DECLARATION

We, Samir Sheriff, Satvik N and Vaishakh B.N. bearing USN number 1RV09CS093 1RV09CS095 and 1RV09CS114 respectively, hereby declare that the project entitled "**Time Series Data Mining Tool**" completed and written by us, has not been previously formed the basis for the award of any degree or diploma or certificate of any other University.

Bangalore

Samir Sheriff
USN:1RV09CS093

Satvik N
USN:1RV09CS095

Vaishakh B N
USN:1RV09CS114

R V COLLEGE OF ENGINEERING

(Autonomous Institute Affiliated to VTU)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled, "**Time Series Data Mining Tool**", which is being submitted herewith for the award of B.E is the result of the work completed by **Samir Sheriff, Satvik N, Vaishakh B N** under my supervision and guidance.

Signature of Guide	Signature of Head of Department
(Mrs. Shanta R)	(Dr. N K Srinath)

Name of Examiner	Signature of Examiner
1:	
2:	

ACKNOWLEDGEMENT

Any achievement, be it scholastic or otherwise does not depend solely on the individual efforts but on the guidance, encouragement and cooperation of intellectuals, elders and friends. A number of personalities, in their own capacities have helped me in carrying out this project work. We would like to take this opportunity to thank them all.

First and foremost we would like to thank **Dr. B. S. Satyanarayana**, Principal, R.V.C.E, Bengaluru, for his moral support towards completing my project work.

We deeply express my sincere gratitude to our guides **Mrs. Shanta Rangaswamy**, Assistant Professor and **Dr. Shobha G**, Professor, Department of CSE, R.V.C.E, Bengaluru, for their able guidance, regular source of encouragement and assistance throughout this project.

We would like to thank **Dr. N. K. Srinath**, Head of Department, Computer Science & Engineering, R.V.C.E, Bengaluru, for his valuable suggestions and expert advice.

We thank our Parents, and all the Faculty members of Department of Computer Science & Engineering for their constant support and encouragement.

Last, but not the least, We would like to thank our peers and friends who provided us with valuable suggestions to improve our project.

Samir Sheriff
8th semester, CSE
USN:1RV09CS093

Satvik N
8th semester, CSE
USN:1RV09CS095

Vaishakh B N
8th semester, CSE
USN:1RV09CS114

ABSTRACT

A time series is a sequence of data points, measured typically at successive points in time spaced at uniform time intervals. Time series analysis comprises methods for analyzing time series data in order to extract meaningful statistics and other characteristics of the data.In the context of statistics,the primary goal of time series analysis is forecasting. In the context of signal processing it is used for signal detection and estimation, while in the context of data mining, pattern recognition and machine learning time series analysis can be used for clustering, classification, query by content, anomaly detection as well as forecasting. This project is aimed making a time series data mining tool which can be used to accomplish the above goals.

This project mainly focuses on analyzing the sea and rainfall level time series. The tool developed can be used to perform anomaly detection, forecasting, similarity detection. The tool has been developed and tested using these data sets.

The tool consists of various algorithms categorized under anomaly detection, forecasting, similarity detection and temporal pattern recognition. The performance of these algorithms were tested on the above data sets and the results are presented.

Every algorithm in this tool requires a set of user-defined parameters that determine the accuracy of the results. The Neural Network in the Forecasting module is the most accurate with 60% accuracy. The CUSUM and Statistical approach in the Anomaly-Detection module requires a threshold to be set by the user. The Temporal Pattern Mining tool requires a fitness threshold to be set by the user, and similarly, the Dynamic Time Warping tool in the Similarity Module requires no parameters.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

A time series is a set of observations Xt , each one being recorded at a specific time t. Discrete-time time series is one in which the set T of times at which observations are made is a discrete set. Continuous-time time series are obtained when observations are recorded continuously over some time interval, e.g., when T0 belongs [0,1]. Examples of time series are the daily closing value of the ECG readings and the annual flow volume of the Nile River at Aswan. Time series are very frequently plotted via line charts. Time series analysis comprises methods for analysing time series data in order to extract meaningful statistics and other characteristics of the data. Time series forecasting is the use of a model to predict future values based on previously observed values.

1.1 DEFINITIONS AND USAGE

Definition of Time Series: An ordered sequence of values of a variable at equally spaced time intervals. TSDM : Time Series Data Mining tool. @TODO ADD DETAILS OF THIS SECTION

1.2 LITERATURE SURVEY

A time series is a collection of observations made sequentially through time. At each time point one or more measurements may be monitored corresponding to one or more

attributes under consideration. The resulting time series is called univariate or multivariate respectively. In many cases the term sequence is used in order to refer to a time series, although some authors refer to this term only when the corresponding values are non-numerical. Throughout this paper the terms sequence and time series are being used interchangeably. The most common tasks of time series data mining methods are: indexing, clustering, classification, novelty detection, motif discovery and rule discovery. In most of the cases, forecasting is based on the outcomes of the other tasks. A brief description of each task is given below.

Indexing: Find the most similar time series in a database to a given query time series.

Clustering: Find groups of time series in a database such that, time series of the same group are similar to each other whereas time series from different groups are dissimilar to each other.

Classification: Assign a given time series to a predefined group in a way that is more similar to other time series of the same group than it is to time series from other groups.

Novelty detection: Find all sections of a time series that contain a different behavior than the expected with respect to some base model.

Motif discovery: Detect previously unknown repeated patterns in a time series database.

Rule discovery: Infer rules from one or more time series describing the most possible behaviour that they might present at a specific time point (or interval).

The temporal aspect of data arises some special issues to be considered and/or imposes some restrictions in the corresponding applications. First, it is necessary to define a similarity measure between two time series and this issue is very important in TSDM since it involves a degree of subjectivity that might affect the final result. A lot of research has focused on defining different similarity measures in order to improve the performance of the corresponding methods. Second, it is necessary to apply a representation scheme on the time series data. Since the amount of data may range from a few megabytes to terabytes, an appropriate representation of the time series is necessary in order to manipulate and analyze it efficiently. The desirable properties that this approach should hold are: (a) the completeness of feature extraction (b) the reduction of the dimensionality curse [1]. More specifically, the method of extraction features should guarantee that there would be no

pattern missed, the number of patterns falsely identified as interesting will be minimized and the dimensionality reduction will be substantial. In many cases also, the objective is to take advantage of the specific characteristics of a representation that make specific methods applicable (i.e. inducing rules, Markov models). Consequently, the majority of the researchers are focused on defining novel similarity measures and representation schemes in order to improve indexing performance. Clustering and classification of time series rely heavily on the similarity measure and the representation scheme selected, thus, there are very few papers proposing a novel algorithm [2]. A recent survey on clustering time series is provided by Liao [3]. Novelty detection is a very important task in many areas. Several alternative terms for novelty have been used, such as, anomaly, interestingness, surprising, faults to name a few. Moreover, many problems of finding periodic patterns can be considered as similar problems. The important point here is to provide a clear and concise definition of the corresponding notion. For instance, Keogh et al. [14] describe a pattern as surprising if the frequency with which it appears, differs greatly from that expected given previous experience. The authors present a novel algorithm, called Tarzan, and provide useful pointers to relevant literature. Recently, Aref et al. [4] focus on discovering partial periodic patterns in one or more databases. They present algorithms for incremental mining (how to maintain discovered patterns over time as the database is being expanded). Motif discovery has only recently attracted the interest of the data mining community [9]. Motifs are defined to be previously unknown, frequently occurring patterns in a time series. These patterns may be of particular importance to other data mining tasks, such as, rule discovery and novelty detection. The recent work of Tanaka et al. [7] proposes a new method for identifying motifs from multi-dimensional time series. They apply Principal Component Analysis to reduce dimensionality and perform a symbolic representation. Then, the motif discovery procedure starts by calculating a description length of a pattern based on the Minimum Description Length principle.

- Indexing Indexing approaches are mostly influenced by the pioneer work of Agrawal et al. [1], generalized by Faloutsos et al. [12]. The emerged framework from these papers, referred as GEMINI, can be summarized in the following steps [11]: extract k essential features from the time series map into a point in k-dimension feature

space organize points with off-the-shelf spatial access method. discard false alarms The first and second step suggests the application of a representation scheme in order to reduce the dimensionality. However, this mapping should guarantee that it would return all the qualifying objects. This implies that the similarity measure in the k-dimension feature space should lower bound the corresponding similarity measure in the original space [8]. The third step is an opened selection, however most of the times R-tree structures are used. Other indexing structures may be vp-trees [7] [9], hB-trees and grid-files. The fourth step is a consequence of the fact that this approach can not guarantee that there will not be returned unqualified objects, thus these false alarms should be discarded in a post processing phase. Recently, Vlachos et al. [8] presented an external memory indexing method for discovering similar multidimensional time series under time warping conditions. The main contribution of this work is the ability to support various distance measures without the need to reconstruct the index. Two approaches with respect to distance measures are taken under consideration, namely, the Longest Common Subsequence (LCS) and the Dynamic Time Warping (DTW). Their indexing technique works by splitting a set of multiple time series in multidimensional Minimum Bounding Rectangles (MBR) and storing them in an R-tree. For a given query, a Minimum Bounding Envelope (MBE) is constructed, that covers all the possible matching areas of the query under time warping conditions. This MBE is decomposed into MBRs and then probed in the R-tree index.

- Time series representation There have been several time series representations proposed in the literature, mainly on the purpose of reducing the intrinsically high dimensionality of time series. We will refer to some of the most commonly used representations. Discrete Fourier Transform (DFT) [1] was one of the first representation schemes proposed within data mining context. DFT transforms a time series from the time domain into the frequency domain whereas a similar representation scheme, Discrete Wavelet Transform (DWT) [8], transforms it into the time/frequency or space/frequency domain. Singular Value Decomposition (SVD)

[5] performs a global transformation by rotating the axes of the entire dataset such that the first axis explains the maximum variance, the second axis explains the maximum of the remaining variance and is orthogonal to the first axis etc. Piecewise Aggregate Approximation (PAA) [3] divides a time series into segments of equal length and records the mean of the corresponding values of each one. Adaptive Piecewise Constant Approximation (APCA) [10] is similar to PAA but allows segments of different lengths. Piecewise Linear Approximation (PLA) approximates a time series by a sequence of straight lines. Recently, more representation schemes have been proposed in order to reduce dimensionality. The first class of these schemes consists of symbolic representations. Lin et al. [11] propose a Symbolic Aggregate Approximation (SAX) method, which uses as a first step the PAA representation and then discretizes the transformed time series by using the properties of the normal probability distribution. Bagnal[5] assess the effects of clipping original data on the clustering of time series. Each point of a series is mapped to 1 when it is above the population mean and to 0 when it is below. This representation is called clipping and has many advantages especially when the original series is long enough. It achieves adequate accuracy in clustering, it efficiently handles outliers and it provides the ability to employ algorithms developed for discrete or categorical data. Megalooikonomou et al. [30] introduce a novel dimensionality reduction technique, called Piecewise Vector Quantized Approximation (PVQA). This technique is based on vector quantization that partitions each series into segments of equal length and uses vector quantization to represent each segment by the closest codeword from a codebook. The original time series is transformed to a lower dimensionality series of symbols. This approach requires a training phase in order to construct the codebook, a data-encoding scheme and a distance measure. Cole et al. [9] provide a work that addresses the task of discovering correlated windows of time series (synchronously or with lags) over streaming data. They concentrate in the case where the time series are uncooperative, meaning that there does not exist a fundamental degree of regularity that would allow an efficient implementation of DFT transformations. The proposed method involves a combination of

several techniques sketches (random projections), convolution, structured random vectors, grid structures, and combinatorial design in order to achieve high performance. Gionis and Mannila [7] introduce a different approach, which is mainly motivated from research on human genome sequences. However, this approach is more general and involves multivariate time series. The notion behind their approach is that, the high variability that some time series very often exhibit, may be explained by the existence of several different sources that affect different segments of this series. More specifically, the task is to find a proper way to segment a time series into k segments, each of which comes from one of h different sources (k ¿¿h). This task is analogous to clustering the points of a time series in h clusters with the additional constraint that a cluster may change at most k-1 times. Gionis and Mannila provide three algorithms for solving this problem and they test them on synthetic and genome data. Finally, Vlachos et al. [3] propose to represent a time series by applying discrete Fourier transformations and retain the k best Fourier coefficients instead of the first few ones. Although this paper is motivated by mining knowledge from the query logs of the MSN search engine, the proposed methods may be applied for time series data mining in general.

- Similarity Measures The definition of novel similarity measures has been one of the most researched areas in the TSDM field. Generally, they are strongly related to the representation scheme applied to the original data. However, there are some similarity measures that appear frequently in the literature. Most of the researchers choices are based on the family of Lp norms, that include the Euclidean distance. Yi and Faloutsos [3] presented a novel and fast indexing scheme when the distance function is any of the arbitrary Lp norms (p = 1, 2, ,). Another similarity measure that attracted a lot of attention, Dynamic Time Warping (DTW), comes from the speech recognition field [6]. The main advantage of this measure is that it allows acceleration-deceleration of a series along the time dimension (nonlinear alignments are possible), however it is computationally expensive. Markov models have been constructed and experimented. Another family of distance measures,

Longest Common Subsequence Measures (LCS), often used in speech recognition and text pattern matching. As an example of this approach, we refer to the work of Agrawal et al. [2] who define two sequences as similar when they have enough, non-overlapping, time-ordered pairs of subsequences that are similar. Li et al. [6] propose an algorithm for fast and efficient recognition of motions in multi-attribute continuous motion sequences. The main contribution of this paper is the definition of a similarity measure based on the analysis of Singular Value Decomposition (SVD) properties of similar multi-attribute motions. The proposed measure deals with noise and takes into account the different rates and durations of each motion. The authors also propose a five-phase algorithm for handling segmentation and recognition in real-time. Sakurai et al. [5] propose the Fast search method for dynamic Time Warping (DTW) that satisfies the following criteria: (a) it is fast (b) it produces no false dismissals (c) it does not pose any restriction on the series length (d) it supports for any, as well as for no restriction on warping scope. Their approach is based on a new lower bounding distance measure. They represent the sequence with approximate segments, not necessary of equal length, and operate on them. Three segments, the lower bound, the upper bound, and the time interval, correspond to each one of these approximate segments. In order to fulfill all of the above criteria, they provide algorithms for dynamic programming and searching adjusted to the properties of this representation. Fu et al. [14] propose a new technique to query time series that incorporates global scaling and time warping. The argument is that most real world problems require the ability to handle both types of distortion simultaneously. The approach is to scale the sequence by a bounded scaling factor and also to find nearest neighbor or evaluate range query by applying time warping. The authors provide definitions and proofs of the necessary lower bounds. Furthermore, there is the expected contribution to defining similarity measures by papers that propose novel representation schemes, since these two tasks are interrelated to each other.

1.3 MOTIVATION

mr. Manju. @TODO Add video link here..

1.4 PROBLEM STATEMENT

Make a paper and publish water data everywhere. :/ @TODO add problem statement.

1.5 OBJECTIVE

The ability to model and perform decision modelling and analysis is an essential feature of many real-world applications ranging from emergency medical treatment in intensive care units to military command and control systems. Existing formalisms and methods of inference have not been effective in real-time applications where trade-offs between decision quality and computational tractability are essential. The objective of this project is to fill the void that exists and help in proper analysis of time varying data.

1.6 SCOPE

The scope of a time series data mining tool is two fold. The first is to obtain an understanding of the underlying forces and structure that produced the observed data. The second is to fit a model and proceed to forecasting, monitoring or even feedback and feed forward control. The time series data mining tool can be used in the following fields.

- **Economic Forecasting**
- **Sales Forecasting**
- **Rainfall Analysis**
- **Stock Market Analysis**
- **Yield Projections**

- **Process and Quality Control**
- **Census Analysis**

1.7 METHODOLOGY

Time series analysis of data requires the user to able to view the different algorithms and the result obtained from each algorithm along with the graphs which help the user understand the time varying nature of the data. Hence, the representation of data becomes very important. Having understood this requirement in the early phase of the project, we adopted a methodology that will accomplish the objectives in a neat and intuitive way. A GUI was developed in the form of Java Server Pages and the back end was coded in Java which helped us exploit the object oriented paradigm in design of algorithms.

1.8 ORGANIZATION OF REPORT

@TODO after all chapters are complete.

Chapter 2

SOFTWARE REQUIREMENTS SPECIFICATION

Software Requirement Specification (SRS) is an important part of software development process. It includes a set of use cases that describe all the interactions of the users with the software. Requirements analysis is critical to the success of a project.

2.1 PRODUCT PERSPECTIVE

Time Series Data Mining tool is a unique product that makes use of different algorithms to predict, view similarities, and points out the anomalies in different time varying data sets. It is built in a pluggable fashion where the only requirement at the users end is the browser and a working internet connection.

2.2 PRODUCT FEATURES

The time series data mining tool has many features that distinguishes it from the others available already in the open world. It provides accurate results using the similarity finding, anomaly finding algorithms. The back propagation neural network helps us predicting the future values. On the front end, the user has options to choose the algorithm

of her choice. Also, the charts which depict the output are carefully plotted using the google charts API which has been made available by Google Inc. Also, Java beans along with servlets and java server pages and best practices of coding have been followed.

2.3 CONSTRAINTS

During the development of this product, constraints were encountered. Some specific constraints under which the time series data mining tool has are :

- Add Constraints
- Add Constraints

2.4 ASSUMPTIONS AND DEPENDENCIES

- It is assumed that the user of this tool has basic understanding of time series data mining.
- Also, the user must have a decent knowledge of the interpretation of line graphs.

2.5 SPECIFIC REQUIREMENTS

This section shows the functional requirements that are to be satisfied by the system. All the requirements exposed here are essential to run this tool successfully.

2.5.1 FUNCTIONAL REQUIREMENTS

The functionality requirements for a system describe the functionality or the services that the system is expected to provide. This depends on the type of software system being developed. The requirements that are needed for this project are :

- The data sets should be normalized so that the algorithms can be applied effectively.

- A good representation of the results should be made available to the users through proper representation media like graphs.
- TODO ADD SOMETHING HERE.

2.5.2 SOFTWARE REQUIREMENTS

DEVELOPERS MACHINE

- Operating System: Windows 7/8, Linux, Mac
- Software Tools : Java, JDK 7.0, Apache Tomcat Server version 7.0
  Web Browser (Mozilla, IE8+, Chrome)
- IDE : Eclipse IDE for J2EE Developers
- API Libraries : JQuery UI and Ajax Libraries (Active Internet Connection)

END USERS MACHINE

- Java Enabled Browser
- Active Internet Connection

2.5.3 HARDWARE REQUIREMENTS

- Processor: Intel Pentium 4 or higher version
- RAM: 512MB or more
- Hard disk: 5 GB

SOFTWARE INTERFACES

The Java Runtime Environment (JRE) is required to run the software.

Chapter 3

HIGH LEVEL DESIGN

The software development usually follows Software Development Life Cycle (SDLC). The second stage of SDLC is the design phase. The design stage involves two substages namely High level design and Detailed level design.

High level design gives an overview of how the system works and top level components comprising the system.

3.1 SYSTEM ARCHITECTURE

This section provides an overview of the functionality and the working of the time series data mining tool. The overall functionality of the application is divided into different modules in an efficient way. The system architecture is shown in Figure 3.1

3.2 DATA FLOW DIAGRAMS

A DFD is a figure which shows the flow of data between the different processes and how the data is modified in each of the process. It is very important tool in software

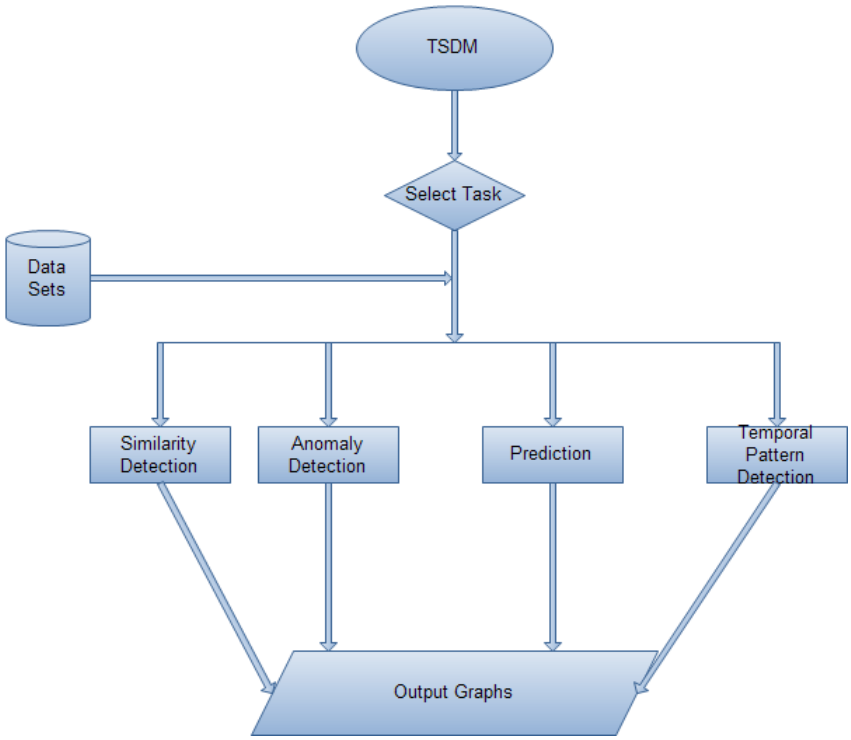

Figure 3.1: System Architecture

engineering that is used for studying the high level design.

There are many levels of DFDs. Level 0 gives the general description and level 1 gives the detailed description. Going higher in the level numbers greater description of the processes will be given.

3.2.1 DFD LEVEL 0

The level 0 DFD is shown in Fig. 3.2 below ehich gives the general operation of the steganographic system There are three major components. Two external entities called sender and receiver and one major system called the steganographic system.

- Sender : The sender is the one responsible to send the data to the receiver. The data to be sent is hidden using the Steganography system. The data is hidden in an image.
- Receiver : The receiver receives the data that is sent to him in the embedded format. The receiver then extracts the data from the embedded format to get the actual data. The receiver makes use of the Steganography system to extract the data.
- Steganography System : This the software which is used to embed the data into the image and also to extract the data from the image.

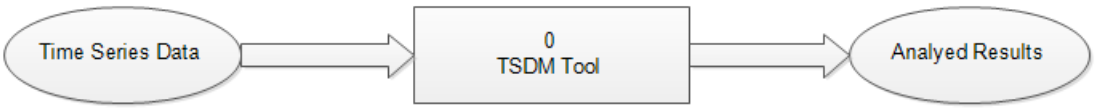

Figure 3.2: Data Flow Diagram Level 0

3.2.2 DFD LEVEL 1

There are two major processes in level 1 DFD as shown in Figure 3.3. The processes invloved in here are :

- Data Embedding: This process represents the actual embedding the data. The inputs given to this process are the data files, cover image. Input image is where the data is to be embedded and the key must also be shared between the sender and the receiver.
- Data Extraction: This process is used to extract the data back from the stego image. This is the reverse process of embedding. Here the input is the stego image and the key.

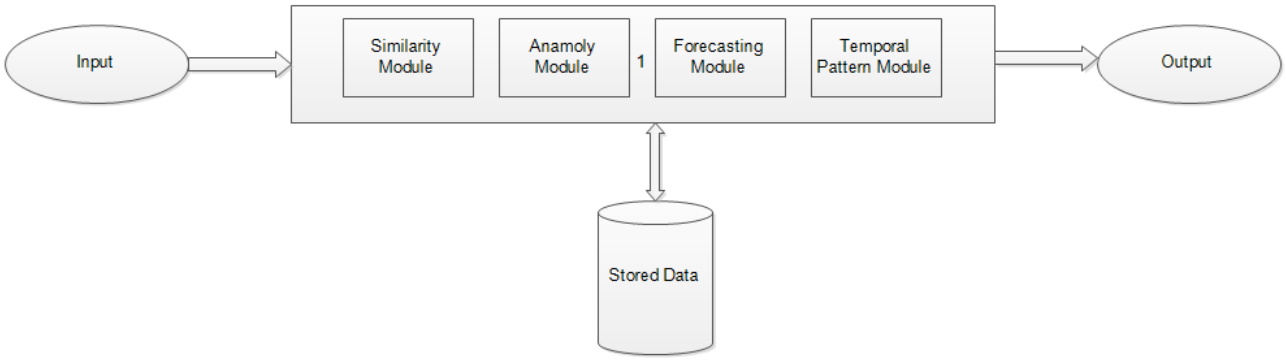

Figure 3.3: Data Flow Diagram Level 1

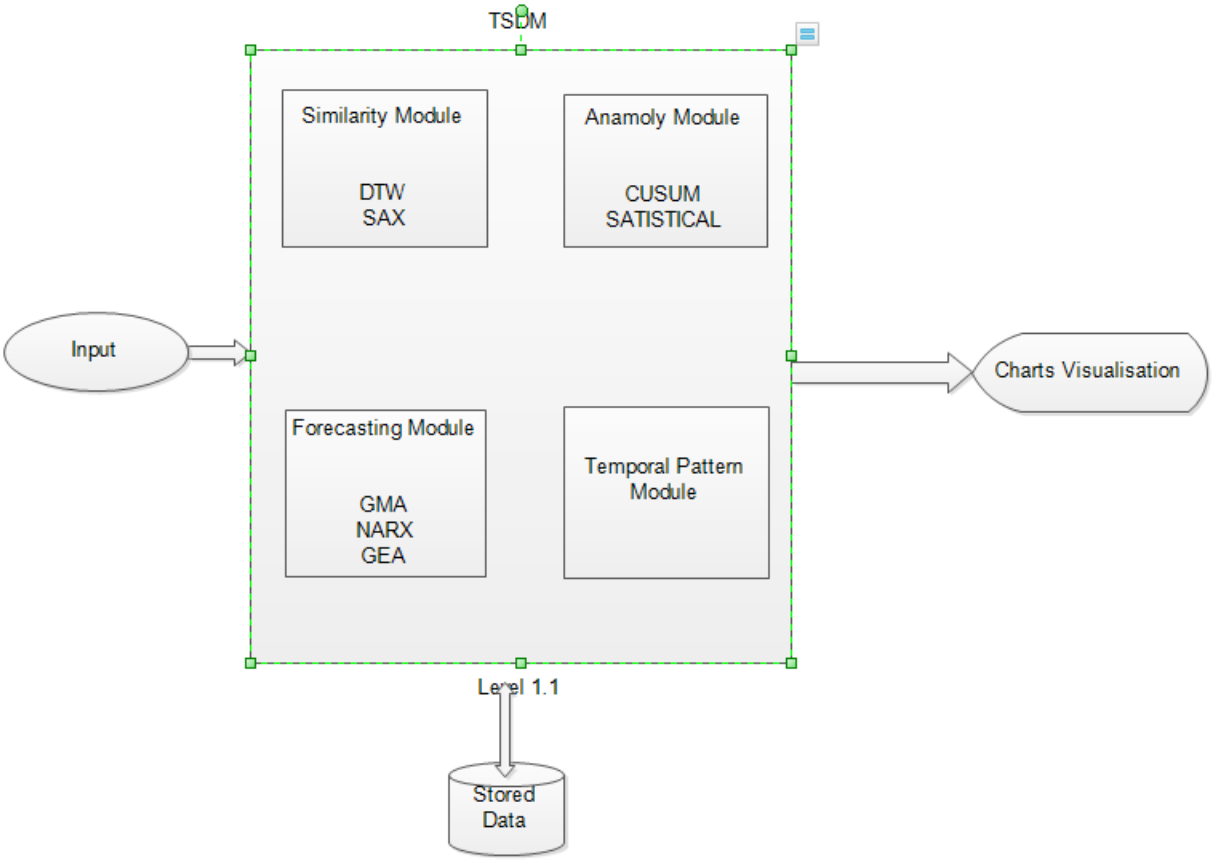

Figure 3.4: Data Flow Diagram Level 1.1

3.2.3 DFD LEVEL 2

Embedding Phase

The major processes in Level 2 DFD of Embed process on the sender side are shown in Figure 3.4.

- Data Holder: Data holder contains all the data bits, which upon invocation will give the required number of bits of data need to be embedded.
- Pixel Retriever: This retrieves the individual pixels from the cover image.
- Processor: Takes the key as the input, encrypts the data and embeds it into the

pixels received from the retriever.

- A final image is formed as a result of this process.

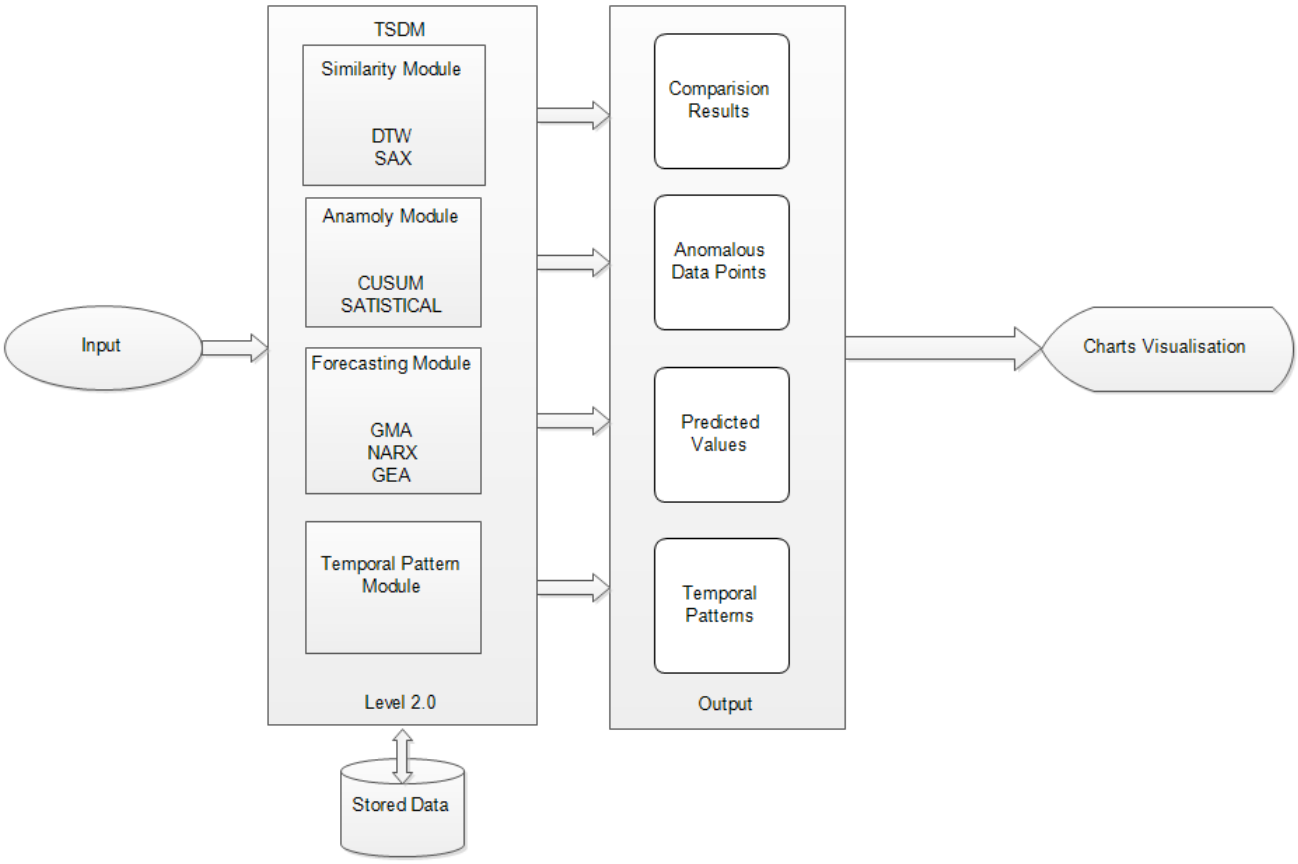

Figure 3.5: Data Flow Diagram Level

Extraction Phase

The major processes in the Level 2 DFD of the Extraction module on the receiver side is shown in Figure 3.5.

- Data Retriever: The data retriever module extracts the data file from the morphed image which it receives. The key should be present for proper retrieval.
- Pixel Retriever: Retrieves the pixels from the morphed image and pprovides it to the Data Retriever.

Chapter 4

DETAILED DESIGN

4.1 STRUCTURED CHART

Structure charts are used to specify the high level design or architecture of a computer program. As a design tool, they help the programmer in dividing and conquering a large software problem, i.e. recursively breaking a problem down into parts that are small enough to be understood by a human brain. The process is called top-down design or functional decomposition.

Programmers use a structure chart to build a program in a manner similar to how an architect uses a blueprint to build a house. In the design stage, the chart is drawn and used as a method for the client and various software designers to communicate. During the actual building of the program, the chart is continuously referred to as master plan. Often, it is modified as programmers learn new details about the program. After a program is completed, the structured chart is used to fix bugs and to make changes.

The entire program starts with user entering his choice of input as to either embed or extract. Based on this, the GUI module responds appropriately with success or failure. The digital media chosen by user is taken as input for embedding phase. The digital media is encrypted using AES algorithm before embedding to stego image.This encrypted data is embedded onto image using embedding algorithm to get the stego image containing

secret data. A key is generated using Diffie Hellman Key exchange which acts as the input for the embedding algorihm.

At the receiver end, the Diffie Hellman key is generated once again. From the stego images, secret data is extracted. It will be in encrypted form. The original input data is extracted by using decompression module of AES algorithm. This gives the hidden file's original content.

4.2 MODULES DESCRIPTION

This section describes the main modules that are used in developing the project. This will help us in understanding the working of individual components.

4.2.1 GUI MODULE

Definition:

This module is the core module which takes the users input to decide upon which operation to be performed. Based upon user choice, the necessary inputs will be taken in this module. If any errors are generated during any operation, then suitable report is displayed to the user.

Resources:

The input files are bitmap images and are chosen to embed data. A session key is generated using Diffie Hellman protocol. Also, the data file to be hidden is taken.

Functionality:

This is the main flowchart which shows all the functions provided by the software. In the beginning, the user is given two options, according to which the user either embeds extracts or exits from the application. This is shown in the decision symbol. According to the decision taken, the operations are performed. If the user chooses to embed, then the functions to select cover image, the secret data, and key generation are performed. Before embedding the secret data encryption is done. After performing these operations, the stego images are sent to the receiver using any wireless medium or LAN. At the receivers end, the receiver implements the functions to extract stego image values, extract secret digits data. After the implementation of these functions, if the user wants to exit from the software, that particular option is also provided to the user. This is shown in Fig 4.2.

Chapter 5

IMPLEMENTATION

The implementation phase of any project development is the most important phase and yields the final solution which solves the problem at hand.The implementation phase involves the actual materialization of the ideas, which are expressed in a suitable programming language. The factors concerning the programming language selection and platform chosen are described in the following sections.

5.1 PROGRAMMING LANGUAGE SELECTION

The programming language chosen must reflect the necessities of the project to be completely expressed in terms of the analysis and the design documents. Therefore before choosing the language, features to be included in the project are decided. The time series data mining project needs the following features in a language to be implemented. Some of the features required are stated as follows:

- J2EE provides us with servlets and JSP which help in dynamically constructing web pages.
- J2EE provides us with Java Beans which help in proper data manipulation.
- JSP and servlets make use of Java backend in a very optimal manner. They have special tags which help us exploit these features.

- Java's core classes are designed from scratch to meet the requirements of an object oriented system.

With these necessities in mind, J2EE is selected as the optimal programming language to implement the project.

5.2 PLATFORM

The TSDM tool was built and designed on Windows Operating system family. They were specifically tested on Windows 7 with Google Chrome and Mozilla Firefox browsers. Because the product is browser based, any user with the browsers mentioned above will be able to run the tool. The product is hence platform independent in the true sense.

5.3 CODE CONVENTIONS

The code standards for the Java programming Language document contains the standard conventions that follows. It includes file names, file organizations, indentation, comments, declarations, naming conventions and programming practices. Code conventions improve the readability of the software.

5.3.1 Naming Conventions

A naming convention is a set of rules for choosing the character sequence to be used for identifiers which denote variables, types, functions, and other entities in source code and documentation. There are several common elements that influence most if not all naming conventions in common use today. They are :

- **Use mixed case to make names readable**
- **Avoid long names (15 characters maximum is a good idea)**
- **Avoid names that are too similar or that differ only in case**
- **Capitalize the first letter of standard acronyms**

- **Use terminology applicable to the domain**
- **Use full descriptors that accurately describe the variable, field, or class**

5.3.2 File Organization

As stated above, this project has been developed using the eclipse JEE IDE. This is a dynamic web project. The project has been organized as follows :

- **Java Resources** - This folder contains the java classes organized in packages. All the algorithms implemented in java are present in this folder under different packages.
- **JavaScript Resources** - This folder contains the javascript library files.
- **Web Content** - This folder mainly contains the jsp, js, html, css files which are used for developing the front webpages.

5.3.3 Class Declarations

The following table describes the parts of a class or interface declaration, in the order that they should appear. @TODO :(

Part of Class/Interface declaration	Notes
Class/interface documentation comment (/**...*/)	Please see the next section
class or interface statement	
Class/interface implementation comment (/*...*/), if necessary	This comment should cont

5.3.4 Comments

- **Implementation Comment Formats**
  - **Block Comments**

```
/*
  * Here is a block comment.
```

```
  */
```

– **Single Line Comments**

```
if (condition) {

     /* Handle the condition. */
     ...
}
```

– **Trailing Comments**

```
if (a == 2) {
     return TRUE;            /* special case */
} else {
    return isPrime(a);      /* works only for odd a */
}
```

– **End-of-Line Comments**

```
if (foo > 1) {

    // Do a double-flip. Harlem Style
     ...
}
else  {
     return false;          // Explain why here.
}
```

- **Documentation Comments**

```
/**
 * The Example class provides ...
 */
```

```
public class Example { ...
```

5.4 DIFFICULTIES ENCOUNTERED AND STRATEGIES USED TO TACKLE

There were a number of challenges that were faced while implementing the Time Series Data Mining tool.Some challenges were challenging and ended up in helping us think innovatively and come up with efficient solutions. Some major problems that were encountered have been stated in brief along with their solutions.

Problem 1

Initially we wanted to build the front end in python using django web framework. But integrating java (back-end) with python had performance issues.(Using Jython interpreter)

Solution

We used JSP (Java server pages) for the front end and solved this problem.

Problem 2

In the initial stages of the project charts4j libraries were used to plot graphs. There were some internal problems with the URL rendering.

Solution

This Problem was solved later by making use of Google's Charts API and java script.

5.5 Module Design

Object-oriented programming (OOP) is a programming paradigm that represents concepts as "objects" that have data fields (attributes that describe the object) and associ-

ated procedures known as methods. Objects, which are instances of classes, are used to inter- act with one another to design applications and computer programs.

Had it not been for the presence of the OOP paradigm, our efforts in this project would have gone in vain, and we do not use that term lightly. Code management was a whole lot easier when compared to our past experience with procedural programming. In this chapter, we describe the different packages that were created by us to efficiently manage our code. Know first that our application consists of four diverse packages, namely:

1. *org.ck.sample*
2. *org.ck.smoother*
3. *org.ck.similarity*
4. *org.ck.forecaster.nn*
5. *org.ck.anomalifinder*
6. *org.ck.tsdm*
7. *org.ck.tsdm.ga*
8. *org.ck.beans*
9. *org.ck.servlets*
10. *org.ck.gui*

We describe each package in detail, in the following sections. We follow a bottom-up methodology for explaining the layout of the classes.

5.5.1 org.ck.sample

This package allows us to efficiently manage and encapsulate the details of the data samples, provided by users, which are required for analysis. It is this class that allows our application to accept generic data sets. It consists of four classes:

1. **DataHolder** - This class keeps track of names of files that contain time series data; the fitness score threshold for the genetic algorithm. It provides this information, when required, to the front-end or back- end of our application. To make a long story short, this class acts like a middleman between the back-end and front-end of our application.

2. **Sample** - This class stores the values of a given time series in various forms - discrete and continuous; normalized and unnormalized; smooth and unsmooth; SAX Representation. It keeps track of all dimensions of a given time-series

5.5.2 org.ck.smoothers

This package consists of a number of smoothing filters that can be used to smoothen time series values stored in an object of the Sample class. These classes also double up as naive forecaster modules. They are based on the concept of Moving Averages.

1. **SmoothingFilter** - This is the parent class of all the other classes in this package. It is also an abstract class. Hence, all the other classes that extend this class must provide implementations for the calculateSmoothedValues() and getAverage() methods compulsorily.

2. **ExponentialMovingAverageSmoother** - This class extends the SmoothingFilter class. Exponential smoothing is commonly applied to financial market and economic data, but it can be used with any discrete set of repeated measurements. The raw data sequence is often represented by x_t, and the output of the exponential smoothing algorithm is commonly written as s_t, which may be regarded as a best estimate of what the next value of x will be. When the sequence of observations begins at time t = 0, the simplest form of exponential smoothing is given by the formulae:

   $s_0 = x_0 \; s_t = \alpha x_t + (1 - \alpha) * s_{t-1}, t > 0$

   where α is the smoothing factor, and $0 < \alpha < 1$.

3. **SimpleMovingAverageSmoother** - This class extends the SmoothingFilter class. A simple moving average (SMA) is the unweighted mean of the previous n datum points.

   $SMA = (t_m + t_{m-1} + t_{m-2} + t_{m-3} + ... + t_{m-(n-1)}) \div n$

   where t_m is the value of the time series at the m^{th} instance of time.

4. **GeometricMovingAverageSmoother** - This class extends the SmoothingFilter class.The Geometric moving average calculates the geometric mean of the previous N bars of a time series.

5.5.3 org.ck.similarity

This module helps in finding similarity patterns (that occur at regular intervals in case of periodic time series), comparing different time series data. SAX and DTW are the main algorithms implemented/used in this module.

1. **DynamicTimeWarper** - Dynamic time warping (DTW) is an algorithm for measuring similarity between two sequences which may vary in time or speed. In general, DTW is a method that allows a computer to find an optimal match between two given sequences (e.g. time series) with certain restrictions. The sequences are "warped" non-linearly in the time dimension to determine a measure of their similarity independent of certain non-linear variations in the time dimension. This sequence alignment method is often used in time series classification. This class implements the aforementioned functionality.

2. **Discretizer** - This Singleton class is used to convert a time series represented by a PAA (Piecewise Aggregate Approximation), to a string representation (SAX - Symbolic Aggregate Approximation)

3. **Approximator** - This class averages out any time series containing continuous values using Piecewise Aggregate Approximation, allowing the time series to occupy as small a space as possible.

5.5.4 org.ck.forecaster.nn

This package contain algorithms/models which can be trained from the past time series data and can be used to predict the future values of a time series. In more practical terms neural networks are non-linear statistical data modeling or decision making tools.

They can be used to model complex relationships between inputs and outputs or to find patterns in data.

1. **Neuron** - In a neural network model simple nodes (which can be called by a number of names, including "neurons", "neurodes", "Processing Elements" (PE) and "units"), are connected together to form a network of nodes hence the term "neural network".

2. **NetworkLayer** - Creating the NN architecture therefore means coming up with values for the number of layers of each type and the number of nodes in each of these layers.

   - The Input Layer
   - The Hidden Layer
   - The Output Layer

3. **NeuralNetwork** - A neural network (NN), in the case of artificial neurons called artificial neural network (ANN) or simulated neural network (SNN), is an interconnected group of natural or artificial neurons that uses a mathematical or computational model for information processing based on a connectionistic approach to computation.

5.5.5 org.ck.anomalifinder

This package contains algorithms that help in indicating anomalous patterns in the time series data analyzed. Anomalies are patterns in time series which deviate from the normal behavior and can indicate fraud/danger depending on the application. For example in an industry which produces the blades, the thickness of the blade can be monitored by a machine as a time series and any deviation from the normal error rate can signal an error in the manufacturing process.

1. **Cusum_VmaskApproch** -

2. **CusumAnomalyMethod** -

5.5.6 org.ck.tsdm

This package contains a set of classes that reveal hidden patterns in time series data and overcome limitations of traditional time series analysis techniques. This Temporal Pattern Mining tool focuses on predicting events, which are important occurrences. This allows the TSDM methods to predict nonstationary, nonperiodic, irregular time series, including chaotic deterministic time series. It makes use of a genetic algorithm, internally.

1. **TSDM** - This class is to be used to find Temporal Patterns in Time Series and maintains status information about the algorithm.
2. **PhaseSpace** - Represents a Q-Dimensional Phase Space of points in the time series
3. **PhasePoint** - A point in a Q-Dimensional Phase Space

5.5.7 org.ck.tsdm.ga

This package takes care of all operations of the Genetic algorithm that is used by the TSDM class of the org.ck.tsdm package.

1. **Genome** - This class takes as input, a chromosome that encodes a given phase space cluster. It keeps track of this chromosome, and provides methods to manipulate this chromosome; to calculate the fitness score of this chromosome; and to throw an exception when the fitness value threshold has been crossed or when the best solution has been discovered.
2. **Population** - As defined earlier, a population is a collection of genomes. And this is exactly what this class is. Initially, the Population class randomly initializes a large number of genomes, of which it keeps track. It provides methods such as roulette selection, reproduction, crossover, and mutation to operate on the population and discover the best genome, and hence, the best decision tree with the appropriate feature subset.

3. **OptimalScoreException** - This class is responsible for catching the best genome as soon as it is discovered, since the best genome should never be allowed to escape. It should be caught and nurtured for future use.

5.5.8 org.ck.beans

JavaBeans are reusable software components for Java. Practically, they are classes that encapsulate many objects into a single object (the bean). They are serializable, have a 0-argument constructor, and allow access to properties using getter and setter methods.

1. **TimeSeriesBean** - This bean stores information about requested values and results of calculations. The results produced by the Similarity, Forecaster, Anomaly Detection and Temporal Pattern Mining modules are stored in an object of this class, and the front-end reads the results from this bean. It is our very own custom class that allows any number of user-defined objects to be stored for communication.

5.5.9 org.ck.servlets

A Servlet is an object that receives a request and generates a response based on that request.

1. **MainController** - Since we've used the MVC design pattern, this servlet is the controller that gets requests from JSP pages, generates results through beans and forwards them to other jsp pages.

2. **AlgorithmUtils** - This class is a utility class that contains methods to run various algorithms in this tool.

5.5.10 org.ck.gui

This package allows each member of our team to test out the functionality of the back-end framework before we connect the front-end to it.

1. **Constants** - A number of constants and enumerations used by all the other classes and packages.

2. **MainClass** - After a method is added to any class, the method is called from the respective team member's method to test it. If it works fine, then the tested method is used in the front-end.

Chapter 6

Software Testing

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product Testing is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

6.1 Types Of Testing

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program input produces valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provides a systematic demonstration that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items: Valid Input: identified classes of valid input must be accepted. Invalid Input: identified classes of invalid input must be rejected. Functions: identified functions must be exercised. Output: identified classes of application outputs must be exercised. Systems / Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identifying business process flows, data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Performance Test

The Performance test ensures that the output be produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot see into it. The test provides inputs and responds to outputs without considering how the software works.

6.2 Test Environment

The testing of the modules was done on machine with the following features.

· **Operating System** : Windows 7

· **JDK Version** : 1.7

· **Browser** : Google Chrome

· **Web server** : Tomcat Version 7

Sl. Number of Test Case	1
Name Of the test	Connection between Tomcat and JDK.
Feature Being Tested	Tomcat Instance Working
Input	Tomcat Catalina Home batch file
Expected Output	Home Page Must Open
Output	Home Page Opens
Remarks	Test case passed

Table 6.1: Test Case 1 : System Test

Sl. Number of Test Case	2
Name Of the test	Working of JQuery and Ajax Libraries
Feature Being Tested	Presence of JQuery Libraries and Ajax Libraries
Input	Nil
Expected Output	Home Page Must Open with Jquery and Ajax Functionality
Output	Home Page Opens as expected
Remarks	Test case passed

Table 6.2: Test Case 2 : System Test

6.3 System Testing

Test Case 1

6.4 Integration Testing

Sl. Number of Test Case	3
Name Of the test	Working of Google Charts API
Feature Being Tested	Charts
Input	Data Files
Expected Output	Scatter and Line Plots must appear
Output	Charts are plotted
Remarks	Test case passed

Table 6.3: Test Case 3 : System Test

Sl. Number of Test Case	4
Name Of the test	Network Testing
Feature Being Tested	Checking Network Connectivity
Input	IP address of any DNS server
Expected Output	Ping Data
Output	Ping Data Obtained
Remarks	Test case passed

Table 6.4: Test Case 4 : Integration Test

Chapter 7

Experimental Analysis and Results

Satvik will do this

7.1 Evaluation Metric

As explained earlier, major module present in the project are as follows :

- **Similarity Detection Module**
- **Prediction and Forecasting Module**
- **Anomaly Detection Module**
- **Temporal Pattern Finder Module**

Since these modules are independent of each other, they have different evaluation metrices. The evaluation metrics used for performance analysis of each module is explained in the following sub sections.

7.1.1 Metrics for Similarity Detection Module

The algorithms implemented under this module are :

- **Dynamic Time Wrapping (DTW) Algorithm**
- **SAX Algorithms**

The evaluation metrics are :

- **Euclidean Distance** In mathematics, the Euclidean distance or Euclidean metric is the "ordinary" distance between two points that one would measure with a ruler, and is given by the Pythagorean formula. By using this formula as distance, Euclidean space (or even any inner product space) becomes a metric space. The associated norm is called the Euclidean norm. Older literature refers to the metric as Pythagorean metric.

  In Cartesian coordinates, if p = $(p_1, p_2, ..., p_n)$ and q = $(q_1, q_2, ..., q_n)$ are two points in Euclidean n-space, then the distance from p to q, or from q to p is given by:

$$d(p,q) = d(q,p) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + ... + (p_n - q_n)^2}$$

$$d(p,q) = d(q,p) = \sqrt{\sum_{i=0}^{n} (p_i - q_i)^2}$$

  In our project, p and q can be visualized as two time series to be compared, where p_iś and q_iś are the time series values. Depending on the distance, the similarity of two or more time series with a give base series is found out. This metric is used in the DTW approach.

- **String Comparison** In SAX Algorithm, the time series is converted into a string of character as explained in section x.x. Given two or more time series, which are represented by strings, a string comparison algorithm is run and the similarity is found out. There are various string comparison algorithms. In this project KMP algorithm has been used.

7.1.2 Metrics for Prediction and Forecasting Module

The algorithms implemented under this module for modeling and forecasting time series are :

- **NARX-Neural Network**
- **Moving Average Forecaster**

- **Moving Geometric Average Forecaster**
- **Moving Exponential Average Forecaster**

Modeling a time series is an regression problem, the evaluation metrics are :

- **Root-Mean-Square Deviation** - The root-mean-square deviation (RMSD) or root-mean-square error (RMSE) is a frequently used measure of the differences between values predicted by a model or an estimator and the values actually observed. These individual differences are called residuals when the calculations are performed over the data sample that was used for estimation, and are called prediction errors when computed out-of-sample. The RMSD serves to aggregate the magnitudes of the errors in predictions for various times into a single measure of predictive power. RMSD is a good measure of accuracy, but only to compare forecasting errors of different models for a particular variable and not between variables, as it is scale-dependent.

  The RMSD of predicted values y_p for times t of a regression's dependent variable y is computed for n different predictions as the square root of the mean of the squares of the deviations:

  $$RMSD = \sqrt{\sum_{i=0}^{n} (y_p - y)^2 / n}$$

  The accuracies of different algorithms are compared and presented in the next section.

7.1.3 Metrics for Anomaly Detection Module

The anomaly detection algorithms require the controller/user to specify various parameters which determine the anomalous points in the time series.

Algorithms implemented under this module are :

- **Cumulative Sum Approach (CUSUM)**
- **Statistical Approach**

These algorithms require a **threshold value** to be specified by the user and depending on this value, anomalous data points are determined.

7.1.4 Temporal Pattern Finder Module

In this module, a Genetic Algorithm has been implemented to optimize the algorithm. @TODO Add Fitness Function here The fitness function used in the GA determine the accuracies of the patterns found. But eventually user intervention is required to interpret the resulting patterns detected by the algorithm. The results are documented in the next section.

7.2 Experimental Dataset

The data sets considered in this project are

- Sea Level Dataset : Indicating the sea level at various times of a day.
- Water Level : Ground Water level data, indicating the ground water level during various moths of an year for upto 5 years.
- Finance Dataset : Consisting of stock index values of Nifty and Vix collected every minuted for a week.(5 days,during market hours).
- ECG Dataset : The ECG voltage values of patients collected every 4ms.(for 10 patients).

All the experiment analysis and results are presented using the **Water Level** data set. As explained earlier, some algorithms require certain parameters which determine their accuracies. (Add some more stuff)

7.3 Performance Analysis

In the previous section, the evaluations metrics for different modules depending on the algorithm were explained. Performance analysis of different modules implemented in this

project are explained in the following sub sections

7.3.1 Similarity Detection

DTW Algorithm

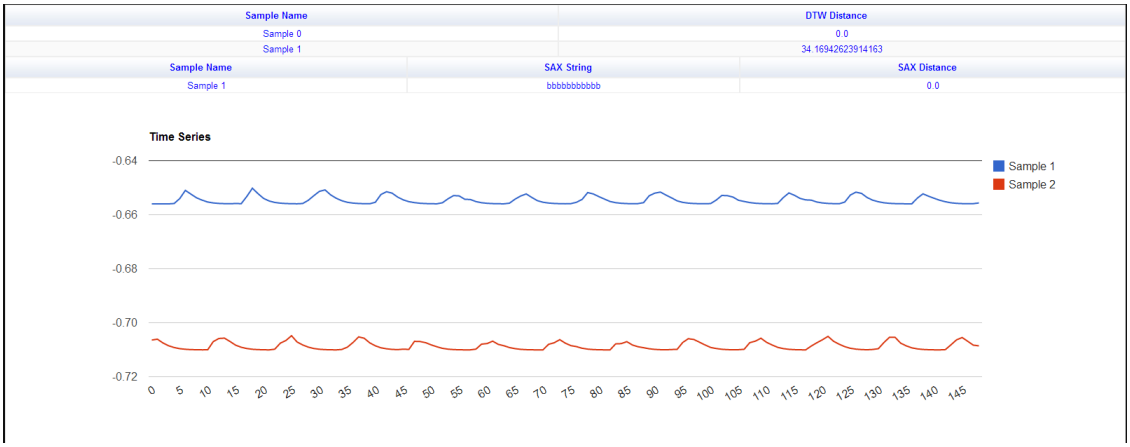

Figure 7.1: Similarity detection for two Samples using DTW Algorithm

DTW Algorithm

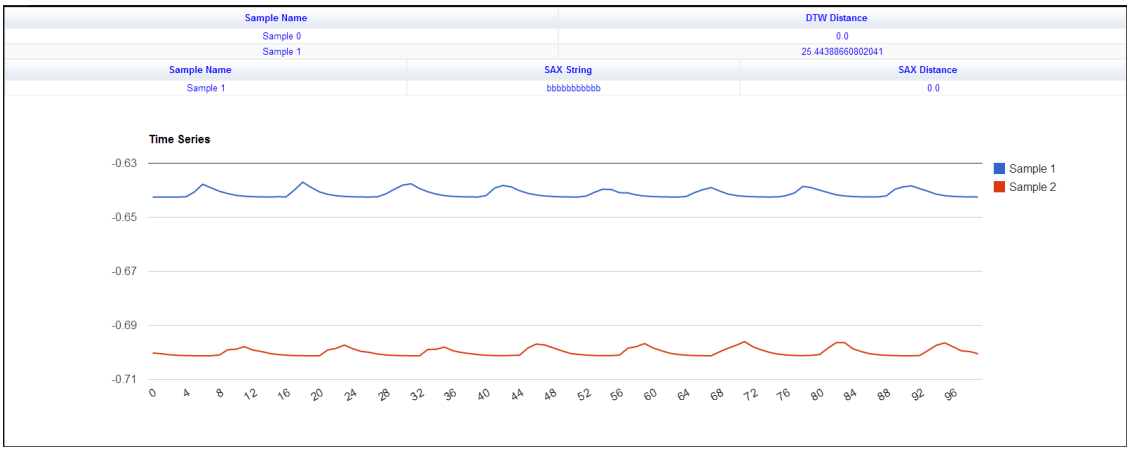

Figure 7.2: Similarity detection for two Samples using SAX Algorithm

7.3.2 Anomaly Detection

Cumulative Sum

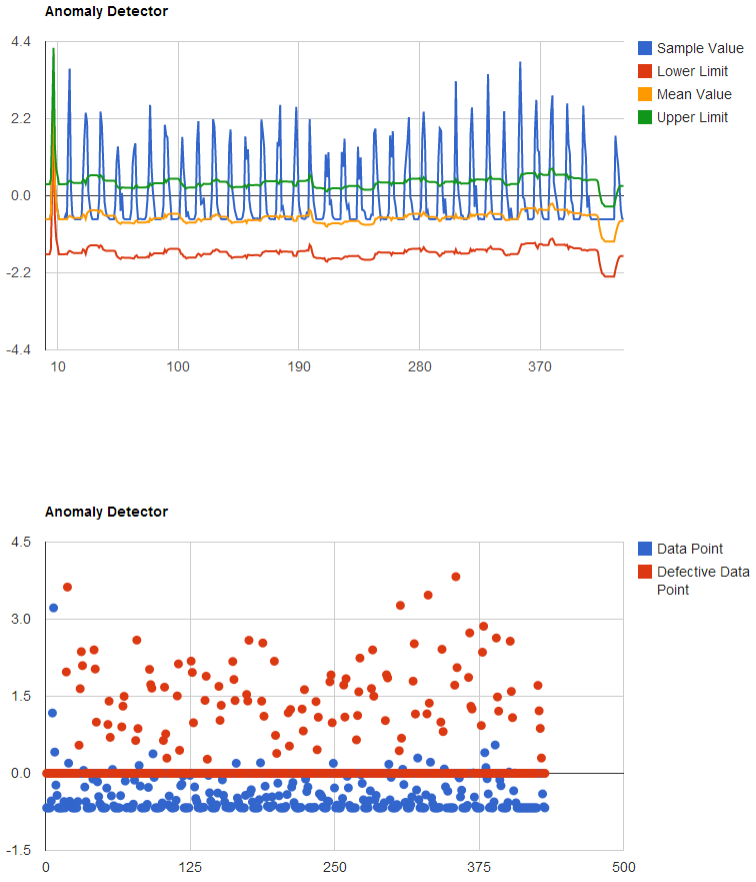

Figure 7.3: Anomaly detection using the Statistical Approach

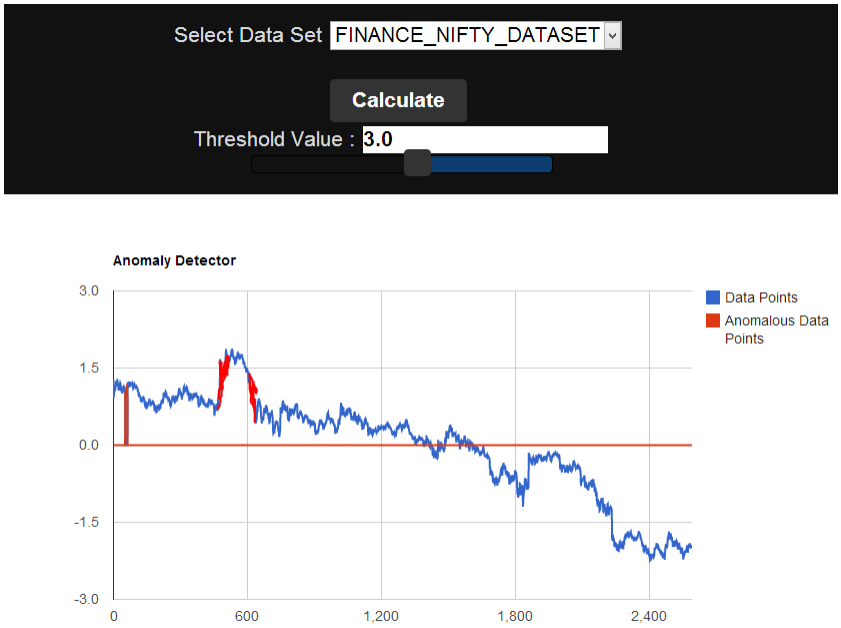

Figure 7.4: Anomaly detection using the Cumulative Sum Approach

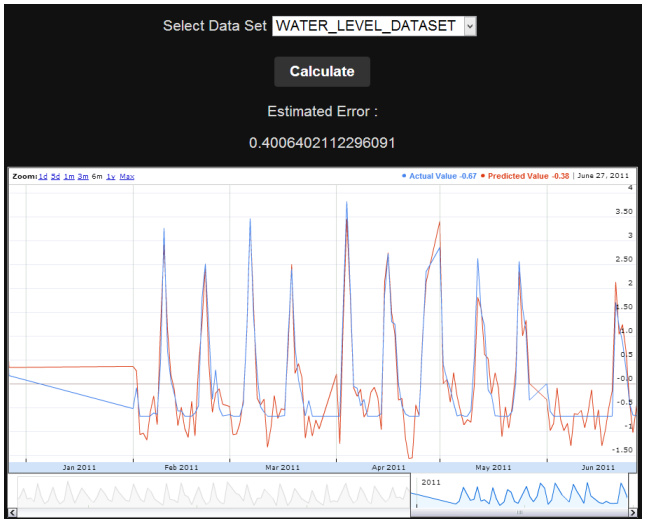

Figure 7.5: Forecasting/Estimation using NARX Neural Network Approach.

SAX Algorithm

7.3.3 Prediction and Forecasting

NARX Neural Network Approach

Moving Exponential Average Method

Simple Moving Average Method

7.3.4 Temporal Pattern Detection

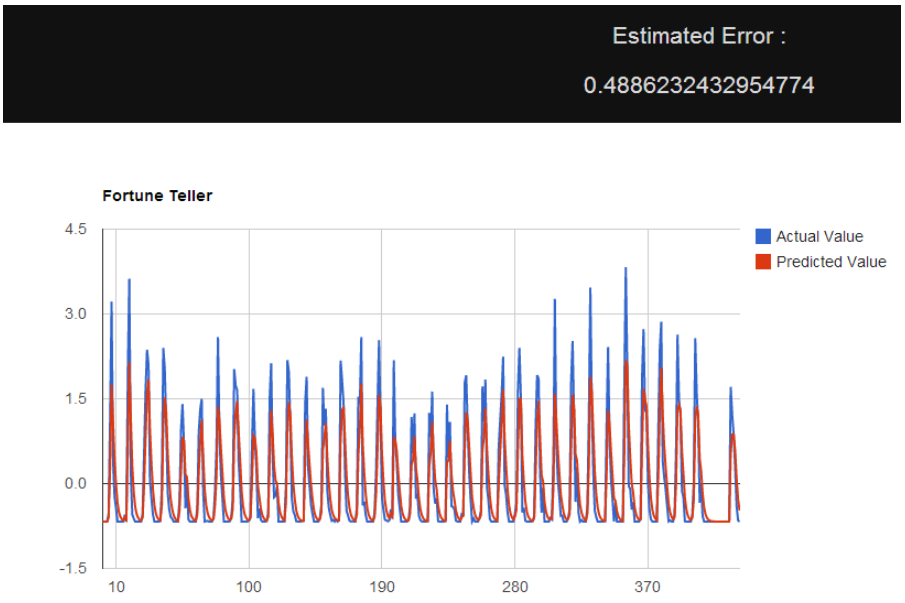

Figure 7.6: Forecasting/Estimation using Moving Exponential Average Method.

Figure 7.7: Forecasting/Estimation with Simple Moving Average Method.

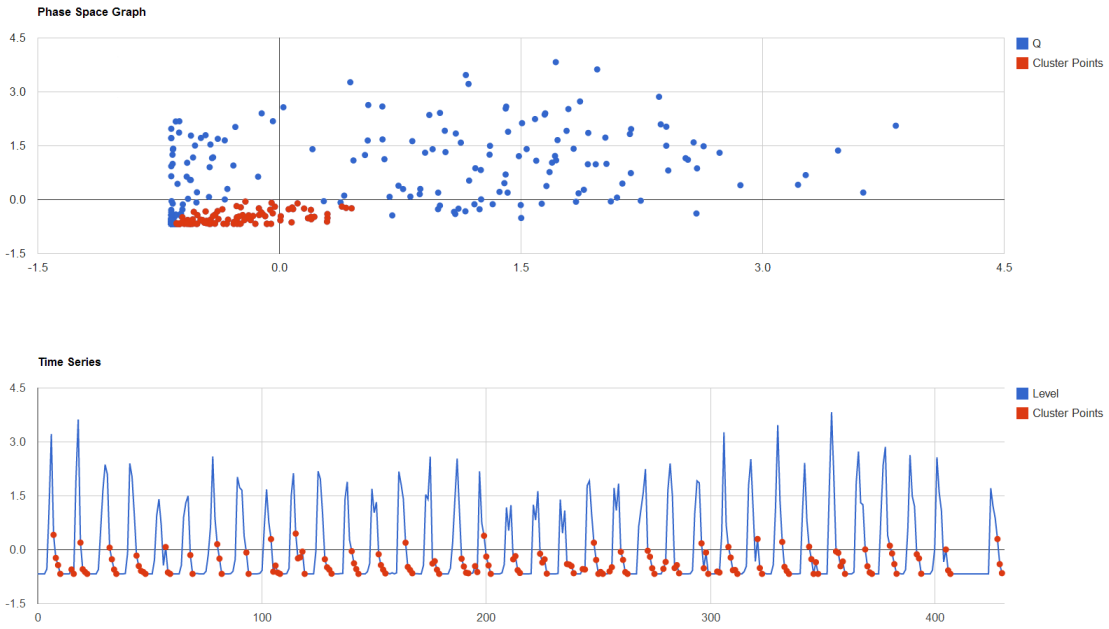

Figure 7.8: Temporal Patterns detected in the Rainfall/Water Data set.

- NARX Neural Network : The accuracy of this algorithms considering various evaluation metrics specified in the previous section is around 60%. This accuracy can be different for different data sets and depending on the specified parameters like learning rate, hidden layer size, the activation function used.
- Moving Exponential Method : This algorithm does not require any special paramters to be specified. The efficiency of this algorithm is around 40% for water data set.
- Simple Moving Average and Geometrical Moving Average methods are naive and are only estimation techniques and have few low accuracies.

2. **Anomaly Detection**

- Statistical Approach
- CUSUM Approach

These anomaly detection algorithms work in specific ways and help in determining the anomalous data points. Depending on application, appropriate algorithm is to

be used.

3. **Similarity Detection**

   - DTW Algorithm : This algorithms helps in comparing two or more time series considering their euclidian distances and exhaustive comparing. This algorithm is computationally expensive but consumes less memory.
   - SAX Algorithm : This algorithm converts the time series to a string and hence consumes more memory than the DTW algorithm. Computationally this algorithm is less expensive.

4. **Temporal Pattern Finder** This module helped in determining interesting/temporal patterns in the time series data. The patterns were highlighted in the graphs. It is upto the user to interpret the patterns/ results shown in the graph.

Chapter 8

CONCLUSION

8.1 Summary

In this project, we were able to successfully implement the Time Series Data Mining Tool for analyzing the time series and test its performance. This tool mainly contains four modules, they are - **Similarity Detection, Forecasting and Prediction, Anomaly Detection and Temporal Pattern Finder**, which we were successful in implementing and testing. The results obtained were presented in the previous chapter.

The description of the modules are below :

- **Similarity Detection** : This module helps in finding similarity patterns (that occur at regular intervals in case of periodic time series), comparing different time series data. SAX and DTW are the main algorithms implemented/used in this module.

- **Forecasting and Prediction** : This module contain algorithms/models which can be trained from the past time series data and can be used to predict the future values of a time series.

- **Anomaly Detection** : This module contains algorithms that help in indicating anomalous patterns in the time series data analyzed. Anomalies are patterns in time series which deviate from the normal behavior and can indicate fraud/danger depending on the application. For example in an industry which produces the blades,

the thickness of the blade can be monitored by a machine as a time series and any deviation from the normal error rate can signal an error in the manufacturing process.

- **Temporal Pattern Finder** : This module helps in finding hidden temporal patterns in a time series. This module can be further extended to implement clustering techniques.

Initially this project mainly focused on analyzing the sea and water level time series. Later this application was extended to any uni-variate time series data. Users can upload the time series data to be analyzed and get the results instantly. Major data sets used were :

- Sea Level Dataset : Indicating the sea level at various times of a day.
- Water Level : Ground Water level data, indicating the ground water level during various moths of an year for upto 5 years.
- Finance Dataset : Consisting of stock index values of Nifty and Vix collected every minuted for a week.(5 days,during market hours).
- ECG Dataset : The ECG voltage values of patients collected every 4ms.(for 10 patients).

In this project, we also analyzed the efficiencies of different algorithms for the same tasks and also compared the results for different data sets.Clearly more work needs to be done.(End this section Properly Boys)

8.2 Limitations

Some of the shortcomings in our project are :

1. The application hangs when analyzing very large data sets (more than 500 MB).
2. The application does not support multi-variate time series.

3. Some algorithms efficient for a particular data set and may not be efficient for other data set. So user intervention is required in selecting an algorithm for a data set.

8.3 Future enhancements

Some of the future enhancements are :

1. The size of the time series data analyzed is in terms of Mega Bytes. For larger dataset(In terms of GBs) or big data, distributed computing technologies like Hadoop can be used.

2. The application can be extended to analyze multi variate time series data.

3. The application could be made more responsive by using Threads and Parallel/-Cloud Computing

4. One more extension could be analyzing twitter post data with respect to time and predicting the trends. This requires NLP, but is an example of time series.

5. Efficient algorithms using Support Vector Models (SVMs) for forecasting, Hidden Markov Model for anomaly detection can be implemented.

6. This application uses static time series data, enhancements can be made to use real time data.(In finance applications)

7. This application can be converted into an mobile application (android, iPhone, iPad) where the users can analyze the time series data on the go and share the results on facebook

Bibliography

[1] Agrawal R., Lin K.-I., Sawhney H. S., Shim K.,*Fast Similarity Search in the Presence of Noise,Scaling, and Translation in Time-Series Databases*

[2] Genetic Algorithm for constructing DT - `http://www.jprr.org/index.php/jprr/article/viewFile/44/25`

[3] Decision Trees - `http://web.cecs.pdx.edu/~mm/MachineLearningWinter2010/pdfslides/DecisionTrees.pdf`

[4] Project brief for the DT using Horse data sets - `https://cs.uwaterloo.ca/~ppoupart/teaching/cs486-spring06/assignments/asst4/asst4.pdf`

[5] Supervised and Unsupervised Discretization of Continous Features - `http://robotics.stanford.edu/users/sahami/papers-dir/disc.pdf`

[6] Naming Convention -*Wikipedia* : `http://en.wikipedia.org/wiki/Naming_convention_(programming)`

[7] Code Conventions for the Java Programming Language - `http://www.oracle.com/technetwork/java/codeconv-138413.html`

Appendices

Appendix A : Screen Shots

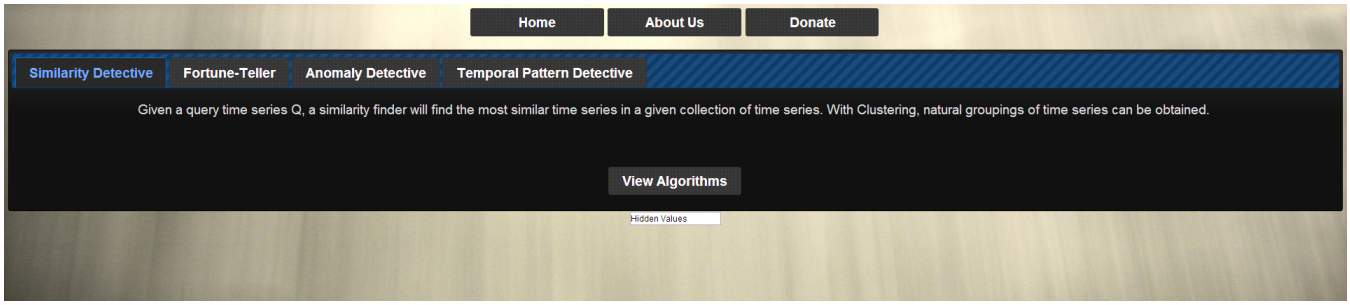

Figure 8.1: TSDM Tool Window - Welcome Screen

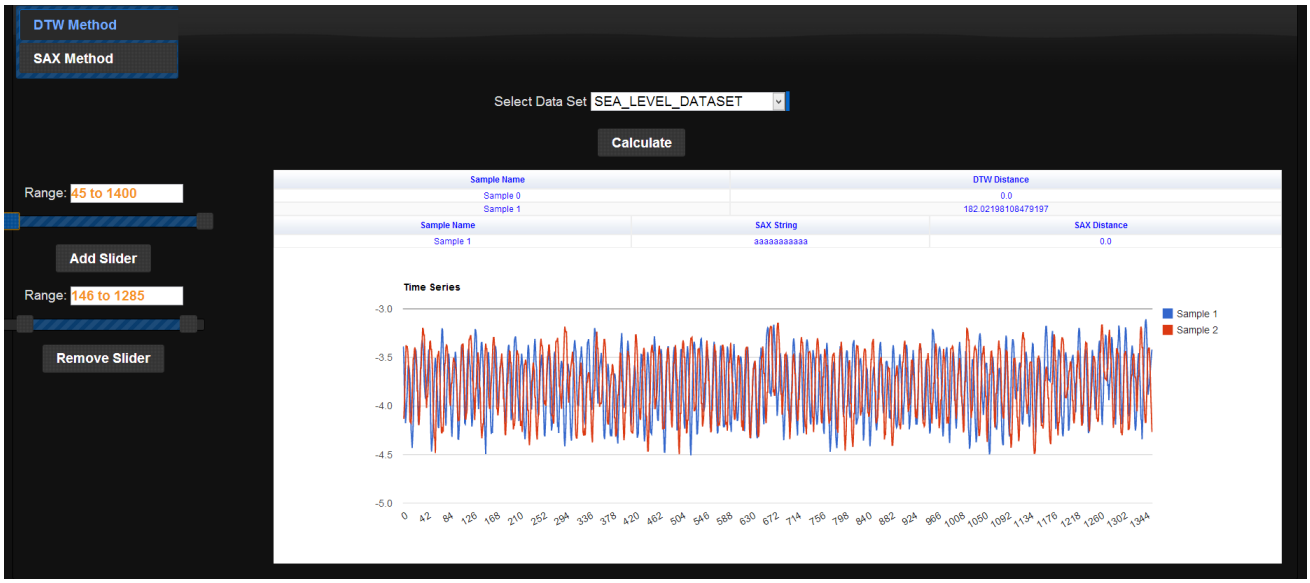

Figure 8.2: Similarity Detection with DTW Algorithm.

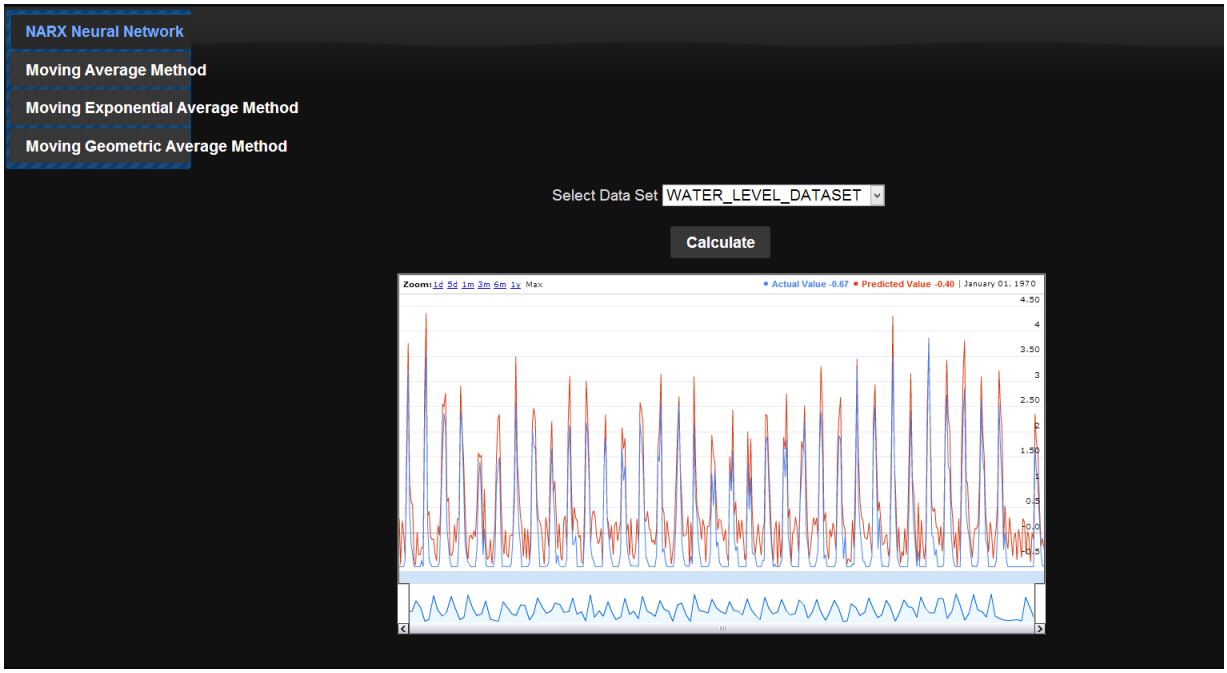

Figure 8.3: Time Series Prediction with NARX NN Model.

Add Paper to be published in IEEE.

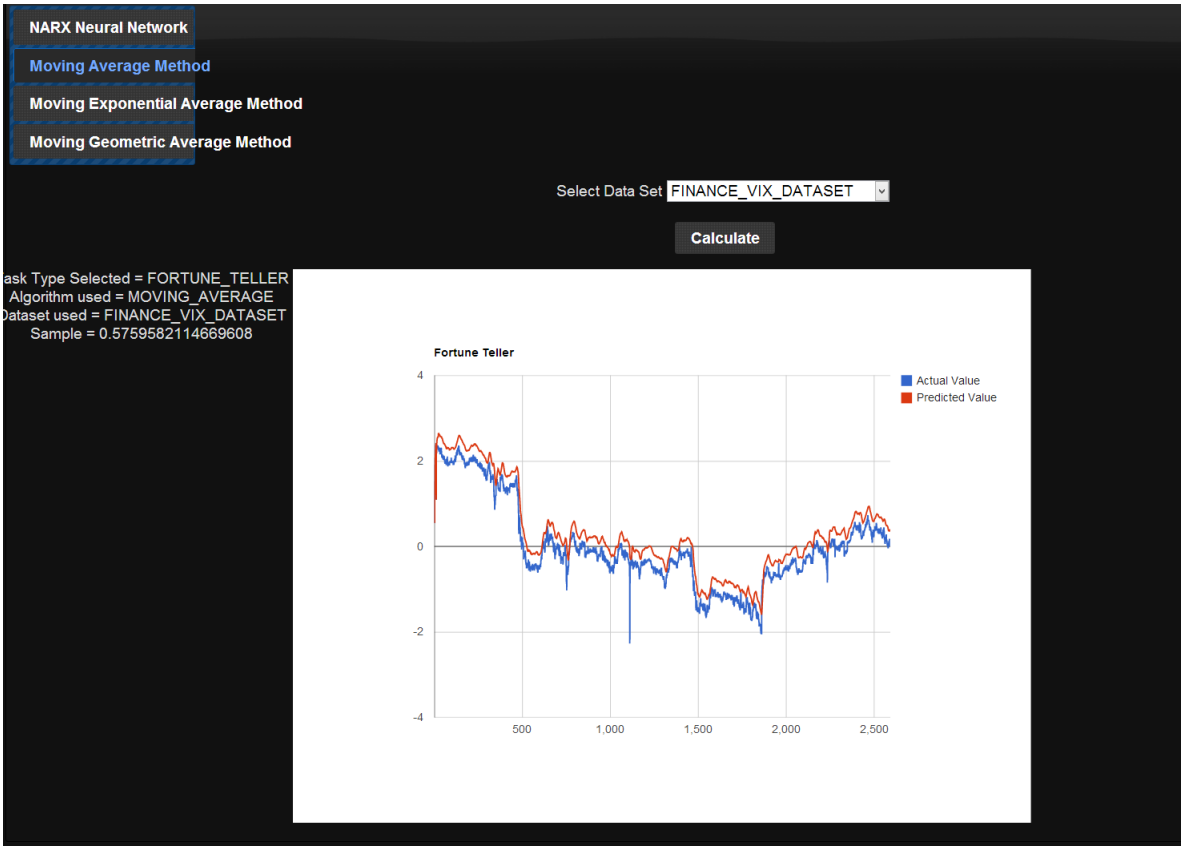

Figure 8.4: Time Series Prediction with Moving Average Method.

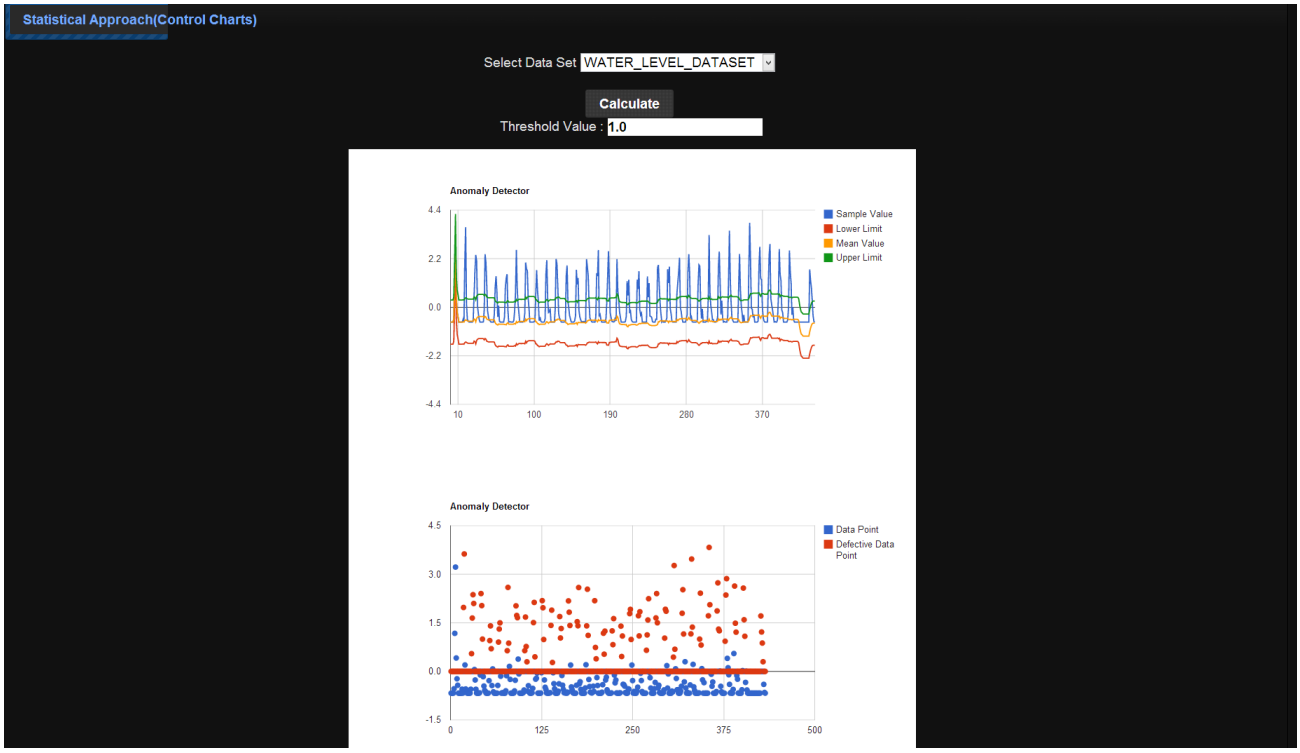

Figure 8.5: Anomaly Detection using Statistical Approach.

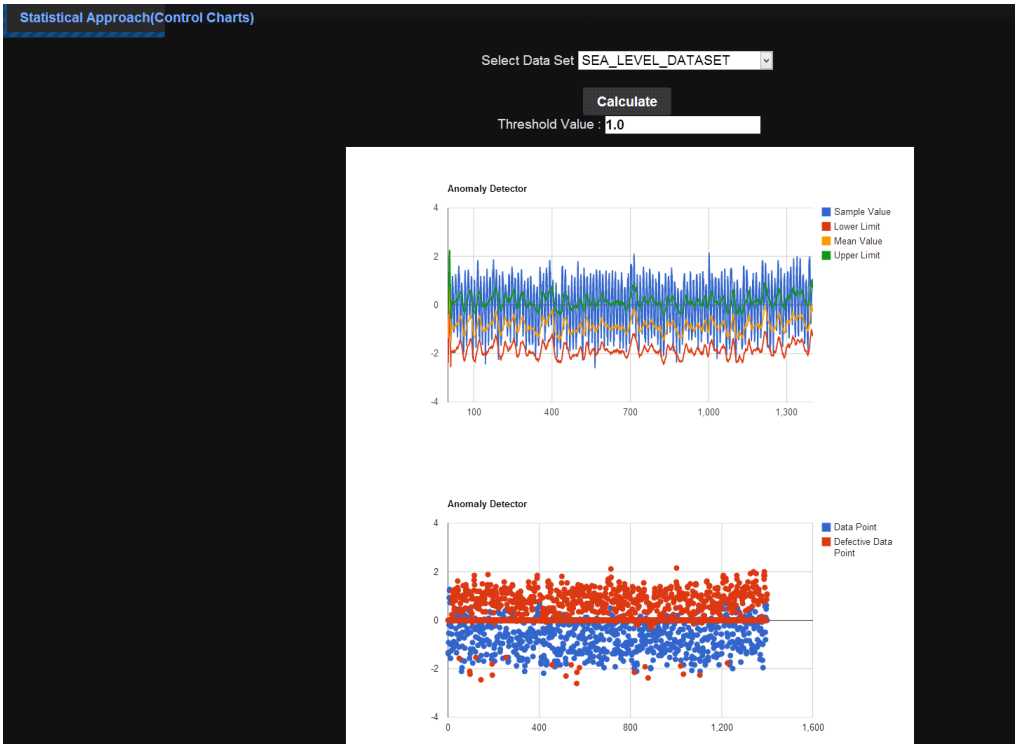

Figure 8.6: Anomaly Detection using Cusum Algorithm.

Figure 8.7: Anomaly Detection using Cusum Algorithm.

Figure 8.8: About the Developers who did this project!