). The first step is the generation of the initial population of artificial features (**I**). The artificial features are transformed from vectors to trees for comprehension purposes (**II**). Such trees encode the artificial features. From these, the artificial functions are translated into mathematical expressions (the arguments of the summation operators) to be evaluated (**III**). Later, the EEG signal, segmented with the sliding window (**IV**) is processed by the artificial features generating the artificial signals (**V**). Later these artificial data are input to the *k*-NN classifier to execute the categorization task (**VI**). In the illustration, label "0" denotes the baseline samples and "1" denotes the preictal (or ictal) samples. In the training phase (*i.e.*, when artificial features are being designed), we can evaluate either the training data or a checking data set (a data set different from the training data intended to resemble the validation data) to calculate the classification error, the performance measure that will drive the GP search (further discussion about the performance index is in Section 4.6). It should be noted that this entire illustration is just to evaluate one individual (*i.e.*, one forest). After all individuals in the initial population are evaluated, the genetic operators take over to create the next population of artificial features. The errors are calculated again. This process is repeated until either a performance goal is reached or the maximum number of generation is met.

4.5 Autocorrelation Function

The *autocorrelation function* [71] of a random process or a signal $q[n]$ is defined as the expected value of the multiplication of $q[n]$ and a time-shifted version of itself. Mathematically, the autocorrelation (discrete version) is defined as follows:

$$R_{qq}[n, n+m] = E[q[n]q[n+m]]. \tag{4.11}$$

What the autocorrelation tells is how quickly the self-similarity of a process (or a given signal) changes with respect to time. To graph an autocorrelogram, Equation (4.11) is evaluated for different lags, m, and the results are displayed in the graph. A

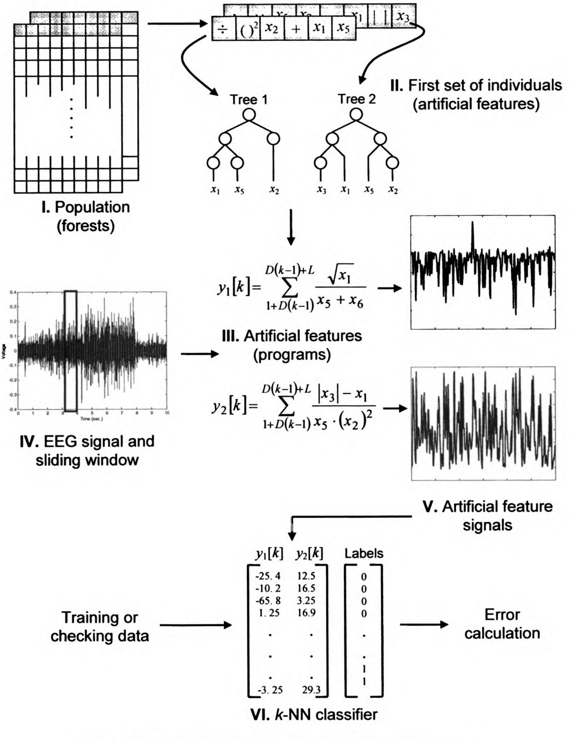

÷
$()^2$
x_2
+
x_1
x_5
x_1
$|\,|$
x_3
II. First set of individuals (artificial features)
Tree 1
Tree 2
x_1 x_5 x_2
x_3 x_1 x_5 x_2
I. Population (forests)
$y_1[k]=\sum_{1+D(k-1)}^{D(k-1)+L}\frac{\sqrt{x_1}}{x_5+x_6}$
III. Artificial features (programs)
$y_2[k]=\sum_{1+D(k-1)}^{D(k-1)+L}\frac{|x_3|-x_1}{x_5\cdot(x_2)^2}$
Voltage
Time (sec.)
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0
1
2
3
4
5
6
7
8
9
10
IV. EEG signal and sliding window
V. Artificial feature signals
$y_1[k]$ $y_2[k]$
Labels
-25. 4 12.5
-10. 2 16.5
-65. 8 3.25
1. 25 16.9
-3. 25 29.3
0
0
0
0
1
1
Training or checking data
Error calculation
VI. *k*-NN classifier

Figure 4.9 Detailed illustration of the GPAF algorithm process.

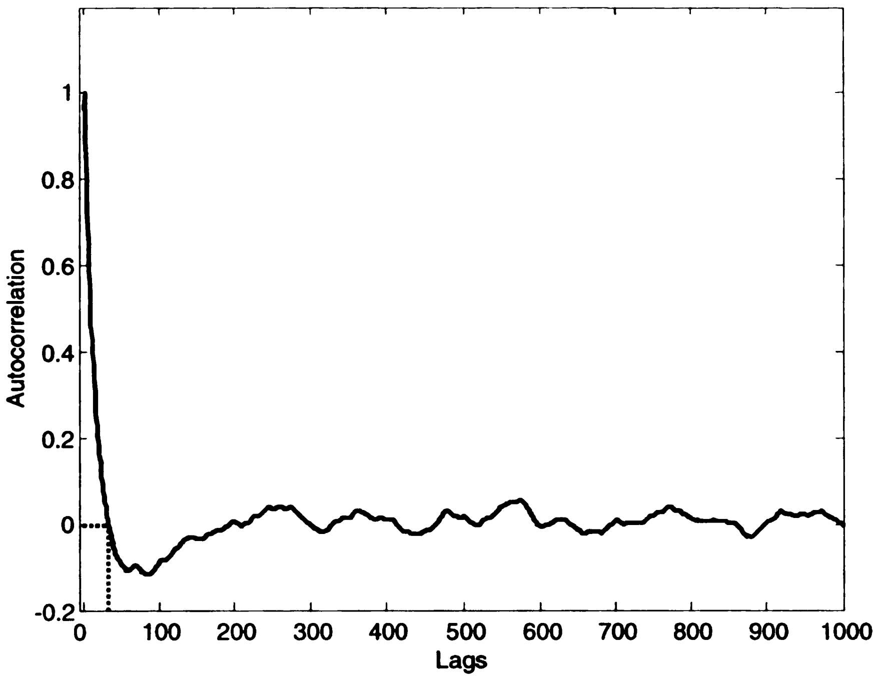


Figure 4.10 Characteristic autocorrelation plot.

characteristic normalized autocorrelation plot is shown in Figure 4.10. It can be noted that the closer *m* (the lag) is to zero, the larger is the autocorrelation.

One parameter that is determined from the autocorrelation plot is the *autocorrelation time*. It is the value of *m* for which the autocorrelation value, $R_{qq}(n,n+m)$, is zero or crosses zero for the first time. In Figure 4.10, this value (point) is indicated, on the lower left side of the plot, with two dashed-lines.

To determine the delay time τ (see Section 4.3), the autocorrelation plot of an EEG preictal signal (selected for training purposes) will be obtained and the autocorrelation time will be visually determined for each patient. The autocorrelation time (delay) value can be represented using time units, *e.g.*, seconds.

4.6 Performance Index

The desirability of the artificial features is quantified using classifier-based performance metrics. These performance metrics drive the search of the GP algorithm for the "best" set of artificial features. The prediction/detection epileptic seizure problem is a classical detection problem, with the sole difference that in a classical detection the test is whether the event at hand is noise or a signal-plus-noise. In our case, we are the test is to know whether the event at hand is nonseizure (analogous to noise) or preseizure (analogous to a signal) data. We can model this more formally by letting

$$\begin{aligned} &H_0\text{: } \mathbf{y} \in \text{NS} \\ &H_1\text{: } \mathbf{y} \in \text{S}. \end{aligned} \tag{4.12}$$

This is the classical binary hypothesis test. Typically, H_0 is known as the null hypothesis, and states that **y** is coming from the nonseizure class NS, whereas H_1 is the alternative hypothesis that states that **y** is coming from the preseizure S class. Figure 4.11 shows an illustration of two conditional PDFs, where $p(\mathbf{y}|\text{NS})$ denotes the conditional PDF for the nonseizure class, while $p(\mathbf{y}|\text{S})$ denotes the conditional PDF for the preseizure class. There are various metrics that can be defined from this illustration. These metrics comprise integrals, determination of PDFs, and class-decision boundaries. Thus, we develop empirical measures of such metrics, which are unbiased estimates based on counting. Counting can be performed either on a point basis or a block basis. In *point-basis* counting, a sample (*i.e.*, a point) in the artificial feature space is counted as one example, whereas in ***block-basis*** counting, a series of artificial feature samples (*i.e.*, a string of

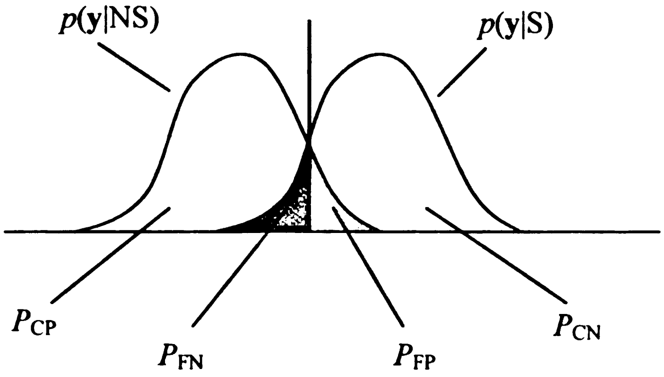


Figure 4.11 PDFs for a binary hypothesis test.

points) is counted as one example. From the above hypothesis, we can define a series of classifier (detector) decisions, which are presented as a confusion matrix in Table 4.2.

Table 4.2 Confusion matrix of the classifier decisions.

	$\hat{\mathrm{S}}$	$\widehat{\mathrm{NS}}$
S	N_{CP}	N_{FN}
NS	N_{FP}	N_{CN}

S and NS are labels for true preseizure and nonpresiezure classes, while their "hats" (with circumflex marks) indicate the classes declared by the classifier, N_{CP} is number of correct positives (preseizure class detections), N_{CN} is number of correct negatives, N_{FP} is number of false positives (false alarms, when the algorithm or device said that a seizure will occur and it does not), and N_{FN} denotes number of false negatives (preseizure class misses, when the algorithm says that a seizure will not occur, then it does). Addtionally, we define $N_{\mathrm{S}} = N_{\mathrm{CP}} + N_{\mathrm{FN}}$, the number of preseizure examples, $N_{\mathrm{NS}} = N_{\mathrm{CN}} + N_{\mathrm{FP}}$, the number of nonpreseizure examples, and $N_{\mathrm{tot}} = N_{\mathrm{S}} + N_{\mathrm{NS}} = N_{\mathrm{CP}} + N_{\mathrm{FN}} + N_{\mathrm{CN}} + N_{\mathrm{FP}}$, the total number of examples. Using the classifier decisions stated above, we can derive a set of empirical formulas to calculate various classifier-based performance metrics. These metrics are provided in Table 4.3. The quantities P_{CP}, P_{FN}, P_{FP}, and P_{CN} are denoted in Figure 4.11.

Table 4.3 Multiple classifier-based performance metrics for the GP cost function.

Metric	Formula
Probability of correct positive (Sensitivity)	$P_{\text{CP}} = \dfrac{N_{\text{CP}}}{N_{\text{S}}}$
Probability of false negative	$P_{\text{FN}} = \dfrac{N_{\text{FN}}}{N_{\text{S}}} = 1 - P_{\text{CP}}$
Probability of correct negative (Specificity)	$P_{\text{CN}} = \dfrac{N_{\text{CN}}}{N_{\text{NS}}} = 1 - P_{\text{FP}}$
Probability of false positive	$P_{\text{FP}} = \dfrac{N_{\text{FP}}}{N_{\text{NS}}}$
Selectivity	$Sel = \dfrac{N_{\text{CP}}}{N_{\text{CP}} + N_{\text{FP}}}$
Probability of correct classification	$P_{\text{C}} = \dfrac{N_{\text{CP}} + N_{\text{CN}}}{N_{\text{tot}}}$
Probability of error	$P_{\text{E}} = \dfrac{N_{\text{FN}} + N_{\text{FP}}}{N_{\text{tot}}} = 1 - P_{\text{C}}$

The selectivity is another definition for specificity quantity [100], which denotes how many of the feature-vectors (or a single feature) categorized as preseizure actually were preseizure. Additionally, we can estimate the prior probabilities empirically as shown in Table 4.4, where $P(\mathrm{S})$ denotes the prior probability of the preseizure data and $P(\mathrm{NS})$ denotes the prior probability of the nonseizure data (baseline).

Table 4.4 Prior probability metrics from the confusion matrix.

Quantity	Formula
Prior probability of preseizure	$P(\mathrm{S}) = \dfrac{N_{\mathrm{S}}}{N_{\mathrm{tot}}}$
Prior probability of nonpreseizure	$P(\mathrm{NS}) = \dfrac{N_{\mathrm{NS}}}{N_{\mathrm{tot}}}$ $= 1 - P(\mathrm{S})$

From Tables 4.3 and 4.4, we can state the performance index (or fitness function) for the GP algorithm. The probability of error can be stated as

$$\begin{aligned} P_{\mathrm{E}} &= P(\mathrm{E} \mid \mathrm{S})P(\mathrm{S}) + P(\mathrm{E} \mid \mathrm{NS})P(\mathrm{NS}) \\ &= P_{\mathrm{FN}}P(\mathrm{S}) + P_{\mathrm{FP}}P(\mathrm{NS}) \end{aligned} \quad . \tag{4.13}$$

Equation (4.13) penalizes the two types of errors: the number of false positives and the number of false negatives, which are scaled by their respective prior probabilities. In some problems, it is possible to use the priors to penalize the errors; however, for prediction and detection of epileptic seizures this may not be enough because $P(\mathrm{S})$ is very small considering the time scales used (seizures just last approximately minutes while the nonseizure data may last hours); and missing of seizures (false negatives) are relatively unacceptable. For this reason, we invoke the *error risk* metric, which assigns risk factors $r > 0$ to adjust the relative costs of the errors. Equation (4.14) shows the error risk metric

$$R_{\mathrm{E}} = P_{\mathrm{FN}}P(\mathrm{S})r_{\mathrm{FN}} + P_{\mathrm{FP}}P(\mathrm{NS})r_{\mathrm{FP}}, \tag{4.14}$$

where r_{FN} is a risk term associated with missing preictal signals and r_{FP} is a risk term associated with declaring false positives. We can interpret the $P(\mathrm{S})r_{\mathrm{FN}}$ and $P(\mathrm{NS})r_{\mathrm{FP}}$ as penalty factors (or weight factors); thus, we can rewrite Equation (4.14) as

$$R_{\mathrm{E}} = P_{\mathrm{FN}}\pi_{\mathrm{FN}} + P_{\mathrm{FP}}\pi_{\mathrm{FP}}. \tag{4.15}$$

Typically, missing a seizure is considered worse than declaring a false alarm; therefore, it is intuitive to set $\pi_{\mathrm{FN}} > \pi_{\mathrm{FP}}$ to compensate for the effects of a low $P(\mathrm{S})$. For example, we can assign $\pi_{\mathrm{FN}} = 0.75$ and $\pi_{\mathrm{FP}} = 0.25$. This makes the GP algorithm to pay attention to both types of errors. However, the exact values of penalty weights can be adjusted according to the patient's untreated seizure frequency, forms of therapy, or other factors.

For a given feature vector or a single artificial feature, the minimum error risk problem is

$$\min_{\Omega} R_{\mathrm{E}}, \tag{4.16}$$

where Ω is the space of all possible class decision boundaries or the space that contains all possible mappings from input feature **y** to classifier output. The optimal decision structure [10] is given by

$$C(\mathbf{y}) = \arg\max_{C_i \in \{\mathrm{S},\mathrm{NS}\}} \{p(\mathbf{y}|\mathrm{S})\pi_{\mathrm{FN}}, p(\mathbf{y}|\mathrm{NS})\pi_{\mathrm{FP}}\}. \tag{4.17}$$

If we define the likelihood ratio, the decision rule declares class S if

$$\frac{p(\mathbf{y}\,|\,\mathrm{S})}{p(\mathbf{y}|\mathrm{NS})} > \frac{P(\mathrm{NS})r_{\mathrm{FP}}}{P(\mathrm{S})r_{\mathrm{FN}}} = \frac{\pi_{\mathrm{FP}}}{\pi_{\mathrm{FN}}}, \tag{4.18}$$

where $\pi_{\mathrm{FP}}/\pi_{\mathrm{FN}}$ is the threshold. The decision rule for the threshold on the posterior probability declares class S if

$$P(\mathrm{S}|\mathbf{y}) > \frac{1}{1 + r_{\mathrm{FN}} / r_{\mathrm{FP}}}. \tag{4.19}$$

If P(S) is very small, this threshold can be very small, as in the maximum-likelihood rule. For example, if P(S) = 0.005 and π_{FN} = 0.75 and π_{FP} = 0.25, values that place more stress on avoiding FNs than maximum likelihood, the threshold is 0.0017.

To discourages the GP algorithm from designing artificial features for which its classifier makes too many errors of one type and too few of the other type, we added a constraint to the minimum risk problem as stated in Equation (4.20):

$$\min_{\Omega}\{R_{\mathrm{E}} \,|\, b \geq B_0\}. \tag{4.20}$$

For instance, the constant-declaration classifier whose output is always class preictal (*i.e.*, it is always declaring that a seizure is occurring) has only 25% error risk:

$$R_{\mathrm{E}} = 0(0.75) + 1(0.25) = 0.25. \tag{4.21}$$

However, this classifier is merely a device that medicates the patient in a prescribed open-loop regime. The balance constraint drives the algorithm to make more useful decisions based on the input artificial feature. In the cost function of the genetic programming system, this contraint is added as a penalty term to the error risk if the balance is less than 50%.

$$\text{Cost Function} = \begin{cases} (0.75)P_{\mathrm{FN}} + (0.25)P_{\mathrm{FP}}, & b \leq 0.5 \\ (0.75)P_{\mathrm{FN}} + (0.25)P_{\mathrm{FP}} + \gamma(1-b), & b > 0.5 \end{cases}, \tag{4.22}$$

where γ is a weight commonly equal to 1 and b is the balance defined as

$$b = 1 - \frac{2|P_{\mathrm{FN}} - P_{\mathrm{FP}}|}{|0.5 - P_{\mathrm{FN}}| + |0.5 - P_{\mathrm{FP}}| + 1}. \tag{4.23}$$

The parameter b is 0 for the worst case when either $(P_{\mathrm{CN}}, P_{\mathrm{FP}}) = (0,1)$ or $(1,0)$, and is 1 in the best case when $P_{\mathrm{FN}} = P_{\mathrm{FP}}$ [21].

Depending on the value of the threshold arrived at by changing the probability or the likelihood ratio, we move the operating point of the classifier along the receiver operating characteristic (ROC [92]; P_{CN} vs. P_{FP}) curve, trading off the ability to detect preseizures for the ability to detect nonpreseizures (and losing optimality as initially defined by the performance metrics). If the risk r_{FP} is increased, the probability threshold is also increased, reducing the propensity of the system to declare S. In contrast, to make the detector (*i.e.*, classifier) more sensitive, the risk r_{FN} should be increased. The complexity of the classifier decision boundaries changes in the n-dimensional space of features as the value of thresholds on the likelihood ratio or the probability changes.

4.7 Benchmark Classifiers

As a frame of reference for our proposed methods, principal components analysis [49] (also known as Karhunen-Loéve transform in engineering) will be implemented as the feature extractor. Principal components analysis (PCA) is a well-known, established nonparametric feature extractor that performs a linear transformation over the input space, returning a subspace with the best (in the least-sum-of-squared-error sense) uncorrelated representation of it. Mathematically, PCA is defined as

$$Y = \Phi^{\mathrm{T}} X, \tag{4.24}$$

where X is the input data and Φ^{T} are eigenvectors associate with the largest eigenvalues. Such eigenvalues are obtained from the covariance matrix of the input data X. The number of eigenvalues (and thus the number of eigenvectors) is related to the number of dimensions to which it is desired to reduce the input space (*i.e.*, X).

The embedding matrix (denoted as Γ in Section 4.3) with n_e columns (equal in number to the embedding dimension), is input to the PCA. That is,

$$X = \begin{bmatrix} x[1] & x[1-\tau] & x[1-2\tau] & \cdots & x[1-(n_e-1)\tau] \\ x[2] & x[2-\tau] & x[2-2\tau] & \cdots & x[2-(n_e-1)\tau] \\ x[3] & x[3-\tau] & x[3-2\tau] & \cdots & x[3-(n_e-1)\tau] \\ \vdots & & & & \\ x[n] & x[n-\tau] & x[n-2\tau] & \cdots & x[n-(n_e-1)\tau] \\ \vdots & & & \ddots & \\ x[n_s] & x[n_s-\tau] & x[n_s-2\tau] & \cdots & x[n_s-(n_e-1)\tau] \end{bmatrix}, \tag{4.25}$$

where $x[]$ is the EEG signal and n is the time index. The parameter n_e is the embedding dimension, whose value defines the number of features input to the PCA. The number of components selected to reduce the dimensionality of the input data X will be same number of artificial features that was needed by the GPAF algorithm to achieve a satisfactory result. Then these components are processed by the sliding window yielding

$$y_j[k] = \sum_{n=1+(k-1)D}^{D(k-1)+L} \theta_j(n), \tag{4.26}$$

where θ_j is the principal component j selected by the PCA, $j = 1, 2, \ldots, n_c$, and n_c is the number principal components. The parameters L and D will be set to the value set for the GPAF features. The summation of all points can be considered an elemental feature (summation is proportional to the mean). For instance, if $n_c = 2$ and letting $\theta_1 = x[n-\tau]$ and $\theta_2 = x[n]$ to be the principal components, the features in Equation (4.27) are obtained.

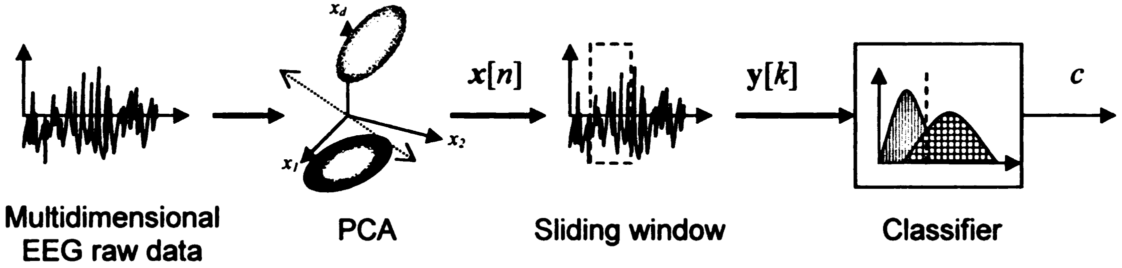


Figure 4.12 Stages of the PCA *k*-NN benchmark classifier.

$$
\begin{aligned}
y_1[k] &= \sum_{n=1+(k-1)D}^{D(k-1)+L} x[n-\tau]. \\
y_2[k] &= \sum_{n=1+(k-1)D}^{D(k-1)+L} x[n].
\end{aligned} \tag{4.27}
$$

Finally, the output of the sliding-window-processed data is input to a *k*-NN classifier that will categorize the data into its respective classes, *i.e.*, baseline or preictal class. Figure 4.12 illustrates the process of the first benchmark classifier. For instance, if we have that for a given patient, the GPAF algorithm found three artificial features, the dimension of *X* will be reduced by the PCA to three. Later, each one of these features will be processed by the sliding window, yielding three feature vectors that are input to the classifier.

A *k*-NN classifier alone is the second benchmark classifier to be used for comparison against the performance of the GPAF algorithm. That is, the matrix *X* in Equation (4.25) (pseudo-state vector) is processed through the sliding window and a vector of size n_e

input directly to a *k*-NN classifier, so there is no dimensionality reduction. That is, the number feature vectors input to the classifier is equal to n_e, the dimension of X.

4.8 Receiver Operating Characteristic (ROC) Curve

For the experiments that we will present in Section 4.10, we will provide the receiver operating characteristic curve plots, an easy, visual way to compare the performance of the GPAF algorithms and the benchmark classifiers. We will also provide the area-under-curve statistic, a measure that allows us an easier way to compare the performance among classifiers. There are two ways to plot an ROC curve: theoretical and empirical. In the theoretical way, the PDFs of both classes (*i.e.*, seizure and nonseizure) are needed. If the more typical ROC curve is going to be plotted (*i.e.*, P_{CP} vs. P_{FN}), we to calculate P_{FA} as

$$P_{\mathrm{FA}} = \int_{\delta}^{\infty} p(\mathrm{y} \mid \mathrm{NS})dy\,, \tag{4.28}$$

where δ is the threshold and P_{CP} as

$$P_{\mathrm{CP}} = \int_{\delta}^{\infty} p(\mathrm{y} \mid \mathrm{S})dy \tag{4.29}$$

for each value of δ from $-\infty$ to ∞ [50]. Each pair (P_{FN}, P_{CP}) corresponding to a value of δ is plotted in the graph. Both axes (P_{FN}, P_{CP}) go from 0 to 1. An ideal classifier would be one for which the ROC curve is a vertical line from coordinates (0,0) to (0,1) and a horizontal line from (0,1) to (1,1), with (0,1) being the point at which the classifier would yield perfect discrimination. However, this framework is not possible because the distributions of the conditional PDFs are unknown. Additionally, the classifier used for the GPAF algorithm and the benchmarks is a nonparametric classifier, which outputs not a probability, but a label. For this reason, to plot the ROC, which may be considered an empirical or estimated ROC, we define a set of equations to estimate the probabilities P_{CP}, P_{FP} using the output of the classifier. Let us define

$$L(\mathbf{y}) = \frac{p(\mathbf{y} \mid \text{S})}{p(\mathbf{y} \mid \text{NS})} \gtrless \delta, \tag{4.30}$$

where $L(\mathbf{y})$ is the likelihood ratio, δ denotes the threshold, and $p(\mathbf{y}|\text{S})$ and $p(\mathbf{y}|\text{NS})$ are the conditional probabilities which are estimated as

$$p(\mathbf{y} \mid \text{S}) = \frac{1}{\sum_{i=1}^{k} d_i^{\text{S}}}$$

and (4.31)

$$p(\mathbf{y} \mid \text{NS}) = \frac{1}{\sum_{i=1}^{k} d_i^{\text{NS}}},$$

where k is the number of nearest neighbors, d_i^{S} is the Euclidean distance between the sample point being tested and the closest preseizure point i, and d_i^{NS} is the Euclidean distance between the sample point being tested and the closest nonseizure point i. It can be noted that the further away are the closest points of a class, the lower is the probability of $\mathbf{y}$ to belong to that class; correspondingly, the closer the k closest points of a given class are, the greater is the probability that $\mathbf{y}$ belong to that class. Equation (4.32) provides an estimation of the conditional probabilities that allow us to calculate and computationally graph the likelihood ratio stated in Equation (4.30). Moving the threshold δ from $(-\infty, \infty)$ allows us to estimate P_{FN} and P_{CP} by counting those nonseizure samples that were misclassified (*i.e.*, exceeded the threshold and were classified as preseizure samples) and those preseizure points that were classified correctly as shown in Table 4.3 (*i.e.*, sensitivity and probability of false positives). For each evaluation of δ, P_{FN} and P_{CP} are calculated obtaining a point in the (empirical) ROC curve plot. To smooth the ROC curve, it will be plotted using point-basis samples. That is, each point in the artificial feature space is counted as one example.

4.9 Function and Terminal Sets for GP

In GP algorithms, the user must specify the function set and terminal set that the GP algorithm will use to create the programs. Table 4.5 shows the functions that will be available for the GP.

Because functions need to satisfy the closure property in GP, the square root, logarithm, and division operators are implemented in a protected way. Protected-division works identically to ordinary division except that it outputs one when its denominator input is zero. In order to avoid complex values, the protected square root operator applies an absolute value operator to the input value before the taking the square root. Also, logarithms are protected versions that output zero for an argument of zero and apply an absolute value operator to the argument to handle negative arguments. By increasing the function set by adding to it more functions (*i.e.*, trigonometric and transcendental functions), we are addressing and implementing the third recommendation to overcome the limitations found in the GFNCAF algorithm experiments in [30]. We recall that in that work, the function set was limited to $\{+, -, \times, \div, \mathrm{I}, |\;|, (\;)^2, \sqrt{\;}\}$. With the function set provided in Table 4.5, we are giving more complexity to the search space and increasing the likelihood to create better artificial features.

Table 4.5 List of functions and their respective symbols for the GP algorithm.

Function	Symbol
addition	$+$
subtraction	$-$
multiplication	$\times$
division	$\div$
square	$(\,)^2$
square root	$\sqrt{[\ ]}$
natural logarithm	ln
logarithm base 2	$\log_2$
logarithm base 10	$\log_{10}$
absolute value	$\lvert\ \rvert$
sine	sin
cosine	cos

As a terminal set, the GP has available $\{x[n], x[n-\tau], ..., x[n-(n_e-1)\tau]\}$, *i.e.*, based on an insight from chaos theory, the number of elements in the terminal set is n_e, the embedding dimension. Again, the delay time τ is determined using the autocorrelation plot as explained in Section 4.5. With this information (embedding vector) as input to the GP algorithm, we are addressing the fourth and last recommendation to overcome the limitations of the GFNCAF algorithm (bulleted early in Section 3.1). By essentially inputting raw data to the GP algorithm, we expect the GP to explore and exploit the EEG data, increasing the chance of finding more informative and successful patterns to discriminate the classes.

In this chapter, the maximum tree depth was set to 10 and the population size to 1000. The initial population was generated using the ramped half-and-half method. The GP algorithm used a crossover operator with a rate of 0.9, fitness proportional selection, and a breeding operator with a rate of 0.1. The GP algorithm was extended to allow it to evaluate not just one tree per individual, but multiple trees per individual (*i.e.*, a forest), allowing us to generate a prespecified number of features for a patient simultaneously. All the trees of the individual were evaluated at the same time; however, there was no crossover among trees (features) of the same individual—crossover was done only among homologous trees. Although it is possible to configure the GP algorithm (software) to decide how many artificial features is going to evaluate per forest (individual), in this work, we limited the algorithm to start looking for one artificial feature. If no satisfactory results were obtained, then we increased the number of artificial features to two. The same procedure was executed if no satisfactory results were obtained. However, features were not incremented to more than three artificial features. To reduce the computing time, LilGP software was used, a popular genetic programming system implemented in C code (the genetic programming algorithm implemented in Chapter 3 could not be used because of the impractically high computing time for the experiments given that it is a Matlab-based code).

4.10 Results

In the experiments in this section, all baseline and preictal epochs were of ten minutes duration, thus containing 120,000 points in each epoch. Preictal segments were selected to end five minutes before the UEO occurred. Figure 4.13 illustrates how the preictal epochs were selected. The length of the sliding window is set to L = 800 points (or 4 s) and D = 400 points (or 2 s). Additionally, the maximum for the embedding dimension n_e was set to 6, which was arbitrarily selected. The number of nearest neighbors for the k-NN classifier was set to k = 5. For training, we used a holdout validation technique. Each patient had for training data two epochs, a baseline epoch and a preictal epoch. In this chapter, baseline epochs are labeled with small letters, whereas the preictal epochs are labeled with capital letters.

Experiments were carried out using point-basis classification. As was stated previously, in point-basis classification each point counts as an example, whereas in the block basis, an entire segment (epoch) of data is considered a sample. Figure 4.14 shows

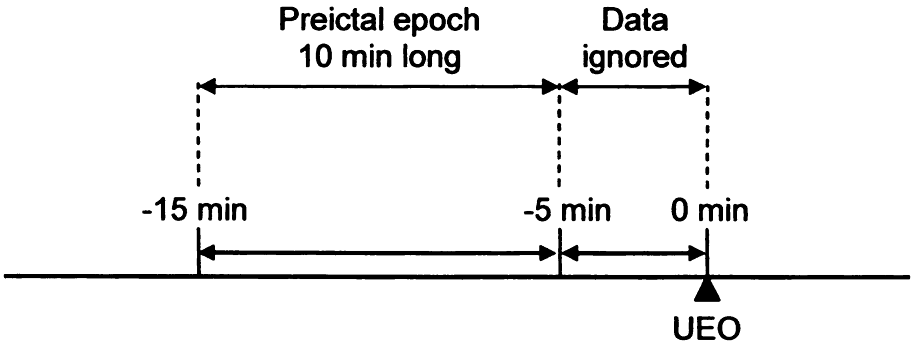


Figure 4.13 Illustration showing from where preictal epochs are extracted. It starts at 15 min before the UEO and ends 5 min before the UEO. The data between 5 min and the UEO are not used.

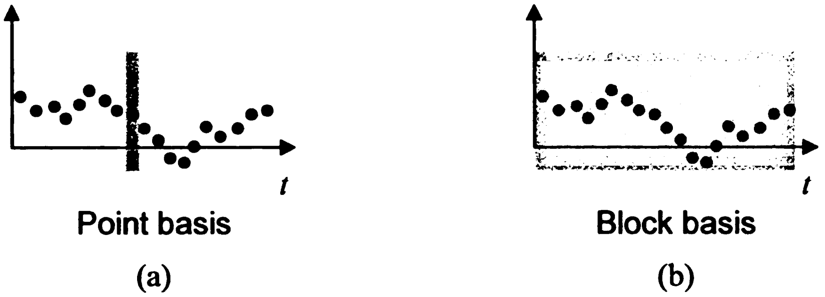


Figure 4.14 Point and block basis examples. A signal with a sampling of Δt is illustrated. (a) in point basis, each point (shaded area) is an example for the classifier while (b) in block basis a segment of points is considered an example (shaded area).

an example of each basis. In the figure, a hypothetical artificial signal sampled at Δt is presented, in which for the point basis, each point is considered an example for the classifier, while in the block basis, a segment (string of points) is considered an example for the classifier. From this, it is intuitive that while the block basis is more informative to make a decision, given that we are using more points, a lot more data will necessary to train the classifier. On the other hand, the point basis is not as informative as the block basis, given that we are using a point as an example; however, there are many data available to train the classifier. Here, a ten-minute long epoch is treated as a block. The problem with reporting results (to the drug delivery component) on the point basis is that we do not want to have an implantable device that medicates the patient each Δt that the classifier says a point in the signal is preictal. The question that comes up is whether we can predict the seizures or not. Therefore, we need to transform the point basis results to a block basis in order to integrate the decision from the classifier over a predetermined, fixed period. When the evolution of classifier's decisions is monitored over a fixed period, the device (classifier) can make a better decision on whether to medicate (*i.e.*, drug) the patient or not. To convert from point basis to block basis, we define a threshold, also known as “chance level”. The chance level to convert from point basis to block basis was set to 65% in order to increase the confidence of the decision beyond the noise floor that a "coin flipper" at 50% would create. If the sum of the point-basis classification results of a preseizure epoch is equal to or exceeds the threshold, then the epoch will be labeled preictal; otherwise, it will be categorized as baseline (nonseizure). Thus, an epoch for which 35% or more of the point-basis classifications classify it as baseline is labeled as baseline by the block-basis classifier; otherwise it is labeled preictal.

In the next subsections, for the sake of simplicity, we refer to the GPAF algorithm, which includes the GP algorithm and *k*-NN classifier, as the GPAF classifier or just GPAF, while the benchmark algorithm that includes the PCA transformation and *k*-NN classifier is called the PCA *k*-NN classifier or simply PCA. Of course, *k*-NN alone refers to a *k*-NN classifier used on full-dimensionality data, without preprocessing by PCA or using GP-evolved artificial features.

4.10.1 Patient A

For this patient, seizures were diagnosed as coming from left anterior hippocampus. Two kinds of seizure were clinically identified—simple partial and complex partial seizures. For the training stage, we used one baseline epoch and one preictal epoch. To validate the artificial feature, we had three baseline and three preseizure epochs available. The time delay obtained from the autocorrelation function was set to $\tau = 32$ (or 0.16 s). Thus, the terminal set for this patient was $\{x[n], x[n\text{–}32], x[n\text{–}64], x[n\text{–}96], x[n\text{–}128], x[n\text{–}160]\}$.

As stated above, the default prediction horizon was set to five minutes before the UEO. However, after several evaluations, the GPAF algorithm could not find a set of artificial features that achieved good performance. Therefore, we moved the prediction horizon to one minute before UEO. The GPAF algorithm was executed again and a set of features was found. Equation (4.32) shows the artificial features $y_1[k]$, $y_2[k]$, and $y_3[k]$:

$$y_1[k] = \sum_{n=1+400(k-1)}^{400k+400} \log_{10}(\log_{10}(x[n-32])).$$

$$y_2[k] = \sum_{n=1+400(k-1)}^{400k+400} \frac{\log_{10}(x[n-160])}{x[n-96]}. \tag{4.32}$$

$$y_3[k] = \sum_{n=1+400(k-1)}^{400k+400} (x[n-128])^4 .$$

Table 4.6 shows the results for the training epochs (point basis). The table also shows the results for the PCA *k*-NN and *k*-NN classifier benchmarks. Results for training data show that the transformation provided by the artificial features achieves an increment on the overall performance of the classifier.

Table 4.6 Results for training data for patient A (point basis).

Epoch	PCA *k*-NN	*k*-NN	GPAF
a	65.55%	65.55%	87.87%
A	67.89%	65.89%	67.10%
Overall	66.72%	65.72%	77.44%

Table 4.7 presents the results for the validation data. The results for validation data are on a block basis using the threshold previously defined (for point basis results, see Appendix I). The table shows the number of preseizure epochs (PSz), the number of baseline epochs (Bs), the number of false positives (FP), the rate of false positives as a percentage (FP rate), the number of false negatives (FN), and the rate of false negatives as a percentage.

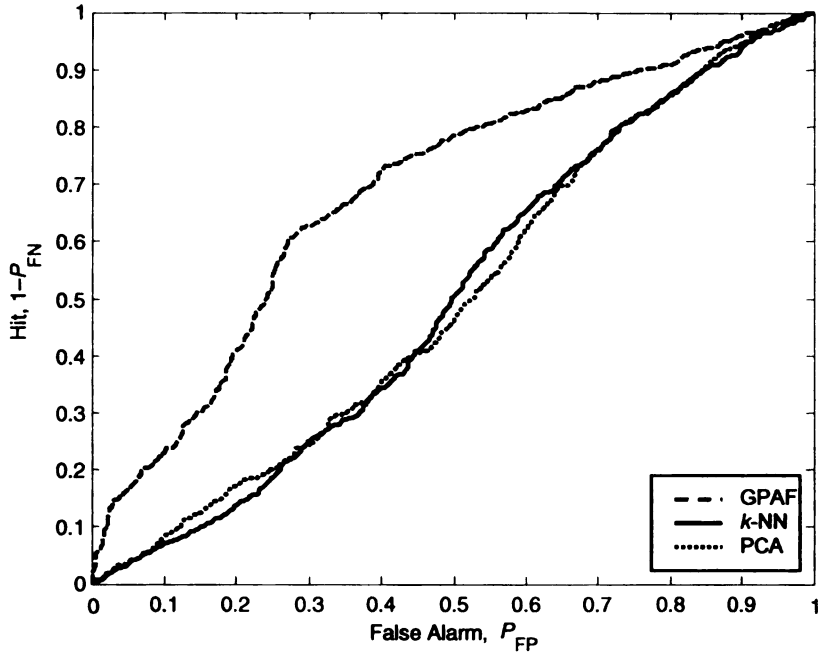


Figure 4.15 Receiver operating characteristic curves from testing data.

Table 4.7 Results for validation data for patient A (block basis).

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	3	3	0	0%	3	100%
PCA	3	3	0	0%	3	100%
GPAF	3	3	0	0%	0	0%

While all classifiers show good performance classifying all the baseline epochs correctly, the GPAF classifier was the only one that classified all three preictal epochs correctly. Figure 4.15 depicts the receiver operating characteristic (ROC) curve for the three different classifiers. Although not close to an ideal curve, it shows that the GPAF classifier exceeds the performance of the benchmark classifiers. Because the ROC curve is a visual tool, we compare the three classifiers by also calculating the area under the curve (AUC) of the ROC, an easier way to compare the performer among classifiers. The AUC values for the classifiers are $\text{AUC}_{k\text{-NN}} = 0.499$, $\text{AUC}_{\text{PCA}} = 0.499$, and $\text{AUC}_{\text{GPAF}} = 0.688$. Theoretically, the classifier cannot achieve an AUC worse than 0.5 because the class labels could always be reversed; thus, the $\text{AUC}_{k\text{-NN}} = 0.501$ and $\text{AUC}_{\text{PCA}} = 0.501$. The AUC statistic shows that the transformation provided by the artificial features produces a better classifier system than the benchmark classifiers.

4.10.2 Patient B

For patient B, seizures were found coming from the right inferior temporal neocortex and were diagnosed as complex partial. There were seven baseline epochs and five preictal epochs for training and validation purposes. One baseline and one preictal epoch were used to train the GPAF algorithm. The delay τ for this patient was set to $\tau = 30$ (or 0.15 s) based on autocorrelation calculations; thus, the terminal set was $\{x[n], x[n\text{–}30], x[n\text{–}60], x[n\text{–}90], x[n\text{–}120], x[n\text{–}150]\}$.Equation (4.33) shows the artificial features found for this patient by way of the GPAF algorithm.

$$\begin{aligned} y_1[k] &= \sum_{n=1+400(k-1)}^{400k+400} x[n]\cdot\left(x[n-30]\right)^2 \cdot \\ y_2[k] &= \sum_{n=1+400(k-1)}^{400k+400} \ln\left(x[n]\right) - x[n-90] + \left(x[n-150]\right)^2 \cdot \end{aligned} \tag{4.33}$$

Table 4.8 shows point basis results for training epochs along with the results for the benchmark classifiers, using a prediction horizon of five minutes. Table 4.9 shows the results for validation data on a block basis (see Appendix I for point basis results). All baseline epochs were classified correctly by all the classifiers. On the other hand, the *k*-NN and PCA *k*-NN classifiers missed two and three preictal epochs, respectively, while GPAF failed to predict only one seizure. Figure 4.16 shows the ROC curves for the three classifiers. The *k*-NN has an $AUC_{k\text{-NN}}$ = 0.735, the PCA *k*-NN an AUC_{PCA} = 0.609, and GPAF a AUC_{GPAF} = 0.874. Looking at the ROC and AUC statistics, it is clear that the GPAF classifier had a better performer.

Table 4.8 Results for training data of patient B (point basis).

	PCA *k*-NN	*k*-NN	GPAF
a	68.46%	59.06%	92.29%
C	72.15%	83.56%	90.94%
Overall	70.40%	71.24%	91.62%

Table 4.9 Results for validation data for patient B (block basis).

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	4	6	0	0%	2	50%
PCA	4	6	0	0%	3	75%
GPAF	4	6	0	0%	1	25%

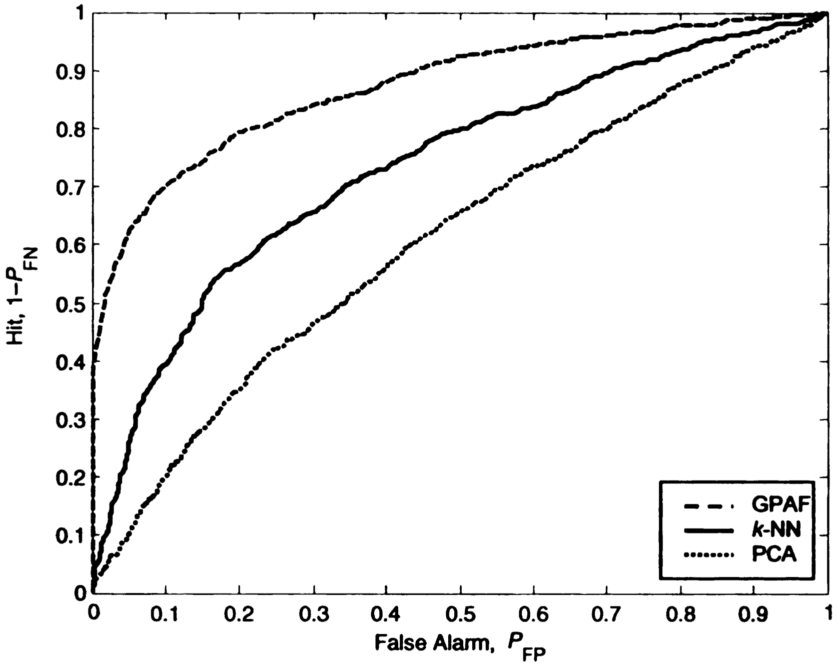


Figure 4.16 Receiver operating characteristic curves from testing data for patient B.

4.10.3 Patient C

This patient has complex partial seizures. Seizures were found to be multifocal, coming from right and left inferior frontal neocortex, left temporal neocortex, and left hippocampus. The delay was set to τ = 31 (or 0.155 s) and thus the terminal set was $\{x[n], x[n–31], x[n–62], x[n–93], x[n–124], x[n–155]\}$. For training and validation, we have eleven baseline and eleven preictal epochs.

For this patient, only one artificial feature was needed. Equation (4.34) shows the GPAF formula derived. As well as being relatively simple, it used only three of the six possible dimensions in the terminal set.

$$y[k] = \sum_{n=1+400(k-1)}^{400k+400} \log_{10}(x[n]) \cdot (x[n-155])^2 \cdot \log_{10}(\cos(x[n-124])). \tag{4.34}$$

Table 4.10 presents the training results of the GPAF equation and benchmark classifiers *k*-NN and PCA *k*-NN (the PCA algorithm reduced the dimensionality of the data to 1). Table 4.11 shows the results for the validation data. The benchmark classifiers classified correctly all the baseline epochs; however, the *k*-NN classifier missed one seizure whereas the PCA *k*-NN classifier missed four seizures. On the other hand, the GPAF classifier achieved a perfect classification. Figure 4.17 shows the ROC curves. We note that both *k*-NN and GPAF achieved near optimal curves (a sharp 90-degree elbow along the axes) in this case. From the figure, we can determine the AUC values, yielding $\text{AUC}_{k\text{-NN}} = 0.952$, $\text{AUC}_{\text{PCA}} = 0.778$, and $\text{AUC}_{\text{GPAF}} = 0.967$. The GPAF algorithm not only found a single artificial feature that achieved perfect classification (in the block basis sense), but also a single feature, which uses just three out of six inputs available, that yielded a higher accuracy classification in a point basis sense (see Appendix I).

Table 4.10 Results for training data for patient C (point basis).

Epoch	PCA *k*-NN	*k*-NN	GPAF
a	85.95%	98.66%	96.66%
D	66.88%	87.27%	97.66%
Overall	76.42%	92.97%	97.16%

Table 4.11 Results for validation data for patient C (block basis).

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	10	10	0	0%	1	10%
PCA	10	10	0	0%	4	40%
GPAF	10	10	0	0%	0	0%

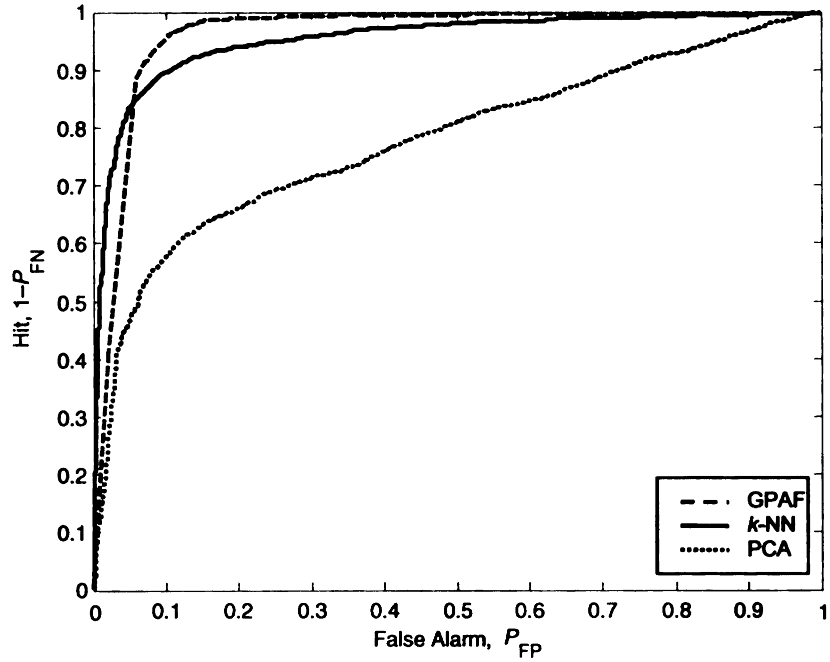


Figure 4.17 Receiver operating characteristic curves from testing data for patient C.

4.10.4 Patient D

This patient was diagnosed with complex partial seizures coming from the right inferior frontal region. The delay for this patient was set at $\tau = 27$ (or 0.135 s). Therefore, the terminal set was $\{x[n], x[n-27], x[n-54], x[n-81], x[n-108], x[n-135]\}$. For training and validation purposes, we had twelve baseline and eleven preictal epochs available. The prediction horizon for this patient was five minutes, the default horizon. Following the training procedure described earlier in this section, one baseline and one preseizure epoch were used. However, when the GPAF algorithm was executed, it found a set of features that achieved acceptable performance over the preictal epochs (six out of ten), but not over some baseline epochs. Because of this, we decided to use two five-minute-duration baseline epochs for the training process (plus the single preictal epoch) thus giving the

GPAF algorithm a better representation of the baseline data of this patient. Under these conditions, two artificial features were found that achieved a more practical performance. Equation (4.35) shows the genetically-designed features.

$$
\begin{aligned}
y_1[k] &= \sum_{n=1+400(k-1)}^{400k+400} \log_{10}\left((x[n-27])^2\right) \cdot x[n-27]. \\
y_2[k] &= \sum_{n=1+400(k-1)}^{400k+400} |x[n-54]| + \frac{x[n-54]}{x[n-81]}.
\end{aligned}
\tag{4.35}
$$

Table 4.12 shows the results for the training epochs. Table 4.13 presents the validation results (block basis) (see Appendix I for point basis results). It shows that the benchmark classifiers, while correctly classifying all baseline epochs, also misclassified all the preictal seizures (ten of them). On the other hand, the GPAF classifier misclassified one baseline epoch and just four preseizure epochs (six seizures were predicted correctly five minutes before the seizure occurred). Finally, Figure 4.18 shows the ROC curve for validation data (point basis). The ROC is not close to the ideal; however, it still does better than the benchmark classifiers. To measure this, we calculated the AUC values yielding AUC$_{k\text{-NN}}$ = 0.519, AUC$_{\text{PCA}}$ = 0.513, and AUC$_{\text{GPAF}}$ = 0.673. Although, the AUC for the GPAF is very far from the ideal 1, it is not nearly as close to the "coin-flipper" case as are the benchmark classifiers *k*-NN and PCA *k*-NN.

Table 4.12 Results for training data for patient D (point basis).

Epoch	PCA *k*-NN	*k*-NN	GPAF
c and *e*	70.57%	70.90%	83.61%
A	76.92%	72.24%	87.96%
Overall	73.75%	71.57%	85.79%

Table 4.13 Results for validation data for patient D (block basis).

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	10	10	0	0%	10	100%
PCA	10	10	0	0%	10	100%
GPAF	10	10	1	10%	4	40%

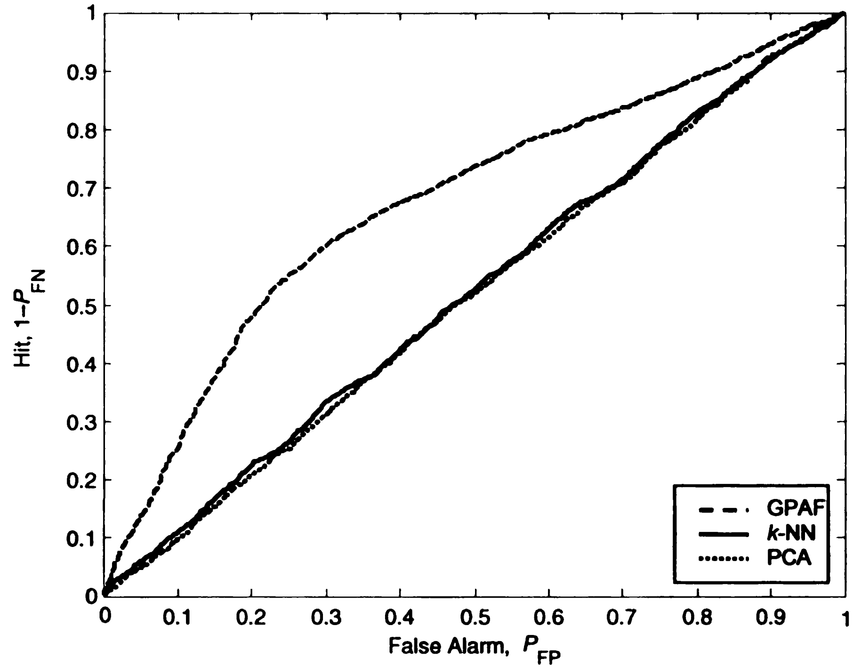


Figure 4.18 Receiver operating characteristic curves from testing data for patient D.

4.10.5 Patient E

Patient E was found to have complex partial seizures coming from the left hippocampus and left anterior temporal cortex. The EEG data include sixteen baseline epochs and the same number of preictal epochs. Preictal epochs were extracted five minutes before the UEO (unequivocal electrographic onset of seizure). The delay factor τ was set to $\tau = 49$ (or 0.245 s). Consequently, the variables in the terminal set are $\{x[n]$, $x[n–49]$, $x[n–98]$, *x*[*n*–147], *x*[*n*–196], *x*[*n*–245]}. For this patient, two artificial features were found, for which equations are shown in (4.36).

$$y_1[k] = \sum_{n=1+400(k-1)}^{400k+400} \log_2\left(\frac{x[n-147]}{x[n]}\right).$$
$$y_2[k] = \sum_{n=1+400(k-1)}^{400k+400} (\log_{10}(x[n]))^2. \tag{4.36}$$

Table 4.14 Results for training data for patient E (point basis).

Epoch	PCA *k*-NN	*k*-NN	GPAF
g	74.16%	84.9%	85.23%
P	76.51%	74.16%	94.3%
Overall	75.42%	79.6%	89.63%

Table 4.15 Results for validation data for patient E (block basis).

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	15	15	0	0%	6	40%
PCA	15	15	1	6.67%	6	40%
GPAF	15	15	1	6.67%	4	26.67%

Table 4.14 shows the results for the benchmark and GPAF classifiers evaluated with the training data. The GPAF algorithm created two artificial features that for the training data achieved about 10% higher correct classification than the benchmark schemes. Table 4.15 presents the results for the validation EEG data. The *k*-NN classifier misclassified one baseline epoch and six preictal epochs. The PCA *k*-NN classified all baseline epochs correctly, however, missed six preictal epochs. Finally, the GPAF classifier failed to categorize one baseline epoch correctly, but also misclassified four preictal epochs; in

other words, it was able to correctly predict two more seizures than the benchmark classifiers did.

Figure 4.19 shows the ROC curve. From this, we can calculate the AUC values yielding $\mathrm{AUC}_{k\text{-NN}} = 0.821$, $\mathrm{AUC}_{\mathrm{PCA}} = 0.749$, and $\mathrm{AUC}_{\mathrm{GPAF}} = 0.884$. Although the *k*-NN classifier had good performance, the GPAF classifier had a still better AUC value. Also, we need to remark that the *k*-NN classifier used all dimensions (six of them) in the terminal set, whereas the GPAF features used just two inputs (states)—that is, $x[n]$ and $x[n–147]$.

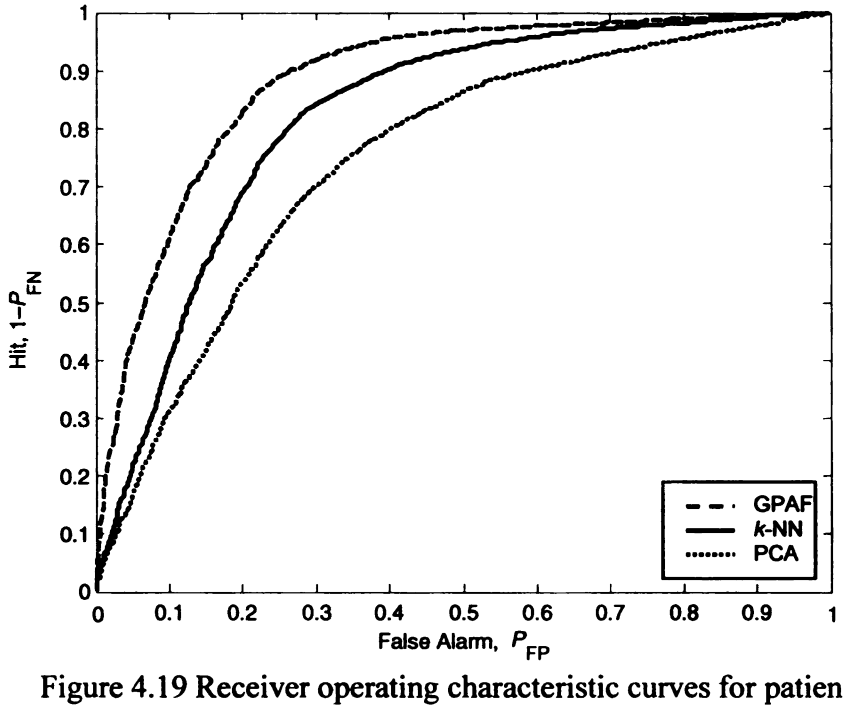


Figure 4.19 Receiver operating characteristic curves for patient E.

4.10.6 Patient F

This patient was diagnosed with complex partial seizures coming from bilateral regions. Some seizures came from the left hippocampus and other seizures came from the right temporal region. This patient, though, had seven seizures in the data; one epileptic seizure occurred very early in the recording, leaving us without enough data to extract ten minutes prior to the UEO mark. Therefore, we had six preictal epochs and eight baseline epochs for training and validation purposes. Because the delay was equal to $\tau = 55$ (or 0.275 s), the elements of the terminal set are $\{x[n], x[n-55], x[n-110], x[n-165], x[n-220], x[n-275]\}$. Equation (4.37) shows the GP artificial feature for this patient. Table 4.16 shows the training results. The artificial feature achieved an overall improvement of 7% in classification accuracy over the benchmark classifiers.

$$y[k] = \sum_{n=1+400(k-1)}^{400k+400} \cos(\log_{10}(x[n] - x[n-55])). \tag{4.37}$$

Table 4.17 shows the results for the validation data. Although the k-NN classifier categorized all baseline segments correctly; it also misclassified one preictal epoch, falling short by 0.79% (see Appendix I, epoch *F*) to be classified correctly at five minutes before the UEO. On the other hand, the PCA k-NN and GPAF classifiers categorized all epochs correctly. However, when the ROC curves are plotted in Figure 4.20, it shows that the ROC curve for the k-NN classifier is closer to the ideal ROC curve than that for the PCA k-NN classifier. In fact, the AUC values are $\text{AUC}_{k\text{-NN}} = 0.979$ and $\text{AUC}_{\text{PCA}} = 0.834$.

Table 4.16 Results for training data for patient F (point basis).

Epoch	PCA *k*-NN	*k*-NN	GPAF
a	84.56%	94.63%	86.91%
A	68.79%	78.19%	96.31%
Overall	76.59%	86.45%	93.65%

Table 4.17 Results for validation data for patient F (block basis).

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	5	7	0	0%	1	20%
PCA	5	7	0	0%	0	0%
GPAF	5	7	0	0%	0	0%

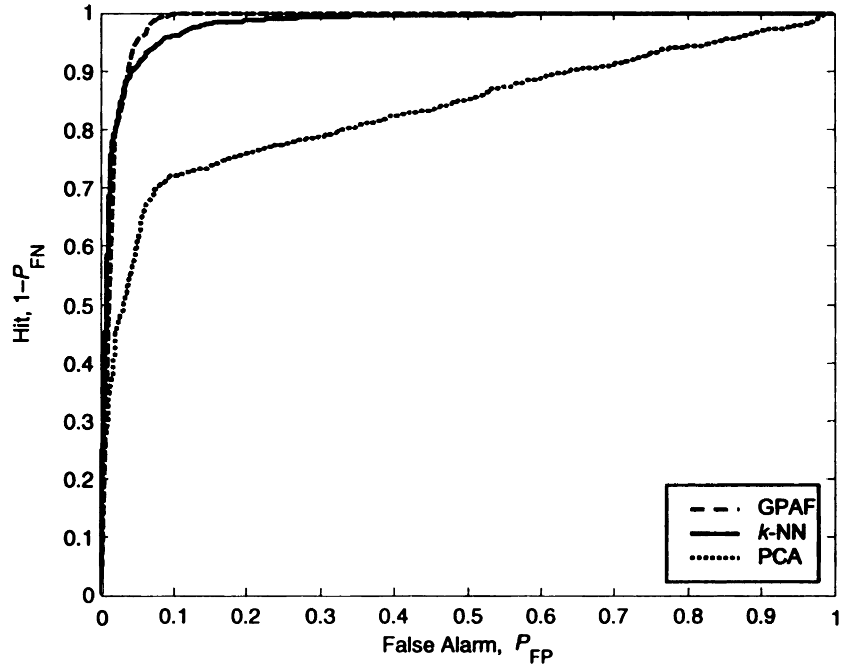


Figure 4.20 ROC curves from testing data for patient F.

None of those, however, surpassed the performance of the GPAF classifier, for which $\mathrm{AUC_{GPAF}} = 0.985$.

Equation (4.37) shows that GP artificial feature used just two dimensions (or states), $x[n]$ and $x[n-55]$, while the *k*-NN classifier necessarily used all six dimensions in the terminal set. In other words, the GPAF algorithm created an artificial that performs slightly better than the *k*-NN classifier while using fewer—in this case, two—dimensions.

On the other hand, although the PCA *k*-NN classifier classified all epochs correctly, it reached an overall point basis classification of 78.2%, while the GPAF classifier reached 94.45%. Therefore, the GPAF is not just classifying epochs correctly, but also classifying those with high-accuracy, allowing a larger margin for error.

4.10.7 Patient G

Like most patients, this patient also suffered complex partial seizures, which came from the left hippocampus and left inferior temporal neocortex. This patient had 31 seizures, of which twenty seizures were subclinical seizures, so only eleven were clinical seizures. Although it is known that subclinical seizures are disruptive to the brain [59], in this work, our target is the prediction of clinical epileptic seizures. In the next chapter, we discuss subclinical seizures more fully, in the context of the seizure *detection* (rather than prediction) problem.

There were eleven preseizure epochs and eleven baseline epochs for training and validation processes. The delay was $\tau = 33$ (or 0.165 s) and terminal set was $\{x[n], x[n-33], x[n-66], x[n-99], x[n-132], x[n-165]\}$. Initially, in the training phase, we used one baseline epoch and one preictal epoch, yielding a set of artificial features that had good performance on preictal data; however, they displayed poor performance on non-training baseline data, classifying some epochs correctly but other epochs completely wrong. Therefore, we decided, again, to use two five-minute-duration baseline epochs from two different timeframes in the training data in order to give to the GPAF algorithm a better representation of the baseline data of this patient. Using these data, the GPAF algorithm was able to design the features shown in Equation (4.38).

$$\begin{gathered}
y_1[k] = \sum_{n=1+400(k-1)}^{400k+400} \log_2(x[n-66]) \cdot |x[n]| . \\
y_2[k] = \sum_{n=1+400(k-1)}^{400k+400} \cos\left((x[n-66])^{\frac{1}{4}} \right) + \left| x[n-99] - (x[n] + \log(x[n])) \right| .
\end{gathered} \tag{4.38}$$

Table 4.18 Results for training data for patient G (point basis).

Epoch	PCA *k*-NN	*k*-NN	GPAF
a and *h*	49.83%	67.89%	79.26%
A	50.17%	38.8%	81.94%
Overall	50%	53.35%	80.36%

Table 4.19 Results for validation data for patient G (block basis).

	PSz	Baseline	FP	FP rate	FN	FN rate
k-NN	8	8	2	25%	4	50%
PCA	8	8	0	0%	8	100%
GPAF	8	8	2	25%	3	37.8%

Table 4.18 shows the results for the GPAF equations. It also shows the results for the benchmark *k*-NN and PCA *k*-NN classifiers. In Table 4.19, the validation data results are presented. The *k*-NN classifier misclassified two baseline epochs and four preictal epochs. The PCA *k*-NN classified all baseline epochs correctly but misclassified all preictal seizures (eight of them). Finally, the GPAF classifier misclassified two baseline and three preictal epochs. The GPAF classifier, compared against the *k*-NN classifier, could predict one more seizure correctly. Figure 4.21 shows the ROC curves. The areas

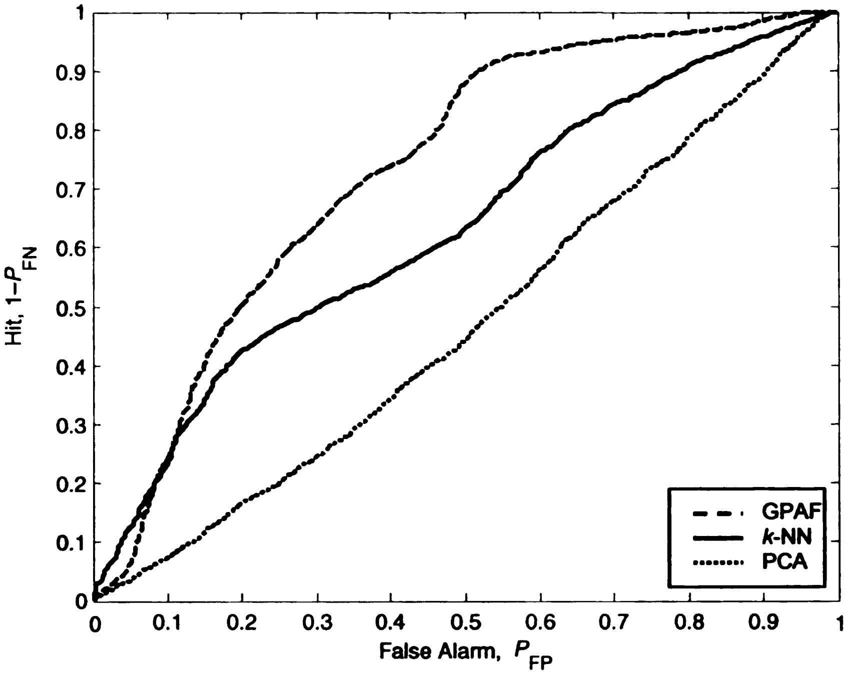


Figure 4.21 Receiver operating characteristic curve for validation data.

under curve are $\text{AUC}_{k\text{-NN}} = 0.632$, $\text{AUC}_{\text{PCA}} = 0.469$ (or 0.531 by the worst binary classifier argument), and $\text{AUC}_{\text{GPAF}} = 0.733$. Again, the GPAF classifier achieved better performance than the benchmark classifiers by obtaining an overall 9.64% increase in correct classification (for the point basis results, see Appendix I) over the *k*-NN classifier and an overall 18.33% increase in correct classification (on the point basis) over the PCA *k*-NN. This means that the GPAF classifier classified epochs correctly with higher accuracy than its benchmark counterparts.

4.11 Discussion

In this section, we summarize the results described in the preceding subsections. Table 4.20 shows the overall results where the performance of the benchmark classifiers and the

GPAF classifier are measured counting all the epochs. We have a total of 59 baseline epochs and 55 preseizure epochs. The table shows that the number of baseline epochs misclassified was small, with the GPAF classifier misclassifying more, 4 out of 59, for a false positive rate of 6.78%. On the other hand, the *k*-NN classifier showed poor prediction performance by missing 27 out of 55 seizures (or it correctly predicted 28 seizures) for a false negative rate of 49.09%. The other benchmark, the PCA *k*-NN classifier, obtained the worst performance, misclassifying 34 of 55 preseizure epochs (it correctly predicted only 21 seizures). It resulted in a 61.82% false negative rate. The high false negative rates yielded by the benchmark classifiers make them impractical for clinical purposes. On the other hand, the GPAF classifier was able to predict 43 preictal epochs out of 55 (it missed twelve of them), yielding a 21.81% false negative rate. These results make the GPAF classifier by far the best of the three classifiers, for this purpose.

Table 4.20 Overall results for all three classifiers.

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	55	59	3	5.08%	27	49.09%
PCA	55	59	1	1.69%	34	61.82%
GPAF	55	59	4	6.78%	12	21.81%

It is clear that the artificial features crafted by the GP algorithm play an important role in getting a better separation of the classes and thus obtaining a better classification. Also, we make clear that the block basis classification depends on what value is set for the threshold used to transform from point basis to block basis classification. For all patients evaluated above, this threshold was set to 65%. However, if we move the threshold to a lower value, we would have an increase in the number of preictal epochs classified correctly. For example, consider Patient D. If we set the threshold to 50%, the block basis classification rate changes drastically. Table 4.21 presents the results with the new threshold. The *k*-NN classifier now misclassifies two additional baseline epochs, but is able to predict four seizures, unlike when using 65%, when all seizures were missed. The PCA classifier misclassified most of the baseline epochs (seven of them) while classifying correctly seven preseizure epochs. Finally, the GPAF classifier misclassified two more baseline epochs and classified correctly one additional preictal epoch (it missed only three seizures). A reduction of the false negative rate was obtained at the expense of an increase of the false positive rate. However, the GPAF still was not drastically affected by the new threshold given its superior point basis classification.

Table 4.21 Results for validation data on block basis for patient D using 50% threshold.

	PSz	Bs	FP	FP rate	FN	FN rate
k-NN	10	10	3	30%	6	60%
PCA	10	10	7	70%	2	20%
GPAF	10	10	3	30%	3	30%

The overall point basis classification using the GPAF classifier was 65.10% (with a standard deviation ± 32.16%), whereas the overall point basis classification for the *k*-NN was 51.49 ± 15.32%, and the overall classification using the PCA *k*-NN was an average of 50.86% with a standard deviation ±15.44%. The mean of the GPAF classifier was roughly 15% over the average of the benchmark classifiers (the standard deviation was high because one baseline and three preictal epochs were poorly classified). This means that the GPAF classifier yielded a high-accuracy classification, so we were able to define the threshold for the block-basis decision with higher confidence.

The overall average of the point-basis classification accuracy of GPAF across all patients was 78.55 ± 25.26%, whereas the PCA *k*-NN classifier achieved 62.46 ± 17.75% and the *k*-NN classifier achieved 69 ± 23.53%. The GPAF classifier surpassed the best benchmark classifier, the *k*-NN, by about 9% in the mean. The GPAF classifiers achieved an average prediction time of 4.43 minutes (before UEO).

4.12 Statistical Test

However, this comparison is not sufficient to establish whether our algorithm is better or not. Therefore, we wished to perform a statistical test. To determine whether the statistical test should be parametric or nonparametric, we plotted the probability density functions (PDF) of the classification of the epochs across all patients (114 epochs in total) for the three different algorithms. Figure 4.22 depicts the PDFs.

From the figure, it is easy to determine that the PDFs are not normally distributed; a nonparametric statistic test should be employed. We selected the Mann-Whitney U test [81], also called the Wilcoxson test. Although nonparametric tests are not as powerful as parametric tests, according to Conover [13], a comparison of the relative efficiency between the Mann-Whitney test and the two-sample *t*-test, which is parametric, is close. We are interested in knowing whether the GPAF algorithm's performance is equal to or better than the benchmark algorithms' performances, across the ensemble of data. Thus, the hypothesis for the statistical test is as follow:

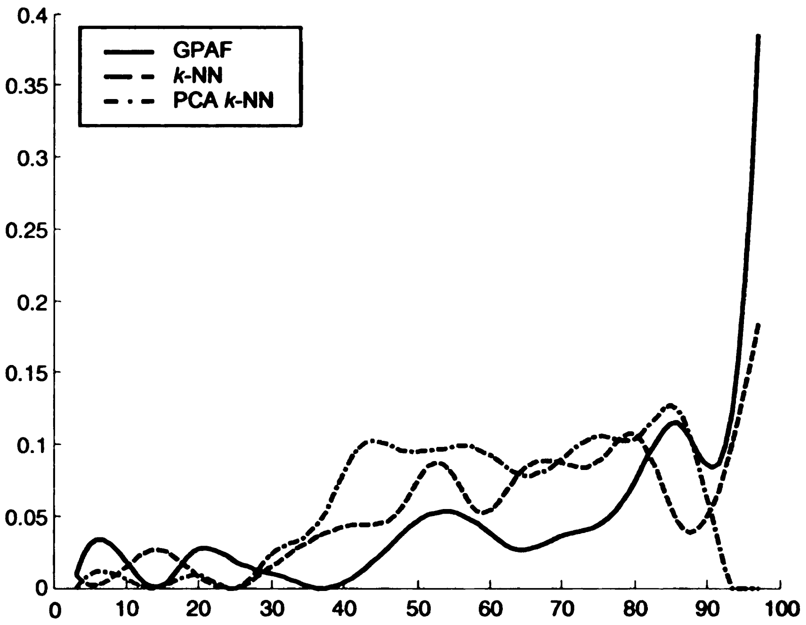


Figure 4.22 PDFs for the percentage of correct classification of the epochs for the three algorithms (from Appendix 1).

$$
\begin{aligned}
&H_0\text{: } E[\mathrm{X}] = E[\mathrm{Y}] \\
&H_a\text{: } E[\mathrm{Y}] < E[\mathrm{X}],
\end{aligned}
\qquad (4.39)
$$

where H_0 is the null hypothesis, H_a the alternative hypothesis, X denotes the epochs processed with the GPAF algorithm, Y denotes the epochs evaluated with the PCA *k*-NN and is $E[]$ is the expected value. Also, a Wilcoxson test was established between the GPAF classifier and the *k*-NN classifier with the hypothesis

$$
\begin{aligned}
&H_0\text{: } E[\mathrm{X}] = E[\mathrm{Z}] \\
&H_a\text{: } E[\mathrm{Z}] < E[\mathrm{X}],
\end{aligned}
\qquad (4.40)
$$

where Z denotes the epochs processed by the *k*-NN classifier. The level of significance α was set to 0.05. Because 114 samples (number of epochs) are considered a large number, the P-value is calculated using a normal (z) distribution (by means of the central limit theorem). For the hypothesis in Equation (4.39), the statistical test value was 6.59 and the P-value was $4.54\times10^{-11} \approx 0$. The critical value (the value that must be exceeded in order to reject the null hypothesis) for one-tailed, $\alpha = 0.05$, was 1.659. Therefore, the null hypothesis is rejected and the alternative hypothesis is accepted. That is, the expected value (or mean) of the GPAF classification is larger than the mean of the PCA *k*-NN results. In the second case, for the hypothesis in Equation (4.40), the statistical test value was 3.61 and the P-value was 3.09×10^{-4}, which indicates that the null hypothesis was rejected and the alternative was accepted, because the critical value was 1.659. In other words, the mean of the GPAF results was larger than that of *k*-NN classifier. Therefore, the statistical tests state that the improvement in the mean performance of the GPAF over the other two methods is statistically significant to a very high level, well beyond the 5% level represented by a critical value of 1.659. Of course, the statistical test results do not necessarily imply that the GPAF algorithm is better than the benchmark classifiers but just simply assure us that the GPAF results are better.

4.13 Generic GPAF Equations

Ideally, it is desirable to find not just a good predictor for a given patient, but also a general set of features that will work for all patients by tuning a set of parameters to customize the predictor for the needs of such patients. Although such an approach has been proposed in the past, little effort has been exerted in this direction—in part because of the lack of an algorithm that allows researchers to explore the precursor patterns to predict seizures directly from the raw EEG data.

In this section, we present the results of an experiment intended to find a set of artificial features or a generic predictor model that yields acceptable results. In the experiment, we set the number of artificial features to three. A holdout validation technique was used for training and validation purposes, where the baseline and preictal epochs were clipped to five minutes, in order to reduce the computation load. Preictal epochs were the five minutes immediately leading to the UEO, except for patient A, for whom EEG data were extracted up to one minute before the UEO. Additionally, we used the same delay values previously determined for each patient in the terminal set. That is, during training, when a given patient was being evaluated, the algorithm set its delay values in the artificial features to integer multiples of that patient's autocorrelation time. The parameters of the GP were set the same as before except for the number of generations, which was increased to 250 because of the complexity of the problem.

Equation (4.41) shows the artificial features found by the GPAF algorithm. Surprisingly and interestingly, the GPAF algorithm came up with a set of artificial features that yielded an acceptable, but also, surprising, result.

$$y_1[k] = \sum_{n=1+400(k-1)}^{400k+400} x[n] + \log_2\left(\ln\left(x[n-2\tau]\right)\right) + \ln\left(x[n-\tau]\right).$$

$$y_2[k] = \sum_{n=1+400(k-1)}^{400k+400} x[n-\tau] + x[n-5\tau]. \tag{4.41}$$

$$y_3[k] = \sum_{n=1+400(k-1)}^{400k+400} \left|x[n-\tau]\right|.$$

Table 4.22 Validation results for seven patients.

Patient	PSz	Bs	FP	FP rate	FN	FN rate
A	3	3	0	0%	1	33%
B	4	6	0	0%	2	50%
C	10	10	0	0%	0	0%
D	10	10	1	10%	3	30%
E	15	15	1	10%	4	26.67%
F	5	7	0	0%	0	0%
G	8	8	2	25%	7	87.5%
Overall	55	59	4	4.5%	17	22.71%

Table 4.22 shows the results for the validation data. Overall, the set of features misclassified just 4 of out 59 baseline epochs for a rate 4.5% of false positives, whereas 17 preictal epochs out of 55 were not classified correctly for a false negative rate of 22.71%.

Comparing these results against those of the previous section, where a set of features was optimized for each patient, reveals that Equation (4.41) misclassified the same number of baseline epochs, while missing five seizures more than were missed collectively by the artificial features shown in the previous section. Most of those misclassifications were on patient G, in which just one seizure was predicted. Similar to results presented before for patients C and F, the Equation (4.41) also achieved perfect classification. The average overall accuracy achieved was 76.79 ± 27.56%, roughly 2% less in the mean than that achieved by the "customized" artificial features, although the standard deviation was also a little higher. The results from Table 4.23 demonstrate that it is possible to optimize a generic model to capture the characteristics of individual patients. These results might pave the way for more research on a general model that captures the dynamics hidden behind the epileptic seizures.

4.14 Training the GPAF Algorithm Using the Leave-One-Out Method

It is generally desirable to train a classifier using the leave-one-out cross validation method. The leave-out-one method (LOO) tends to extract from a given dataset the model with the highest classification accuracy, as it uses the largest possible training set against which to perform each remaining testing classification, but at the price of the highest requirements for computation. This computational expense is precisely the reason that the LOO method was not chosen as the training method for most runs of the GPAF algorithm. Nonetheless, as a way to reinforce this work, we provide the results for a patient evaluated with the GPAF algorithm using a leave-one-out method.

Data from patient B were selected for the experiment. The total number of epochs available, twelve, created too much processing load for use in a full LOO method; therefore, three baseline epochs (*a*, *c*, *e*) and three preictal epochs (*C*, *D*, *E*) were randomly selected from the twelve epochs available for this patient. Since only these six datasets will be used for both training and validation, the analysis corresponds to LOO for that universe of six datasets, and demonstrates the sorts of gains in accuracy available with more computational resources.

To train, we set the algorithm to leave one baseline and one preictal epoch out for validation and to use the remaining four for training, doing this with all combinations of baseline and preictal epochs. The parameters for the GP algorithm and *k*-NN classifier were set the same as before. The number of features was set to two, thus producing two artificial features. The terminal set for the patient was the same, given that the delay value was unaltered.

Equation (4.42) shows the equation found by the GPAF algorithm using LOO cross validation for patient B. The first artificial feature $y_1[k]$ is the signal energy of the state $x[n\text{–}60]$. Table 4.23 provides the results using the LOO cross-validation, where P_c denotes the percent of correct classification.

$$\begin{aligned} y_1[k] &= \sum_{n=1+400(k-1)}^{400k+400} (x[n-60])^2 . \\ y_2[k] &= \sum_{n=1+400(k-1)}^{400k+400} x[n-90] \cdot (x[n-150] - x[n-120]). \end{aligned} \tag{4.42}$$

Table 4.23 Results using leave-one-out cross validation.

Epochs	P_c
a, C	79.3
c, D	84.31
e, E	85.14
a, D	77.63
a, E	68.78
c, C	73.96
e, C	80.63
c, E	78.46
e, D	86.14
Average	79.59 ± 5.94 %

According to the results from the table, using the LOO method, the GPAF equations achieved good, low-variance classification results even when epoch *E*, which in holdout validation was misclassified, is included (see Appendix I, patient B). These results show that to design good artificial equations with high accuracy and low-variability, the leave-one-out method can definitely be useful. However, when running many experiments, such as was done here, across many patients, it would have required an impractically large computer resource. In this work, there were large archives of EEG signals, containing many epochs, but use of all of them would have consumed more computer time than was available. This LOO run demonstrates that better accuracy could be obtained for clinical application if significant computing resources were available for analyzing each patient's EEG signals, or if a satisfactory cross-patient model could be found.

Finally, we want to remark that the GPAF algorithm did not just design artificial features with unknown physical meanings, but was able to produce some interpretable equations that are well known to researchers in epilepsy. In this case, the GPAF algorithm designed a feature measuring the energy of the signal, which is one of the conventional features used for prediction and detection of epileptic seizures.

4.15 Comparison with Another Proposed Method

Because the benchmarks used for comparison with the GPAF algorithm in this chapter are neither proven nor well-known methods for extracting features to predict epileptic seizures, in this section we would like to compare our results with another reported method that used the same EEG database. In Chapter 2, we briefly discussed a method proposed by D'Alessandro *et al.* [15], in which a GA was used to find a set of conventional features (and channels or electrodes) that best predicted seizures, while at the same time having a low false positive rate. Table 4.24 shows the average results reported in their work, where "baseline tested" is the number of ten-minute-duration epochs used for the experiments, "preictal tested" is the number of preictal epochs used for the experiments, "FP rate" is the fraction of baseline epochs misclassified, "FN rate" is the fraction of preictal epochs misclassified, and FPH is the rate of false positives per hour. The table also displays the results obtained by the GPAF algorithm's customized equations and the generic (cross-patient) GPAF equations (denoted as "Gen. GPAF"). The table shows that both GPAF-generated sets of features had better performance than the D'Alessandro's method. However, it is clear that we tested a lower number of baseline epochs. Additionally, D'Alessandro *et al.* reported an average prediction time of 3.46 minutes (before UEO), whereas the GPAF and generic GPAF achieved an average prediction time of 4.43 minutes (before UEO).

We need to consider that in [15], the authors used 70% (they had 46 preictal epochs) of the preseizure data available for training, while the rest was used for validation purposes. For this reason, surrogate preictal data sets were artificially generated to increase the amount of the preictal data available. The authors did not indicate how many surrogate epochs were generated (therefore, the preictal tested cell just indicates that more than 46 seizures were used). In contrast, we used roughly 11% percent of the preictal data available (we had 62 preictal epochs in total) for training purposes, and 89%

of the data for validation purposes. This reliance on only a smaller set of training data appears to be another advantage of the GPAF method.

Even though the comparison is not under completely identical conditions, since training and validation sets are not the same, the data are drawn from the same pool, and the results favorable over pre-existing methods. Results do not demonstrate that our algorithm outperforms D'Alessandro's approach in all respects, but simply indicate that the GPAF algorithm produced promising results that are worthy of further investigation.

Table 4.24 Comparison of D'Alessandro's algorithm and GPAF results.

System	Number Patients	Baseline Tested	Preictal Tested	FP Rate	FN Rate	FPH
D'Alessandro	4	276	+46	9.53	37.5	0.278
GPAF	7	59	55	6.78	21.81	0.068
Gen. GPAF	7	59	55	6.78	30.91	0.068

4.16 Summary

This chapter presented the application of the GPAF algorithm, introduced in Chapter 3, to design of a set of artificial features helping to predict epileptic seizures. The theory behind the reconstruction of the state-space trajectory for a chaotic system was explained early in the chapter. We reconstructed the state-space trajectory for a time-varying empirical model of the EEG. Those reconstructed dynamic variables were input to the GP algorithm, which designed a set of artificial features to separate the baseline data from the preictal data. The discrimination task was carried out by the classifier component of the GPAF algorithm—in this case, a *k*-nearest neighbor classifier was used. By implementing the GP algorithm as the search engine for artificial features, using the delay embedding scheme as input data to the GP, and adding more functions to the function set of the GP, we addressed every one of the suggested recommendations by the author in [30] and stated in Section 3.1.

The algorithm was evaluated over seven patients: six with a prediction horizon of five minutes, and one with prediction horizon of one minute before the UEO. Results from two benchmark classifiers were used to compare against the results of the GPAF algorithm. In all the experiments carried out, the GPAF-designed features surpassed the performance of the benchmark classifiers, yielding high accuracy on a point basis as well as on a block basis.

Although the training for the customized artificial equations was carried out using holdout validation, because of computing resource limitations, we carried out an experiment using a reduced LOO cross-validation on six epochs from patient B, which produced a pair of artificial features that yielded good results with high confidence.

Results in this chapter also illustrate that, although a unified model for the epilepsy prediction problem is still far from trivial, the design of optimized, nontrivial features for each patient makes viable a custom-implantable device with the artificial features integrated into the chip.

Finally, we evaluated an algorithm to design a generic set of artificial features that works across patients, using only one pre-calculated patient-specific characteristic, the delay parameter. The results were much better than expected, yielding three simple equations that predicted the epileptic seizures across patients, better than the benchmark classifiers. These equations are a step forward in the search for a general model for epilepsy prediction.

In the next chapter, we use the GPAF algorithm to design features for detection (rather than prediction) of epileptic seizures—a relatively easier problem, yet still far from trivial.

Chapter 5

Detection of Seizure Onset by Means of GPAF

In this chapter, we discuss the development of features that can help detect, with high accuracy, the onset of seizures. We then present the application of the GPAF algorithm for the detection case. It will be implemented in different patients such that the algorithm can create a feature that has a detection rate similar to or better than that of the benchmark classifier.

5.1 Why Detect Onset of Seizures?

At first glance, detection of seizure onset does not seem very beneficial, from a medical point of view, if it is compared with the obvious benefit offered by prediction of such attacks. However, the truth is that detection of seizure onset is as important as prediction of seizures. There is one benefit from a diagnostic perspective and two potential benefits from a therapeutic perspective:

- *To diagnose what area of the brain is generating the seizures.* Patients with epileptic disorders that are untreatable with current medications are candidates for surgery. Evaluation of surgical candidates consists of hospitalization, stopping all anti-seizure medications and any other such treatment, inserting EEG probes, and observing patients to determine in what area of the brain seizures are generated. If seizures are generated in critical areas of the brain, the subject is rejected for surgery; otherwise, surgery is possible. In some cases, the patient presses a button each time he or she feels that an attack is imminent. This procedure can be done by only about 50% of the patients with MTLE who maintain volitional control of their finger at the start of seizure [5]. Monitoring also includes video recordings in order to visually confirm clinical events. Review of such video is an exhaustive, tedious, and time consuming process. In the video recording analysis, labeling of seizures is limited to those that are clinical seizures (*i.e.*, the patient exhibits clinical symptoms of seizure during the convulsion); therefore, subclinical seizures—those where the patient does not exhibit any externally observable symptoms—are not labeled as seizures. Developing an algorithm that can detect all the (clinical and subclinical) seizures and record the time when the seizures occurred, regardless of the patient's display of clinical symptoms, would give the neurologists a more accurate "profile" of the brain of the patient. It could also allow performing surgeries after a shorter period of observation.

- *Early detection can trigger immediate therapy.* If early detection of seizures can be achieved by a detection algorithm, therapy (*e.g.*, drug or an electrical stimulus) from an implantable device could be initiated, avoiding a seizure's complete evolution and allowing the patient to continue doing ordinary daily activities like driving or talking, without any clinical consequence like loss of consciousness or suffering of convulsions.
- *A chronic therapeutic effect on patients.* Just as traditional medications, used to attempt to control or relieve a given condition, must often be administered over an

extended time period (*e.g.*, antibiotics usually require days to weeks to kill bacteria), similarly extended treatment may be needed to see all the beneficial therapeutic effects of an implantable detection device firing an electrical stimulus each time the device detects epileptiform activity (*i.e.*, abnormal behavior of the EEG signals). It is hypothesized that such extended treatment may have a chronic therapeutic effect on the patient, reducing the number of epileptic seizures that are suffered and thus improving the quality of life of the patient.

5.2 Ictal EEG Signals

Unlike preictal signals, in which no obvious sign is visible, in *ictal* signals, noticeable changes can be observed throughout the evolution of the seizure. This means that detection of EEG seizures is somewhat easier than prediction of seizures. The ictal period includes the Unequivocal Electroencephalographic Onset (UEO) of the seizure, the indicator observable by EEGers that denotes that a seizure is starting. Figure 5.1 depicts an ictal signal of two minutes duration.

Patients' data used for these experiments were the same as in Chapter 4; thus, so are their clinical profiles. In the training phase of the experiments, we used baseline and ictal epochs of two minutes duration, thus containing 24,000 points each (recall EEG signals were sampled at 200Hz). For ictal epochs, we used one minute of data before the UEO and one minute of data after it (for example, see Figure 5.1). The training set for each patient was three baseline epochs and three ictal epochs.

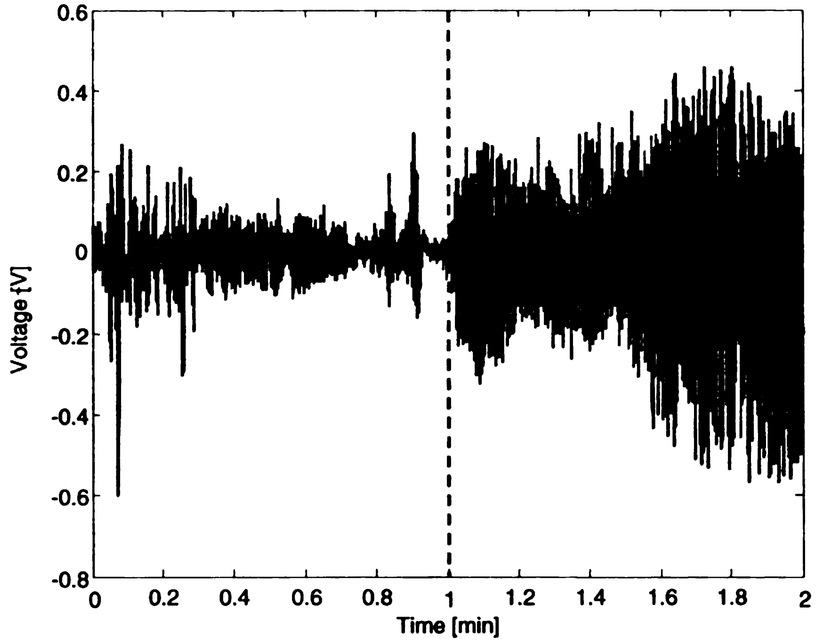


Figure 5.1 A two-minute duration ictal signal. The dashed line indicates the UEO.

Whereas in Chapter 4 the extracted baseline and preictal epochs of ten minutes duration were used for validation purposes, in this chapter, in the validation stage, we used the entire EEG archive available for each patient. Because the detector is less sensitive than the predictor, the detector should be able to ignore artifacts that might be considered by the predictor as precursors of an epileptic seizure. Yet, after classification of the validation data was completed, further analyses were conducted to verify that any false positives were not due to one of those artifacts.

5.3 Detection with GP Artificial Features

Although detection of seizures is not as hard as prediction of seizures, the problem is still far from trivial. Many algorithms and methods have been developed to do such tasks, of which several were cited and briefly discussed in Chapter 2; however, most algorithms still fail to detect some seizures. In this chapter, we apply the genetic programming artificial feature (GPAF) algorithm to create a feature that finds a pattern or gives a transformation that delivers more separability between baseline and ictal classes, with a similar or better classification rate than the benchmark classifier that was explained in Section 4.7. In other words, this transformation will make the task of the classifier (*i.e.*, categorization) easier; therefore allowing detection of the seizures with higher accuracy. The implementation of the GPAF algorithm for detection is exactly as was explained in Section 4.4, with the GP algorithm using the same cost function as presented in Section 4.6.

Unlike the prediction case, where a whole preictal epoch was labeled as belonging to one class, in detection, we need an additional labeling detail because the ictal data are clipped. For ictal epochs, the first minute before the UEO was labeled as baseline data and the rest of the epoch (*i.e.*, the minute of EEG data after the UEO) was labeled as ictal. Figure 5.2 depicts the new classification labeling (the EEG signal was normalized to one). Figure 5.2 makes this clear. The one-minute-EEG data before the UEO has an amplitude similar to baseline data. Given that for the detection case a large part of the information used to detect seizures is provided by the amplitude of the signal, it is reasonable that our previous labeling (of the whole epoch as belonging to one class) would be confusing to the GPAF (or any other) algorithm.

The GP algorithm used the same fitness function as in the previous chapter. Therefore, we measured metrics such as the numbers of false positives and false negatives. In this chapter, we sought to achieve a false negative rate of 10% or less and no more than two false positives per day (or a false positive rate of 0.0833 per hour).

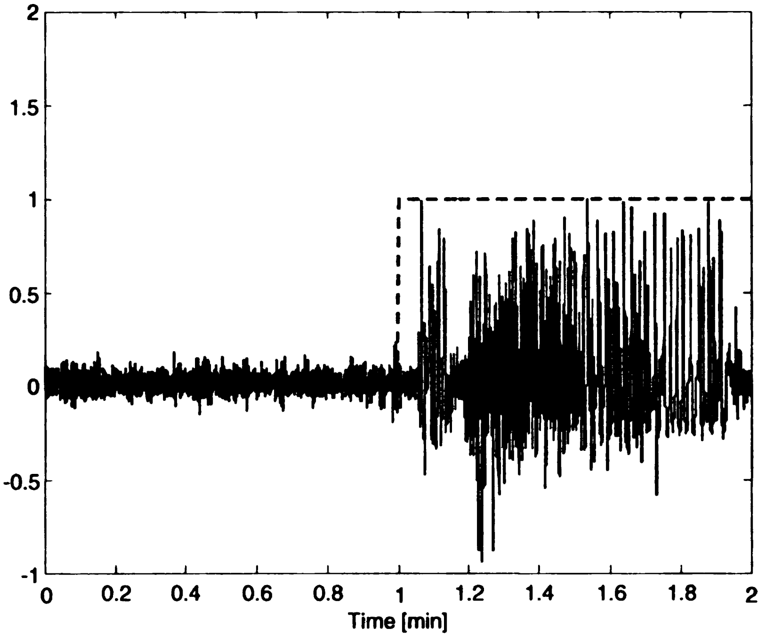


Figure 5.2 Classification labeling for a two-minute ictal epoch. The dashed-straight line (step) indicates the classification for this epoch.

5.4 Benchmark Feature

In the case of seizure prediction, a benchmark had to be created against which to compare the GPAF algorithm. However, in the seizure detection case, a benchmark was already available. It is a conventional (handcrafted) feature called ***line length*** or ***curve length*** that may well be the best one implemented to date for detection of epileptic seizures [25]. Equation (5.1) shows the mathematical expression.

$$LL(k) = \sum_{n=k-L+1}^{k} |x[n] - x[n-1]|. \tag{5.1}$$

Although the equation is simple, the feature can detect abrupt changes in a given signal; and that is precisely the characteristic activity that present when a seizure occurs. That is, the line length feature is sensitive to sudden changes in the amplitude (it is the discrete analog of the absolute value of the derivative of the function) and frequency. Line length linearly tracks AM and FM activity in a signal.

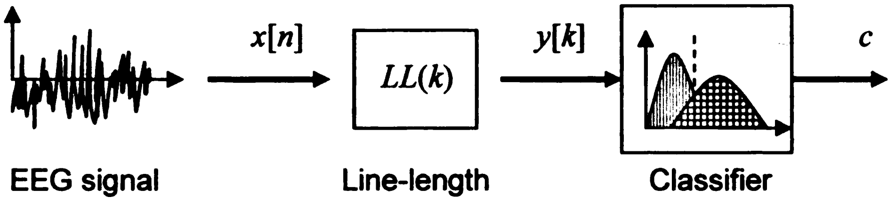


Figure 5.3 Line length feature benchmark classifier.

In this chapter, we compare the results of the GPAF algorithm against the results provided by the k-NN classifier after processing the EEG data using the line length feature. Figure 5.3 shows an illustration of the benchmark classifier using the line length feature, where $x[n]$ is the EEG signal, $y[k]$ is the output of the line length, ($y[k]$ is calculated from Equation (5.1), and we have used the $y[k]$ notation to be consistent with previous equations and diagrams), and c denotes the output label of the classifier. This benchmark does not use the delays used for the GPAF algorithm. However, Equation (5.1) can also be viewed as a state-space reconstruction using two dimensions $x[n]$ and $x[n-1]$ (that is, the embedding dimension is two), with the delay $\tau = 1$.

5.5 Decision Integration Window

As in the seizure prediction experiments, detection experiments were conducted using a point basis, where each point in an artificial feature epoch counts as one example, *i.e.*, each point *k* is an example for the classifier (see Figure 4.14). Similarly, if, for instance, an implantable device (with GPAF features integrated) classified each *t* seconds over an incoming EEG signal, the device could not be allowed to medicate the patient based on the decision that the classifier makes each Δt seconds. The device needs a longer, fixed-length window, so that it can observe the past evolution of the point-basis classification during a defined period to decide whether or not a patient is suffering a seizure. Even when this window is buffered, the point basis creates too much flickering at the binary output. We denoted the length of this window as L_{DIW}. This parameter controls a tradeoff between the number of false positives and the detection delays of seizures (thus, it also controls the number of seizures that are detected; however, because most seizures are detected sooner or later given the sudden and large changes on the EEG signal's amplitude, what is also of great interest is how early those seizures can be detected). How the length of the window affects the number of false positives, false negatives, and detection delay is as follows:

L_{DIW}	Small	Large
False Positives	Increase	Decrease
False Negatives	Decrease	Increase
Detection Delay	Decreases	Increases

This parameter can be fixed as a constant for all patients, that is, the same L_{DIW} value for all the patients. However, this parameter allows more flexibility in our approach, to make the system tunable for better patient-specific performance. Here we selected L_{DIW} by trial-and-error, but the ability of the GP algorithm to automatically select the optimal parameter will be investigated in the future.

The detection algorithm uses the concept of a time tolerance. A detection will be considered to be made correctly up to one minute before and through one minute after the UEO. If the detector detects the seizure earlier than a minute before UEO, it is considered a false positive; whereas if the seizure is not detected within one minute after the UEO, it is considered a false negative.

5.6 Results

The feature-sampling periods for creating the data sets were different. Since the relative amplitude variability of the EEG signals at the moment of an attack is much higher than the relative variability of the preictal EEG (see Section 4.2), features were generated every 0.125 seconds. Therefore, the length of the sliding window was set to $D = 50$ points (0.25 s) and $L = 200$ points (1 s). The same sliding window parameters were used for the curve length feature. Thus, the number of points in an epoch processed by the GP-generated equation κ was 477 points. Holdout validation was used as training technique. The training data consisted of three baseline and three ictal epochs for each patient. Euclidean distance was set as the metric and the number of nearest neighbors for the k-NN classifier was set to $k = 5$, a value typically used. The classifier was set to output one when ictal data are present (that is, a seizure is being detected) and zero otherwise. Finally, as stated before, the embedding dimension was set to six.

In the experiments, the L_{DIW} parameter was set identical for both the GPAF algorithm and the curve length benchmark for each patient.

In these experiments, we set the number of trees for each individual in the GP module to two; that is, the GPAF algorithm created two artificial features for each patient. Also, the GP algorithm used the same set of functions presented in Section 4.8. The maximum depth for a tree was set to 10 and the population size to 1000 individuals. The initial population was generated using the ramped half-and-half method. A crossover operator with a rate of 0.9, fitness proportional selection, and a mutation operator with a rate of 0.1 were used.

5.6.1 Patient A

This patient suffered six seizures through 45.0 hours of EEG recording. Based on autocorrelation analysis of the ictal signal, the delay was set to $\tau = 19$ (or 0.095 s) and the terminal set was $\{x[n], x[n-19], x[n-38], x[n-57], x[n-76], x[n-95]\}$. Equation (5.2) shows the artificial features found by the GPAF algorithm for the detector of this patient's seizures. Table 5.1 provides the results for GPAF and curve length equations, where FP denotes the number of false positives, FPH is the rate of false positive per hour, FN denotes the number of false negatives, L_{DIW} is the length of the decision integration window, ADT denotes the average delay of detection time, Sz is the number of seizures that occurred throughout the EEG archive, and HP is the number of hours of EEG recordings processed.

This patient had a high average of detection time because one seizure was detected with an outlier delay of 35.75 s. For this patient, the traditional curve length measure outperformed GPAF, overall.

$$y_1[k] = \sum_{n=1+50(k-1)}^{50k+150} \frac{(x[n-76])^2}{\log(x[n-19])}.$$

$$y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} x[n-95]\cdot x[n-76]. \tag{5.2}$$

Table 5.1 Results for validation data for patient A.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	6	0.133	0	3.75 s	15.67 s	6	45.0
Curve length	5	0.111	0	3.75 s	7.67 s	6	45.0

5.6.2 Patient B

Through 46.2 hours of EEG recording, this patient suffered 5 seizures. The delay was τ = 11 (or 0.055 s) and terminal set was {$x[n]$, $x[n-11]$, $x[n-22]$, $x[n-33]$, $x[n-44]$, $x[n-55]$}. Equation (5.3) shows the equations designed by the GPAF algorithm. Additionally, Table 5.1 provides the results obtained from the validation of the GPAF equations and the curve length feature. The table shows that using the GPAF equations, just one false positive occurred. GPAF was able to detect all five seizures that occurred. In addition, the GPAF-based detector was able to achieve a lower average detection delay.

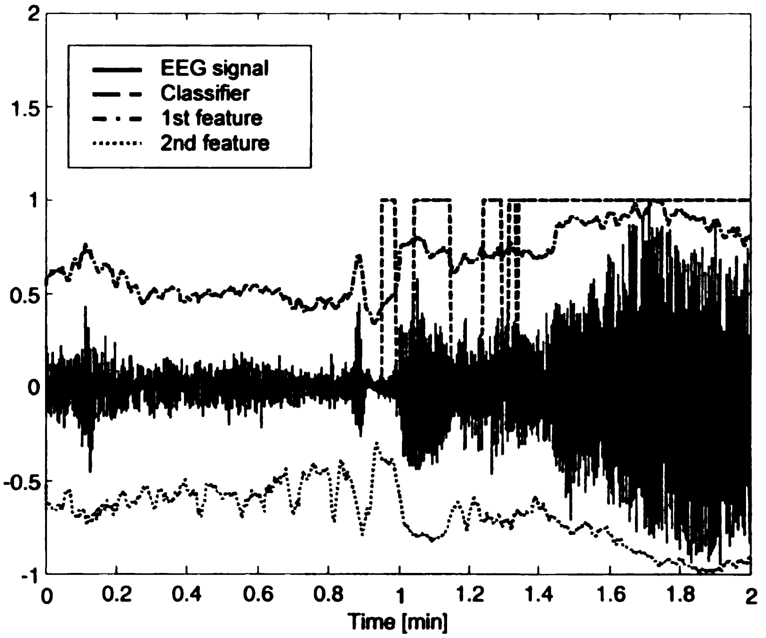


Figure 5.4 A seizure from patient B, which depicts a beta-buzz pattern.

$$y_1[k] = \sum_{n=1+50(k-1)}^{50k+150} \log_{10}\left(x[n-11]-x[n-22]+\sqrt{x[n-11]}\right).$$

$$y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} \log_2\left(\frac{\log_2(x[n-11])}{x[n-33]+x[n-44]}\right). \tag{5.3}$$

Table 5.2 Results for validation data for patient B.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	1	0.022	0	3.75 s	1.15	5	46.2
Curve length	1	0.022	0	3.75 s	4.35	5	46.2

A good average of detection delay was achieved because this patient had a characteristic pattern called "beta-buzz" in which the amplitude dramatically drops before the UEO. Figure 5.4 shows the characteristic pattern, which occurs a few seconds before the UEO that is located exactly at one minute in the plot. Summarizing across measures, GPAF never under-performed the curve length detector, and outperformed it on average time to detection.

5.6.3 Patient C

This patient had an archive of 65.7 hours of EEG recording, in which eleven clinical seizures were diagnosed. The delay was $\tau = 19$ (or 0.095 s) and terminal set was $\{x[n], x[n-19], x[n-38], x[n-57], x[n-76], x[n-95]\}$. The pair of artificial features found by the GPAF algorithm is in Equation (5.4). Table 5.3 shows the results for the validation data, where zero false positives and thus a zero false positive rate per hour occurred and all eleven seizures were detected. Finally, though not better than the curve length, the artificial equations produced a low average of detection delay.

$$
\begin{aligned}
y_1[k] &= \sum_{n=1+50(k-1)}^{50k+150} \left(x[n] - \frac{\sqrt{x[n-76]}}{x[n-95]} \right)^2 . \\
y_2[k] &= \sum_{n=1+150(k-1)}^{150k+50} \left| \frac{(x[n-19] - x[n])^2}{\left|\log_{10}(\log_{10}(x[n-38]))\right|} \right| .
\end{aligned}
\qquad (5.4)
$$

Table 5.3 Results for validation data for patient C.

System	FP	FPH	FN	LDIW	ADT	Sz	HP
GPAF	0	0	0	3 s	1.93 s	11	65.7
Curve length	0	0	0	3 s	–1.82 s	11	65.7

5.6.4 Patient D

This patient has 156.4 hours of EEG recording in which eleven clinical seizures were found. The delay was $\tau = 9$ (or 0.045 s) and the terminal set was $\{x[n], x[n-9], x[n-18],$

$x[n-27]$, $x[n-36]$, $x[n-45]\}$. Equation (5.5) shows the artificial functions found by the GPAF. Some very interesting results were found for this patient. The first artificial equation $y_1[k]$ is the curve length feature, but with delay of 9 instead of 1. As seen in Table 5.4, the GPAF equations and the curve length both performed well, producing no false positives or false negatives, and the average detection delay (ADT) with GPAF was lower than that with curve length.

$$y_1[k] = \sum_{n=1+50(k-1)}^{50k+150} |x[n-9]-x[n]| .$$

$$y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} |x[n]| . \tag{5.5}$$

Table 5.4 Results for validation data for patient D.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	0	0	0	2.75 s	2.1 s	11	156.4
Curve length	0	0	0	2.75 s	4 s	11	156.4

5.6.5 Patient E

This patient has the longest EEG archive, lasting 197.3 hours, in which fifteen seizures were found. The delay was $\tau = 9$ (or 0.045 s) and the terminal set was $\{x[n], x[n-9], x[n-18], x[n-27], x[n-36], x[n-45]\}$. The pair of artificial features is presented in Equation (5.6). Table 5.5 provides the results for both the artificial features and the curve length feature. Both of them produced the same number of false positives, providing a false positive rate of 0.025 per hour, which is acceptable. All seizures were detected, with the

GPAF features producing a slightly better average delay of detection although this difference does not appear to be statistically significant.

$$
\begin{aligned}
y_1[k] &= \sum_{n=1+50(k-1)}^{50k+150} x[n-18]\cdot x[n-36].\\
y_2[k] &= \sum_{n=1+50(k-1)}^{50k+150} x[n]+x[n-9]+x[n-36]\cdot x[n-45].
\end{aligned}
\tag{5.6}
$$

Table 5.5 Results for validation data for patient E.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	5	0.025	0	4.25 s	10.9 s	15	197.3
Curve length	5	0.025	0	4.25 s	11.05 s	15	197.3

5.6.6 Patient F

The entire EEG database of this patient lasts 66.2 hours, during which six clinical seizures were identified. The delay was $\tau = 20$ (or 0.1 s) and the terminal set was $\{x[n], x[n-20], x[n-40], x[n-60], x[n-80], x[n-100]\}$. Equation (5.7) shows the artificial features designed by the GPAF algorithm. Table 5.6 shows results obtained with the GPAF equations and the curve length feature. For this patient, the GPAF-designed equations produced three false positives, whereas the curve length feature produced fifteen of them. The average detection delay was lower for the GPAF equations.

On the other hand, both algorithms failed to detect one seizure previously reported. However, the GPAF algorithm could detect three clinical seizures, of which one is depicted in Figure 5.5, and one subclinical seizure that had not been previously reported; whereas the curve length feature detected an extra subclinical seizure (that is, the GPAF features did not detect the additional subclinical seizure). The clinical seizures lasted roughly one minute, whereas the subclinical seizures lasted 16 to 18 seconds.

$$y_1[k] = \sum_{n=1+50(k-1)}^{50k+150} \log\left(\sqrt{x[n-100]} - x[n-20]\right).$$
$$y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} \left| \frac{x[n-100]}{x[n] \cdot \log_{10}\left(x[n-60]\right)} \right|. \qquad (5.7)$$

Table 5.6 Results for validation data for patient F.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	3	0.045	1	3.75 s	–9.21 s	7	66.2
Curve length	4	0.060	1	3.75 s	5.33 s	7	66.2

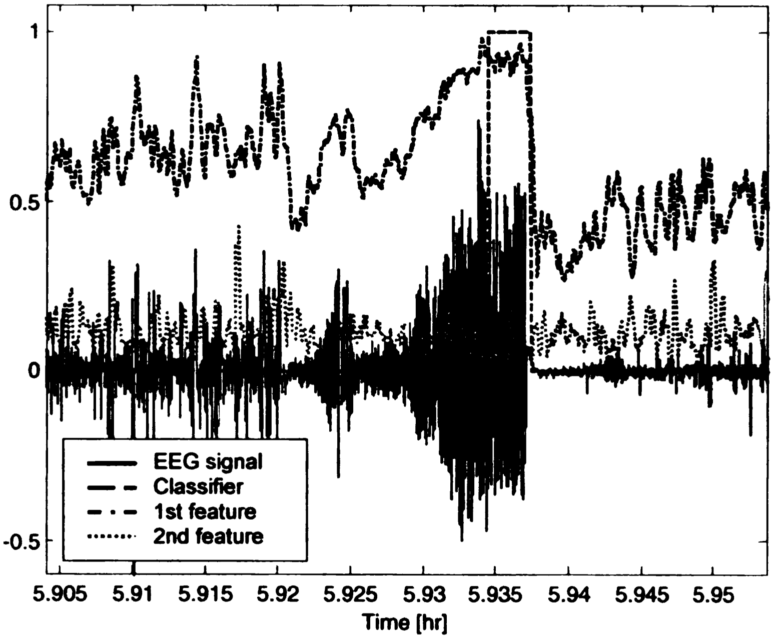


Figure 5.5 Clinical seizure previously unreported for patient F.

5.6.7 Patient G

This patient had available 153.8 hours of EEG for the training and validation procedures. This patient suffered 31 seizures, of which twenty were subclinical. Unlike in the prediction problem, here we also included the subclinical seizures so that the GPAF algorithm could design a detector that also warns about such seizures. Subclinical seizures are epileptic seizures, but of duration usually no more than 30 seconds and without overt, clinically evident symptoms, *i.e.*, they are detected only with EEG. Figure 5.6 shows a subclinical seizure.

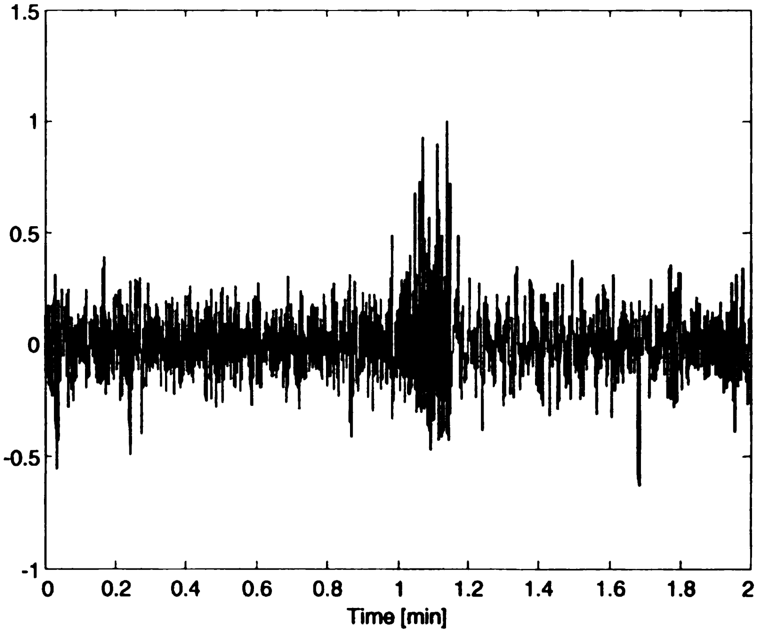


Figure 5.6 Subclinical seizure. The signal is normalized to one.

The delay was $\tau = 15$ (or 0.045 s) and the terminal set was $\{x[n], x[n\text{–}15], x[n\text{–}30], x[n\text{–}45], x[n\text{–}60], x[n\text{–}75]\}$. Equation (5.8) shows the artificial features found by the GPAF and Table 5.7 shows the results for the validation data. Although the number of false negatives was zero, the number of positives was so high that the artificial equations were impractical (see below). A similar result was obtained for the curve length feature.

$$y_1[k] = \sum_{n=1+50(k-1)}^{50k+150} \left|x[n-30]\right| + \left(\left(x[n-45]\right)^{12} + \left(\log\left(x[n]\cdot x[n-60]\right)\right)^2\right)^2 .$$

$$y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} \sqrt{x[n-60]}. \tag{5.8}$$

Table 5.7 Results for validation data for patient G.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	—	—	0	2.75	0.105	31	153.8
Curve length	—	—	0	2.75	3.07	31	153.8

We realized that, unlike the other patients, this patient had many subclinical seizures. Figure 5.2 illustrates what was happening. The classification labeling that was used before was also being applied to subclinical seizures, but Figure 5.6 shows that once a subclinical seizure ends, the amplitude of the signal goes back to normal. In fact, the amplitude of the signal after the seizure is similar to the amplitude before the seizure. Therefore, to address this problem, we redefined the classification labeling for the subclinical seizures. The classifier should output one just when the seizure is occurring, and zero both before and after the seizure. Also, this patient had some seizures that created a "catch 22"—the situation when neurologists, who are trying to place the intracranial electrodes at the seizure foci, do not know exactly where those are located. Because of this, sometimes the electrodes are misplaced, creating delays in the EEG recordings of seizure activity of up to tens of seconds. For this reason, because our observation of the data suggested the existence of such delays, for six seizures, we corrected the UEOs forward in time by 6 to 12 seconds.

Although the GPAF algorithm was run again to see what features would be designed using the revised information, we also evaluated Equation (5.8) again using the new classification labeling. Table 5.8 shows the results for the validation data. The number of false positives is approximate, given that there were still too many events for an accurate false positives count. Additionally, approximately eight hours of EEG recording, during which there were no seizures, were highly misclassified; therefore, the real number of false positive is a lot larger. Increasing the length of the DIW did not yield an acceptable solution; given the setting DIW = 2.75 s, four seizures were still missed (although all were subclinical).

Table 5.8 Results for validation data using the new classification labeling.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	85	0.553	4	2.75	8.361	31	153.8

The GPAF algorithm was run using the new classification labeling. Equation (5.9) shows the artificial features found. Table 5.9 provides the new results for the validation data. Clearly, these artificial features are achieving good performance. The two seizures missed by the GPAF equation were both subclinical seizures. However, an additional unreported subclinical seizure was detected by both methods. Additionally, a low false positive rate was achieved. In contrast, the curve length feature had a high false positive rate (although the number of false positives is an approximation, as explained above). In fact, the number of false positive for the curve length feature was larger because misclassified almost eight hours of EEG recording. If the parameter L_{DIW} was increased, several seizures were missed.

$$
\begin{aligned}
y_1[k] &= \sum_{n=1+50(k-1)}^{50k+150} x[n-75]\cdot x[n-60] + \frac{x[n-60]}{x[n-30]}. \\
y_2[k] &= \sum_{n=1+50(k-1)}^{50k+150} \frac{(x[n-60])^2}{\ln(x[n-45])}.
\end{aligned}
\tag{5.9}
$$

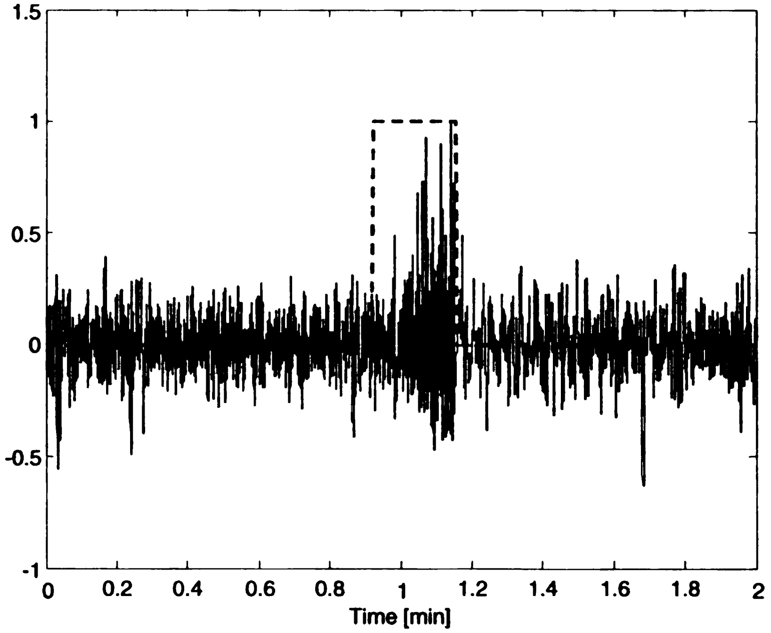


Figure 5.7 Classification labeling for subclinical seizures. The dashed-straight line indicates the ideal classification for this epoch.

Table 5.9 Results for validation data for patient G using new classification labeling.

System	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
GPAF	11	0.072	2	2.75	5.81	31	153.8
Curve length	81	0.527	0	2.75	3.40	31	153.8

5.7 What Causes False Positives?

It is natural to wonder about the anomalies that tend to make the GPAF classifier produce false positives. Some of the false positives occur because artifacts are present, as discussed below. In the results of the previous section, artifact-type false positives were not counted, however, in this section we show pictures of how a false positive is produced by an artifact. In the following figures, the means of the artificial features were subtracted to shift the artificial signals to zero for easier comparison. Figure 5.8 shows an artifact, called blackout, detected in the EEG recording of patient B. The picture shows that as soon as the blackout starts, the artificial features change suddenly, making the detector "fire." Blackouts were excluded as false positives given that such artifacts occur because of the recording equipment and process. Another artifact that leads to false positives is full-scale swings. Figure 5.9 depicts a full-scale swing detected by the artificial features for patient D. Once again, it represents a flaw in the EEG recording system.

However, not all false positives are due to artifacts; some result from brain-generated events. Figures 5.10 and 5.11 show seizure-like events that occurred in patient E. These "false start" events may last just a few seconds. If their duration is more than 30 seconds, they can be considered epileptic seizures. Both graphs illustrate that the detector was "firing" during the false start (the frequency increased). Additionally, the EEG of patient E also exhibited "delta trains," whose frequency components are between 0.5 and 4.0 Hz. Figure 5.12 presents an example of a delta train.

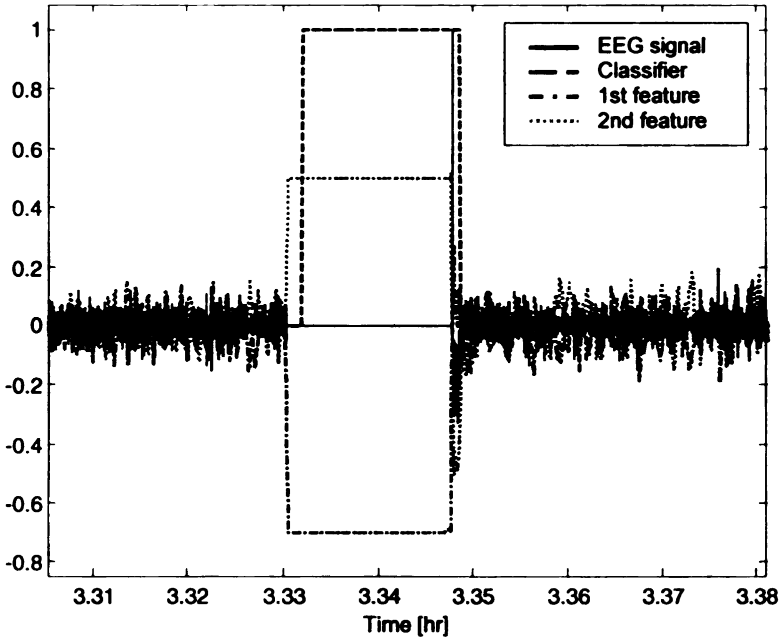


Figure 5.8 Blackout occurred in the EEG recording of patient B.

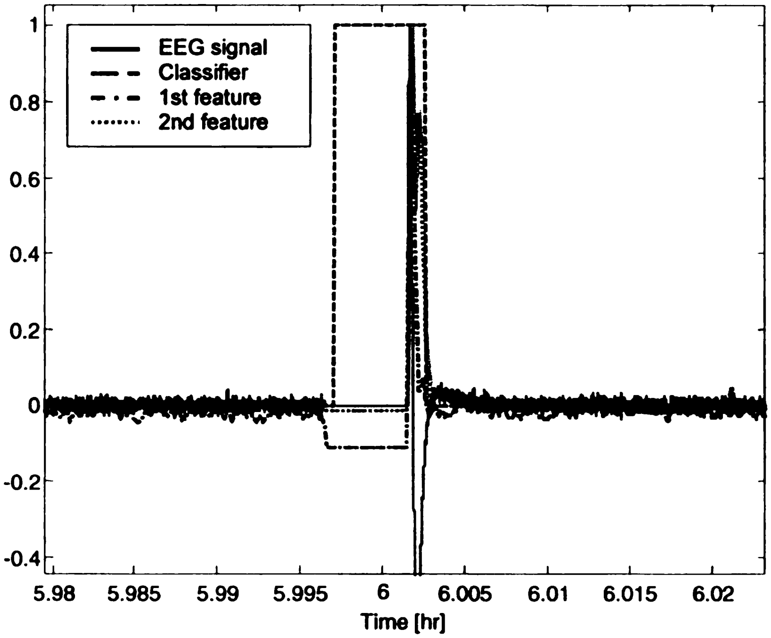


Figure 5.9 Full-scale swing occurring during EEG recording of patient D.

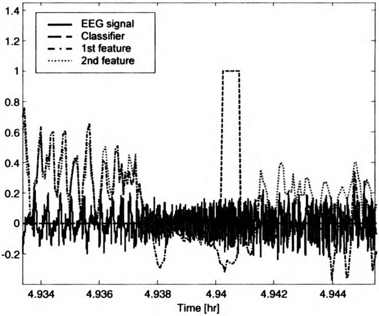


Figure 5.10 False start occurring during the EEG recording of patient E.

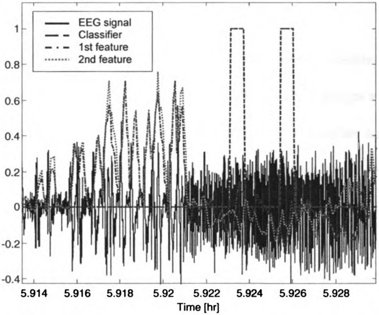


Figure 5.11 False-start occurring during EEG recording of patient E.

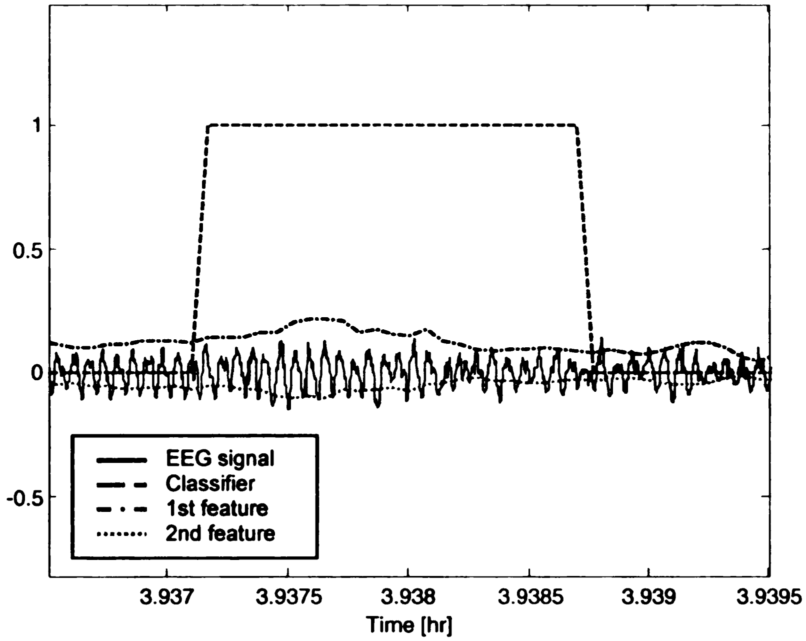


Figure 5.12 Delta train in EEG recording of patient E.

5.8 A Generic Detector

A universal, generic detector, such as the curve length feature, is desired because it can be applied to any patient without the need for extracting data to design a detector for the seizures of the patient at hand. Using only a small amount of baseline and ictal data, the detector should able to detect most seizures and at the same time have a low false positive rate. In this section, we show a general detector designed by the GPAF algorithm.

For this experiment, we used only one baseline epoch and one ictal epoch of two minutes duration from each available patient. Therefore, the training data set consisted of seven baseline and seven ictal epochs (a total of 28 minutes of data). The training ictal epoch for patient G was a subclinical seizure. The holdout method was used for training.

In searching for a generic detector, we also fixed the delay parameter by obtaining an average from the delays calculated previously. The average delay was 16. Therefore, the terminal set was {*x*[*n*], *x*[*n*–16], *x*[*n*–32], *x*[*n*–48], *x*[*n*–64], *x*[*n*–80]}. The L_{DIW} parameter, which is not specifically tied to any feature, was adjusted for each patient and was not necessarily the same value used for the customized detector in the previous section. The parameters for the GP algorithm and *k*-NN classifier were set as before. Equation (5.10) shows the artificial equations found by the GPAF algorithm.

$$y_1[k] = \sum_{n=1+150(k-1)}^{150k+50} (x[n-64]-x[n-80])^2 + \frac{x[n]\cdot\sin(\sin(x[n]))\cdot x[n-48]}{(x[n-16])^2 - \log_{10}(x[n-16])} - \log_2(x[n-64]).$$

$$y_2[k] = \sum_{n=1+150(k-1)}^{150k+50} x[n] + x[n-80] + x[n-32]\cdot x[n-16] - \cos(x[n-48]). \tag{5.10}$$

Table 5.10 provides the results for the seven patients using the validation data, their respective whole EEG archives. The results were acceptable for most patients, having similar performance to the curve length feature. Five patients out of seven met the specifications of a 0.0833 of false positive rate and less than 10% of false negative.

Table 5.10 Results for validation data for the general detector.

Patient	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
A	3	0.067	0	3.75	15.08	6	45.0
B	2	0.043	0	3.75	0.1	5	46.2
C	0	0	0	3.25	0.16	11	65.7
D	0	0	0	2.5	1.55	11	156.4
E	25	0.159	0	6.25	19.13	15	197.3
F	8	0.121	1	6.25	16.17	7	66.2
G	2	0.013	2	2.75	6.22	31	153.8
Overall	5.71	0.058	0.429	4.07	8.34	86	730.6

For patient E, a 0% false negative rate was achieved; however, this patient had a flurry of EEG seizure-like bursts in the form of false starts and delta trains, which caused all of the false positives. Because of this, the length of the decision window was increased to filter out several false positives; however, this decision affected the average detection delay.

Patient F had a rate of 14.29% false negatives. However, three clinical and two subclinical unreported seizures previously were detected (found by the customized and curve length feature presented in Subsection 5.6.6); additionally, the detector found an additional subclinical seizure, which was not detected by either the customized or curve length detector. If all seizures were counted, the false negative rate for this patient drops to 7.69%. On the other hand, this patient had many epileptiform discharges (spike-and-wave patterns) that lasted only a few seconds. These electrical discharges made the generic detector fire, producing several false positives.

Finally, a similar situation occurred with the results for patient G. Two subclinical seizures were missed, yielding a false negative rate of 6.45%. However, a previously unreported subclinical seizure was found. A low false positive rate per hour was achieved. Additionally, a series of nineteen epileptiform discharges—spike-and-wave EEG patterns—was also detected.

5.9 A Detector with $\tau = 1$

It is clear, as mentioned previously, that the curve length feature exhibits quite good detection performance for all patients. Even for patient D, where the GPAF found the curve length, the original curve length equation performed slightly better than the GPAF equation. (Note that it was not possible for the GPAF search to find the identical curve length feature, because it was not given the possibility to use $\tau = 1$ in its terminal set; instead, for previously justified reasons, τ was set using autocorrelation data.) Therefore, it is natural to ask if the better performance was because of the maximum correlation that the curve length feature displays; that is, the delay equals one. Therefore, we selected patient G to rerun the GPAF algorithm, but with delay $\tau = 1$. The set of terminals or delays was $\{x[n], x[n-1], x[n-2], x[n-3], x[n-4], x[n-5]\}$. Patient G was selected because that patient exhibited the most false negatives. The rest of the parameters was set as before. Equation (5.11) shows the artificial features found by the GPAF algorithm.

$$\begin{gathered} y_1[k] = \sum_{n=1+50(k-1)}^{50k+150} \left| x[n] - x[n-1] \right| - \log_{10}\left(\log_2\left(x[n-1] - x[n-3]\right)\right). \\ y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} x[n]. \end{gathered} \tag{5.11}$$

Perhaps not surprisingly, the algorithm found a pair of features in which the first, $y_1[k]$, contains the term $|x[n] - x[n-1]|$, which is precisely the curve length feature; whereas the second term in the same equation is a "tuning" or "noise" term, given that its relative magnitude is so small that it has little effect. The second feature was not contributing to better discrimination, given that when the GPAF features were compared with the curve length, the latter was still performing better. When it was removed, the curve length feature and $y_1[k]$ in Equation (5.11) performed similarly, with the difference of 0.25s in lead or delay of detection of some seizures caused by the log term in the GPAF feature.

In a second experiment, we ran the GPAF algorithm to find a generic set (that is, using the EEG data epochs from all patients) of artificial features, in the same way as in the previous section, but with the delay equal to one. Equation (5.12) shows the equations found by the GPAF algorithm, whereas Table 5.11 provides the results for the validation data.

$$\begin{gathered} y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} (x[n-3])^2 . \\ y_2[k] = \sum_{n=1+50(k-1)}^{50k+150} x[n-3] \cdot (x[n] + x[n-5]). \end{gathered} \tag{5.12}$$

Unlike the generic detector found before, these equations are simpler; however, according to the table, the general artificial detector achieved good performance. Except for one patient, the patients had an average of two false positives or fewer per day. At the same time, a low average false negative rate was achieved. Patient F had the most seizures missed. Patient F had the longest decision window. That is, because the detector was being triggered by epileptiform discharges (in the form of spike-and-wave artifacts) not lasting more than three seconds, in order to filter out these EEG discharges, the length of the window was increased, at the expense of two (reported) clinical seizures missed. However, it detected three clinical and one subclinical seizures, but another subclinical seizure that had been previously detected by the curve length was missed. On the other hand, although postictal activity was not counted as false positives, it was observed for patient C using Equation (5.12). However, such activity was not seen using the customized detector (*i.e.*, Equation (5.4)) or the generic detector (*i.e.*, Equation (5.10)).

That means that this set of features is very sensitive for patient C. Additionally, all the false positives for patient F were false starts. Such EEG patterns also affected the previous set of artificial features.

Table 5.11 Validation data results for the general detector with delay equal one.

Patient	FP	FPH	FN	L_{DIW}	ADT	Sz	HP
A	2	0.044	0	3.75	14.42	6	45.0
B	2	0.043	0	3.75	25.2	5	46.2
C	0	0	0	3.75	−1.3	11	65.7
D	0	0	0	2.5	9.34	11	156.4
E	12	0.061	0	3.75	5.65	15	197.3
F	13	0.196	2	5	9.75	7	66.2
G	13	0.084	1	2.75	4.53	31	153.8
Overall	6	0.061	0.429	3.61	9.66	86	730.6

5.10 Training by Leave-One-Out Method

All the training phase for detection (as well for prediction) was done using a holdout method (or split-sample validation). As previously mentioned, a problem with this method is that there is variability, given that the classification is tied to the training data selected for use. To avoid (or at least to reduce) this effect, a leave-one-out method can be used; however, is computationally intensive. Because of this, we did not use the leave-one-out (LOO) method for most experiments; however, we will show here results for patient G using a form of the LOO method. Unlike prediction, where we were using EEG epochs for training and validation purposes, in detection, we had an entire EEG archive—in this case, patient G had 153.8 hours of EEG recording—which made it impractical to segment all the data and run the algorithm with two-minute segments. Therefore, our

LOO implementation was very limited, using three baseline and three ictal epochs (two subclinical were included, one of them was the one missed by the all the detectors when patient G was processed) of two-minute duration. Although the length of the epochs is shorter for detection, we needed to calculate more points, given that the sliding window is 0.25 s long.

The parameters for the GP algorithm and *k*-NN classifier were the same used to obtain the customized artificial features for patient G. Equation (5.13) provides the GPAF-designed features. The results are shown in Table 5.12, where P_c denotes the percent of correct classification FP denotes false positives, and FN is false negatives. The length of the decision window was set to 2.5 s. Two false positives were detected among all the combinations, and one seizure was misclassified, which was the one missed by all the detectors previously tested. By looking at the mean and standard deviation, which was ± 7.19 %, we can state that a detector with high confidence was found by the GPAF algorithm.

$$y_1[k] = \sum_{n=1+150(k-1)}^{150k+50} (x[n-20])^2 .$$
$$y_2[k] = \sum_{n=1+150(k-1)}^{150k+50} x[n-30]\cdot(x[n-50]-x[n-40]) . \tag{5.13}$$

It seems clear that using the LOO method for training produces models—detectors, in this case—that are more accurate, but on the other hand, they are computationally expensive. High-end computers are needed in order to make LOO practical for this application, considering the amount of data that must be processed, at least when conducting many experiments in a short timeframe.

Table 5.12 Results for the leave-one-out method.

Set	P_c	FP	FN
1	96.13	0	1
2	82.34	1	0
3	81.57	0	0
4	95.40	0	0
5	81.51	0	0
6	94.25	0	0
7	82.55	0	0
8	96.34	1	0
9	95.72	0	0
Overall	89.53	0.22	0.11

5.11 Discussion

Section 5.6 shows that the GPAF algorithm can produce or design sets of equations with similar or better performance to that of a conventionally human-crafted feature—in this case, the curve length. For the customized artificial features, just four seizures were missed (with three being subclinical seizures, one of which was found in this work by other detectors), for a rate of 4.35% false negatives (88 of 92 seizures caught, counting the previously unreported ones), which more than satisfies the requirement set, being no more than 10%. Also, in 703.6 hrs (roughly 30.4 days) of EEG recordings, just 23 false positives were signaled, for a rate of 0.0315 false positives per hour. The false positive requirement was 0.0833 false positive per hour so the system exceeded the expectation. The curve length feature achieved an average 0.1314 false positives per hour. However, this value was affected most by the results in patient G. Also, it missed just one previously reported seizure on patient F, yielding a false negative rate of 1.09%. Note, however, that GPAF was using two equations (artificial features) while the curve length is just one equation, allowing the GPAF equations more flexibility.

A generic detector for all patients was also successful. This model was sufficiently general to allow use of a fixed (average) delay parameter. The overall false positive rate per hour was a low 0.055, and only three seizures were missed, of which two were subclinical. However, this model was also able to detect an additional subclinical seizure in patient F. This yielded a low false negative rate of 3.22% (90 seizures of 93 were detected, after counting the seizures that had not been previously reported). This general detector was created by the GPAF algorithm with just one baseline and one ictal epoch of two minutes duration from each patient, a very limited dataset (due to computational expense) compared with the entire body of data available. Time (generations) limits on the evolution of GPAF features were also imposed, for the same reason. Our point is that a better model may be obtained if more data, more generations, and a leave-one-out method were used. With an experiment, we demonstrated that a high-accuracy, low-variance model could be designed by using a larger dataset with leave-one-out cross validation.

Similar results were obtained by the generic model with the delay set to one, yielding an overall false positives per hour rate of 0.058, and missed seizure rate of 4.35% (88 of

92 seizures were detected, adding the previously unreported seizures). Clearly, for seizure detection, exploring the maximum correlation (that is, by using $\tau = 1$) to design new artificial features may also lead to better artificial detectors.

Similar to prediction, in detection, the number false positives and false negatives (and thus the average of detection delay) is affected by the length of the decision window. Although the length can be fixed, it seemed beneficial to fix the length for each patient. Additionally, the design of artificial features from the GPAF algorithm is affected by where we start to label the ictal class. Initially, we were labeling the ictal epochs one minute before the UEO in order to give to the GPAF algorithm a way to track some sort of signal that indicates that an epileptic seizure was imminent. However, artificial features then produced too many false positives, because part of the data was in conflict with the post-UEO data, as was explained earlier in this chapter. Therefore, we decided to start labeling data as ictal class only after UEO. After that, good artificial features were obtained. However, we also learned that the classification should not start for all patients at UEO. For example, patient B has a characteristic pattern before UEO, known as beta-buzz, in which amplitude collapses and frequency increases considerably. This pattern was precisely the one that made the detector fire, allowing an early detection of the seizures. However, in general, the length of the decision integration window, the classification labeling, and the possible feature complexity provided by the GP algorithm provide much flexibility to design good artificial features. First, the GP algorithm tries to explore the search space looking for the "optimal" artificial features, based on the data and the classification labeling. Later, the points misclassified can be corrected using the decision integration window, using its length parameter to control the tradeoff between false positives and negatives. This architecture allows us to design detectors that pay attention to the EEG patterns and specifics of each patient.

Unlike in the prediction case, where the GPAF algorithm far outperformed the benchmark classifiers, in this chapter we cannot claim that the GPAF algorithm outperformed the curve length feature; rather, the GPAF algorithm did better for some patients and the curve length feature did better for other patients. However, the GPAF algorithm has a desirable property: it provides room for improvement. By providing more data, adding a term in the fitness function that considers in some better way the detection delay, and using high-end computers to enable use of more data and more thorough search, there is a high probability that the GPAF algorithm will find better artificial features that respond to the characteristics of a particular patient than does one universal feature. Additionally, we can increase the number of artificial features, for additional discriminating power.

Finally, this work has shown that the GPAF algorithm is not only able to create artificial features with no known physical meaning, but also is able to find some features that are known to be good features—specifically, this was the case for patient D and patient G with delay set to one. Although not the central object of this study, these results confirmed the claims by Esteller *et al.* [25] in the sense that the line length feature is a good and efficient feature for detection of epileptic seizure onsets.

5.12 Summary

In this chapter, we presented the GPAF algorithm for the seizure detection problem, demonstrating its use in several experiments. These experiments yielded detectors that had low false positive rates per hour and low false negative rates (in total, 88 of 92 seizures were detected). In addition to customized equations that were compared with the curve length feature and had similar or better results than that benchmark, we also presented a generic detector that showed good performance by detecting 90 seizures (clinical and subclinical, including the seizures discovered during the experiments) out of 93, and by being able to detect an extra seizure that was not detected by either the other GPAF-generated detectors or by the benchmark.

Artificial features were also designed using a terminal set with delay τ set to 1. In the first such experiment, just using EEG data for patient G, the GPAF algorithm found an expression that resembled the curve length feature, the benchmark feature. In a second experiment, this time to find a generic detector, the GPAF algorithm came up with a set of simple equations that yielded a low false positive rate and detected 88 of 92 seizures (counting even those that were discovered in these experiments).

Finally, an experiment was run with the GPAF algorithm using a leave-one-out method on an expanded dataset as the training process. Although this method is of limited practical value because of its computational intensiveness, this experiment showed that a detector with low variability can be found by such a process.

Chapter 6

Conclusions and Recommendations

In this dissertation, we proposed the genetic programming artificial features (GPAF) algorithm, intended to overcome the reported limitations of the genetically found, neurally computed artificial features (GFNCAF) algorithm. The GPAF algorithm was first tested on a problem of design of an artificial feature that discriminates between parallel and nonparallel vectors. Results were compared with those of the GFNCAF algorithm. Later, we used the GPAF algorithm to design artificial features for epileptic seizure prediction directly from the raw intracranial EEG data via state-space reconstruction for seven patients. In the last chapter, the methods were applied to machine design of epileptic seizure detectors.

6.1 Benefits of the Algorithm

The GPAF algorithm, along with the space-state reconstruction technique, allows researchers and medical personnel to design a set of features that take into account the characteristics or patterns that are "hidden" in the intracranial EEG signals of a given patient in order to predict or detect epileptic seizures. Similar to an optometric test that prepares the prescription for eyeglasses for a patient, a set of features optimized for the needs of a patient could someday be programmed into a chip to be implanted in the patient by a surgical procedure, providing warnings or automatically administering medications or electrical stimulation.

On the other hand, the GPAF algorithm could also be applied in several other fields of study, such as signal analysis, optimization, intelligent control, pattern recognition, and biomedical engineering. The computational framework described in this dissertation also represents an advance in the development of new analyses and practical techniques to predict epileptic seizures, while paving the way for application to other important problems that remain unsolved, such as prediction of earthquakes or cerebral infarct.

The GPAF algorithm offers a modest contribution in advancing the area of pattern recognition. Previous work, in which preprocessing or transformation of the input data to the classifier has been recognized as allowing achievement of better classification, has relied upon scaling the input features (that is, multiplying original features by coefficients) or selecting the most informative features (that is, turning on or off the original features) usually using a genetic algorithm. In this work, we have extended the meaning of feature extraction by allowing the GP algorithm to create artificial features, based on use of case-specific but algorithmically determined terminal function sets that allow enough flexibility to make either simple or complex transformations. This makes the task of the classifier easier.

Additionally, in this work we dealt with a dynamic system, the brain. Using state-space reconstruction allows us to work directly in the time-domain—avoiding the need to make some transformation to another domain like frequency, for example—while trying to capture the deterministic dynamics underlying what appears to be random behavior of the EEG signals. Although it does not make sense to apply state-space reconstruction in a dynamic system in which the state variables are already well-known, the GPAF algorithm could still be used for tasks that include diagnosis, estimation, and detection, by highlighting patterns that can be confused with other normal behavior or are submerged in undesired anomalies like noise.

6.2 Conclusions

In this dissertation, we have proposed an algorithm aimed at capturing a very broad set of artificial feature characteristics, thus, intended to overcome the limitations of the traditional features that have been used in the past for prediction and detection of epileptic seizures and to overcome the limitations of the feature creator algorithm GFNCAF. The genetic programming artificial features algorithm extracts relevant information from the EEG data. Looking at the raw EEG data—in this case, through state-space reconstruction—to look for patterns was valuable, endowing the algorithm with the ability to pay attention to the dynamics of each individual patient as a system. At the same time, with the inclusion of a genetic programming (GP) module as the search engine to look for the artificial features, the GPAF algorithm overcame the limitations of the GFNCAF algorithm by having vastly expanded control over the structure (connections, functions, and inputs) of the features.

Experiments showed that predictive patterns vary from patient to patient, given that the artificial features designed were different on a per-patient basis in structure as well as in number of features needed for good prediction. Thus, it was also shown that, although prediction of epileptic seizures could be difficult, seizures can be predicted with a reasonable time margin before onset, potentially allowing intervention using a medical treatment.

A generic (cross-patient) detector was designed that could easily be programmed into an implantable chip.

With the GPAF algorithm, we are automating much of the process involved in designing customized predictors and detectors. Once the patient is instrumented for EEG recording, the EEG data can be fed to the GPAF algorithm, which can design the predictor or detector (artificial features). Later, these equations could be programmed into an implantable chip, which could then intelligently trigger various forms of intervention.

Finally, this work has shown that the GPAF algorithm is not only able to create artificial features without needing to ascribe physical meaning to the signals, but also is able to find some man-made features that are known to be good features.

6.3 Limitations and Recommendations

Albeit this work was intended to overcome the shortcomings of the GFNCAF algorithm, we still found some limitations, which we summarize below along with some recommendations, providing to the GPAF algorithm more room for improvement.

- *Increasing Computing Performance*: Although we ran the experiments on a 3GHz Pentium IV PC, it was not enough to run the GPAF algorithm for an acceptable number of generations. According to Banzhaf *et al.*, GP population sizes typically should range between 500 and 5000 individuals. They also mention that for good search on challenging problems, the population size should be even larger. Our population sizes ranged between 500 and 1000 individuals, and the number of generations was no larger than 200. However, it is well known that the GP algorithm is computationally intensive, consuming a large amount of memory and CPU time. Because of this, there were also limitations on the amount of EEG data that could be used for training in each run, with consequent limitations on the GP algorithm to look for a better set of features. In order to increase considerably the memory and CPU time resources available, parallel computing or high-performance computers (supercomputers) could be considered for such purposes.

- *Extend GPAF Algorithm Capability*: In this work, the GPAF algorithm was limited to designing or finding the arguments (GP program) of a summation, and could not produce a set of artificial features that contained embedded summations or other vector-handling operators, because the GP algorithm used was single-typed (scalar-typed). For instance, the GPAF algorithm presented in this work cannot produce artificial features that include quotients of summations, because the GP algorithm does not handle vectors inside the programs. This shortcoming limited the flexibility of the GP algorithm and thus the complexity of the artificial features that it could explore or design. To allow the GPAF algorithm to produce more complex features, a strongly-typed genetic programming system could be implemented as the search engine for artificial features—*i.e.*, a GP algorithm that contains mechanisms to prevent policy violations (enforce well-formed formulas) in mixing different types of data like vectors and scalars, for example. With this, the GPAF algorithm would have much more flexibility to create the simplest as well as very complicated features, probably yielding more accurate results.

- *Use All EEG Channels*: In this work, the intracranial EEG data available came from one channel per patient. Similar to the approach used by D'Alessandro *et al.*, we could provide multiple channels to the GPAF algorithm and let it decides which channels are more "informative" to extract patterns in order to predict (or detect) the epileptic seizures.
- *Support Vector Machine as classifier*: an SVM could be implemented as a classifier component for the GPAF algorithm. An SVM processes the data first and projects them into a high-dimensional space by means of a nonlinear function. This classifier may yield a better separation of the classes, increasing the capability of the GPAF algorithm.

Appendix I

This appendix shows the point-basis results for all the epochs available evaluated with the benchmark classifiers, as explained in Section 4.9, and GPAF algorithm, for each patient. Small letters denote baseline epochs, whereas capital letters denote preictal epochs.

Patient A

Point-basis classification results for patient A.

Epoch	PCA *k*-NN	*k*-NN	GPAF
b	50.5	53.51	61.54
c	46.82	52.51	57.86
d	45.82	53.85	57.19
B	55.18	47.49	80.33
C	49.83	45.15	69.23
D	54.52	42.14	73.91
Overall	50.45	49.11	66.68

Patient B

Points-basis classification results for patient B.

Epoch	PCA *k*-NN	*k*-NN	GPAF
b	63.88	57.86	82.61
c	67.22	72.91	89.63
d	67.22	67.89	86.96
e	63.88	53.85	88.29
f	67.56	62.21	81.61
g	66.56	51.51	77.26
B	59.2	85.62	85.62
C	40.8	57.19	66.56
D	67.89	87.63	94.65
E	34.78	54.85	53.17
Overall	59.9	65.15	80.64

Patient C

Points-basis classification results for patient C.

Epoch	PCA *k*-NN	*k*-NN	GPAF
b	87.63	99.67	100
c	50.17	79.59	100
d	84.62	98.99	40.8
e	87.96	100	70.9
f	76.92	95.65	99.0
g	86.62	100	100
h	87.96	99.33	100
i	88.63	100	87.63
j	72.58	91.97	100
k	90.3	100	97.32
A	80.93	89.63	100
B	69.57	83.27	99.66
C	51.17	76.92	87.63
E	63.88	81.94	81.61
F	75.25	87.29	99.0
G	55.85	77.59	97.99
H	30.1	15.38	96.99
I	75.59	89.29	99.0
J	67.22	85.28	96.66
K	70.57	89.29	100
Overall	72.68	87.27	92.71

Patient D

Points-basis classification results for patient D.

Epoch	PCA *k*-NN	*k*-NN	GPAF
a	77.93	75.25	96.66
b	78.26	68.9	96.33
d	52.17	47.83	47.16
f	38.46	71.24	96.65
g	46.15	64.88	91.30
h	43.14	64.88	90.97
i	41.81	58.53	45.82
j	42.81	35.12	20.74
k	44.48	43.14	51.84
l	40.47	63.88	88.63
B	10.70	12.37	0.33
C	53.85	50.50	74.92
D	56.86	34.45	8.03
E	63.21	44.15	53.18
F	57.19	55.18	86.29
G	57.86	60.20	77.93
H	61.54	51.17	86.96
I	31.77	34.78	17.07
J	58.19	47.16	86.29
K	59.87	46.15	84.95
Overall	50.86	51.49	65.10

Patient E

Points-basis classification results for patient E.

Epoch	PCA *k*-NN	*k*-NN	GPAF
a	31.88	42.28	26.85
b	56.71	77.52	72.82
c	82.55	99.66	100.0
d	86.58	100.0	100.0
e	48.32	64.43	70.47
f	89.93	97.31	100.0
h	87.85	82.21	100.0
i	85.9	100.0	100.0
j	46.64	73.15	57.05
k	80.87	74.83	100.0
l	85.23	100.0	100.0
m	86.58	95.97	100.0
n	91.95	100.0	100.0
o	58.39	71.81	53.69
p	59.06	80.87	74.5
A	62.75	41.61	79.53
B	70.13	68.12	93.29
C	57.05	48.99	48.99
D	74.16	79.53	92.95
E	78.19	75.5	94.63
F	74.83	78.52	92.28
G	76.51	74.16	94.3
H	79.53	74.16	94.97
I	36.58	28.86	17.11
J	79.87	79.87	97.32

K	76.85	78.86	97.32
L	79.19	79.19	98.32
M	43.62	36.91	49.66
N	44.3	31.21	78.52
O	20.81	4.03	9.4
Overall	67.79	71.28	79.8

Patient F

Points-basis classification results for patient F.

Epoch	PCA *k*-NN	*k*-NN	GPAF
b	74.24	94.63	83.56
c	75.59	94.97	86.58
d	71.57	90.27	66.56
e	88.63	100.0	100.0
f	85.28	100.0	100.0
g	83.95	100.0	100.0
h	83.28	100.0	100.0
B	72.57	89.63	97.99
C	76.58	88.29	99.33
D	78.93	92.98	100.0
E	80.2	89.3	100.0
F	67.89	64.21	99.33
Overall	78.2	92.02	94.45

Patient G

Points-basis classification results for patient G.

Epoch	PCA *k*-NN	*k*-NN	GPAF
b	61.54	80.6	60.2
c	48.49	55.52	9.03
d	50.84	60.54	26.76
e	52.84	69.23	71.57
f	54.18	67.22	59.2
g	61.54	80.6	79.26
i	41.14	11.37	94.65
j	37.46	13.38	96.66
B	42.48	59.53	29.77
C	48.91	66.89	79.6
D	42.81	63.21	90.97
E	48.5	61.54	84.28
F	45.15	67.56	51.17
G	50.84	68.23	90.64
H	46.82	66.56	89.97
I	38.46	19.4	51.51
Overall	48.25	56.94	66.58

Appendix II

Glossary

ANN: *Artificial neural network* – a network of interconnected units, called neurons, which activate or fire an output signal depending on the input signal. The network is trained using a training algorithm, *e.g.*, backpropagation algorithm, and training data. Neural networks are also known as universal approximators, given that they are able to approximate any function if sufficient nodes and training set data are provided.

EA: *Evolutionary algorithm* – a family of computer algorithms inspired by natural selection processes. In addition to GP, those algorithms include the Genetic Algorithm (GA), Evolutionary Programming (EP), and Evolution Strategies (ES). These algorithms are population-based and use one (or all) of the so-called genetic operators: selection, crossover, and mutation.

Epilepsy: (ĕp'ə-lĕp'sē) – a neurological condition that makes people susceptible to seizures. A seizure is a term used to describe a change in sensation, awareness, or behavior brought about by a brief electrical disturbance in the brain. Seizures vary from a momentary disruption of the senses, to short periods of unconsciousness or staring spells, to convulsions.

GA: *Genetic algorithm* – a general purpose, global optimization search algorithm that emulates the natural selection process by using genetic operators: selection, crossover, and mutation.

GFNCAF: *Genetically found neurally computed artificial feature* – an algorithm proposed in [31], which uses a GA and a classifier to create relevant artificial features from a given set of conventional features or raw data directly. The algorithm also may reduce the dimensionality of the feature space.

GP: *Genetic programming* – an extension of GA, genetic programming (GP) uses the genetic operators to evolve programs (solvers), unlike GA, which evolves solutions to solve problems, among them real-world applications. One of the most important aspects of GP is its variable length representation, which is designed to find simple as well as complex programs.

GPAF: *Genetic programming artificial feature* – an algorithm that uses a GP in combination with a classifier to construct artificial features from a set of classical features or raw data. This approach overcomes some of the limitations of the GFNCAF algorithm.

Ictal: (ĭk'təl) – period or signal in which a seizure occurs.

***k*-NN**: *k-nearest neighbor classifier* – a supervised, nonparametric classifier which uses a distance measure, commonly Euclidean distance, to categorize an input pattern depending on the class value of the maximum number of samples closer (nearest neighbors) to the input pattern. Typically, *k* is assigned to be three or five.

PFA: *probability of false alarm* – probability that the next negative class feature vector sample (which for practical purposes is approximately all samples) will be a false positive.

PFN: *probability of false negative* – probability that the next positive class feature vector sample will be a false negative.

Preictal: (prē-ĭk'təl) – signal refers to an EEG signal from a time period just prior to an epileptic seizure.

UCO: *unequivocal clinical onset* – time when an observer can see the consequences of a seizure in the person, for example, a convulsion.

UEO: *unequivocal EEG onset* – time of seizure onset identified in an EEG signal by an electroencephalographer.

BIBLIOGRAPHY

[1] "Device to cut seizures gets F.D.A.'s nod," *Dow Jones,* Washington, July 16, 1997.

[2] A. A. Adewuya, *New Methods in Genetic Search with Real-Valued Chromosomes*, Master's Thesis. Massachusetts Institute of Technology, Cambridge, MA, 1996.

[3] W. Banzhaf, P. Nordin, R. E. Keller, and F. D. Francone, *Genetic Programming: An Introduction*, San Francisco, CA: Morgan Kaufman Publishers, Inc., 1998.

[4] C. Bigan, "A system for neural networks detection and automatic identification of EEG epileptic events," *IEE Colloquium in Intelligent Methods in Healthcare and Medical Applications*, No.1998/514, pp. 13/1-13/4, 1998.

[5] A. Block and R. Fisher, "Can patients perform volitional motor acts at the start of a seizure?," *Journal of Clinical Neurophysiology*, vol. 16, no. 2, pp. 141-145, Mar. 1999.

[6] V. E. Bondarenko, "Epilepsy-like phenomena in chaotic neural networks," *IEEE International Conference on Neural Networks 1996*, vol. 2, pp. 774-777, 8-11 Nov. 1996.

[7] D. A. Campbell and L. W. Cahill, "Transient discrimination in nonlinear time series using linear-phase tome-delay neural networks," *IEEE International Symposium on Circuit and Systems 1996*, vol. 2, pp. 524-527, 12-15 May 1996.

[8] E. Chang, R. Lippmann, and D. Tong, "Using genetic algorithms to select and create features for pattern classification," *Proceeding IEEE International Joint Conference on Neural Networks*, vol. 3, pp. 747-752, 1990.

[9] H. Chen, S. Zhong, and D. Yao, "Detection singularity value of character wave in epileptic EEG by wavelet," *IEEE International Conference on Communications, Circuits and Systems and West Sino Expositions 2002*, vol. 2, pp. 1094-1097, 29 Jun.-1 Jul., 2002.

[10] V. Cherkassky and F. Mulier, *Learning from Data: Concepts, Theory, and Methods*. NY: Wiley, 1998.

[11] L. O. Chua and L. Yang, "Cellular neural networks: Theory". *IEEE Transactions on Circuits and Systems-I*, Volume: 35, no. 10, pp. 1257-1272, Oct. 1988.

[12] L. O. Chua and L. Yang, "Cellular neural networks: Applications". *IEEE Transactions on Circuits and Systems-I*, Volume: 35, no. 10, pp. 1273-1290, Oct. 1988.

[13] W. J. Conover, *Practical Nonparametric Statistics*, 3rd eds. Wiley, John & Sons, Inc. 1998.

[14] M. D'Alessandro, *The Utility of Intracranial EEG Feature and Channel Synergy for Evaluating the Spatial and Temporal Behavior of Seizure Precursors*. Ph.D. Dissertation, Georgia Institute of Technology, Atlanta, 2001.

[15] M. D'Alessandro, R. Esteller, G. Vachtsevanos, A. Hinson, J. Echauz, and B. Litt, "Epileptic seizures prediction using hybrid feature selection over multiple intracranial EEG electrode contacts: a report of four patients," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 5, pp. 603-615, May 2003.

[16] M. D'Alessandro, G. Vachtsevanos, R. Esteller, J. Echauz, D. Sewell, and B. Litt, "A systematic approach to seizure prediction using genetic algorithm and classifier based feature selection," *14th International Conference on Digital Signal Processing 2002*, vol. 2, pp. 603-606, 1-3 July 2002.

[17] M. D'Alessandro, G. Vachtsevanos, A. Hinson, R. Esteller, J. Echauz, and B. Litt, "A genetic approach to selecting the optimal feature for epileptic seizure prediction," *Proceedings of the 23rd Annual EMBS International Conference 2001*, pp. 1703-1706, Oct. 25-28, 2001.

[18] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd edition. New York, NY: John Wiley & Sons, Inc., 2001.

[19] R. C. Eberhart, R. W. Dobbins, and W. R. S. Webber, "Neural network design considerations for EEG spike detection," Proceedings of the 1989 15th Annual Northeast Bioengineering Conference, pp. 97-98, 1998.

[20] J. Echauz, *Wavelet Neural Networks for EEG Modeling and Classification*. Ph.D. dissertation, Georgia Institute of Technology, Aug. 1995.

[21] J. Echauz, S. Kim, V. Ramani, and G. Vachtsevanos, "Neuro-fuzzy approaches to decision making: A comparative study with an application to check authorization," *Journal of Intelligent and Fuzzy Systems*, no. 6, pp. 259-278, June 1998.

[22] Epilepsy Foundation of America, Landover, MD, http://www.efa.org.

[23] Epilepsy Foundation of America, Landover, MD, Epilepsy Facts & Figures, http://www.efa.org/education/facts.html, 1999.

[24] L. J. Eshelman and D. J. Shaffer, "Real-coded genetic algorithms and interval-schemata". *Foundations of Genetic Algorithms 2.* San Mateo, CA: Morgan Kaufman, pp. 187-202, 1993.

[25] R. Esteller, J. Echauz, T. Tcheng, B. Litt, and B. Pless, "Line length: An efficient feature for seizure onset detection," *Proceedings of the 23th Annual International Conference of the IEEE Engineering in Medicine and Biology Society 2001*, pp.1707-1710, Oct. 25-28 2001.

[26] R. Esteller, G. Vachtsevanos, J. Echauz, M. D'Alessandro, C. Bowen, R. Shor, and B. Litt, "Fractal dimension detects seizures onset in mesial temporal lobe epilepsy," *Proceedings of the First Joint BMES/EMBS Conference: Serving Humanity, Advancing Technology,* Atlanta, GA, pp. 442, Oct. 13-16, 1999.

[27] R. Esteller, G. Vachtsevanos, J. Echauz, T. Henry, P. Pennell, C. Epstein, R. Bakay, C. Bowen, and B. Litt, "Fractal dimension characterizes seizure onset in epileptic patients," *Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing 1999*, vol. 4, pp.2343 – 2346, 1999.

[28] K. J. Falconer, *Fractal Geometry: Mathematical Foundations and Applications*, John Wiley & Sons, 2003.

[29] P. J. Flegg and A. Kirton, "Seizure-alerting and -response behaviors in dogs living with epileptic children," *Neurology*, 64 p. 581, Feb 2005.

[30] H. Firpi, *Genetically Found, Neurally Computed Artificial Features with Applications to Epileptic Seizure Detection and Prediction.* Master's Thesis, University of Puerto Rico-Mayagüez, Mayagüez, PR, 2001.

[31] H. Firpi and J. Echauz, "Genetically found, neurally computed artificial features from relevant and irrelevant data," *Group Technology/Cellular Manufacturing World Symposium 2000*, San Juan, PR, Mar. 27-29, pp. 327-331, 2000.

[32] H. Firpi and E. Goodman, "Swarmed feature selection," *Proceedings of Emerging Technologies and Applications for Imagery Pattern Recognition (AIPR 2004)* Washington DC, pp. 112-118, 2004.

[33] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* NY: IEEE Press, 1995.

[34] D. B. Fogel, *Evolutionary Computation: The Fossil Record.* New York, NY: IEEE Press, pp. 301-309, 1998.

[35] L. J. Fogel, *Intelligence through Simulated Evolution: Forty Years of Evolutionary Programming.* New York, NY: John Wiley & Sons, Inc., 1999.

[36] K. Fukunaga, *Introduction to Statistical Pattern Recognition.* NY: Academic Press, 1972.

[37] R. Gilmore and M. Lefranc, *The Topology of Chaos: Alice in the Stretch and Squeezeland.* John Wiley and Sons Inc., 2002.

[38] D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning.* Reading, MA: Addison-Wesley, 1989.

[39] J. C. Goswami and A. K. Chan, *Fundamentals of Wavelets: Theory, Algorithms, and Applications*, John Wiley & Sons, Inc., 1999.

[40] R. L. Haupt and S. E. Haupt, *Practical Genetic Algorithms.* New York, NY: John Wiley & Sons, Inc., 1998.

[41] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd edition. Upper Saddle River, NJ: Prentice-Hall, Inc., 1999.

[42] R. Harikumar and B. Sabarish Narayanan, "Fuzzy techniques for classification of epilepsy risk level from EEG signals," *Conference on Convergent Technologies for Asia-Pacific Region*, vol. 1, pp. 209-213, Oct. 15-17, 2003.

[43] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, pp. 66-72, July 1992.

[44] J. H. Holland, "Genetic algorithms and the optimal allocation of trials," *SIAM J. Computing*, vol. 2, no. 2, pp. 88-105.

[45] L. Iasemidis, D.-S. Shiau, W. Chaovalitwongse, J. C. Sackellares, P. M. Pardalos, J. C. Principe, P. R. Carney, A. P. Prasad, B. Veeramani, and K. Tsakalis, "Adaptive epileptic seizure prediction system," *IEEE Transactions on Biomedical Engineering*, vol. 50, no. 5, May 2003.

[46] L. Iasemidis, R. Savit, and J. Sackellares, "The use of dynamical analysis of EEG frequency content in seizure prediction," *Proceeding American Electroencephalographic Annual Conference*. New Orleans, LA. pp. (abstract), 1993.

[47] B. H. Jansen, "Quantitative analysis of electroencephalograms: Is there chaos in the future?," *International Journal of Biomedical Computing*, vol. 27, pp. 95-123, 1991.

[48] K. Jerger et al., "Early seizure detection," *Journal of Clinical Neurophysiology*, vol. 18, no. 3, pp. 259-268, 2001.

[49] R. A. Johnson and D. W. Wichern, *Applied Multivariate Statistical Analysis*, 5th ed. Prentice Hall, 2002.

[50] S. M. Kay, *Fundamentals of Statistical Signal Processing: Detection Theory*. vol. 2. Upper Saddle River, NJ: Prentice Hall, 1998.

[51] J. M. Keller, R. Gray, and J. A. Givens Jr., "A fuzzy *k*-nearest neighbor algorithm," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 15, no. 4, pp. 258-263, 1985.

[52] D. H. Kil and F. B. Shin, *Pattern Recognition and Prediction with Applications to Signal Characterization*. Woodbury, NY: AIP Press, 1996.

[53] M. E. Kirlangic, D. Pérez, S. Kudryavtseva, Griessbach, G. Henning, and G. Ivanova "Fractal dimension as a feature for adaptive electroencephalogram segmentation in epilepsy," *Proceedings of the 23rd Annual EMBS International Conference 2001*, pp. 1573-1576, Oct. 25-28, 2001.

[54] A. Kirton, E. Wirrell, J. Zhang, and L. Hamiwka, "Seizure-alerting and -response behaviors in dogs living with epileptic children," *Neurology*, 62, pp. 2303-2305, Jun. 2004.

[55] M. Kotani, M. Nakai, and K. Akazawa, "Feature extraction using evolutionary computation," *Proceedings of the Congress on Evolutionary Computation 1999*, pp. 1230-1236, 1999.

[56] J. R. Koza, "Hierarchical genetic algorithms operating on populations of computer programs," *Proceedings of the 11th International Joint Conference on Artificial Intelligence*, vol. 1, pp. 768-774, 1989.

[57] J. R. Koza, *Genetic Programming: On Programming of Computers by Means of Natural Selection*. Cambridge, MA: MIT Press, 1992.

[58] J. R. Koza, *Genetic Programming II: Automatic Discovery of Reusable Programs*. Cambridge, MA: MIT Press, 1994.

[59] S. A. Lee, D. D. Spencer, and S. S. Spencer, "Intracranial EEG seizure-onset patterns in neocortical epilepsy," *Epilepsia*, vol. 41, no. 3, pp. 297-307, 2000.

[60] B. Litt, R. Esteller, J. Echauz, M. D'Alessandro, R. Shor, T. Henry, P. Pennell, C. Epstein, R. Bakay, M. Dichter, and G. Vachtsevanos, "Epileptic seizures may begin hours in advance of clinical seizures: A report of four patients," *Neuron*, vol. 29, no. 4, 2001.

[61] S. Luke and L. Panait, "Lexicographic parsimony pressure," *Proceedings of Genetic and Evolutionary Computation Conference 2002*, San Francisco, CA: Morgan Kaufmann, pp. 829-836, 2002.

[62] T. McConaghy, H. Leung, and V. Varadan, "Functional reconstruction of dynamical systems from time series using genetic programming," 26th Annual Conference of the IEEE Industrial Electronics Society, 2000, vol. 3, pp. 2031-2034, 2000.

[63] S. Mehta, R. W. Koser, and P. Venziale, "Wavelet analysis as a potential tool for seizure detection," *Proceedings of the IEEE-SP International Symposium on Tome-Frequency and Time-Scale Analysis 1994*, pp. 584-587, 1994.

[64] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolutionary Programs*, 2nd edition, New York, NY: Springer-Verlag, 1994.

[65] E. Micheli-Tzanakou, *Supervised and Unsupervised Pattern Recognition: Feature Extraction and Computational Intelligence*. New York, NY: CRC, 2000.

[66] M. Mitchell, *An Introduction to Genetic Algorithms*, MIT Press, 1998.

[67] M. M. Mirbagheri, K. Badie, R. M. Hashemi Golpayegani, and M. Amir Ahmadi, "A neural network approach to EEG classification for the propose of differential diagnosis between epilepsy and normal EEG states," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 14, pp. 2649-2650, Oct. 29-Nov. 1 1992.

[68] A. H. Nayfeh and B. Balachandran, *Applied Nonlinear Dynamics: Analytical, Computational, and Experimental Methods*, John Wiley & sons, Inc., 1994.

[69] J. J. Niederhauser, "Simple electric model for epilepsy (SEMEY)," *IEEE Potentials*, vol. 21, issue 1, pp. 35-39, Feb.-Mar. 2002.

[70] C. Niederhoefer, F. Gollas, A. Chernihovskyi, K. Lehnertz, and R. Tetzlaff, "Detection of seizure precursors in the EEG with cellular neural networks," *Epilepsia*, vol. 45, suppl. 7, p. 245 (abstract), 2004.

[71] S. J. Orfanidis, *Optimum Signal Processing: An Introduction*, 2nd edition. Englewood Cliffs: NJ: Prentice Hall, 1996.

[72] Ö. Özdamar, C. N. Lopez, and İ. Yaylali, "Detection of transient EEG patterns with adaptive unsupervised neural networks," *IEEE International Biomedical Engineering Days*, pp. 192-197, 1992.

[73] S. K. Pal, S. Bandyopadhyay, and C. A. Murphy, "Genetic algorithms for generations of class boundaries," *IEEE Transactions on System, Man and Cybernetics*, vol. 28, pp. 816-828, 1998.

[74] H. S. Park, Y. H. Lee, D. S. Lee, and S. I. Kim, "Detection of epileptiform activity using wavelet and neural network," *Proceedings of the 19th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp.1194-1197, Oct. 30-Nov. 2 1997.

[75] T. S. Parker and L. O. Chua, "Chaos: A Tutorial for Engineers," *Proceedings of the IEEE*, pp. 982-1008, Aug. 1987.

[76] E. A. Patrick and F. P. Fischer III, "A generalized *k*-nearest neighbor rule," *Information and Control*, vol.: 16, no. 2. pp. 128-152, 1970.

[77] N. Pradhan and P. K. Sadasivan, "Validity of dimensional complexity measurements of EEG signals," *International Journal of Bifurcation and Chaos*, vol. 7, no.1, pp. 173-186, 1997.

[78] W. S. Pritchard and D. W. Duke, "Measuring chaos in the brain: a tutorial review of nonlinear dynamical EEG analysis," *International Journal of Neuroscience* (preprint), Apr. 1992.

[79] W. Punch, E. Goodman, M. Pei, L. Chai-Shun, P. Hovland, and R. Enbody, "Further Research on Feature Selection and Classification Using Genetic Algorithms", *ICGA 1993*, pp. 557-564.

[80] I. Rechenberg, *Evolutionsstrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution*. Frommann-Holzboog, Stuttgart, 1973.

[81] J. A. Rice, *Mathematical Statistics and Data Analysis*, 2nd eds. Duxbury Press, 1994.

[82] M. Risinger et al., "Ictal localization of temporal seizures with scalp-sphenoidal recordings," *Neurology*, vol. 39, pp. 1288-1293, 1989.

[83] Z. Rogowski, I. Gath, and E. Bental, "On the prediction of epileptic seizures," *Biological Cybernetics*, vol. 42, pp. 9-15, 1981.

[84] D. Ruelle, *Evolution and Strange Attractors: The Statistical Analysis of Time Series for Nonlinear Deterministic Nonlinear Systems*. Cambridge: Cambridge University Press, 1989.

[85] H. P. Schwefel, *Evolution and Optimum Seeking: The Sixth Generation*, Wiley, John & sons, 1993.

[86] D. A. Scott and S. J. Schiff, "Predictability of EEG interictal spikes," *Biophysical Journal*, vol. 69, pp. 1748-1757, Nov. 1995.

[87] S. Silva and J. Almeida "GPLab: A genetic programming toolbox for Matlab," *Proceedings of the Nordic MATLAB Conference 2003*, pp. 273-278, 2003.

[88] S. Silva and E. Costa, "Dynamic limits for bloat control: Variations on size and depth," *Proceedings of GECCO 2004*, Berlin, Germany: Springer Verlag, pp. 666-677, 2004.

[89] J. Sherrah, R. Bogner, and A. Bouzerboum, "Automatic selection of features for classification using genetic programming," *Proceedings of Australian and New Zealand Conference on Intelligent Information Systems 1996*, pp. 284-287, 1996.

[90] R. Sowa, F. Mormann, A. Chernihovskyi, C. Niederhoefer, R. Tetzlaff, C Elger, and K. Lehnertz, "Seizure prediction: Measuring EEG phase synchronization with cellular neural networks," *Epilepsia*, vol. 45, suppl. 7, p. 244 (abstract), 2004.

[91] J. C. Sprott, *Chaos and Time-Series Analysis*, New York, NY: Oxford Univesity Press, 2003.

[92] M. D. Srinath, P. K. Rajasekaran, and R. Viswanathen, *Introduction to Statistical Signal Processing with Applications*. Englewood Cliffs, NJ: Prentice Hall, 1996.

[93] S. D. Stearns, "On selecting features for pattern classifiers," *Third International Joint Conference on Pattern Recognition*, Coronado, CA, 1976.

[94] L. Szilágyi, Z. Benyó, and S. M. Szilágyi, "A new method for epileptic waveform recognition using wavelets decomposition and artificial neural networks," *Proceedings of the Second Joint EMBS/BMES Conference*, Houston TX, pp. 2025-2026, Oct. 23-26, 2002.

[95] F. Takens, "Detecting strange attractors in turbulence," *Dynamical Systems and Turbulence*, *Warwick 1980 Lecture Notes in Mathematics 898.* Berlin, Germany: Springer-Verlag, pp. 336-381, 1981.

[96] R. Tetzlaff and D. Weiβ, "Cellular neural networks for the anticipation of epileptic seizures," *IEEE International Symposium on Circuit and Systems 2002*, vol. 4, pp. IV-177-IV180, May 26-29, 2002.

[97] R. Tetzlaff, C. Niederhofer, and P. Fischer, "Feature extraction in epilepsy using a cellular neural network based device – first results," *Proceedings of the International Symposium on Circuit and Systems 2003*, vol. 3, pp. III-850-III-853, May 25-28, 2003.

[98] J. Theiler, "On the evidence for low-dimensional chaos in an epileptic electroencephalogram," *Physics Letters A*, vol. 196, pp. 335-341, Sept. 1995.

[99] P. Wahlberg, A. Grennberg, and G. Salomonsson, "Feature extraction from multichannel EEG transients for clustering purposes," *IEEE 17th Annual Conference Engineering in Medicine and Biology Society*, vol. 2, pp. 915-916, Sept. 20-23 1995.

[100] W. R. S. Webber, B. Litt, K. Wilson, and R. Lesser, "Practical detection of epileptiform discharges (EDs) in the EEG using an artificial neural network: A comparison of raw and parameterized EEG data," *Electroencephalography and Clinical Neurophysiology*, vol. 91, pp. 194-204, 1994.

[101] I. Yaylali, H. Koçak, and P. Jayakar, "Detection of seizures from small samples using nonlinear dynamic system theory," *IEEE Transactions on Biomedical Engineering*, vol. 43, no. 7, pp. 743-751, July 1996.

[102] J. Yen and R. Langari, *Fuzzy Logic: Intelligence, Control, and Information.* Upper Saddle River, NJ: Prentice Hall, 1999.

[103] J. Zhu, N. Hazarika, A. C. Tsoi, and A. A. Sergejew, "Classification of EEG signals using wavelet coefficients and an ANN," *Workshop of Brain Electric and Magnetic Topography*, Sydney, Australia. Feb. 1994.

[104]B. Zupan, M. Bohanec, J. Demsar, and I. Bratko, "Feature transformation by function decomposition," *IEEE Intelligent Systems*, pp. 38-43, March/April 1998.

Index

MICHIGAN STATE UNIVERSITY LIBRARIES

3 1293 02736 1983