Design of Algorithms for Gene Predictions

A Thesis

Submitted for the award of degree of

Doctor of Philosophy

By

Srabanti Maji
(Reg. no. 90703505)

Department of Computer Science and Engineering

Thapar University

Patiala 147 004, Punjab, INDIA

URL: www.thapar.edu

March 2013

CERTIFICATE

This is to certify that the thesis entitled **"Design of Algorithm for Gene Prediction"** being submitted by **Mrs. Srabanti Maji** to the **Department of Computer Engineering, Thapar University**, Patiala for the award of degree of **Doctor of Philosophy**, is a record of bonafide research work carried out by her. Mrs. Srabanti Maji has worked under my guidance and supervision and has fulfilled the requirements for the submission of this thesis, which to my knowledge has reached the requisite standard.

The results embodied in the thesis have not been submitted in part or full to any other University or Institute for award of any degree or diploma.

I wish her success for her future endeavors.

Deepak Garg
28/03/13

(Deepak Garg)
Associate Professor
Department of Computer Science & Engineering
Thapar University, Patiala

ACKNOWLEDGEMENTS

I am highly grateful to the authorities of Thapar University, Patiala for providing this opportunity to carry out the present research work. I am highly thankful to my supervisor Dr. Deepak Garg, Associate Professor, Department of Computer Science & Engineering, Thapar University, Patiala who gave me opportunity to associate with him. As an outstanding teacher, he has given me the benefit of his guidance throughout the research work.

I am also thankful to Dr. Pramod Kumar Bajpai, Distinguished Professor and Dean, Research and Sponsored Projects, Thapar University, Patiala for his valuable comments and suggestions during research work.

Heartiest thanks to Dr. K. K. Raina, Director, Thapar University, Patiala, Dr. Maninder Singh, Associate Professor and Head, Department of Computer Science & Engineering, Thapar University, Dr. N. Tejo Prakash, Associate Professor, Department of Biotechnology, Thapar University, and Dr. R. K. Sharma, Professor, School of Mathematics & Computer Applications, Thapar University for their encouragement and support during this research work.

I am also thankful to some of the fellow researchers for the valuable support extended during research work. These include Dr. Malay Bhattacharyya, Dr. Ramakrishna, at Department of Machine Intelligence, Indian Statistical Institute, Kolkota. I am obliged to my fellow colleague Mr. Gaurav Madhu, for the valuable inputs and support provided by him from time to time.

I express my gratitude to family members for their patience, sacrifice, and support throughout the research work. My firm faith in almighty has led me to face all the problems with smile and helped me to cross all the hurdles which came in my way during the journey to success. I would like to thank The God for not letting me down during the crisis and showing me the silver lining in the dark clouds.

Last but not the least my parents deserve special thanks for their love affection and blessings showered on me to undertake and successfully complete this research work leading to Ph.D. degree.

Date: March 2013

Place: Patiala

Srabanti

(Srabanti Maji)

ABSTRACT

Identification of coding sequence from genomic DNA sequence is the major step in pursuit of gene identification. In the prediction of splice site, which is the separation between exons and introns, though the sequences adjacent to the splice sites have a high conservation, but still, the accuracy is lower than 90%. Therefore, here, both approaches – Conventional as well as Computational Intelligences (CI) have been pursued to predict the splice site in DNA sequence of the Eukaryotic organism and, both have been evaluated and compared in terms of their performance.

In the conventional approach, i.e., "Hidden Markov Model (HMM) System", the model architecture includes the probabilistic descriptions of the splicing, translational, and transcriptional signals. Splice sites predictor based on Unique Hidden Markov Model (HMM) is developed and trained using Modified Expectation Maximization (MEM) algorithm. A 12 fold cross validation technique is also applied to check the reproducibility of the results obtained and to further increase the prediction accuracy. The proposed system is able to achieve the accuracy of 98% of true donor site and 93% for true acceptor site in the standard DNA (nucleotide) sequence.

The second proposed method, based on combination of conventional and computational intelligences, namely, "Markov Model 2 Feature – Support Vector Machine (MM2F-SVM)" consists of three stages – initial stage, in which a second order Markov Model (MM2) is used; intermediate, or the second stage in which principal feature analysis (PFA) is done; and the third or final stage, in which a support vector machine (SVM) with Gaussian kernel is used. The first stage is known as "feature extraction"; the second stage is called "feature selection" and, the final stage is known as "classification". The model is proficient of indicating the reliability of each predicted splice site with high accuracy. The accuracy of this method, when tested on standardized sets of human genes, is shown to be significantly better than some of the existing methods as it correctly identified maximum 98.31% of the true donor sites and 97.88% of the false donor sites in the test dataset; 97.92% of the true acceptor sites and 96.34% of the false acceptor sites in the test data set.

The applications of the program to identify splice site in newly sequenced genomic regions and to identify the alternative splice sites are also explained along with appropriate examples.

Contents

List of Symbols

Symbol	Description
X	Base in Hidden Markov Model
Y	Various states in the Hidden Markov model
T	Various transitions in the Hidden Markov model
P(Y)	Discrete state probability
P(T)	Transition probability
C_{site}	A candidate sequence
THV	Predefined threshold value
F	Flag variable
L	Length of the candidate site
MOD_t	True Acceptor HMM unit
MOD_f	False Acceptor HMM unit
N	The set of sequences that are randomly picked from the positive training data set and negative training data set.
N^t	The sequence collection containing the remaining sequences in the positive training data set, after taking some positive training dataset for M.
N^f	Represent the remaining sequences in the negative (non-coding) training data set, after picking some negative training data set for M
P	Subset of sequence collection N
L_b	The positive lower bound
S_n^{mem}	The sensitivity during the MEM training
S_p^{mem}	The specificity during the MEM training
S_{total}	S_{total} represent the total number of states in the Acceptor Model.
$T_{in}^{(t)}$	Total number of true acceptor sites that have been input into True Acceptor HMM Unit
$T_{in}^{(f)}$	Total number of false acceptor sites that have been input into False Acceptor HMM Unit
$FLAGHMM_i$	A flag indicating whether C_{site} is a true acceptor site or not.
S_n^{true}	Sensitivity or TPR of the HMM
S_n^{false}	Specificity of the HMM
ACC	Accuracy of the Hidden Markov Model
$f\ tr_i^{(t)}(b_i, b_{i+1})$	State transition probabilities in True Acceptor HMM/True Donor HMM Unit
$f\ tr_i^{(f)}(b_i, b_{i+1})$	State transition probabilities in False Acceptor HMM/False Donor HMM Unit

List of Abbreviations

Abbreviation	Description
DNA	Deoxyribonucleic Acid
RNA	Ribonucleic Acid
mRNA	messenger RNA
MM	Markov Model
MM2	Second Order Markov Model
HMM	Hidden Markov Model
EM	Expectation Maximization
MEM	Maximum Expectation Maximization
SVM	support vector machine
NN	Neural Networks
GA	Genetic Algorithms
SRM	Structural Risk Minimization
ERM	Empirical Risk Minimization
TSS	Transcriptional Start Site
TF	Transcription Factor
MM2F-SVM	Markov Model Two Feature Selection-Support Vector Machine
PFA	Principal Feature Analysis
ORFs	Open Reading Frames
PWMs	Positional Weight Matrices
WAM	Weight Array Model
MDD	Maximal Dependence Decomposition
CHMM	Class Hidden Markov Model
GHMM	Generalized Hidden Markov Model
IMM	Interpolated Markov Model
AAT	Analysis and Annotation Tool
DP	Dynamic Programming
WMM	Weight Matrix Method
GSA	Gene Structure Assembly
DAG	Directed Acyclic Graph
CI	Computational intelligence
CBR	Case-based reasoning

TRRD	Transcriptional Regulation Data
ANNs	Artificial Neural Networks
LMS	Least Mean Square
MSE	Mean Square Error
DGSF	Dragon Gene Start Finder
SA	Simulated Annealing
CC	Correlation Coefficient
EP	Evolutionary Programming
ES	Evolution Strategies
PSO	Particle Swarm Optimization
ACO	ant-colony optimization
DE	Differential Evolution
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
TPR	True Positive Rate
MEM	Modified Expectation Maximization
VEIL	Viterbi Exon-Intron Locator
E-step	Expectation step
M-step	Maximization step
ROC	Receiver Operating characteristic
PFA	Principal Feature Analysis
FSS	Feature Subset Selection
EDA	Estimation of Distribution Algorithm
WMM	Weight Matrix Models
CP	Conditional Probability
TFBS	Transcription Factor Binding Sites
PVLMM	Permuted Variable Length Markov Models
NNPP	Neural Network for Promoter Prediction
BPNN	Back Propagation Neural Networks
HS3D	Homo Sapiens Splice Sites Data set
PFS	Positional Feature Selection

PCA	Principal Component Analysis
MM0	Zero Order Markov Model
MCC	Matthews's correlation coefficient
CV	Cross Validation

List of Figures

List of Tables

Preface

The work done in this study has been presented in **six chapters**. After defining the problem and rationale for the study in **chapter 1**, background and the existing scenario of algorithms (for gene prediction) is described.

In **chapter 2**, literature survey on the gene identification is given. This chapter also contains the methods for determining the coding parts of a sequence and constructing the whole gene from its start site and stop codon. As sequence data continues to be generated at a logarithmic rate, the dependence on effective *in-silico* gene prediction methods also increased. The current state of eukaryote gene prediction methods, their strengths, weaknesses and future direction are also discussed in this chapter.

In **chapter 3**, analysis of gene prediction algorithms and tools is discussed. The existing (conventional) as well as computational methods to identify gene(s) and various gene predictors are also compared. The usefulness of automatic gene prediction has been discussed and various gene identification tools are based on computational intelligence approaches have also been explained.

Chapter 4 is about Hidden Markov Model (HMM) for splicing junction sites identification in DNA sequences. The HMM based splice sites predictor is developed and trained using Modified Expectation Maximization (MEM) algorithm. Identification of coding sequence from genomic DNA sequence is the major step in pursuit of gene identification. In the eukaryotic organism, to identify the gene structure, accurate labeling of the mentioned segments is necessary. In general, the predicted accuracy of splice site is lower than 90%; though the sequences adjacent to the splice sites have a high conservation. As the accuracy of splice site recognition has not yet been satisfactory (adequate), therefore, much attention has been paid to improve the prediction accuracy and improvement in the algorithms used. A 12 fold cross validation technique is also applied to check the reproducibility of the results obtained and to further increase the prediction accuracy. Ultimately, the proposed system was able to achieve the accuracy of 98% of true donor site and 93% for true acceptor site in the standard DNA (nucleotide) sequence.

In **chapter 5**, Hybrid approach using Support Vector Machine (SVM) and Markov Model 2 (MM2) in the identification of splice site identification is explained and Second Order Markov Model Feature - Support Vector Machine (MM2F-SVM) based splice sites predictor is discussed. This proposed method consists of three stages – feature extraction, feature selection, and the final classification, in which a support vector machine (SVM) with Gaussian kernel is used. Superior performance has been noticed for the proposed model as compared to the other similar splice site prediction programs existing already. For validation of the results, 12-fold cross validation

experiment is applied for this model also. The ability of this MM2F-SVM system to correctly identify 98.31% (max.) of the true donor sites and 97.88% of the false donor sites in the test dataset; 97.92% of the true acceptor sites and 96.34% of the false acceptor sites in the test data set has been reported.

In **chapter 6**, the conclusions of the entire work are drawn. This chapter also includes the recommendations for further research work.

At the end, all the references cited in the thesis have been listed.

CHAPTER – 1

INTRODUCTION

Introduction

It is generally said that we are living in an 'ics' era e.g. informatics, proteomics, genomics etc. Due to the explosion of the vast amount of data in almost all fields of sciences, it has become need of the hour that this data should be handled in a more meaningful manner (informatics). Out of this need, several sub branches of the basic sciences have emerged in the recent past as a special field, for e.g. chemo informatics, bioinformatics, medical informatics, etc.

The problem of interpreting nucleotide sequences by computer, in order to provide uncertain annotation on the location, structure, and functional class of protein-coding genes, is the basic gene identification problem [1]. The identification of protein coding genes is noticeably influenced by the knowledge of other significant features of the sequence. The gene identification is usually considered to be autonomous as compared to most of the other sequence analysis because of the complexity in considering the automatic annotation problem as a logically integrated process.

Eukaryotic gene regulation is a complex process. It still seems a difficult aim to predict from DNA sequence the path of the key biochemical reactions of gene expression: transcription, splicing and translation. Presently, the success of gene identification algorithms is measured in terms of the degree to which they correctly predict the amino acid sequence of protein products and, some hint of product function like sensitivity, specificity and accuracy; making a transition from studying primarily components of genes to studying genes and genomes in their entirety. Therefore, the issue of selecting an appropriate language in which to convey and incorporate the knowledge gained from the component calculations is one of the most dynamic areas in computational gene identification.

A genomic sequence is a string composed of four different nucleotides, A, T, G and C, which codifies in group of three, called codons that are amino acids that form the proteins and are necessary for all organisms to live. A very large number of computational solutions for the gene identification problem have been reported which are the valuable resources for the human genome program and for the molecular biology community. A gene is a structure that codifies the proteins [2, 3]. In prokaryotes, it is a sequence of codons between a start codon (ATG) and a stop codon (TAA, TAG or TGA) whereas in eukaryotes, the structure is more complex. The coding sequence is usually broken by non-coding sequences, called introns that are removed during the transcription in a process called splicing [4]. The coding sections are called exons. In this manner, the eukaryotic gene begins with first exon, then any number of intron/exon pairs, and ends with a last exon which finishes with a stop codon. This is called an open reading frame (ORF). The eukaryotic genes are composed by a single exon. The boundary between an exon and an intron is called a splice donor

site and that between an intron and an exon, a splice acceptor site. The actual gene has the sequences of nucleotides before start codon and after stop codon, known as the untranslated terminal regions (UTRs). However, it is not uncommon in gene recognition to use the term “gene” when referring only to the coding part of it, since that part only determines the protein structure [5].

Gene recognition, gene structure prediction or gene finding, all of these three terms consist of determining those parts of a sequence which are coding and constructing the whole gene from its start site to its stop codon. Here, we are concerned with the work related to eukaryotic gene recognition, as it is significant, useful and complex as well. There are two basic approaches to predict the gene structure [6]; first one is homology based approaches that search for similar sequences in databases of known genes and are usually called extrinsic methods. The growing number of sequenced genomes and known genes is increasing the potential of homology based methods. However, it is clear that only genes that are somewhat similar to known genes can be identified in this way. Furthermore, when using homology based techniques, it is very difficult to establish the complete structure of the gene, as the exact bounds of the exons are not easy to determine with certainty. The second approach, usually known as intrinsic approach includes two basic methods: ab-initio and de novo [7]. Both are based on obtaining the features that characterize a coding region and/or the functional sites, and using them to find the correct structure of the unknown genes. Ab-initio methods use only the information of the genome to be annotated (the target genome), whereas de novo methods add information of one or more related genomes (the informant genomes). An essential characteristic for gene finding in genome sequencing projects is the occurrence of splice sites in the gene sequences. In sequencing of known structural elements, the signals observed are explored by the latest available computational techniques. The key aspect in the systematic study of eukaryotic genes is the accurate prediction of splice sites, which is the partition between exons and introns and further depends largely on exactly locating the splice sites. Genome annotation is a necessity and a multi-step process in itself. The steps involved in genome annotation can be grouped into three categories: nucleotide-level (gene prediction or identification), protein-level (structure determination of proteins), and process-level annotation (mechanism of biochemical reactions). Among these three categories, nucleotide- level annotation is the most significant, because it primarily deals with gene annotation, a fundamental step in molecular biology [8]. Therefore, partitioning them into promoters, genes, intergenic region, regulatory elements, etc. for interpreting long unidentified genomic sequence are required to be modified from the conventional techniques became essential [9].

The basic structural and resourceful unit of all living organisms is called the cell, which may be classified into two types – eukaryotic and prokaryotic. The prokaryote cell is simpler and smaller

than a eukaryote cell. The nucleus is present in eukaryote, not in prokaryote. The gene structure in eukaryotic organism consists of promoter, intron, start codon, exons and stop codon.The promoter, intron, start codon, exons and stop codon are present in the gene structure of eukaryotic organism. Identification of the coding region is done by the presence of exon and in case of non-coding region; it is done by the presence of intron. The size of intron sequences are in the range of 80-10000 nucleotides or more. In protein synthesis, introns are removed from the sequence during the process of transcription and translation. In all known intron sequences, the consensus sequences at both ends of an intron are almost the same. From DNA, pre-mRNA is produced through transcription process that contain all the necessary information of the gene sequence in which protein is encoded, but only before it is fully converted (or processed) into mRNA.

Dissimilar modification of a protein can arise when single exon is bounced or if only one out of two splice sites is used from an exon. This is called alternative splicing [10]. Essential mechanism for splice site selection in alternative splicing is the changeability in signal strength [11].

1.1 Genome Analysis

Genomics is a science that tries to describe a living organism in terms of the sequence of its genome. Genomics leads to certain developments that provide the facility to generate time specific gene expression data [12]. Genome analysis entails the prediction of genes in uncharacterized genomic sequences. However, the pace of genome annotation is not matching the pace of genome sequencing. Experimental genome annotation is slow and time consuming. Therefore, the objective is to be able to take a newly sequenced uncharacterized genome and break it up into introns, exons, repetitive DNA sequences, transposons etc. and other elements.

The various components of Genome Analysis are:

- Gene Evaluation: Given a DNA sequence, gene evaluation is done to estimate the part of it which codes for a protein and the portion of it which is junk DNA.
- Genome Classification: Classification of the junk DNA as intron, untranslated region, transposons, dead genes, regulatory elements etc.
- Gene Prediction: Prediction of the coding regions in a newly sequenced genome into the genes (coding) and the non-coding regions.

1.1.1 Importance of Genome Analysis

Several genetic disorders like Huntington's disease**,** Parkinson's disease, sickle cell anemia etc. are caused due to mutations in the genes or a set of genes inherited from one generation to another. There is a need to understand the cause for such disorders. An understanding of the genome

organization may lead to concomitant progresses in drug-target identification. If the genome for humans and a pathogen, a virus causing harm is identified, comparative genomics can predict possible drug-targets for the invader without causing side effects to humans.

Genome Analysis is important in SNP (Single nucleotide polymorphisms) discovery and analysis. SNPs are common DNA sequence variations that occur when a single nucleotide in the genome sequence is changed. SNP occurs every 100 to 300 bases along the human genome. The SNP variants promise to significantly advance our ability to understand and treat human diseases. Mice and humans contain approximately the same number of genes – about 28,000 protein coding regions. The chimp and human genomes vary by an average of just 2% i.e. just about 160 enzymes. The genome projects will have additional benefits that at present can only be guessed at. For e.g. we think that most of the intergenic DNA has no function, but perhaps this is because we do not know enough about it. There is one final reason for genome projects. The work stretches current technology to its limits. Genome analysis therefore represents the frontier of molecular biology, territory that was inaccessible just a few years ago.

In this research work, various conventional approaches of gene identification viz. HMM, Bayesian Networks and Dynamic programming are explained along with the review of some of the computational intelligence techniques. The recent developments in gene identification tools, especially those based on computational intelligence techniques like Neural Networks and Genetic Algorithms have been highlighted.

1.2 Lacunae in Existing Methods of Gene Identification

The primary product of the Human genome Project is the nucleotide genomic sequence, but still a major short-term importance will be the amino acid sequences of the proteins encoded in the genome. Although, computational gene identification in eukaryotic organism continues to be an active field of research, but still contains certain drawbacks. These are described as follows:

There are problems in prediction of protein coding regions and functional sites of the gene. These programs promise highly accurate prediction, but at the cost of greater computational time. In addition, their accuracy has not been properly quantified on testing dataset. The effect of the degree of similarity between the candidate homology and the genomic sequence also deserve careful evaluation. Due to unavailability of sufficient genomic sequence, there is lack in the reliability of the predictions obtained by such gene identification programs. That concerns both developers of the program as well as user of that tool. Sometimes selection of the sequence based on some specific criteria is not properly mentioned. Division between testing and training data sets is unclear; to evaluate the real value of the different programs, their strengths, their weaknesses, and the

particular problem for which a given program may be particularly suited is difficult for the user. In addition, there is a lack of annotated large genomic sequences. Although many prediction tools are available for genomic sequence analysis and annotation, there has been a little effort to collectively evaluate them till date.

1.3 Objectives

The basic goal of this work is to develop a new generation technique of splice site identification, which would be proficient of predicting the number of splice sites is a DNA sequence by using statistical and/or machine learning procedure. Presently, the approaches like GENSCAN, GENIO, GeneId3, HMMgene, VEIL, NNSplice, GENIO, NetGene2 etc. are well accepted for the purpose.

The main objectives of this research work are as under:

1. To study the existing algorithms and look for further improvements
2. To design new approaches for accurate prediction of genes in a DNA sequence
3. To implement and validate new algorithms for gene predictions**.**

In order to improve in the existing algorithm, complete analysis of existing algorithms and the tools used in the process of splice site prediction is performed taking into account, their accuracy measures which are executed here with utmost care. In creating the design of the novel approach, the basic model architecture was observed minutely and then, design of algorithm to generate and create specific program is modified for the desirable model. Further, to performed validation check, newly developed model were trained and tested with reliable dataset collected from existing sequence data bank. Their performance is compared with existing splice site predictor. In addition to the obvious goal of improving predictive accuracy, multiple additional model properties are considered as mentioned in the following chapters.

CHAPTER – 2

LITERATURE REVIEW

After achieving the objectives of Human Genome project, researchers mainly focused their attention on the vast amount of the data that are available and started exploring this data to solve many common problems related to a better understanding of how genes, proteins behave in environments (for e.g. healthy against diseased environment). In the past few years, an elevated increase in the genomic primary sequence data for a broad range of organisms has been noticed [13]. The translation of data into knowledge is the key for future biological research and a great challenge as well. Watson and Crick in 1953 discovered the double-helical structure of DNA, and within short period, researchers achieved a detailed understanding of the molecular methodology involved in gene replication and expression. In 1970s, direct access to the sequence of gene became possible through the invention of DNA sequencing and cloning [11].

A huge number of publications have been done over the last 25 years describing several methods for recognizing protein coding genes in DNA sequence, and new, more comprehensive algorithms, based on the repertoire of existing techniques, continue to be developed. Amongst all the gene recognition algorithms, the core is one or more *coding measures* — functions which produce, given any sample sequence, a number or vector intended to measure the degree of resemblance of a 'typical' exonic DNA.

Bioinformatics essentially grew out of this requirement of managing and extracting a huge amount of information which can later be used in solving many common problems, especially related to drug design [14-17]. Three research communities mainly Biologists, Mathematicians and Computer Scientists joined their hands to solve these interesting problems. In the field of bioinformatics, gene identification from large DNA sequence is known to be a significant setback. However, the human genome project was completed in April 2003, in which main focus was given on the identification of gene in eukaryotic genomes, but the accurate number of genes encoded by the human genome are still unknown [12, 18]. Computational Gene prediction is relatively simple for the prokaryotes where all the genes are converted into the corresponding mRNA and then into proteins. The process is more complex for eukaryotic cells where the coding DNA sequence is interrupted by random sequences called introns. The mathematical approach in the segment of molecular biology and genomics is gaining a lot of attention and is an interesting research area for many scientists [18-20].

The methods for gene-finding which are being used nowadays are more precise and reliable than the earlier tactics. The advances in gene finding through dynamic programming, decision trees and Hidden Markov Model (HMM), are also studied [21]. The available gene prediction programs and methods are also reported and summarized [7, 22, 23]. The existing methods for gene prediction are also studied and compared [23, 24]. A comprehensive review of prediction methods for functional sites, protein coding genes, tRNA etc. is also reported [25]. A summary of a few techniques based on computational gene identification tools is also reported [26]. Catherine and some other groups of

researchers have provided a review of the existing approaches of gene identification in eukaryotic organisms, their advantages as well as the limitations [6, 27, 28]. A large number of gene identification tools are available publicly in the Web site http://www.nslij-genetics.org/gene/. A new algorithm for gene identification, CONTRAST is also studied and reported [29]. In recent times, a method is proposed [30] which uses statistical methods to combine the gene predictions of ab-initio gene finders, protein sequence alignments, expressed sequence tag and cDNA alignments, splice site predictions, and other evidences. The gene identifier Combiner integrates multiple gene prediction programs and a large number of evidences are available in a typical annotation pipeline including evidence from proteins, ESTs, cDNAs and splice site predictions [30]. Although methods to predict potential protein coding regions on genomic DNA sequences came into existence since 1980s, the first program to assemble potential DNA coding regions into translatable mRNA sequences were not available until the early 1990s [31]. From the recent past, there are several programs available for biology scientists. GRAIL is the one amongst them, which is widely used today and is available on the BLAST web site for gene structure detection (BLAST: http://www.ncbi.nlm.nih.gov) [31, 32].

Hidden Markov Models (HMM) have been applied successfully in various applications, viz. speech recognitions [33]. An HMM model is a type of process in which some of the details are unknown or hidden and is stochastic in nature. This process uses a number of states and probabilistic state transitions and is usually represented by a graph in which transitions are represented by edges and states by vertices. Individual states are denoted by Y, which are associated with a discrete output probability distribution, P(Y). Transition probability is the probability of going from a certain state to the next state. Thus, the sum of the probabilities of all the transitions from a given states *s* to all other states must be 1. Markov and HMMs are gaining popularity in bioinformatics research for nucleotide sequence analysis [24, 34-37]. For prokaryotes gene identification, Borodovsky *et al.* [38] effectively applied this HMM technique. Eukaryotic promoter detection algorithm using a Markov transition matrix was proposed by Audic and Claverie [39]. A new technique VEIL (Viterbi Exon-Intron Locator) was developed by Salzberg [40] and Henderson *et al.* [41] to identify translational start site and splice sites in eukaryotic mRNA. The HMM based gene predictor GeneScout was developed by Yin *et al* [42], to detect translational start site and mRNA splicing junction sites. Our proposed technique used in this manuscript differs from Salzberg's and others, in which two different HMM are used; one for 5' and another for 3'splice sites. Every model consists of two elements; one for false sites and another for true sites. Other approaches consisting of multiple evidence types can be found in the Eu-Gene [43] and GAZE [44] systems. After gaining popularity in the conventional methods, much more attention is being paid on computational intelligence techniques like support vector machine (SVM) due to more accuracy. Several programs

based on similarity searches have emerged during the last decade lead by Gelfand et al. with PROCRUSTES [45]. Most of these programs are based on the principle of combining the similarity information with signal information obtained by signal sensors, which are used to refine the region boundaries. These programs succeed to rectify all the weaknesses of the sensors used but these may fail in case when non-canonical splice sites are present. A process satisfies the Markov property if one can explain the future predictions based on its present; or, the present status, based on the process's history [21, 46].

Identifying splice site on the basis of models of site / signal recognition and sequence data supported by experimental confirmations has been described in a multi-agent system named AMELIE [47]. The NetPlantGene and AMELIE are two independent systems devoted to the recognition of splice sites in plant and human genomes respectively [48]. The HMM system, proposed by Salzberg *et al.* [40] is used to predict translation start site and splice site in the eukaryotic genes. They also developed Viterbi exon-intron locator (VAIL) [49], which is also an HMM based eukaryotic gene predictor tool. To signify the consensus and degeneracy features of splicing sites in eukaryotic genes, an effective HMM has been developed by Michael *et al.* [31], which is utterly trained by using expectation maximization (EM) algorithm. A 12-fold cross-validation method was also used to calculate the performance of the system. The Feature Subset Selection (FSS) by using Estimation of Distribution Algorithm (EDA) is established [50], which have shown superior performance in classification of splice sites [51] in case of Arabidopsis thaliana. An FSS, based on wrapper algorithm and united with SVM is used for splice site prediction [52]. Another newer technique, EDA based feature ranking, was used for splice site identification in rapid feature selection process [53]. The SVM was also used by Brown *et al.* [54] to calculate the useful function for microarray gene expression by classifying genes and the existing data sets. To trace the signals in ribosome binding sites, splice site and promoter section, one computational technique was developed by Rodger Staden [55]. This method allocates separate values to all bases at every position of the identification sequence to specify the comparative significance of each base, which is performed by weight matrix models (WMM). Zhang and Marr [56] introduced WAM that oversimplified the conventional WMM and integrated the dependencies between contiguous locations. Splice site can be identified by applying four stochastic regular grammar (SRG) inference algorithms controlled by generalization parameters with 10-way cross-validation to choose the best grammar for each algorithm [57]. To identify Translational Start Sites (TSS) and splice sites junction in eukaryotic mRNA, Salzberg established conditional probability (CP) matrices [40]. Gene finding model GENSCAN was introduced by Burge and Karlin [2, 58] and tested on human and vertebrate genes. It has the blend of the double-stranded nature of the model and the ability to deal with inconsistent numbers of genes, particularly useful for study of

long human genomic sequences. The maximal dependence decomposition (MDD) was also developed by them for modeling useful signals in DNA sequence, which recommended that there were strong relationship between some of two or three precise positions with base constraints and probably relate to the splice site recognition. To identify transcription factor binding sites (TFBS) and splice sites, Huang and Zhao [59] had used permuted variable length Markov models (PVLMM) that can confine the potentially important dependencies with locations. For the prediction of splice site region in human pre-mRNA an artificial neural networks (ANN) had been applied [60] . A time-delay neural network model which is a type of feed-forward neural network had shown their application to promoter annotation in the Drosophila melanogaster genome, and name of the tool was neural network for promoter prediction (NNPP) [61].

It was already recommended that a large improvement in the recognition of splice sites is possible if that model will be using hybrid architecture, like one of the foundations is statistical models like WMM, MM1, MDD etc., is joined with other signal methods. GeneSplicer [62] is such type of method, where second order Markov models (MM2) are united with MDD. Probabilistic parameters of first order MM are joint with a support vector machine (SVM) to predict splice site [63]. The addition of RNA structure information surely increased the accuracy of eukaryotic splice site finding was examined by using Markov models of zero to second orders [64]. Rajapakse and Ho *et al.* [33] merged a typically MM2 and back propagation neural networks (BPNN) to establish another splice site predictor.

As far as Genetic Algorithms (GAs) are concerned, the initial one was proposed by J. H. Holland, where the genetic algorithm applies the principles of natural evolution to finding an optimal solution to an optimization problem [65]. A comparison of GA with Simulated Annealing (SA) was done by Gunnels et al. [66] and found that the GA based method rapidly converged to a good solution as it was able to take the benefit from the extra information to create superior local maps which were useful in constructing good global maps. Using a GA, Alexander *et al.* [67] designed the sets of appropriate oligonucleotide probes capable to predict new genes belonging to a defined gene family within a cDNA or genomic library. This approach requires the low homology to identify functional families of sequences with little homology, which is the major advantage. A new approach for identifying promoter regions of eukaryotic genes using a GA with an implementation on *Drosophila melanogaster* is described [68] in which realizing the genetic algorithm to search for an optimal partition of a promoter region into local non-overlapping fragments and selection of the most significant dinucleotide frequencies for the obtained fragments.

Support Vector Machines (SVMs) are the set of related supervised learning methods used for classification and regression [69]. The SVMs have been developed by Vapnik [69] and gained popularity due to many promising features such as better empirical performance. The formulation

uses the Structural Risk Minimization (SRM) principle, which has been shown to be superior [70] to traditional Empirical Risk Minimization (ERM) principle, used by conventional neural networks. SRM minimizes an upper bound on the expected risk, whereas ERM minimizes the error on the training data. It is this difference which gives SVM the quality of generalization and ease of training with a greater ability, which is the primary requirement in statistical learning. SVMs were developed to solve the classification problem, but recently they have been extended to solve regression problems [71]. The concepts of fuzzy sets and fuzzy logic for the gene identification were introduced in the 1960s by Lotfi A. Zadeh [72, 73] as a generalization of conventional set theory. This model deals with quantifying the imprecision and uncertainty that is not easily captured by standard mathematical models. Some of the reasons for utilizing SVMs in Bioinformatics are – these have a strong widespread application in machine learning for classification and, they can target relevant data positions automatically [74]. Other applications of SVM in bioinformatics are – identification of human signal peptide cleavage sites, secondary structure of protein and multi-class protein fold detection. Till date, the most popular techniques in use for splice site recognition are Markov models which need the labor-intensive selection of information resource; SVM, support vector kernels [40, 62, 75-83].

Theoretically, ideal gene predictor should have the ability to recognize the exact boundaries of all the attributes common to most eukaryotic protein-coding genes. The specific sequences which are there between the introns and exons can be identified by gene prediction algorithms. Many gene prediction programs focus solely on identifying the protein-coding regions of a gene, that is, the region between the start codon and an in-frame stop codon. However, the regions upstream and downstream of the protein coding exons are imperative regulatory regions, and therefore, should be included in attempts to define the boundaries of a gene.

2.1 Biological Background

Deoxyribonucleic acid (DNA) and proteins are biological macromolecules built as long linear chains of chemical components. Proteins are assembled from a number of building blocks, called amino acids (out of which 20 are used in practice) using information encoded in genes and are responsible for structural behavior. Every protein has its own unique amino acid sequence that is specified by the nucleotide sequence of the gene encoding this protein. Amino acids are the organic molecules containing an amine group, a carboxylic acid group, and a side-chain that varies from amino acid to amino acid. It is important that the instructions for building the proteins on reproduction of an organism are reproduced accurately and completely, which are maintained by the organic molecules known as nucleotides. Amino acid characters are advantageous over nucleotide characters due to increased character-state space for amino acids [84].

The nucleotide in DNA consists of a sugar (deoxyribose), one of four bases (cytosine (C), thymine (T), adenine (A), guanine (G)), and a phosphate. Cytosine and thymine are pyrimidine bases, while adenine and guanine are purine bases. The sugar and the base together are called a nucleoside. Nucleotides are monomer building block in the nucleic acids [85]. The two most common types of nucleic acids are deoxyribo-nucleic acids (DNA) and ribonucleic acid (RNA). DNA is found mainly in the nucleus of the cell, while RNA is found mainly in the cytoplasm of the cell although it is usually synthesized in the nucleus. In DNA, because of the presence of hydrogen bonds, A pairs with T and G pairs with C. DNA contains the genetic codes to make RNA and the RNA in turn then contains the codes for the primary sequence of amino acids to make proteins. It also plays a fundamental role in different biochemical processes of living organisms in two respects. First, it contains the templates for the synthesis of proteins, which are essential molecules for any organism. The second role in which the DNA is essential to life is as a medium to transmit hereditary information (namely, the building plans for proteins) from generation to generation. Nucleotides are arranged in two long strands that form a spiral called double helix. The structure of the double helix is similar to ladder, with the base pairs forming the ladder's steps and the sugar and phosphate molecules forming the vertical sidepieces of the ladder. A molecule of DNA is organized in the form of the two complementary chains of nucleotides wound in a double helix. The DNA double helix is stabilized primarily by two forces – hydrogen bonds between nucleotides and base-stacking interactions among the aromatic nucleobases. Twin helical strands form the DNA backbone. One of these strands is called the sense strand, and other the anti-sense strand or the template strand. The anti-sense strand contains the genetic code of a gene, and is transcribed. Generally, at any place in a DNA molecule, either of the two strands may be serving as the anti-sense strand. A gene is the basic building block of a living organism residing on the stretch of DNA that codes for a specific type of protein. Gene holds the information to build and preserve an organism's cells and transfer genetic information to their offspring. Although genes lie linearly along chromosomes but they are not always contiguous. The regions of DNA in between cluster of genes are called the Intergenic regions, which contain a few or no genes. Sometimes, some intergenic DNA act to control genes that are close by, but most of it has no currently known function. Because protein-coding genes are responsible for protein synthesis, therefore they are also called coding regions [86]. Intergenic regions constitute approx. 95% of the genomic strand and protein coding genes constitute 2% of the total DNA [87, 88]. All living organisms can be divided into two groups depending on their fundamental cell structure – prokaryotes and eukaryotes. In prokaryotes, the coding genes are not separated by protein non-coding regions, i.e. introns; but in case of eukaryotes, protein-coding regions, i.e. exons are separated by introns [87, 89]. Eukaryotic gene structure is shown in Figure 2.1.

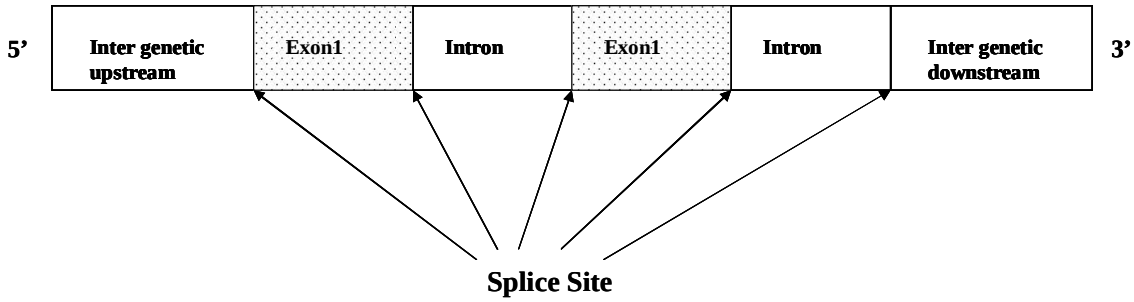

Figure 2.1 Eukaryotic gene structure

In case of Eukaryotes intron-exon separators are called Splice sites. Central dogma of biology states that is represented by four major stages. These are –

1. The DNA replicates its information in a process, called 'replication' that involves many enzymes.
2. The DNA codes for the production of messenger RNA (mRNA) during transcription.
3. The mRNA is processed by splicing mechanism and migrates from the nucleus to the cytoplasm.
4. Messenger RNA carries coded information to ribosomes. The ribosomes 'read' this information and use it for protein synthesis through translation process.

Proteins do not code for the production of protein, RNA or DNA. They are involved in almost all biological activities, structural or enzymatic[90]. All these processes are depicted in Figure 2.2.

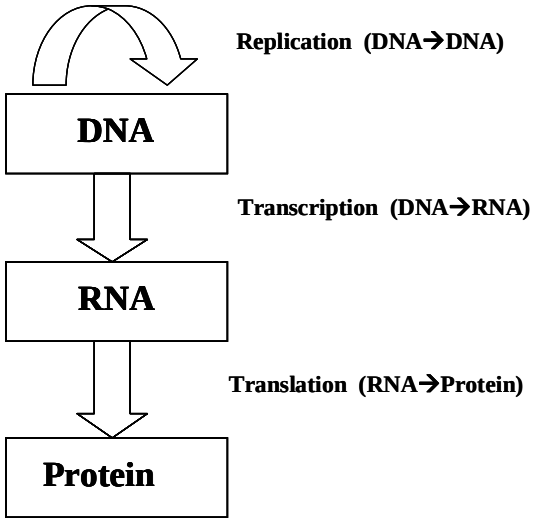

Figure 2.2 Central dogma of biology

In the eukaryotic gene expression, a gene is transcribed from DNA to pre-mRNA, through RNA processing, pre-mRNA produces mRNA which include splicing, capping, polyadenylation of the transcript, then it is transported to cytoplasm from the nucleus by the process of translation [20].

CHAPTER – 3

ANALYSIS OF GENE PREDICTION ALGORITHM AND TOOLS

The basic characteristic of a eukaryotic gene is the organization of its structure into introns and exons. In general, exons can be divided into four classes – 5' exons, 3' exons, internal exons, and intronless exons. According to their coding content, they can be further subdivided into 12 subclasses which have different statistical properties. In the gene finding process, identifying 5' splice site is the most cumbersome task due to difficulty of identifying the promoter and transcriptional start site (TSS) in DNA sequence. On a very high level, genes in human DNA and many other organisms have a relatively regular structure. At TSS, there is a beginning of a gene which is followed by the exon. Transcription initiation and promoter activation in eukaryotic cells is a complex process. The different steps of gene transcription starts with the binding of several transcription factors (TFs) on the "upstream" promoter that could be as far as ~ 1 kilo base pairs (kb) upstream of the start site in conjunction with enhancers, silencers and insulators, controls the binding of a pre-initiation complex on the core promoter that lies approximately 100 base pairs (bp) on either side of the TSS [20]. This helps in opening up the double helix, followed by a movement of the RNA polymerase from the 3' end to 5' end on the anti-sense strand of the DNA [91].

A set of three consecutive nucleotides in an mRNA or DNA is called a codon which codes for a specific amino acid. As there are a total of 64 possible codons, but only 20 different amino acids, coding redundancy exists. Splice site and promoter recognition processes are still limited. Moreover, there is no strong consensus sequence at the splice junctions. The knowledge of number of genes in the human genome is an estimation only till date and this value too, is getting revised frequently [92]. These and several other issues make the task of identifying genes extremely challenging.

The statement of a computational gene prediction problem can be expressed as:

Provide a DNA sequence as input

$S = (s_1, s_2, \ldots s_n) \in \sum^*$,

where $\sum = \{'A','C','G','T'\}$.

The output of the 'input' given as above may emerge the accurate labeling of each element in S belonging to a coding region (exon), intergenic region or non-coding region (intron) [8].

3.1 Overview of the Conventional Gene Prediction Techniques

The conventional techniques for gene prediction can be divided into identifying the evidence for gene and integrating the various evidences of genes for predicting the gene structure as shown in Figure 3.1 [6].

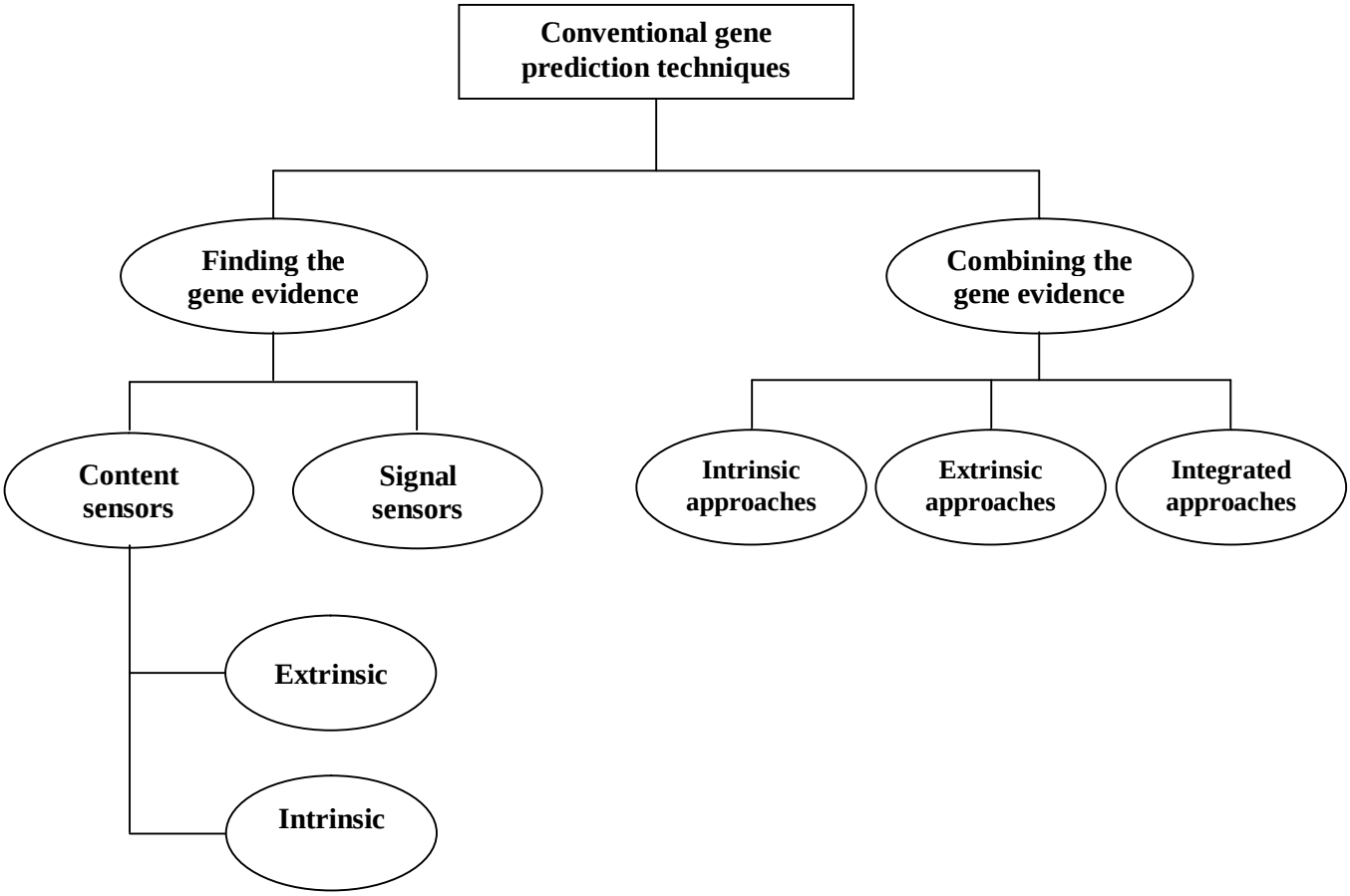

Figure 3.1 Gene prediction using conventional techniques.

3.1.1 Evidence Discovery

Here, we are considering the problem of finding genes coding for a protein sequence in eukaryotes only. The problem of finding genes in prokaryotes presents different types of difficulties (there are no introns and the intergenic regions are small, but genes may often overlap each other and the translation starts are difficult to predict correctly). Functionally, a eukaryotic gene can be defined as being composed of a transcribed region and of regions that cis-regulate the gene expression, such as the promoter region which controls both the site and the extent of transcription and is mostly found in the 5’ part of the gene. The currently existing gene prediction software looks only for transcribed region of genes, which is then called `the gene'. Signal sensors and content sensors are two fundamental types of information those are presently locate genes in genomic sequence.

Content Sensors

Content sensors are the frequencies of nucleotide, dinucleotide and trinucleotide frequencies (e.g., in exons and introns), and triplet frequencies. The triplet frequencies or nucleotide frequencies at three different positions of the triplet are usually dissimilar. Content sensor does not include site specific information [93]. Content sensors try to classify a DNA region into different types, e.g.

coding versus non-coding [6] and are of two types – extrinsic content sensors and intrinsic content sensors.

Extrinsic content sensors

Extrinsic content sensors are segregated from the training from the balanced set of non-coding regions. A large number of similarities between protein, DNA sequence and genomic sequence regions in the database are identified by using local alignment techniques such as Smith-Waterman algorithm [94], FASTA [95] and BLAST [96] to determine the transcribed or coding region. Similarities between protein sequences, genomic DNA and DNA sequences provide information about exons/introns location. The limitation of extrinsic content sensors is the poor quality of database, insufficient accuracy and missing small exons.

Intrinsic content sensors

These content sensor were defined initially, for prokaryotic genomes in which two types of regions were generally taken into consideration – first, the regions that code for a protein and will be translated, and second, the intergenic regions. The intrinsic content sensors are characterized by the fact that three successive bases in the correct frame define a codon that will be translated into specific amino acid in the protein. Since prokaryotic sequences does not contain stop codons, therefore in order to find out the potential coding sequences, the sufficiently long open reading frames (ORFs) approach is utilized. In case of eukaryotic sequence, the translated region will be very short and stop codon will be present [97]. Therefore, various other measures have defined to more delicately describe whether the sequence is ‘coding' for a protein – nucleotide composition and especially G+C content, hexamer frequency, base occurrence periodicity, codon composition etc. In general, most currently existing programs use two types of content sensors: one for coding sequences and one for non-coding sequences, i.e. introns, untranslated terminal regions (UTRs) and intergenic regions [98].

Signal Sensors

Signal sensors are specific functional sites those are present inside or at the boundaries of various genomic region and take part in various levels of protein encoding gene expression. In other words, these are the measures that try to predict the presence of the functional sites specific to a gene [6]. The basic and natural approach for the purpose is to search for a match with a consensus sequence with possible variations, the determination of the consensus being made from multiple alignments of the sequences. This type of method is used for splice site prediction in SPLICEVIEW [99] and a logitlinear model based approach (SplicePredictor) [100]. The positional weight matrices (PWMs) offer another flexible representation of signals which captures the probability of appearance of a base in a particular location. PWMs are also known as inhomogeneous zero order Markov Model

which follows the rule of a classical zero order Markov Model preposition. In order to capture possible dependencies between adjacent positions of a signal, one may use higher order Markov models. An inhomogeneous higher order Markov Model is the Weight Array Model (WAM), which was first proposed by Zhang and Marr [56] and later used by Salzberg [40], who applied it in the VEIL [41] and MORGAN [101] software. A modified WAM is also used in Genscan [2] tool to identify acceptor splice sites, and a second order WAM is used to represent branch point information. An alphabetical list of currently available splice site detection programs is presented in Table 3.1.

Table 3.1 Splice site prediction programs

Program	Organism	Method
GeneSplicer	Arabidopsis, human	HMM + MDD
NETPLANTGENE (http://www.cbs.dtu.dk/services/NetPGene/)	Arabidopsis	NN
NETGENE2 (http://www.cbs.dtu.dk/services/NetGene2/)	Human, C.elegans, Arabidopsis	NN + HMM
SPLICEVIEW (http://l25.itba.mi.cnr.it/~webgene/wwwspliceview.html)	Eukaryotes	Score with consensus
NNSPLICE0.9 (http://www.fruit¯y.org/seq_tools/splice.html)	Drosophila, human or other	NN
SPLICEPREDICTOR (http://bioinformatics.iastate.edu/cgi-bin/sp.cgi)	Arabidopsis, maize	Logitlinear models: (i) score with consensus; (ii) local composition
BCM-SPL (http://www.softberry.com/berry.phtml; http://genomic.sanger.ac.uk/gf/gf.html)	Human, Drosophila, C.elegans, yeast, plant	Linear discriminant analysis

HMM - Hidden MM; MDD - Maximal Dependence Decomposition; NN - Neural Networks

3.1.2 Evidence Combination for Gene Identification

Several types of signal sensors may be utilized for splice site identification of a given DNA sequence. Since the splice sites and translation starts and stops define the boundaries of coding regions, therefore these evidences can be combined to identify the gene structures, which are different from earlier approaches for identifying respective exons [6]. Theoretically, these signals indicate a prospective gene location, viz. intron, exon or its coding part. Correct gene structure must satisfy certain characteristics like there are no overlapping exons, coding exons must be compatible with frame, and in-frame stop at the junction will not be generated by integrating two successive

coding exons. Various gene identification tools like Soderlund [27] and of Gelfand [28] are capable of using the above mentioned techniques even for identifying the complex gene structure.

This approach can be divided into three categories – intrinsic approach, extrinsic approach, and combined approach.

Intrinsic / Ab-initio Approaches

These approaches try to locate all the gene elements which occur in a genomic sequence including probable partial gene structure at the border of the sequence. Maximum intrinsic gene finders use dynamic programming (DP) to predict the most likely gene structure according to the evidence defined by both signal sensors and content sensors. These type of gene modeling approaches can be implemented with chart language [102] and said to be exon based or signal based depending on whether a gene structure is considered to be an assembly of segments defining the coding part of the exons or by the presence of a succession of signals separated by 'homogeneous' regions, respectively [103]. Gene assembly is separated from coding segment prediction step in case of exon based category. The main objective is to identify the highest scoring gene, which is simply the summation of the scores of the assembled segments. The segment assembly process may be defined as the search for a finest path in a directed acyclic graph where vertices represent exons and edges represent compatibility between exons. The search is made by using Viterbi algorithm [104], which produces a most likely gene structure and is known to be a specific instance of the older Bellman shortest path algorithm [105]. This approach is adopted in the programs viz. GeneId [106], GenView [107], GAP III [108], FGENES [109] and DAGGER [110]. They are presented in Table 3.2.

Table 3.2 Intrinsic / Ab-initio approaches gene prediction programs

Program	Organism	Gene Model
GeneId3 (http://www.imim.es/geneid.html)	Vertebrates, plants	DP
EuGene (http://www.inra.fr/bia/T/EuGene)	Arabidopsis	DP
DAGGER	Vretebrates	Directed acyclic graph
GeneParser (http://beagle.colorado.edu/~eesnyder/GeneParser.html)	Vertebrates	DP
Genie (http://www.fruitfly.org/seq_tools/genie.html)	Drosophila, Human, other	GHMM,DP
GenomeScan	Vertebrates	GHMM,DP
GENSCAN (http://genes.mit.edu/GENSCAN.html)	Vertebrates, Arabidopsis, maize	GHMM,DP

GENVIEW2 (http://123.itba.mi.cnr.it/~webgene/wwwgene.html)	Human, Mouse, diptera	DP
HMMgene (http://www.cbs.dtu.dk/services/HMMgene/)	Vertebrates, C.elegans	CHMM
MORGAN (http://www.cs.jhu.edu/labs/compbio/morgan.html)	Vertebrates	DP
VEIL (http://www.cs.jhu.edu/labs/compbio/veil.html)	Vertebrates	DP

CHMM, class HMM; GHMM, generalized HMM; IMM, interpolated MM; MM, Markov model.

Extrinsic / Homology approaches

All the programs under this class may be considered as complexities of the classical Smith-Waterman local alignment algorithm which are usually referred as spliced alignment programs. Most of these programs are enlisted in Table 3.3.

Table 3.3 Extrinsic / Homology approaches gene prediction programs

Program	Organism	Databank
EbEST (http://ares.ifrc.mcw.edu/EBEST/ebest.html)	Human, other	dbEST
CEM	Human, Mouse	Two genome sequence
AAT (http://genome.cs.mtu.edu/aat.html)	Primates, rodents, other	cDNA, Protein
GeneSeqer (http://bioinformatics.iastate.edu/cgi-bin/gs.cgi)	Arabidopsis, maize, generic plant	dbEST or EST database or Proteins
GENQUEST (http://compbio.ornl.gov/Grail-bin/EmptyGenquestForm)	Human	dbEST, SwissPort, Prosite
ORFgene2 (http://125.itba.mi.cnr.it/~webgene/wwworfgene2.html)	Human, Mouse, Drosophila, Aspergillus, Arabidopsis	SwissPort
ProGen (http://www.anchorgen.com/pro_gen/pro_gen.html)	Prokaryotes, Escherichia coli	Two genome sequences
PredictGenes (http://cbrg.inf.cthz.ch/Server/subsection3_1_8.html)	Prokaryotes, Plants	
SYNCOD (http://125.itba.mi.cnr.it/~webgene/wwwsyncod.html)	Human, Mouse, Arabidopsis, Aspergillus	BLASTN output
TAP (http://sapiens.wustl.edu/~zkan/TAP/)	Human, Mouse, Drosophila	dbEST

AAT, analysis and annotation tool; ORF, open reading frame; TSS, transcription start site; DP, dynamic programming; WAM, weight array matrix; WMM, weight matrix method

Combined approaches

Now when the added values are provided by database similarities, the researchers are combining extrinsic and intrinsic approaches in presently available gene prediction tools. Also, the updates are

added into the older software to include information from homology [111]. Considering this approach, Gene Structure Assembly (GSA) program evolved by combining AAT [112] and Genscan whose results are better than those obtained from these programs used separately. In this regard, TWINSCAN [113] is an important program worked upon. GenomeScan [114] is an extension of Genscan which incorporates the similarity with a protein recovered by BLASTX or BLASTP. Accuracy of GenomeScan is better than Genscan and BLASTX used separately. A highly integrative approach is used in the EuGene [115] program, which is a combination of NetGene2 and SplicePredictor for splice site prediction, NetStart [116] for translation initiation prediction, IMM based content sensors and similarity information for protein, cDNA and EST matches. These concepts are used by many authors to obtain desirable results like DIGIT (http://ismb01.cbs.dtu.dk/GeneFinding.html#A303) integrates FGENESH, Genscan and HMMgene.

3.2 A Few Conventional Approaches

In this section, brief overviews of some conventional gene identification techniques are discussed without going too deeply into their mathematical parts and algorithm [117].

3.2.1 Hidden Markov Models

Hidden Markov Models (HMM) are statistical model used to characterize types of physical systems. A good HMM can accurately models the real world source of the observed data and has the ability to simulate the source. Machine Learning techniques based on HMMs have been successfully applied to problems including speech recognition, optical character recognition, and problems in computational biology. It consists of two stochastic processes – first, a Markov chain, which is characterized by state probability and transition probability, and state of the chain is hidden; second, produces emissions observable at every moment and dependent on a state dependent probability distribution [118]. In case of DNA sequence, a Markov model assumes that the probability of appearance of a given base (A, T, G or C) at a given position depends only on the k previous nucleotides, where k is called the order of the Markov model. Such a model is defined by the conditional probabilities $P(X \mid k \text{ previous nucleotides})$, where X = A, T, G or C. By using a Markov model, one can then simply compute the probability of the sequence generated according to this model [119]. HMM are more successful because they can naturally accommodate uneven length models of sequence regions because maximum biological data has variable length properties [120, 121]. They are used for motif finding [122], multiple sequence alignment [123] and identification of protein structure [124]. As shown in Figure 3.2, there are states representing exons and introns with specific states to the model aspects of the gene parse; with the states being squares

and transitions as arrows between the states. Some of the examples of the HMM based gene identification tools are Genscan [2], Genie [125] and HMM Gene [126].

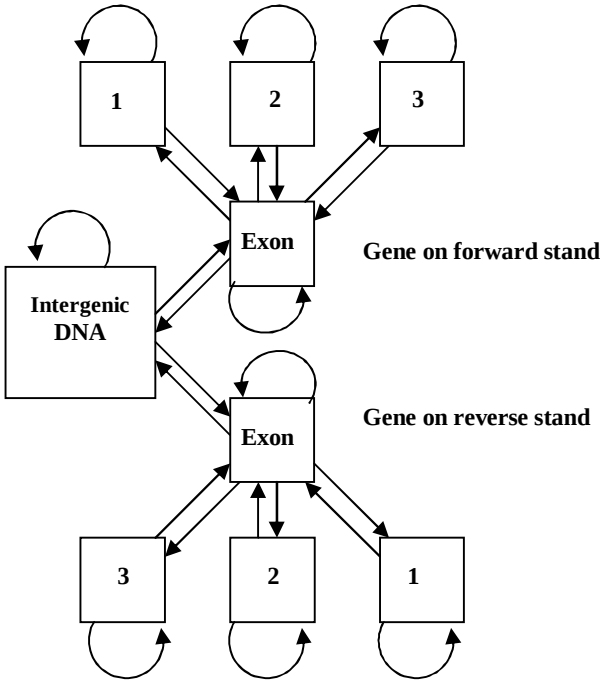

Figure 3.2 Hidden Markov Models for Gene in DNA sequence. The HMM is divided into 2 parts: genes on forward or reverse stand of DNA sequence. Every gene model contains a central exon state which has an emission of nucleotides tuned to recognize protein coding regions. Introns are interrupting exons, three introns states are used, because there are three relative positions at which an intron can interrupt a codon of DNA base. Introns are separated by their "phase"-1, 2, 3.

If we assume the set of internal state is $P = \{'c',\ 'n'\}$, where 'c' indicates coding and 'n' indicates non-coding internal states and the set of emissions is the set of four DNA bases:

$$X = \{'A',\ 'T',\ 'G',\ 'C'\}$$

By using HMM, we are basically solving three basic problems – first, evaluation problem, in which we calculate the probability that a given model will generate a given sequence of observations; second, decoding problem, in which it will calculate most likely hidden state from a sequence of observations; third, learning problem, in which it will identify the optimal model by knowing a sequence of observations [127]. To solve evaluation problem, forward algorithm will find the probability of emission distribution starting from the beginning of the sequence and to find the probability of emission distribution which starts from the end of the sequence, backward algorithm is used. In case of decoding problem, Viterbi algorithm is used to identify the sequence of internal state that has the highest probability and to identify the position of the internal state that has highest probability, posterior decoding algorithm is used. Learning problem is solved by Viterbi training to find the optimal model based on the most probable sequence and Baum-Welch algorithm, which identify the suitable model based on the sequence of most probable internal state [127].

For providing training to the HMM models, the introns, exons and other non-coding regions in the training set are separated out and trained separately [128]. The Viterbi and Expectation Maximization (EM) algorithms are used for computing with HMM during its training and testing [104, 129]. There are a number of standard techniques for training hidden Markov models, out of which the best is the Baum–Welch method. In this method, the model topology is fixed, and all of the output probabilities and transition probabilities [36] are initialized to random values. The E-M algorithm re-estimates all of these probabilities once being presented with a set of DNA sequences. The probability of observing a sequence E of emissions given a likelihood function of λ (HMM) is given by $P(E \mid \lambda)$. The E-M algorithm is certain to converge to a locally optimal estimate of all the probabilities in the model. Generally, it is assumed that the multiple observations in the training data are independent of each other, but, this assumption of independence may not always hold in practice. A formal treatment of HMM training without imposing the independence assumption is available [130]. If a sequence is given with a trained HMM, the Viterbi algorithm will be able to find the most probable sequence of states through the model for that particular sequence. Additionally, it calculates the probability of the model producing the sequence via that path [131].

3.2.2 Dynamic Programming Approach

Nearly all integrated gene prediction methods use dynamic programming approach to combine candidate exons and other scored regions and sites into a complete gene prediction with maximal total score. This is an efficient mathematical technique that can be used to find optimal 'path' or routes to multiple destinations. Dynamic programming belongs to a special class of optimization or minimization techniques. There are a number of characteristics in all dynamic programming

(i) The problem can be divided into stages with a decision required at each stage.

(ii) Each stage has a number of states associated with it.

(iii) The decision at one stage transforms one state into a state in the next stage.

(iv) Given the current state, the optimal decision for each of the remaining states does not depend on the previous states or decisions.

(v) There exists a recursive relationship that identifies the optimal decision for stage j, given that stage $j+1$ has already been solved.

(vi) The final stage must be solvable by itself.

The dynamic programming algorithm is a well-established procedure to identify the coding region and optimal pathway among a series of weighted steps. GeneParser [132] which employs dynamic programming technique, uses coding measures and signal strengths to calculate scores for all subintervals in the test sequence. Then, a dynamic programming approach is used to predict the most suitable combination of exons and introns. The use of dynamic programming in gene

prediction is also reviewed by Gelfand and Roytberg [133], who suggested 'vector dynamic programming' to combine multiple exon quality indices without the time-consuming training of a neural network. These approaches have been implemented in CASSANDRA [134]. GREAT [135] is a program to identify protein-coding segments in the DNA sequence. The GenView system [107] is again based on the prediction of splicable ORFs ranked by the strength of their splice signal and their coding potential ('in phase' hexamer measure). The best gene structure is then constructed using dynamic programming to sift through the numerous possible exon assemblies. GRAIL II [136], GeneParser [132], FGENESH [109], GAP III [108] and recent versions of GeneId [106] also use dynamic programming approach. A brief review of the dynamic programming in gene finding is provided [22].

3.2.3 Bayesian Networks

A Bayesian network is a probabilistic graphical model is a type of statistical model that represent a collection f random variables and their conditional dependencies via a directed acyclic graph (DAG) G consisting of nodes corresponding to a random variable set $X = \{X_1, X_2, ..., X_n\}$ and ages between nodes, which determine the structure of G and hence the joint probability distribution of the whole network [137, 138]. Over the last decade, the Bayesian network has become a popular representation for encoding uncertain expert knowledge in expert systems [139]. An example of a Bayesian network is shown in Figure 3.3. An arc between variable M and E_1 denotes conditional dependency of E_1 on M, as determined by the direction of arc. Additionally, Bayesian network includes a quantitative measure of dependencies. For each variable and its parents, this measure is defined using a conditional probability function.

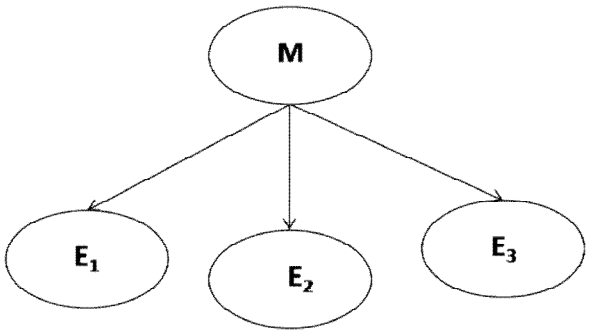

Figure 3.3 An Example of a Bayesian Network

In this example, one such measure is probability $P(E_1 | M)$. This model defines a specific factorization of the joint probability distribution function over the variables in the network. Hence, Figure 3.3 defines $P(M, E_1, E_2, E_3)$ as

$$P(M, E_1, E_2, E_3) = P(E_1 | M)\ P(E_2 | M)\ P(E_3 | M)\ P(M)$$

When used in conjunction with statistical techniques, the graphical model has several advantages for data analysis, like –

- handles situations where some data entries are missing
- can be used to gain understanding about a problem domain and to predict the consequences of intervention
- serves as ideal representation for combining prior knowledge (which often comes in causal form) and data
- an efficient and principled approach for avoiding the over-fitting of data

The methods of learning probabilities in a Bayesian network, with and without the complete data are also studied [139]. A Bayesian network framework for combining gene predictions from multiple systems is given [140], where the approach adopted is that of combining the advice of multiple experts.

3.3 Identification of Gene using Computational Intelligence Approaches

Computational intelligence (CI) is a set of nature-inspired computational methodologies and approaches to solve complex problems of the real word applications to which the conventional approaches are infeasible and/or ineffective. It includes neural networks, fuzzy logic systems, and evolutionary computation [141]. While some techniques within computational intelligence are often counted as artificial intelligence techniques, e.g. genetic algorithms, or neural networks; there is a clear difference between these techniques and traditional, logic based artificial intelligence techniques. In general, typical artificial intelligence techniques are top-to-bottom where, the structure of models, solutions, etc. is imposed from above. Computational intelligence techniques are generally bottom-up, where order and structure emerges from an unstructured beginning. Such methods can be used to develop robust models, either of their own or integration with standard statistical approaches. This is useful especially in data mining, where modeling is the basic component of scientific understanding. They are also useful to solve biological problems as well as bioinformatics problems. Here we are discussing some of the Computational Intelligence techniques along with those that are frequently used for gene identification.

3.3.1 Case Based Reasoning (CBR)

Case-based reasoning (CBR) is the process of solving a new problem based on the solution of past similar problem. In other words, in CBR, a reasoner remembers a previous situation that is similar to the current one and uses them to solve a new problem [142]. It is a model of reasoning where the systems' expertise is embodied in a library of past cases, stored as a case base already experienced by the system. CBR does not encode explicitly as rules, or implicitly as decision boundaries. CBR has been formalized as a four step process [143] –

1. **Retrieve**: It involves retrieving the solution for a given problem from the memory cases (set of similar cases). A case consists of a problem, its solution, and typically annotations about how the solution is derived.
2. **Reuse**: It involves mapping of solution from the previous case to solve target problem and adapting the solution as needed to fit the new best possible solution.
3. **Revise**: After 'reuse', revise involves testing the new solution in the real word (or a simulation) and revise, if required.
4. **Retain**: After successfully adapting the solution of the target problem, retaining involves storing the resulting experience as a new case in the memory.

An application of CBR to the gene-finding problem has investigated [144]. It makes use of a case library of nucleotide segments that have previously been categorized as non-coding (intron) or, coding (exon) in order to locate the coding regions of a new DNA strand. A similarity metric for nucleotide segments is established and results of multiple cases are combined to categorize entire new DNA strands. Overton and Haas [145] initially describe a case-based system that used grammar for describing the feature of genes such as exon, intron, promoter region etc. For efficient working of CBR system, it is necessary to be able to compare the case (e.g., a known exon), with the target strand of DNA in which we want exons to be identified. Costello and Wilson [146] have investigated a number of possible approaches to similarity, including longest common subsequence and sequence alignment methods to develop a CBR approach to find a gene in the mammalian DNA. Provided a measure of sequence similarity, it is required to employ the case library segments such that it will enable us to separate regions of a DNA sequence and identify them as possible protein coding regions. Since library exons are likely to be much shorter than a new strand, an approach that combines many retrieved cases in order to achieve the new solution was adopted [147]. Three measures for classifying an individual nucleotide were adopted –

1. Nucleotide activation score that is normalized by the maximum nucleotide activation score
2. Number of best possible matches the nucleotide can participate in, normalized by the maximum nucleotide participation score
3. Product of first two measure in combination to identify the coding status.

The CBR framework has been applied to the problem of annotating genes and the regulatory elements in their proximal promoter regions. CBR has several advantages such as the problem with sparse data is nowhere more evident than in the study of patterns in DNA necessary for the regulation of gene expression; and it is the objective of this type of research to discover these mechanisms. A database, EpoDB [148] is designed for the study of gene regulation during differentiation and development of vertebrate red blood cells. In this work, the data related to red blood cells was extracted from SWISS-PROT, GenBank, transcriptional regulation data (TRRD)

and expressions level data GERD to create a combined and more accurate view. The objective is to create an informatics system for the study of gene expression in red blood cells and its functional analysis differentiation, called erythropoiesis. The functionality of EpoDB involves the capability to extract features and subsequences, bioWidget viewers and integrated analysis tools to display sequences and features in graphical format. It can be accessed at http://cbil.humgen.upenn.edu/epodb/. A detailed survey of the applications of CBR is also provided in applications of molecular biology for gene identification [149].

3.3.2 Artificial Neural Networks

Artificial Neural Networks (ANNs), usually called neural networks (NN), are mathematical models or computer algorithms based on modeling the neuronal structure of natural organisms. A neural network consists of an interconnected group of artificial neurons and it processes information using a connectionist. They are stimulus-response transfer functions that accept some input and yield some output [150]. They are typically used to learn an input-output mapping over a set of examples. It is an adaptive system that changes the structure of neurons based on internal or external information that comes through the network during the learning phase. In general, if given sufficient complexity, there exists an ANN that will map every input pattern to its appropriate output pattern, so long as the input output mapping is not one-to-many. ANNs are therefore well suited for use as detectors and classifiers.

Multilayer perceptrons, also known as feed-forward networks, are the most common architecture used in supervised learning applications in which exemplar patterns are available for training. In case of DNA sequence, ANNs are trained over set-up example like features of given nucleotide sequence with an output being a decision concerning the similarity that whether it is coding or non-coding. Modern neural network include non-linear processing features interconnected by variable or fixed weights. Each computational node sums N weighted inputs, subtracts a threshold value, and passes the result through a logistic function. Kohonen self-organizing maps, recurrent networks etc. networks are useful for a specific problem set and researcher should familiarize themselves with their use before deciding the best possible solution as shown in Figure 3.4.

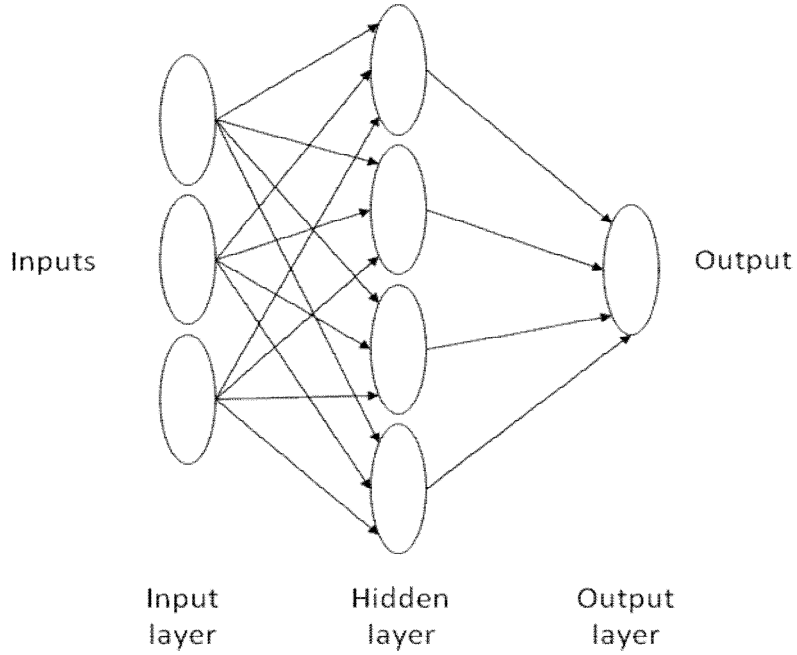

Figure 3.4 A typical artificial neural network representation consists of input layer, hidden layer and output layer

Single perceptrons form decision regions are separated by a hyperplane. If the different data classes being input are linearly separable, a hyperplane can be positioned between the classes by adjusting the weights and bias terms. If the input data are not linearly separable, a least mean square (LMS) solution is typically generated to minimize the mean square error (MSE) between the calculated output of the network and the actual desired output. After selecting a specific architecture like the type of network, number of nodes per layer, number of layers and connection between nodes, a training set for the input pattern is developed. The collection of uneven weights on the ANN identifies the desired output for each presented pattern. Then, every ANN can be scored in the light of fitness metric that reduces the squared error between the target value and actual value. Three learning patterns are associated with ANNs – supervised learning, unsupervised learning and reinforcement learning. Supervised learning technique need the use of a collection of training example and their real output. These examples are used to develop a model which relates features about the given input with the output decision. This approach is used to target pattern in the database having numerous features. In case of unsupervised learning, the target patterns are unknown. ANNs must be tuned to make correct decision in the absence of known fact. Clustering techniques are sometimes used in care of unsupervised learning to identify the similarity between object in a data-set. Reinforcement learning approach allows the machine or software to learn suitable behavior based on the feedback from the given environment. This technique can be used with supervised and unsupervised approaches. All the above mentioned techniques require optimization of a model in light of a fitness function. Back propagation method is an example of supervised learning that use gradient descent to minimize the squared error between ANN output and actual target value [151, 152]. After proper and sufficient training, the best ANN is prepared to

handle the testing sample to measure the accuracy of true positive. For testing the validity of ANN trained model, another test set is used to calculate sensitivity and specificity of the model in light of data which was not used during model building process. A previous attempt at computer-aided gene recognition, such as the well-known GRAIL software, used an ANN to combine a number of coding indicators calculated within a fixed sequence window [153]. Fickett and Tung [98] noted that at the core of most gene recognition algorithm are one or more coding measures: functions that calculate, for any window of the sequence, a number or vector intended to measure the "codingness" of the sequence. Common examples of these measures include codon usage, base composition vector, etc. An exon recognition method includes both a coding measure and a method of deciding between "coding" or "non-coding" regions for each vector. Such an approach to evolve ANNs capable of identifying coding and non-coding regions is available [154]. ANNs combined with a rule based system has been used for splice site prediction in human Arabidopsis Thaliana by using a joint prediction scheme where the prediction of transition regions between introns and exons regulates a cut-off level for local splice site assignment [155].This is followed by a rule based refinement that uses splice site confidence values, prediction scores, coding context, and distances between potential splice sites. This work has been further improved by the incorporation of the information regarding the branch point consensus sequence found by a non-circular approach using Hidden Markov Models [156].

For the analysis of Drosophilia melanogaster genome a time delay architecture which is based on feed forward neural network has applied [61]. The E. Coli gene prediction by locating the promoters of genes are performed by neural network based multi-classifier system [157].There are similar other applications of neural network for gene identification [158].The gene identification tool Dragon Gene Start Finder (DGSF) [159, 160] uses promoter identifier to approximate the transcription start site (TSS). Second system estimates the occurrence of CpG islands on the DNA sequence. Then identified TSS and CpG islands generate several signals. The identification whether the combination of the CpG island and the identified transcription start site indicates the presence of gene starts or not, is done by neural network with A4-layer. A detailed review and application of ANN in bioinformatics is discussed [161].

3.3.3 Decision Trees

A decision tree is a decision supporting tool which uses a graph or decision model and their possible consequences, resource cost and utility including chance event outcomes. Decision trees are helpful to identify a strategy most likely to reach an objective and commonly used in operation research, especially in decision analysis [162]. They accurately differentiate between coding and non-coding DNA for sequences ranging from 54 to 162 base pairs in length [163]. An advantage of decision trees over techniques such as linear discriminent analysis is that they perform more

functions of feature selection automatically, the user can enter a large number of features, including irrelevant data, and the decision tree algorithm will use only a subset in building the tree. In that observation, the task of distinguishing between subsequences that are either entirely encoding or entirely non-coding was addressed. An integrated system MORGAN [101] is a tool to identify genes in the vertebrate DNA sequences that include its decision tree routine and algorithms for splice site identification and its performance on a standard database. It uses an OC1 decision tree system made for separating coding and non-coding DNA. Depending on a separate scoring function, the optimal segmentation takes a subsequence and indicates whether an exon is present in the given sequence or not. In MORGAN, the scoring functions are the collection of decision trees which are combined to give the estimate of a probability. The internal nodes of a decision tree are property values that are tested for each sub sequence passed to the tree which can be various coding measures (e.g., hexamer frequency) or signal strengths. MORGAN correctly identifies 58% of the coding exons, i.e. both the beginning and the end of the coding regions in a DNA sequence. Another well-known gene finder, GlimmerM [164] developed specifically for eukaryotes, uses decision trees hybridized with Interpolated Markov Model (IMM) and dynamic programming. This system is based on bacterial gene finder Glimmer. It selects the best combination from all the possible exons using dynamic programming to consider for inclusion in a gene model. The best gene model is a combination of the strength of the splice sites and the scores of the exons produced by IMM. A scoring function is built on the basis of decision trees to estimate the probability that a DNA subsequence is coding or not. The types of subsequences, which are estimated, are: introns, initial exons, internal exons, final exons and single exons. The average value of the probabilities obtained with the decision trees is calculated and used to produce a smoothed estimate of the probability that the given subsequence is of a particular type. When the IMM score over all coding sequences exceeds a preset value (threshold), only then the gene model is accepted.

3.3.4 Genetic Algorithms

Genetic Algorithms (GAs) are computer programs that impersonate the processes of biological evolution to solve the problems and to model evolutionary systems. In a genetic algorithm (GA), the problem is encoded in a series of bit strings that are manipulated by the algorithm, whereas in an evolutionary algorithm, the decision variables and problem functions are directly used. GAs are playing an increasingly important role in studies of complex adaptive systems, ranging from adaptive agents in economic theory to the use of machine learning techniques in the design of complex devices and integrated circuits. The fundamental GA can be described [66, 165] in a very simple way as follows –

(a) Start with a randomly generated population of N L-bit chromosomes (generate suitable solutions for the problem)

(b) Calculate the fitness function $f(x)$ of each chromosome x in the original population

(c) Create a new population by repeating following steps until the new population is completed

(i) Selection: Select two parent chromosomes in population for reproduction according to their fitness

(ii) Crossover: Exchanges subsequences of two chromosomes to form new offspring (children)

(iii) Mutation: Randomly flips some bits in a chromosome

(iv) Fitness: Evaluate the fitness $f(x)$ of each chromosome x in the new population

(d) If the end condition is satisfied, stop, and return the best solution in current population

(e) Go to step 2

(This assumes that N is even; if it is odd, one offspring may be discarded at random).

3.3.5 Support Vector Machine

Support Vector Machines (SVMs) belong to a family of generalized linear classifiers. In another terms, Support Vector Machine (SVM) is a classification and regression prediction tool that uses machine learning theory to maximize the predictive accuracy while automatically avoiding over-fit to the data. SVMs can be defined as the systems which uses hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. SVM became famous when, using pixel maps as input; it gave accuracy comparable to sophisticated neural networks with elaborated features in a handwriting recognition task [166]. It is also being used for many applications, such as hand writing analysis, face analysis and so forth, especially for pattern classification and regression based applications. Learning with structural risk minimization is the central idea behind SVMs, and this is elegantly accomplished by obtaining the separating hyperplane between the binary labeled data sets (± 1) that separates the labeled data sets with a maximum possible margin [167-169]. SVM has been found to be successful when used for pattern classification problems. Applying the Support Vector approach to a particular practical problem involves resolving a number of questions based on the problem definition and the design involved with it. One of the major challenges is that of choosing an appropriate kernel for the given application [70]. Figure 3.5 shows support vector machine with hyperplane and margin.

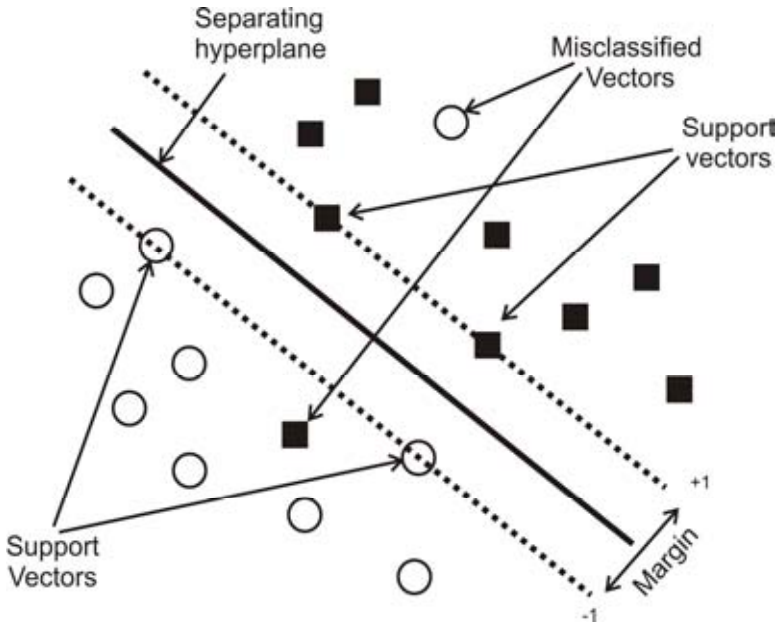

Figure 3.5 Support Vector Machine (SVM) with hyperplane and margin.

There are standard choices such as a Gaussian or polynomial kernel that are the default options, but if these prove ineffective or if the inputs are discrete structures more elaborate kernels will be needed. By implicitly defining a feature space, the kernel provides the description language used by the machine for viewing the data. Once the choice of kernel and optimization criterion has been made the key components of the system are in place. The major strengths of SVM are the training is relatively easy. No local optimal, unlike in neural networks. It scales relatively well to high dimensional data and the trade-off between classifier complexity and error can be controlled explicitly. The weakness includes the need for a good kernel function [170].

3.3.6 Fuzzy system

In contrast with the traditional logic theory where binary sets have two-valued logic, true or false, fuzzy logic variable can have true value that ranges in degree between 0 and 1. Due to this, fuzzy system has been extended to handle the concept of partial truth where true value may range from completely true and completely false. This notion fits very well with many pattern recognition problems where the classes to be separated do not have precisely defined membership criteria. In bioinformatics, for example, membership of a particular gene to a gene cluster may not be accurately defined and may indeed be improperly defined based on an arbitrary threshold of expression needed for classical approaches. Here, fuzzy system can be used for clustering or classification [171] and, used to manage uncertainty in rule-based representations, and rule conflict resolution [172] where the underlying logic of the representation is significant to the end-user. For example, a micro-array analysis system might have rules such as:

IF the expression of gene A is HIGH

THEN the predicted malaria prognosis is LOW

or

IF the expressions of gene A and gene B are

both MOSTLY ON

THEN the decision of malaria is TRUE

A broader review of fuzzy systems and their application to bioinformatics is given along with sample problems and papers [173-175].

3.3.7 Evolutionary Computation

Traditionally, there are many subdivisions of evolutionary computation or evolutionary programming (EP) [176] and evolution strategies (ES) [177]. At the phenotypic level, evolutionary programming and evolution strategies were visualized as pensiveness of Darwinian evolution. The latest derivations and approaches of evolutionary computation includes genetic programming (GP) [178] which represents individuals as tree structures of mathematical expressions, particle swarm optimization (PSO) [179] in which the populations of solutions is abstracted as a swarm of interacting particles with relative velocity through the search space directed by the value of each particle and the particles in neighborhood, ant-colony optimization (ACO) [180] which abstracts the individual solutions at ants that migrate through the solution space based on the trails left by other ants in the population, and others such as differential evolution (DE) [181, 182] a simple and efficient method of global search. These approaches are mostly similar in that they are all nature-inspired, maintain a population of solutions for the problem under consideration, impose some set of random variations to those solutions, and use a method of selection to determine which solutions are to be eliminated from the population, leaving the remainder to serve as ‘parents’ for the next generation of ‘offspring’ solutions. Evolutionary algorithms have been shown to possess asymptotic global convergence properties [183, 184] and thus they are very attractive methods for function optimization. Natural evolution can be considered as a population-based optimization process, the simulation of which on a computer results in a robust method for optimization, as shown in Figure 3.6.

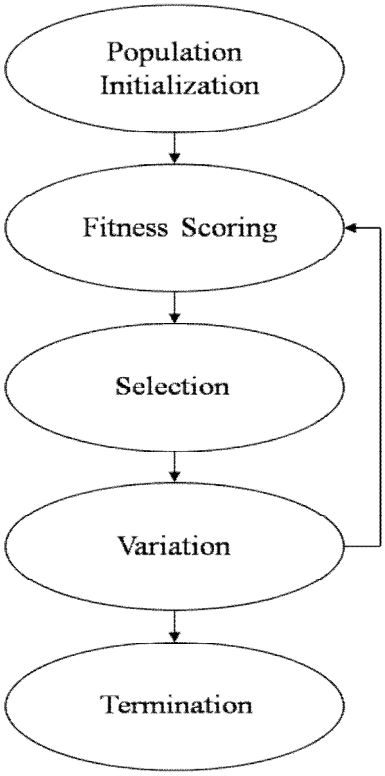

Figure 3.6 A standard flowchart for evolutionary algorithms

A population of solutions is built for the problem under consideration, taking care of the appropriate representation for the problem. Each solution is scored with respect to a fitness function, which may be mean squared error between predicted and actual values in the case of model optimization. In other problems such as transcription factor, this could be a complex equation of various terms and associated weights representing important aspect of the problem. Once all solutions have been scored, a selection method is used to eliminate inadequate solutions from the population. The remaining solutions serve as 'parents' for the next generation of 'offspring' solutions. In order to generate offspring solutions, different operators are applied. This process is repeated until a termination criterion is satisfied.

Evolutionary algorithms requires defining a cost function by the user so that alternative solutions can be scored appropriately, and for many real-world problems, defining a suitable cost function requires its own significant expertise. For rest of the problems, this may require its own research and development. A broader review of applications of evolutionary computation in bioinformatics including sample papers is available [185, 186].

3.4 Accuracy Measures and their Comparative Performance

Many researchers have compared the accuracy of the gene predictions by various programs. The programs were tested on all subset of programs of the same test sequence which gave a fair idea of the differences in the predictive powers of the tools. A detailed comparative study of a number of

computer programs for the prediction of gene structure in genomic sequences is provided [22, 24]. The gene prediction models classification performance was evaluated by various quantitative variables. These are true positive (TP), true negative (TN), false positive (FP) and false negative (FN) [187] as shown in Table 3.4.

Table 3.4 Definition of TP, TN, FP and FN

	Predicted positive	**Predicted negative**
Real Positive	True positive (TP)	False negative (FN)
Real Negative	False positive (FP)	True negative (TN)

Therefore TP is number of coding nucleotides predicted as coding, TN is the number of non-coding nucleotides predicted as non-coding, FP represent the number of non-coding nucleotides predicted as coding and FN is number of coding nucleotides predicted as non-coding. The sensitivity (S_N) or true positive rate (TPR) is the percentage of correct prediction of coding nucleotides and specificity (S_P) is the percentage of the prediction of non-coding nucleotides.

$$Sensitivity(S_N) = \frac{TP}{TP + FN}$$

$$Specificity(S_P) = \frac{TN}{TN + FP}$$

Accuracy (ACC) is the proportion of the DNA sequence in the test data set that are classified correctly which tells the capability of the gene predictor to assign coding and non-coding nucleotides sequences into appropriate categories.

$$ACC = \frac{TN + TP}{TN + FP + TP + FN}$$

All test sequences taken from human genes and are available [153]. Test set I contained sequences used in the testing of GRAIL and GeneID, while Test set II contained genes with complete protein coding regions and at least two exons. These sequences were first used and were without pseudogenes, multiple coding sequence fields and putative coding sequence fields, or alternative splicing forms [188].

At the nucleotide level, FGENEH performed with sensitivity = 77% and specificity = 88%, while the sensitivity and specificity of GRAIL 2 program were found to be 72% and 87% respectively [189] and FGENEH performed with sensitivity = 61% and specificity = 64%, while the sensitivity and specificity of GRAIL 2 program were found to be 36% and 43% at the exon level. An advancement of GRAIL 2 / GAP was actually not proven superior than GRAIL II in terms of sensitivity, specificity and missed exons, but in terms of wrong exons, GRAIL II / GAP (10%) was

better to GRAIL (28%). Performance of GeneID and GeneParser was not better than any other programs in any respect. MORGAN's performance was approximately same as that of GRAIL II and GAP. The best overall performance was shown by gene structure prediction model GENSCAN (sensitivity = 86%; specificity = 81%) and individual exon finder program MZEF (sensitivity = 86%; specificity = 86%) [190]. GENSCAN was found a little bit better than MZEF in terms of wrong exons and missed exons scores. William et al. [191] predicted the accuracy of the gene finders UNVEIL, a relatively recent HMM based gene finder, GENSCAN, Exonomy and GlimmerM for 300 genes based on full-length *A. thaliana* cDNAs and found that UNVEIL performs well in terms of nucleotide accuracy, exon accuracy, and whole-gene accuracy. The nucleotide accuracy, exon specificity, exon sensitivity of the four gene finders was found to be 94%, 75% and 74% respectively for UNVEIL, 95%, 63%, 61% for Exonomy, 93%, 71%, 71% respectively for GlimmerM and 74%, 80% and 75% respectively for Genscan. GRAIL and Evolved ANN both are based on neural network methodologies and the sensitivity of Evolved ANN program is found to be much higher than GRAIL but specificity and correlation coefficient of GRAIL is greater than Evolved ANN. On the whole, GENSCAN and MZEF perform better than any other program. Though a limitation in interpreting the results is that the test sets vary in size, and complexity or (G+C) composition, but some researchers studied, in detail, the comparative performance of different tools [22, 189].

There are some drawbacks found in the above mentioned approaches for gene prediction. The major limitation is that only about half of the discovered genes have significant homology to the genes in the databases. Another limitation with HMM method is that it has a little knowledge of gene structures, especially for new sequencing genomes. Furthermore, the current set of known genes is limited and certainly does not represent all potential features or their organizational gene themes. Recently, some techniques in physics and signal processing have been applied to overcome this limitation. The accuracy of the predictions can be measured at three different levels: coding nucleotide sequence, exonic structure, and protein product [192]. Exon level assessment mainly provides how accurately the sequence signals (splice sites, start codon, and stop codon, etc.) are identified. The accuracy can be measured by Sensitivity (S_N), Specificity (S_P), False Negative Rate (FNR), and False Positive Rate (FPR); the equations for FNR and FPR are given by:

$$FNR = 1 - S_N = \frac{FN}{TP + FN}$$

$$FPR = 1 - S_P = \frac{FP}{TN + FP}$$

Some popular programs for exon level prediction performances have been shown in the table 3.5, which is tested on [24] dataset.

Table 3.5 Exon level accuracy comparison of gene identification programs

Program	Sensitivity (S_N)	Sensitivity (S_P)	FNR (1- S_N)	FPR (1- S_P)
GENSCAN	0.78	0.81	0.09	0.05
FGENEH	0.61	0.64	0.15	0.12
GeneID	0.44	0.46	0.28	0.24
Genie	0.55	0.48	0.17	0.33
GenLang	0.51	0.52	0.21	0.22
GeneParser2	0.35	0.40	0.34	0.17
GRAIL2	0.36	0.43	0.25	0.11
SORFIND	0.42	0.47	0.24	0.14
Xpound	0.15	0.18	0.33	0.13

The evaluation system of gene identification still needs certain improvement to conquer the insufficiency of any individual gene prediction program. In this respect, integrated approach presented here is proved to constantly outperform the best individual gene finder.

3.5 Results

In bioinformatics, the gene identification is a challenging task and obviously still in improvement, especially, for larger genomes. In last few decades, various methods of gene identification based on HMM and dynamic programming have been developed. As gene identification leads to a structural annotation of the genomes which is then used for experimentation, the value addition to the identifications will be given for each predicted gene. Given the difficulty of the problem, computational intelligence based methods have also been applied in recent times because of their robustness and ability to handle noisy and incomplete/uncertain data. FGENES / FGENESH (species specific gene prediction tool estimation programs) uses Viterbi algorithm to search for optimal path. GRAIL and GRAIL 2 uses neural network for gene prediction, GRAIL 2 being the advancement on GRAIL. GeneMark uses one-homogeneous model for protein coding DNA and homogeneous Markov Model for non-coding DNA. GenomeScan use integrated approaches in database similarities while MORGAN uses decision trees and dynamic programming. GenScan and UNVEIL use Hidden Markov Model for the purpose. Genie use GHMM and SPLICEVIEW and SplicePredictor uses signal sensor methods. AAT use integrated approach in data similarities while DAGGER gene recognition is based on DAG shortest path. An equal number of coding and non-coding nucleotides are contained in the training sets used for various gene finding methods. But, it has been found that only about 2 % of human DNA is coding and the rest is non-coding. Recently, the promoter is considered as appearing in the intergenic region (immediately upstream of the

gene), and not overlapping with it, thus simplifying the reality. There is a requirement of the databases which are not redundant contain reliable and relevant annotations, and provide all necessary links to further data. Although there exists various problems in gene finding, the comparative genome approach seems to be a very promising not only in the field of gene prediction but also for the identification of regulatory sequences and the decoding of junk DNAs.

CHAPTER – 4

Hidden Markov Model for Splicing Junction Sites Identification in DNA Sequences

The main function of eukaryotic gene structure predictors is to pin point the locations of all start codons, stop codons, exons and introns in every gene and this step is considered as the rate-limiting step in the gene identification. In predicting splice site (which is the separation between exon and intron), the initial task is finding exons and introns. Splice site junction identification means the identification of donor site (5' boundary containing dinucleotide GT) and acceptor site (3' boundary containing dinucleotide AG) of introns [31-33, 62]. The success in gene prediction largely depend on the accuracy in finding the splice site junction, and thus, the removal of the introns from the DNA sequence to get coding regions is possible [62]. Bioinformatics unite the capability and knowledge of researcher from computational and biological areas, and locate a familiar stage for people from these backgrounds to work collectively to decipher gene annotation challenges [12]. Splice site in eukaryotic DNA sequence is shown in Figure 4.1.

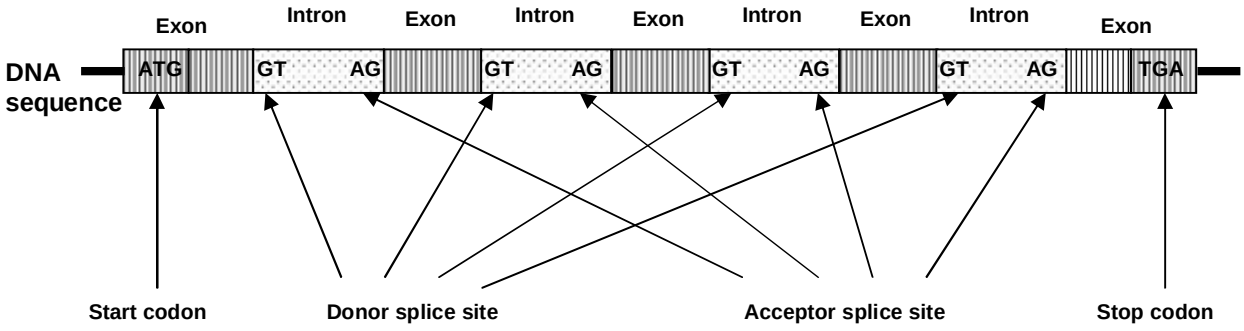

Figure 4.1 The splice sites (Donor site and Acceptor site) in eukaryotic DNA sequence.

A Hidden Markov Model (HMM) is a generalization of a Markov chain, in which each ("internal") state is not directly observable (hence the term hidden) but produces ("emits") an observable random output ("external") state, also called "emission", according to a given stationary probability law. An HMM consists of two stochastic processes. The first stochastic process is a Markov chain that is characterized by states and transition probabilities. The states of the chain are externally not visible, therefore "hidden". Another stochastic process will generate emissions, which is observable at every instant. It is dependent on state probability distribution. In case of HMM, the term "hidden" not indicates the parameter of the model, but it indicates the state of the Markov Chain [193].

4.1 Dataset Collection

To build reliable expanded Hidden Markov Model for the detection of human splice sites, high-quality datasets must be used. Splice site dataset is collected from the website http://www.fruitfly.org/sequence/human-datasets.html. There is a collection of 2381 true donor sites and 2381 true acceptor sites from a set of 462 annotated multiple-exon human genes. After removing junk sequence (splice sites that contained base positions not labeled with A, T, C, G but

with other symbols) there remained 2379 true donor sites and 2379 true acceptor sites, which were used as the true dataset. Hence, every acceptor site has a conserved AG di-nucleotides and every donor site has a conserved GT di-nucleotides. We also collected a large database of 300,062 false donor sites and 400,314 false acceptor sites from the 462 annotated genes and used it as the 'false dataset'.

Afterwards, we used a 12-fold cross-validation in our dataset to estimate the splice site detection accuracy of all the models. Cross validation is a standard experimental technique in which each model is verified by randomly partitioning the data into several subsets [42, 194]. We tested each subset (testing data) with the parameters trained by the other twelve subsets (training data) under the splice site model. After completing all these operations we took the average of the twelve predictive accuracy measures corresponding to the 12 testing/training data pair. Our proposed HMM system is trained with sequences which contains 2179 true site and 275,055 false sites, tested with 200 true sites and 25,005 false sites, for every time in the cross validation testing.

4.2. Proposed Models

In our proposed models for the identification of acceptor and donor splice sites, the splice site classification problem is subdivided into two – acceptor splice site classification and donor splice site classification. Two different models are constructed for the identification of acceptor splice sites and donor splice sites respectively. For simplicity, some basic notations have been provided in the 'list of symbols'.

4.2.1 Donor Site Hidden Markov Model (HMM) for 5' Splice Site

The nucleotide sequences must pass through this model to move from exon model to intron model. 11 nucleotide bases with GT are included in the conserved sequences which are almost consistent to all the donor sites [3, 60]-[195], an example of which is shown below:

ATGACGTGACC

The di-nucleotides GT are located in position 6 and 7 respectively. The exon-intron boundary occurs between stages 5 and 6, 1-3 is a start codon and so 4-5 are the part of exon and 6-11 are the part of intron. The location of G and T is 6 and 7 respectively in all the true 5' splice sites [196]. There is an 11-base non-donor sequence also present in which the G and T are located at position 6 and 7 respectively as a "false donor site". The motive of our proposed algorithm is to identify whether the given sequence (candidate) is a true donor site or a false donor site.

In our proposed donor HMM for identifying true donor site, 11 states and a set of transitions is used, which is represented as a digraph where vertices depicts the states and edges depicts the transitions. At each state, the model generates a base 'X' in {A, G, C, T} accordance with the state

and transition probabilities, with the exception of states 6 and 7. At the state 6, the donor HMM consistently generates base $X = G$, and at state 7, $X = T$. Every state Y is coupled with an output probability distribution, P(Y). We can simply observe that the value of P(Y) is 1 for states 6 and 7. The transition probability of HMM to make a transition is denoted as P(T). At state 5, every base has a constant transition, $P(T) = 1$, to the base **G** at state 6. Similarly, at state 6, the base **G** has a constant transition, $P(T) = 1$, to the base **T** at state 7. The donor site HMM for 5' splice site is shown in Figure 4.2.

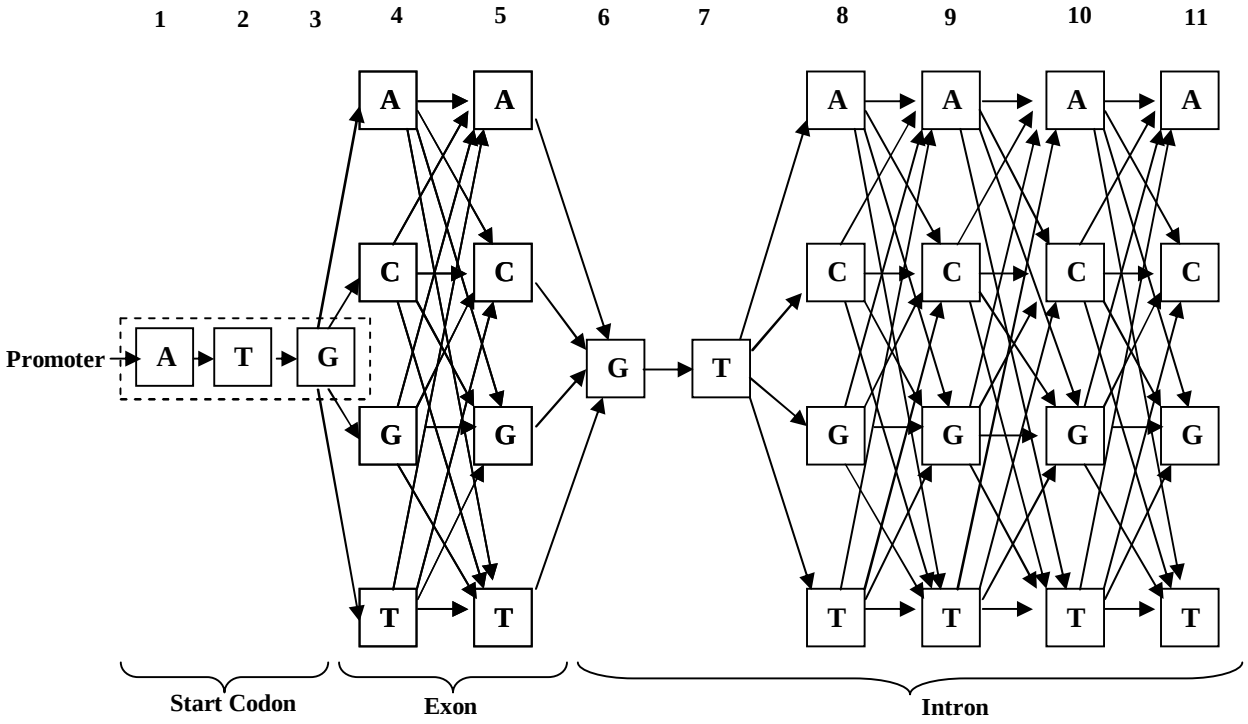

Figure 4.2 The Donor site HMM for 5' splice site.

4.2.2 Acceptor Site Hidden Markov Model (HMM) for 3' Splice Site

In DNA, the acceptor sites are the preserved boundary sequences at 3' splice sites which include 17 nucleotide bases with AG almost consistent to all acceptor sites [3, 197, 198], for example,

CTATCCTTCTCACAGGG

In an acceptor site, nucleotide A and G are located at positions 12 and 13 respectively [199]. There is also a non-acceptor sequence, in which the location of A and G are 12 and 13, which are considered as false acceptor site. Therefore, the proposed algorithm attempts to identify whether the given sequence is true donor site or false donor site. The acceptor HMM for 3' splice site is used to express the basic properties of true acceptor sites.

In our proposed acceptor HMM for identifying true acceptor site, 17 states and a set of transitions is used, which is represented as a diagram where vertices depicts the states and edges depicts the

transitions. In a nucleotide sequence, states 1 to 13 belong to an intron and state 14-17 belong to an exon. At each state, the model generates a base 'X' in {A, G, C, T} accordance with the state and transition probabilities, with the exception of states 12 and 13. At the state 12, the acceptor HMM consistently generates base $X = A$, and at state 13, $X = G$. Every state Y is coupled with an output probability distribution, P(Y). We can simply observe that the value of P(Y) is 1 for states 12 and 13. The transition probability of HMM to make a transition is denoted as P(T). At state 11, every base has a constant transition, $P(T) = 1$, to the base A at state 12. Similarly, at state 12, the base **G** has a constant transition, $P(T) = 1$, to the base G at state 13. The acceptor site HMM for 3' splice site is shown in Figure 4.3.

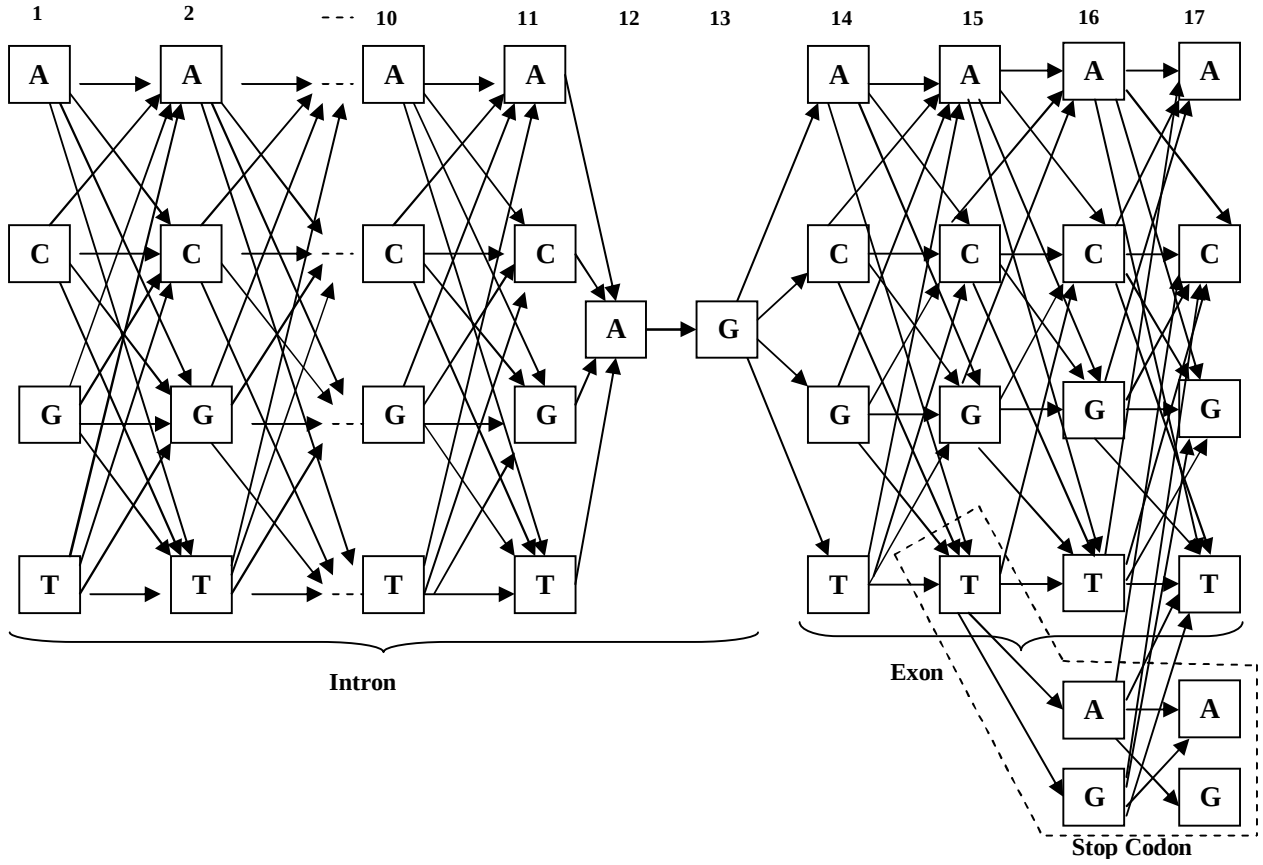

Figure 4.3 The Acceptor HMM for 3' splice site.

4.3 Unit Creation for Each Model

The number of false splice sites present is much larger than the number of true splice sites in vertebrate DNA sequence. To identify their difference, for Donor HMM System, we have created two programs – True Donor HMM Unit and False Donor HMM Unit. Similarly, for Acceptor HMM System, another two programs – True Acceptor HMM Unit and False Acceptor HMM Unit are created. The True splice site HMM Unit is the integration of True Donor HMM Unit and True Acceptor HMM Unit; and in a similar manner, False Splice site HMM Unit is the combination of

False Donor HMM Unit and False Acceptor HMM Unit. Here, we assume that C_{site} represents the given DNA sequence, and MOD_t and MOD_f denotes True splice site HMM element/Unit and false splice site HMM Unit respectively. For splice site predication, true site and false units are used to classify the given sequence into appropriate categories. We assume that the probability of donor site is $P(X=1|C_{site},MOD_t)$ when the given sequence is processed by True Donor HMM unit and the probability of the non-donor site as $P(X=0|C_{site},MOD_f)$ when it is processed by False Donor HMM Unit. The training data for true and false splice sites are used to give training to the true splice site HMM Unit and false splice site HMM Unit respectively. To calculate the result of C_{site}, initially we run True Donor HMM Unit to obtain the probability of being a donor site sequence and then, False Donor HMM Unit to obtain the probability of non-donor sequence. After comparing these values, our given C_{site} is assigned to false donor category or true donor category. Figure 4.4 and Figure 4.5 shows the creations of True Splice site HMM unit and False Splice site HMM unit.

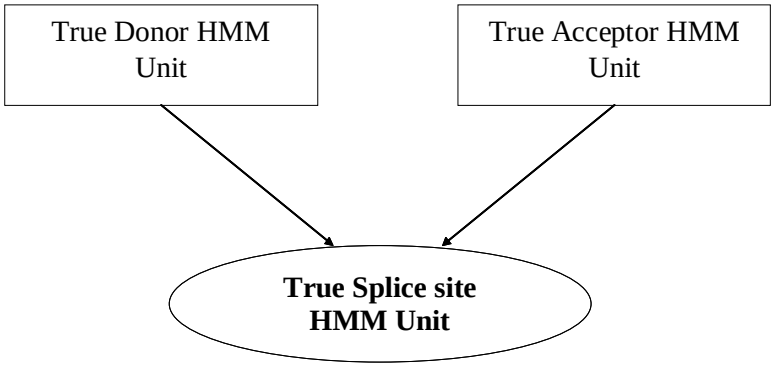

Figure 4.4 The True Splice site HMM Unit.

To train these HMM Units, we used modified expectation maximization (MEM) algorithm. In the basic EM algorithm, a set of unaligned sequence and a motif length are provided as input resulting in a probabilistic model for motif [200-203]. Also, each iteration consist of two steps namely expectation step (E-step) and the maximization step (M-step). But, many of pre-trained values, such as those in splice junction models, are fixed and can not be modified by the EM algorithm, whereas, as our dataset contains splicing junction sites of same length which may be aligned to each other, therefore, we developed the proposed MEM algorithm for training a HMM with fixed topology. In this MEM algorithm, we trained the module iteratively to get the maximum value of specificity, i.e. the fraction of correctly classified sites or until positive training data set, N^t and negative training data set N^f become empty with the condition that he value of sensitivity, S_n^{mem} during the training period remains constant.

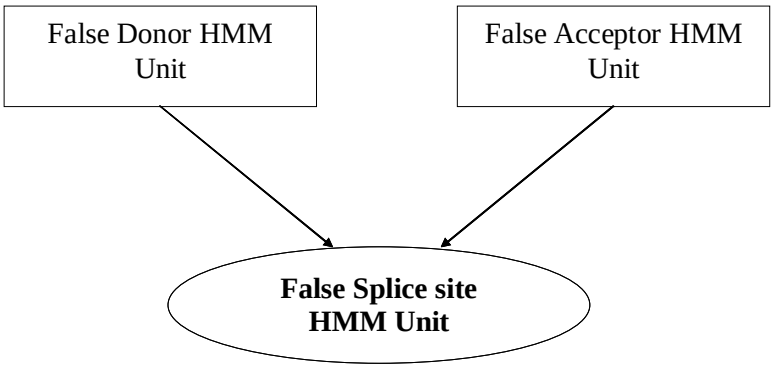

Figure 4.5 The False Splice site HMM Unit.

Assuming that all these sequences may be aligned to each-other; our designed Modified EM (MEM) algorithm works in the following manner: Initially, the value of all the transition probabilities P(T) and state probabilities P(Y) are set to 0 and the HMM Unit topology is constant. Then the first subset of positive training data (e.g. 120 sequences) is given as input to the True Donor HMM Unit; the numbers of the individual bases at each state and from present state to the next are recorded. Afterwards, the prior probabilities for all the states and transitions are calculated in the True Donor HMM Unit. After getting the prior probabilities, we provided another subset of positive training data to the True Donor Unit, and all the subsequent probabilities are re-adjusted. After this, we calculated the differences, *diff*, for all the probabilities between the earlier and subsequent probabilities. If some of the *diff* are larger than a predefined threshold value (THV), set the current posterior probabilities as the new prior probabilities, and the new data set is then run through the True Donor HMM Module again to further refine the probabilities. This training process is repeated until the changes in all probabilities in the True Donor HMM Unit are smaller than the THV. The False Donor HMM Unit is trained using the negative training data in the same way as for the True Donor HMM Unit.

4.4 Algorithms

Three efficient algorithms – Forward, Viterbi and Expectation Maximization (EM) are used for HMM computation. The proposed algorithms can be used mutually for the Donor HMM System and Acceptor HMM System. Initially, Acceptor HMM System and its related units are created, and then, the True Acceptor HMM Unit and False Acceptor HMM Unit are formed accordingly. The algorithms for the Donor HMM System are developed in the similar manner.

4.4.1 Training Algorithm

In the training algorithm, N represents the set of sequences which are arbitrarily selected from the positive and negatively training data sets, contains about 200 true acceptor sites and 19,000 false acceptor sites. The sequences in N are labeled as P_v if it is collected from positive training dataset

or N_v if it is from negative training dataset. Apart from N, the sequences in the remaining dataset are labeled as N^t if it is taken from positive training data set and N^f if taken from the negative training data set and, P is the subset of N. The sum of sequences in N^t and N^f exceeds about eleven times the number of sequences in N.

The algorithm converges in the training phase by advancing iteratively. A few sequences from N^t and N^f are removed by the algorithm at each iteration, and inputs those into True Acceptor HMM Unit and False Acceptor HMM Unit. Then, the algorithm determines the sequences those are located in the subset P. During the MEM training, let S_n^{mem} represents the sensitivity, which is the ratio between the number of true acceptor sites in P and the total number of true acceptor sites in N; and S_p^{mem} represents the specificity, which is the ratio between the number of true acceptor sites in P and the total number of sequences in P. Here, it is important to note that $P \subseteq$(belongs to) N and the goal of the MEM training is to train the Units repeatedly to get a maximal value of S_p^{mem} until N^t and N^f is emptied, provided S_n^{mem} remains constant. Here we have taken the value of S_n^{mem} = 0.92 for the purpose.

Specifically, S_{total} represents the total number of states in the Acceptor Model and $b_i \left(b_i \in \{A, G, C, T\}\right)$ be the base at state i, $1 \le i \le S_{total}$ and $tr_i\ (b_i, b_{i+1})$, $1 \le i \le S_{total} - 1$ be the transition from state i to state $i+1$. The topology for the Acceptor HMM System is fixed, and all of the transition probabilities and state probabilities are initialized to random values. Then we selected one twelfth of the sequences from N^t and provided as input into the True Acceptor HMM Unit. At the same time, one twelfth of the sequences from N^f are selected and these are fed as input into False Acceptor HMM Unit. The number of the individual bases b_i and the number of individual transitions from one state to the next state, $tr_i\ (b_i, b_{i+1})$ are recorded at each state. Then we calculated the post probabilities for all the states and transitions in True Acceptor HMM Unit and finally, the False Acceptor HMM Units are computed. Considering $T^{(t)}tr_i\ (b_i, b_{i+1})$ as the total number of transitions from a base b_i at state i to a base b_{i+1} at state $i+1$ in True Acceptor HMM Unit and, $T_{in}^{(t)}$ be the total number of true acceptor sites that have been input into True Acceptor HMM Unit, the state transition probabilities, $f\ tr_i^{(t)}(b_i, b_{i+1})$, in True Acceptor HMM Unit can be calculated from the following equation:

$$f\ tr_i^{(t)}(b_i, b_{i+1}) = \frac{T^{(t)}tr_i(b_i, b_{i+1})}{T_{in}^{(t)}}. \qquad (1)$$

Similarly, if $T^{(f)}\,tr_i\,(b_i, b_{i+1})$ is the total number of transitions from a base b_i (at state i) to a base b_{i+1} (at state $i+1$) in False Acceptor HMM Unit and $T_{in}^{(f)}$ is the total number of false acceptor sites that have been input into False Acceptor Unit, then, the state transition probabilities, $f\ tr_i^{(f)}(b_i, b_{i+1})$ in False Acceptor Unit can be calculated from the following equation:

$$f\ tr_i^{(f)}(b_i, b_{i+1}) = \frac{T^{(f)} tr_i(b_i, b_{i+1})}{T_{in}^{(f)}} \qquad (2)$$

Subsequently, all sequences contained in N, which are unlabeled, are considered as input to the True Acceptor and False Acceptor HMM Units. Let $P\left(\text{True} \mid Y,\ N^{(t)}\right)$ represents the probability of a sequence Y (acceptor sequence) in set N and $P\left(\text{True} \mid Y,\ N^{(f)}\right)$, the probability of Y (non-acceptor sequence). In order to calculate $P\left(\text{True} \mid Y,\ N^{(t)}\right)$, the probability of sequence Y must be known by using True Acceptor HMM Unit, which can be computed as follows:

$$p(Y \mid True, N^{(t)}) = \prod_{i=1}^{S_{total}-1} ftr_i^{(t)}(b_i, b_{i+1}),\ b_i \in \{A,\ G,\ C,\ T\}. \quad (3)$$

The proposed MEM algorithm uses Bayesian Theorem for which the basic equations are as follows:

According to Bayesian Theorem

$$p(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} \qquad (4)$$

Now, for calculating $P\left(\text{True} \mid Y,\ N^{(t)}\right)$ from $P\left(Y \mid \text{True}, N^{(f)}\right)$,

$$P(True \mid Y, N^{(t)}) = \frac{P(Y \mid True, N^{(t)})P(True)}{P(Y)} \qquad (5)$$

where $P(True) =$ prior probability (assumed to be a constant),

$P(Y) =$ product of the individual base probabilities in the sequences (see eq. 6 and 7)

$P(Y)$ can be derived for calculating $P\left(\text{True} \mid Y,\ N^{(t)}\right)$,which is written as

$$P(Y) = \prod_{i=1}^{S_{total}} P(b_i \mid True, N^{(t)}). \qquad (6)$$

The product of discrete base probabilities $P(Y)$ in the in the sequences can be derived for calculating $P\left(\text{False} \mid Y,\ N^{(f)}\right)$ as written

$$P(Y) = \prod_{i=1}^{S_{total}} P(b_i \mid False, N^{(f)}), \tag{7}$$

Similarly, equations can be derived for calculating $P\left(\text{False} \mid Y,\ N^{(f)}\right)$ as follows:

$$P(Y \mid False, N^{(f)}) = \prod_{i=1}^{S_{total}-1} ftr_i^{(f)}(b_i, b_{i+1}),\ b_i \in \{A,\ G,\ C,\ T\}, \tag{8}$$

$$P(False \mid Y, N^{(f)}) = \frac{P(Y \mid False, N^{(f)})P(False)}{P(Y)} \tag{9}$$

Assuming the probability ratio of sequence Y in the dataset N is represented by *pr*

$$pr = \frac{P(True \mid Y, N^{(t)})}{P(False \mid Y, N^{(f)})}. \tag{10}$$

Once the *pr* is calculated for each sequence in set N, then the sequences in set N is sorted in the descending order according to their respective *pr* values. If the total number of positive sequences in set N is S_{pt}, we select the *pr* value for $S_{pt} * S_n^{mem}$ th positive sequence and use that value as the positive lower bound, denoted by L_b. The sensitivity S_n^{mem} of 200 positive sequences in set N is 0.92, so L_b is the *pr* value of the 184th positive sequence. A sequence Y $\in$ N into set P is assigned by the MEM algorithm if the *pr* value for $Y \geq L_b$. Let $T_{(P+N)}$ be the number of positive sequences in set N. Assume that the number of positive sequences in N that are assigned into set P are $T_{(TP)}$. Then, sensitivity during the MEM training will be given by

$$S_n^{mem} = \frac{T_{(TP)}}{T_{(P+N)}}. \tag{11}$$

And, let $T_{(pp)}$ be the total number of sequences in N that are assigned into P. Then, by definition, specificity during the MEM training will be given by

$$S_P^{mem} = \frac{T_{(TP)}}{T_{(PP)}}. \tag{12}$$

To increase S_p^{mem}, the entire probabilities are adjusted in the re-estimation procedure hidden in the Donor Model Acceptor System and the new sequences in N^t and N^f are chosen and removed. These sequences are then run through True Acceptor HMM Unit and False Acceptor HMM Unit again and the probabilities are further refined. This process is repeated until the value of S_p^{mem} is maximized or the value of N^t and N^f become zero. Now, the positive lower bound L_b that

maximizes S_p^{mem} will be considered as output and used in the detection phase for splicing junction sites. In the training period, MEM algorithm is used, which is depicted in Pseudocode 4.1.

INPUT:

Untrained HMM site unit (including a true site unit and a false site unit);

Positive training data set, N^t;

Negative training data set, N^f;

MEM testing data set, N;

OUTPUT:

Fully trained HMM site unit and L_b;

ALGORITHM:

max:= false ;

do begin

max := true ;

if N^t is not empty **then begin**

remove one twelfth of the sequences from N^t and input them into the true site unit;

for $i = 1$ **to** $S_{total} - 1$

calculate $ftr_i^t(b_i, b_{i+1})$ as in Equation (1);

end;

if N^f is not empty **then begin**

remove one twelfth of the sequences from N^f and input them into the false site module;

for $i = 1$ **to** $S_{total} - 1$

calculate $ftr_i^f(b_i, b_{i+1})$ as in Equation (2);

end;

for each sequence $Y \in N$ **do begin**

calculate $P\left(\text{True} \mid Y, N^{(t)}\right)$ as in Equation (5);

calculate $P\left(\text{False} \mid Y, N^{(f)}\right)$ as in Equation (9);

calculate *pr* as in Equation (10);

end;

select L_b ;

calculate S_p^{mem} according to L_b ;

if (S_p^{mem} is not maximum) or (either N^t or N^f is non-empty) **then**

max := false ;

end;

while max

Pseudocode 4.1 The MEM Algorithm in training phase.

4.4.2 Splice Site Junction Detection Algorithm

The implication of a candidate acceptor site is a 17-base sequence section with the bases A and G at locations 12 and 13 respectively [3]. A section $C_{site,}$ of 17-bases (referred as $b_1, b_2, ..., b_{17}$ respectively) is taken as the input of the site detection algorithm, which is extracted from a genomic DNA sequence Y. The indication whether the C_{site} starting at position i of the genomic DNA sequence Y is a true acceptor site or not is estimated/verified by the output of the site detection algorithm, which is a flag given by $FLAGHMM_i$.

Now, considering that $f\ tr_j^{(t)}(b_j, b_{j+1})$ be the probability of a transition from base b_j to base b_{j+1} $(1 \le j \le 16)$, of C_{site} using True Acceptor HMM Unit, a flag variable F may be defined as 1 if C_{site} belongs to a true site category, otherwise, it will be 0. Let L be the length of the candidate site C_{site} (L is 17 for acceptor sites and 11 for donor sites) and $P\left(C_{site} \mid F=1,\ N^{(t)}\right)$ be the probability of the candidate site C_{site} with the condition that it is an acceptor site processed by True Acceptor HMM Unit, then

$$P(C_{site} \mid F=1, N^{(t)}) = \prod_{j=1}^{n-1} ftr_j^{(t)}(b_j, b_{j+1}),\ \ b_i \in \{A,\ G,\ C,\ T\} \quad (13)$$

Therefore, according to Bayesian theorem,

$$P(F=1 \mid C_{site}, N^{(t)}) = \frac{P(C_{site} \mid F=1, N^{(t)})P(F=1)}{P(C_{site})}. \quad (14)$$

$P(F=1)$ can be treated as a constant [4] while examining a set of sequences to detect true acceptor sites. Then, $P(C_{site})$, the product of the individual base probabilities for $b_1, b_2, ..., b_n$ will be

$$P(C_{site}) = \prod_{j=1}^{n} P(b_j \mid F=1, N^{(t)}),\ \ b_i \in \{A,\ G,\ C,\ T\}. \quad (15)$$

In the same way as we've followed for True Acceptor HMM Unit, the False Acceptor HMM Unit can be calculated by eq. (16), (17), and (18) with flag variable, F = 0 and $N^{(t)}$ replaced by $N^{ft)}$.

The probability $P\left(C_{site} \mid F=0,\ N^{(f)}\right)$ of the candidate site C_{site} with the condition that it is an acceptor site processed by False Acceptor HMM Unit is

$$P(C_{site} \mid F=0, N^{(f)}) = \prod_{j=1}^{n-1} ftr_j^{(f)}(b_j, b_{j+1}),\ \ b_i \in \{A,\ G,\ C,\ T\}. \quad (16)$$

The False Acceptor HMM Unit is used to compute $P\left(F=0 \mid C_{site}, N^{(f)}\right)$, which is the probability of C_{site} being a false acceptor site with the condition that it is processed by False Acceptor HMM Unit, can be written as

$$P(F=0 \mid C_{site}, N^{(f)}) = \frac{P(C_{site} \mid F=0, N^{(f)})P(F=0)}{P(C_{site})}, \qquad (17)$$

The probability $P\left(F=0\right)$ can be treated as a constant, while examining a set of sequences to detect false acceptor sites. Then, $P\left(C_{site}\right)$, the product of the individual base probabilities for b_1, b_2,..., b_n will be

$$P(C_{site}) = \prod_{j=1}^{n} P(b_j \mid F=0, N^{(f)}), \; b_i \in \{A, \; G, \; C, \; T\} \qquad (18)$$

Now, provided the candidate acceptor site C_{site} starting at position i in the DNA sequence Y is given, the proposed algorithm will find the two most likely sets of states through the two HMM Units for C_{site}. Then, the algorithm will calculate $P\left(F=1 \mid C_{site}, N^{(t)}\right)$ and $P\left(F=0 \mid C_{site}, N^{(f)}\right)$. Based on the scoring function, a score *sr* is assigned to the candidate site, as shown below:

$$sr = \frac{P(F=1 \mid C_{site}, N^{(t)})}{P(F=0 \mid C_{site}, N^{(f)})}. \qquad (19)$$

After evaluating *sr* and L_b, a flag $FLAGHMM_i$, is assigned to the candidate site C_{site} and calculated. If $sr \geq L_b$, the value of $FLAGHMM_i$ will be 1 and the C_{site} is considered as true acceptor site, and if it is 0, then it is considered as a false acceptor site. The Acceptor Splice site junction classification algorithm is given in Pseudocode 2.

INPUT:

A candidate acceptor site C_{site} of an unlabelled genomic DNA sequence starting at position i

OUTPUT:

/* $FLAGHMM_i$ is a flag indicating whether C_{site} is a true acceptor site or not. */

$FLAGHMM_i$;

ALGORITHM:

Calculate probability of transition of C_{site} using True Acceptor HMM Unit

$P(F=1 \mid C_{site}, N^{(t)})$ as in equation (14);

by calculating probability of transition of C_{site} using False Acceptor HMM Unit

$P(F=0 \mid C_{site}, N^{(f)})$;

Calculate *sr* as in Equation (19);

Calculate $FLAGHMM_i$;

Pseudocode 4.2 Acceptor Splice site junction classification algorithm

4.5 Results and Discussion

The classification performance of the models is measured in terms of their sensitivity S_n^{true} (TPR), and specificity S_n^{false} [5, 204]. In the classification performance, TP, TN, FP, and FN stand for true positive rate, true negative rate, false positive rate, and false negative rate respectively [3, 205] (As defined Table 3.4). The state transition probabilities for the Acceptor and the Donor HMM Systems are shown in tables 4.1 to 4.4.

Our proposed system increased the differences between the true splice site and false splice sites to the maximum as verified from the results which are shown in Tables 4.5 and 4.6. A 12-fold cross validation technique is applied to identify the splice site prediction accuracy, and the average results for all the 12 test sets are shown. Their classification efficiency was evaluated by various quantitative variables - (i) true positive (TP): the number of correctly classified splice site, (ii) true negative (TN): the number of correctly classified non-splice site, (iii) false positive (FP): the number of incorrectly classified splice site, and, (iv) false negative (FN): the number of incorrectly classified non-splice site. The sensitivity S_n^{true} or True Positive Rate (TPR), defined as the fraction of correctly classified true acceptor (or true donor) sites among the total number of true acceptor (or true donor) sites in the test data, is shown in following equation:

$$\text{TPR or } S_n^{true} = \frac{TP}{TP + FN} \qquad (20)$$

Analogously, the specificity S_n^{false} is defined as the fraction of correctly classified false acceptor (or false donor) sites among the total number of false acceptor (or false donor) sites in the test data, i.e.

$$S_n^{false} = \frac{TN}{TN + FP}, \qquad (21)$$

Table 4.1 Various State Transition Probability values for True Acceptor HMM Unit.

State Transition Probabilities $ftr_i^{(t)}(b_i, b_{i+1})$																
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
$A \rightarrow A$	0.011	0.011	0	0.01	0.021	0.01	0	0.011	0.01	0	0.019	null	null	0.021	0.01	0.001
$A \rightarrow G$	0.012	0.012	0	0.012	0	0	0	0	0	0	null	1.001	null	0.021	0	0.01
$A \rightarrow C$	0.041	0.041	0.031	0.021	0.031	0.031	0.031	0.031	0.041	0.031	null	null	null	0.021	0.171	0.12
$A \rightarrow T$	0.052	0.031	0.031	0.032	0.032	0.031	0.051	0.011	0.031	0.011	null	null	null	0	0.041	0.011
$G \rightarrow A$	0.022	0.01	0.021	0.011	0.011	0.012	0	0.011	0	0.011	0.011	null	0.031	0.01	0	0.024
$G \rightarrow G$	0.031	0.032	0.041	0.032	0.032	0.021	0.031	0.032	0.011	0.022	null	null	0.481	0.031	0	0.23
$G \rightarrow C$	0.043	0.043	0.051	0.031	0.032	0.031	0.031	0.052	0.032	0.011	null	null	0.141	0.022	0.23	0.11
$G \rightarrow T$	0.052	0.051	0.042	0.052	0.052	0.041	0.062	0.041	0.041	0.011	null	null	0.091	0.011	0.031	0.1
$C \rightarrow A$	0.042	0.032	0.032	0.032	0.031	0.051	0.021	0.041	0.021	0.031	0.781	null	null	0.131	0.011	0.03
$C \rightarrow G$	0.031	0.031	0.011	0.022	0.022	0.011	0.031	0.011	0.011	0.011	null	null	null	0.061	0	0.021
$C \rightarrow C$	0.142	0.14	0.142	0.139	0.141	0.162	0.169	0.182	0.24	0.249	null	null	null	0.162	0.241	0.302
$C \rightarrow T$	0.187	0.151	0.171	0.179	0.182	0.162	0.149	0.191	0.169	0.21	null	null	null	0.082	0.071	0.224
$T \rightarrow A$	0.022	0.021	0.032	0.021	0.022	0.021	0.021	0.031	0.011	0.021	0.181	null	null	0.061	0	0.02
$T \rightarrow G$	0.072	0.071	0.052	0.062	0.049	0.092	0.062	0.04	0.031	0.039	null	null	null	0.152	0	0.069
$T \rightarrow C$	0.15	0.153	0.179	0.172	0.182	0.162	0.191	0.191	0.182	0.139	null	null	null	0.121	0.142	0.1
$T \rightarrow T$	0.151	0.192	0.171	0.159	0.201	0.172	0.142	0.141	0.161	0.21	null	null	null	0.11	0.061	0.15

Table 4.2 Various State Transition Probability values for False Acceptor HMM Unit.

State Transition Probabilities $ftr_i^{(f)}(b_i, b_{i+1})$																
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
$A \to A$	0.051	0.051	0.051	0.062	0.062	0.062	0.062	0.082	0.052	0.062	0.279	null	null	0.052	0.042	0.043
$A \to G$	0.081	0.101	0.101	0.082	0.11	0.082	0.082	0.09	0.13	0.111	null	1	null	0.091	0.061	0.076
$A \to C$	0.052	0.051	0.042	0.052	0.071	0.031	0.061	0.061	0.041	0.041	null	null	null	0.051	0.042	0.051
$A \to T$	0.052	0.041	0.042	0.031	0.052	0.041	0.042	0.031	0.031	0.031	null	null	null	0.031	0.011	0.044
$G \to A$	0.081	0.071	0.082	0.11	0.071	0.082	0.08	0.071	0.091	0.071	0.359	null	0.191	0.041	0.071	0.078
$G \to G$	0.101	0.101	0.134	0.11	0.11	0.132	0.121	0.121	0.121	0.191	null	null	0.43	0.122	0.122	0.102
$G \to C$	0.081	0.071	0.072	0.071	0.071	0.082	0.071	0.051	0.061	0.071	null	null	0.181	0.051	0.083	0.04
$G \to T$	0.061	0.051	0.051	0.062	0.052	0.052	0.062	0.051	0.052	0.041	null	null	0.21	0.191	0.032	0.2
$C \to A$	0.081	0.071	0.072	0.071	0.061	0.071	0.071	0.071	0.072	0.051	0.282	null	null	0.042	0.072	0.065
$C \to G$	0.031	0.021	0.041	0.032	0.032	0.032	0.022	0.031	0.041	0.031	null	null	null	0.031	0.031	0.063
$C \to C$	0.061	0.071	0.061	0.061	0.061	0.081	0.071	0.082	0.071	0.062	null	null	null	0.082	0.102	0.078
$C \to T$	0.081	0.071	0.079	0.061	0.071	0.062	0.071	0.061	0.062	0.062	null	null	null	0.061	0.021	0.068
$T \to A$	0.042	0.032	0.022	0.042	0.032	0.031	0.041	0.031	0.031	0.031	0.082	null	null	0.032	0.091	0.04
$T \to G$	0.071	0.101	0.081	0.091	0.091	0.092	0.071	0.081	0.091	0.071	null	null	null	0.071	0.151	0.122
$T \to C$	0.042	0.061	0.052	0.042	0.042	0.051	0.051	0.062	0.031	0.041	null	null	null	0.041	0.052	0.067
$T \to T$	0.042	0.041	0.041	0.042	0.042	0.042	0.031	0.041	0.041	0.031	null	null	null	0.032	0.011	0.066

Table 4.3 Various State Transition Probability values for True Donor HMM Unit.

State Transition Probabilities $ftr_i^{(t)}(b_i, b_{i+1})$										
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$A \to A$	0.201	0.33	0.2	0.041	null	null	null	0.04	0.012	0.01
$A \to G$	0.036	0.061	0.053	0.513	0.082	null	null	0.063	0.032	0.022
$A \to C$	0.021	0.061	0.022	0.011	null	null	null	0.021	0.01	0.011
$A \to T$	0.011	0.062	0.04	0.042	null	null	null	0.022	0.022	0.021
$G \to A$	0.084	0.372	0.131	0.02	null	null	null	0.01	0.12	0.11
$G \to G$	0.069	0.041	0.02	0.111	0.81	null	null	0.101	0.13	0.131
$G \to C$	0.054	0.02	0.031	0	null	null	null	0.01	0.12	0.112
$G \to T$	0.087	0.01	0.021	0.01	null	1.012	null	0	0.461	0.01
$C \to A$	0.024	0.02	0.227	0.021	null	null	null	0.02	0.011	0.01
$C \to G$	0.088	0	0.022	0.07	0.021	null	null	0.03	0	0.031
$C \to C$	0.062	0	0.041	0.011	null	null	null	0.021	0.021	0.01
$C \to T$	0.032	0.01	0.051	0.021	null	null	null	0.021	0.022	0.011
$T \to A$	0.025	0.01	0.021	0	null	null	0.501	0	0	0.01
$T \to G$	0.073	0.022	0.041	0.121	0.081	null	0.441	0.071	0.021	0.02
$T \to C$	0.011	0	0.032	0.01	null	null	0.031	0.01	0.011	0.01
$T \to T$	0.033	0	0.031	0.011	null	null	0.032	0	0.011	0

Table 4.4 Various State Transition Probability values for False Donor HMM Unit.

State Transition Probabilities $ftr_i^{(f)}(b_i, b_{i+1})$										
i	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
$A \to A$	0.062	0.051	0.081	0.081	null	null	null	0.061	0.062	0.06
$A \to G$	0.022	0.052	0.072	0.082	0.282	null	null	0.071	0.071	0.07
$A \to C$	0.055	0.041	0.051	0.02	null	null	null	0.051	0.042	0.04
$A \to T$	0.042	0.051	0.051	0.081	null	null	null	0.052	0.062	0.061
$G \to A$	0.066	0.091	0.071	0.071	null	null	null	0.052	0.062	0.061
$G \to G$	0.051	0.11	0.072	0.079	0.271	null	null	0.071	0.071	0.07
$G \to C$	0.044	0.072	0.061	0.021	null	null	null	0.051	0.052	0.051
$G \to T$	0.045	0.091	0.052	0.091	null	null	null	0.052	0.071	0.062
$C \to A$	0.008	0.061	0.081	0.081	null	null	null	0.071	0.072	0.07
$C \to G$	0.021	0.011	0.021	0.02	0.081	null	null	0.021	0.02	0.021
$C \to C$	0.072	0.071	0.071	0.02	null	null	null	0.071	0.071	0.07
$C \to T$	0.068	0.081	0.061	0.12	null	null	null	0.081	0.089	0.082
$T \to A$	0.053	0.043	0.051	0.041	null	null	0.181	0.049	0.051	0.05
$T \to G$	0.083	0.062	0.091	0.092	0.38	null	0.36	0.091	0.081	0.082
$T \to C$	0.069	0.061	0.072	0.021	null	null	0.21	0.69	0.062	0.06
$T \to T$	0.075	0.091	0.072	0.092	null	null	0.247	0.081	0.082	0.078

The similar calculations are also used for identifying false positive Rate (FPR) as the fraction of incorrectly classified true acceptor (true donor) sites among the total number of false acceptor (or false donor) sites in the test dataset, i.e.

$$FPR = \frac{FP}{TN + FP}, \qquad (22)$$

Accuracy (ACC) is a parameter of the test which is the proportion of the candidate site in the given test data those are classified correctly (or accurately) and gives a fair idea that whether the proposed system can classify the true and false splice sites into right categories. Accuracy is calculated by the formula:

$$ACC = \frac{TN + TP}{TN + TP + FN + FP} \qquad (23)$$

Acceptor HMM System can correctly identify 95% of the true acceptor sites and 92% of the false acceptor sites in the test data, as shown in Table 4.5. Similarly Donor HMM System can able to predict 95% of the true donor sites and 97% of the false donor sites in the test data set, as depicted in Table 4.6. Accuracy of the candidate acceptor sites is 92%, and for donor sites value is 97% [6]. Their accuracy performances are shown in Figures 4.7 and 4.8 respectively.

The result of our proposed HMM System (Acceptor HMM System and Donor HMM System) on the test data were compared with NNSplice (http://www.fruitfly.org/seq_tools/splice.html), GENIO (http://genio.informatik.uni-stuttgart.de/GENIO) using our test data for the comparison. Both of these splice site predictors offer a web page (already mentioned) where the DNA sequences can be submitted for the generating the results of the data, therefore, we submitted our dataset to each of the websites, and used the default parameters to predict the results. Table 4.7 shows the overall comparison of our proposed HMM System with two of the systems – NNSplice and GENIO.

Table 4.5 The Acceptor HMM performance for 3' Splice site prediction

Set	No of true acceptor	No of false acceptor	TP	FP	TN	FN	Sensitivity S_n^{true}	Specificity S_n^{false}	FPR	Accuracy *ACC*
1	208	19782	190	1028	18754	18	0.9134	0.9480	0.0519	0.9476
2	200	21531	184	1273	20258	16	0.92	0.9408	0.0591	0.9406
3	209	21001	195	1299	19702	14	0.9330	0.9381	0.0618	0.9380
4	210	18965	197	1301	17664	13	0.9380	0.9313	0.0686	0.9314
5	203	18966	193	1297	17669	10	0.9507	0.9316	0.0683	0.9318
6	200	22000	191	1598	20402	9	0.9550	0.9273	0.0726	0.9276
7	208	21343	199	1587	19756	9	0.9567	0.9256	0.0743	0.9259
8	213	21457	206	1573	19884	7	0.9671	0.9266	0.0733	0.9270
9	206	20876	199	1578	19298	7	0.9660	0.9244	0.0755	0.9248
10	212	18790	206	1485	17305	6	0.9716	0.9209	0.0790	0.9215
11	209	17986	203	1490	16496	6	0.9712	0.9171	0.0828	0.9177
12	209	18003	203	1498	16505	6	0.9712	0.9167	0.0832	0.9174
Average							0.9512	0.9290	0.0709	0.9293

Table 4.6 The Donor HMM performance for 5' Splice site prediction

Set	No of true donor	No of false donor	TP	FP	TN	FN	Sensitivity S_n^{true}	Specificity S_n^{false}	FPR	Accuracy *ACC*
1	208	16242	194	200	16042	14	0.9326	0.9876	0.0123	0.9869
2	200	15101	188	199	14902	12	0.94	0.9868	0.0131	0.9862
3	209	16261	196	231	16030	13	0.9377	0.9857	0.0142	0.9851
4	210	13411	198	235	13176	12	0.9428	0.9824	0.0175	0.9818
5	203	12301	192	266	12035	11	0.9458	0.9783	0.0216	0.9778
6	200	15235	190	348	14887	10	0.95	0.9771	0.0228	0.9768
7	208	17221	200	397	16824	8	0.9615	0.9769	0.0230	0.9767
8	213	15815	205	382	15433	8	0.9624	0.9758	0.0241	0.9756
9	206	15863	199	399	15464	7	0.9660	0.9748	0.0251	0.9747
10	212	13123	206	378	12745	6	0.9716	0.9711	0.0288	0.9712
11	209	14331	204	452	13879	5	0.9760	0.9684	0.0315	0.9685
12	209	14354	209	487	13867	5	0.9766	0.9660	0.0339	0.9662
Average							0.9552	0.9776	0.0223	0.9773

Table 4.7 Accuracy of acceptor and donor splice site detection compared for HMM System, NNSplice and GENIO on the human test dataset.

	Splice site predictor	Sensitivity	Specificity	False Positive Rate (FPR)
Acceptor Site	HMM System (all data)	0.9512	0.9290	0.0709
	NNSplice (all data)	0.6419	0.9483	0.05165
	GENIO (all data)	0.7959	0.9523	0.0476
Donor Site	HMM System (all data)	0.9552	0.9776	0.0223
	NNSplice (all data)	0.7116	0.9367	0.0633
	GENIO (all data)	0.8624	0.9406	0.0594

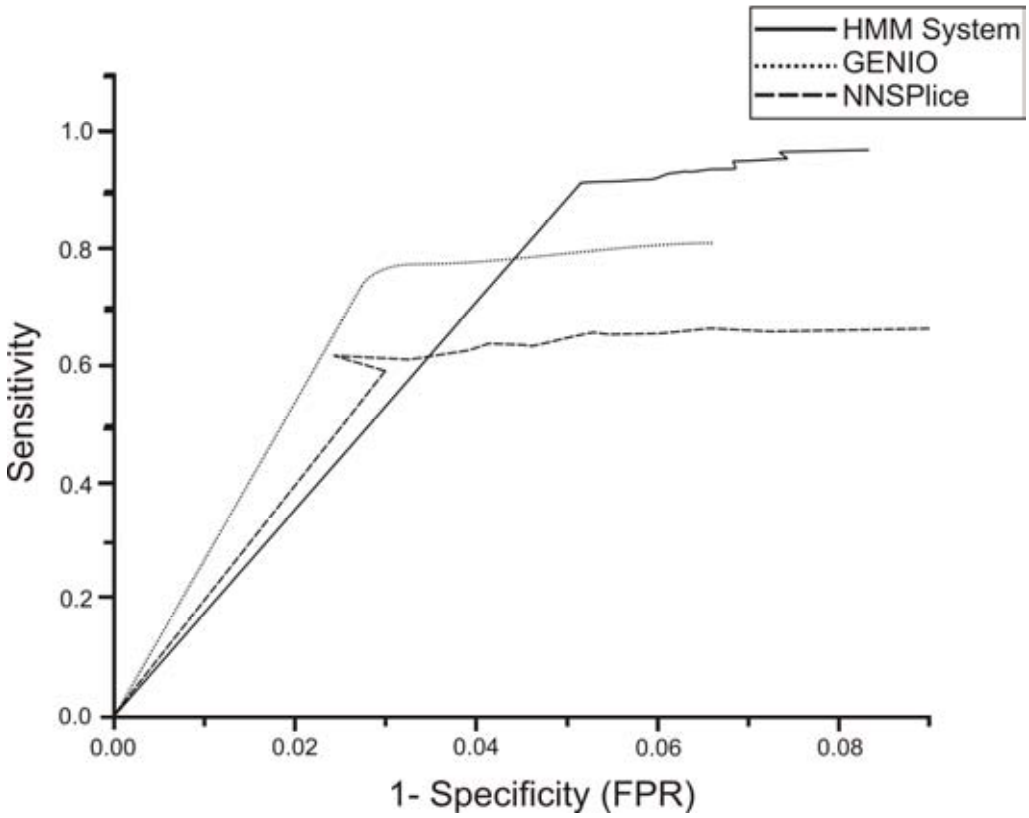

Figure 4.6 Receiver operating characteristic (ROC) curve showing the comparison of performance between HMM System, GENIO and NNSplice Acceptor test dataset.

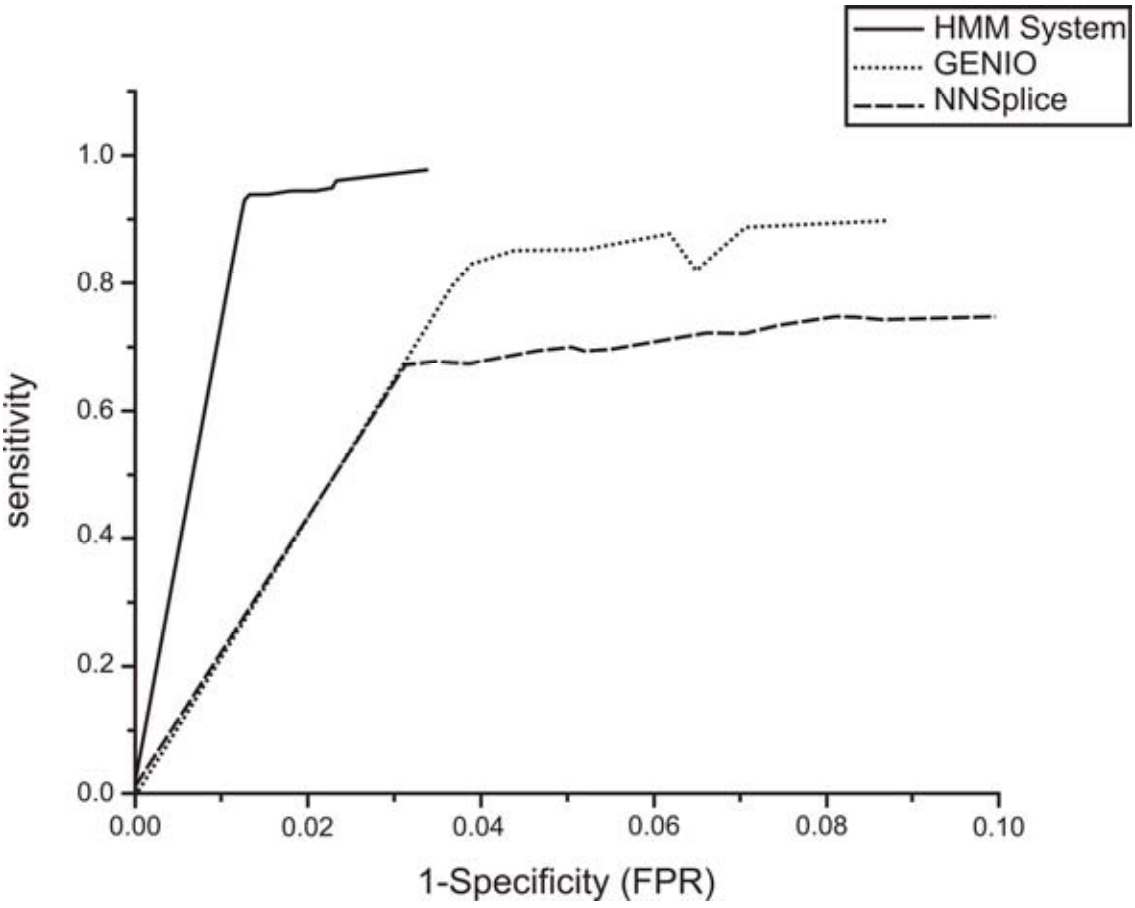

Figure 4.7 Receiver operating characteristic (ROC) curve showing the comparison of performance between HMM System, GENIO and NNSplice Donor test dataset.

CHAPTER – 5

Hybrid Approach using SVM and MM2 in Splice Site Junction Identification

Identifying the presence of splice site within DNA sequence is the initial step in accurately prediction of gene structure. Biology researchers have extensively studied the laboratory procedures such as PCR on cDNA libraries, northern blot, sequencing, etc. to identify the accurate gene structure. But, due to presence of large number of hidden genes, it is impossible to describe all of them by using experiments only in the lab. Hence, lab experiments are combined with bioinformatics approaches in the modern researches [206]. Numerous bioinformatics and computational approaches have been applied for gene prediction with the help of gene splicing. Some of the examples include probabilistic approaches, support vector machine and neural network approaches, discriminant analysis and the information theoretic approaches.

In the process called splicing, introns are removed and exons are retained in the mRNA and the reactions in splicing process are catalyzed by spliceosome. Within the intron, an acceptor site (3' end of the intron) and a donor site (5' end of the intron) are essential for splicing. The splice donor site includes invariant sequence GT at the 5' end of the intron with a larger and less preserved region. The splice acceptor site at the 3’ end of the intron terminates the intron with nearly invariant AG sequence (refer Figure 4.1). This is known as GT–AG law [207]. And as mentioned earlier, the identification of these acceptor and donor sites (splice sites prediction) is a crucial step in the gene identification process.

The proposed method, Second Order Markov Model Feature – Support Vector Machine (MM2F-SVM) consists of three stages – initial stage, in which a second order Markov Model (MM2) is used, i.e. feature extraction; intermediate, or the second stage in which principal feature analysis (PFA) is done, i.e. feature selection; and the final or the third stage, in which a support vector machine (SVM) with Gaussian kernel is used for final classification. While comparing this proposed MM2F-SVM model with the other existing splice site prediction programs, superior performance has been noticed for the proposed model.

5.1. Evaluation Datasets

We have accomplished several simulations to evaluate the performance of the proposed algorithms using three standard and publicly available splice site datasets.

We used publically available HS3D (Homo Sapiens Splice Sites data set) as our first dataset [194]. It is a dataset of intron, exons and splice sites extracted from Genbank, which was derived from Human genes. Length of each splice site sequence is 140bp.There are 2796 true donor and 271937 pseudo donor sites which contain ”GT” dinucleotides and there are 2880 true acceptor and 329374 pseudo acceptor sites which contain “AG” dinucleotides. In case of donor splice site GT dinucleotide is conserved at positions -71 and -72 of the sequences, and for acceptor splice site AG

is conserved at positions -69 and 70 of the sequences. The ration between the number of true splice site and pseudo splice site is 1:1 and we used this dataset to extract features for further modeling

The second dataset is known as DGSplicer [80]. This true dataset is created by extracting a collection of 2381 real acceptor sites and 2381 real donor sites from 462 annotated multiple-exon human genes from [208]. Two of the donor splice sites and one acceptor splice site were excluded from the collection to form a set of 2380 real acceptor sites and 2379 real donor sites as those three splice sites contained symbols other than A, C, G, and T. From 462 annotated human genes a large collection of 400314 pseudo acceptor sites and 283062 pseudo donor sites were collected and used as the false dataset. The window size for the donor splice site is 18 nucleotides {-9 to +9} with consensus GT at positions +1 and +2, which includes the last 9 bases of the exon and first 9 bases of the succeeding intron. The acceptor splice sites have a window of 36 nucleotides {-27 to +9} with consensus AG at positions -26 and -27, which includes the last 27 nucleotides of the intron and first 9 nucleotides of the succeeding exon.

To verify the effectiveness of our method, we performed additional evaluation on the third dataset named NN269 [209], it consists of 1324 confirmed true acceptor sites, 1324 confirmed true donor sites, 5552 pseudo acceptor sites and 4922 pseudo donor sites collected from 269 human genes. The window size of donor splice sites is 15 nucleotides {-7 to +8} with consensus GT at positions +1 and +2. This includes the last 9 bases of the exon and first 6 bases of the succeeding intron. The acceptor splice site have a window size of 90 nucleotides {-70 to +20} with consensus AG at positions -69 and -70. This includes the last 70 nucleotides of the intron and first 20 nucleotides of the succeeding exon, which is available at [210]. This data set is split into a training set and a testing set. The training dataset contains 1116 true acceptor, 1116 true donor, 4672 pseudo acceptor, and 4140 pseudo donor sites. The test data set contains 208 true acceptor sites, 208 true donor sites, 881 false acceptor sites, and 782 false donor sites. In NN269 donor splice sites, GT is conserved at positions 8 and 9 of the sequences; and for acceptor dataset, AG is conserved at positions 69 and 70 of the sequences.

5.2 Overview of the Projected Model

Our proposed model MM2F-SVM include a number of separate modules and sub modules that were anticipated to capture properties of DNA and specially designed to identify splice site. Splice site corresponds to the donor splice site and acceptor splice site, so splice site categorization process is subdivided into two classification modules – donor splice site classification and acceptor splice site classification process. Further, for the recognition of acceptor splice sites and donor splice sites, two different models are assembled which consist of three phases (or sub modules).Model employs several important aspects, these are (1) appropriate features encoding

scheme, (2) feature selection or ranking method, and (3) parameters optimization. The basic subsequent processing steps are outlined in the following:

1. **Feature extraction**: Positional probabilistic descriptions of different orders are constructed and a pool of candidate features is generated.
2. **Feature selection**: The discriminative power of each feature is assessed and the most informative features are selected using PFA.
3. **Classification step**: The SVM classifier is trained on the probabilistic parameters.

The proposed model architecture is described in Figure 5.1.

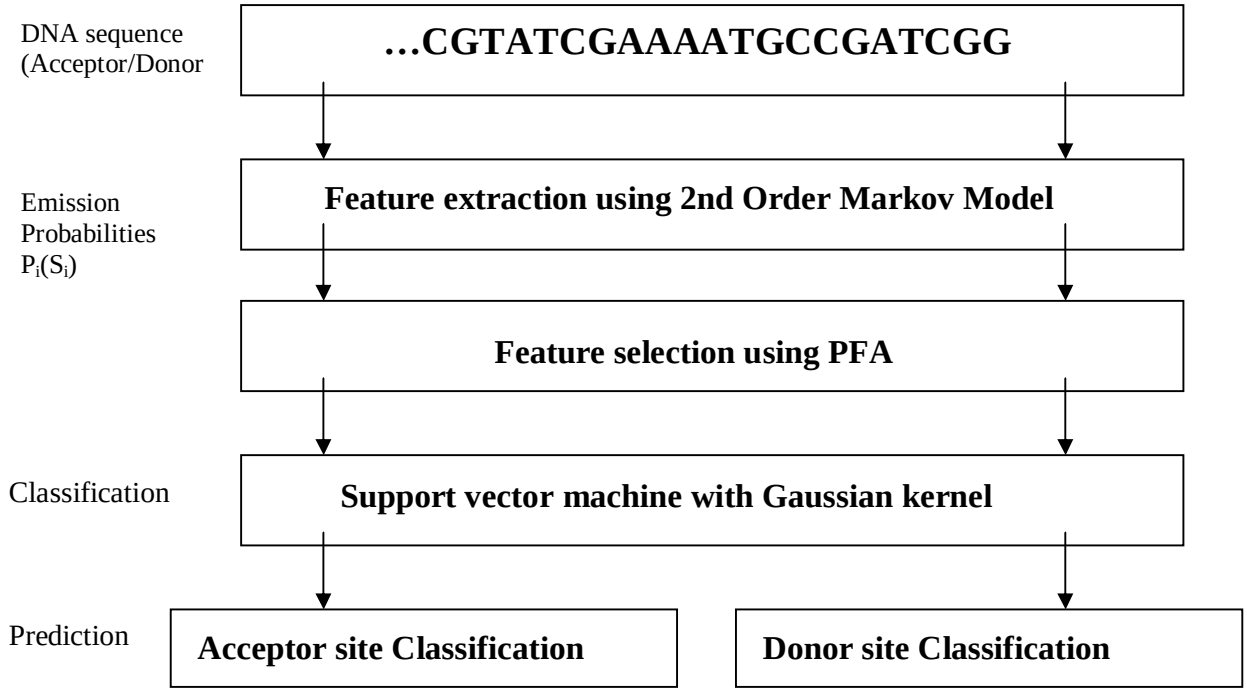

Figure 5.1 The MM2F-SVM Model. The input DNA sequence is preprocessed by 2nd order MM, PFA based feature selection. An SVM with Gaussian kernel function takes these parameters as its input for the splice site prediction.

5.3 Feature Extraction

In Markov process, the probability of the given condition in the given instant is likely to be presumed from information about the previous conditions [211]. A Markov chain represents a statistical system that undergoes transitions from one state to another between a limited or unlimited number of possible states and indicates next state depends only on the present state. A simple and existing Markov Model for DNA sequence is shown in Figure 5.2. The systems which follow this specific type of characteristic are called Markov property, behavior of the Markov chains are described by transition probability matrix. Every element of the matrix signifies probability of passage from a specific condition to a next state. In Markov model, we need a learning set of sequences on which these probabilities will be predictable. By using this technique

we can simply calculate the likelihood of the sequence, i.e. the probability that the sequence has been produced in accordance with this model.

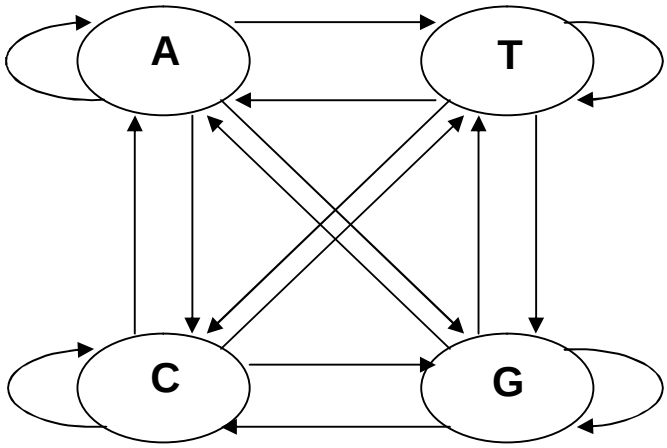

Figure 5.2 Markov Model for DNA Sequence

In a DNA sequence, every nucleotide corresponds to a state in the Markov chain, where observed state variables are derived from the symbol $X_{DNA} = \{A, T, G, C\}$. If length of the MM is L, then this probabilistic model describes the probability distribution of sequences of states $S_1, S_2 ..., S_L$ through transition probabilities, where transition probability $P(S_I = q \mid S_{I-1} = p)$ describes the probability of state $S_I = q$ (given, state $S_{I-1} = p$). A Markov model is used to capture the inter-dependencies among successive states in order to extract a set of probabilistic features [212]. If K is the order of MM, then likelihood of a sequence in this model is shown below:

$$P(S_1, S_2, ..., S_L) = \prod_{I=1}^{L} P_I(S_I \mid S_{I-1}) \qquad (1)$$

The ensuing model allocates different transition probabilities for each position. In the proposed method MM2F-SVM employs 2nd order Markov model (MM2) to built the probabilistic feature set, which has transition probabilities of format $P(S_I = q \mid S_{I-1} = p, S_{I-2} = v)$, can be described by the collection of parameters:

$$\left\{ P(S_I \mid S_{I-1}, S_{I-2}) : S_I, S_{I-1}, S_{I-2} \in X_{DNA}, I = 1, 2, .., L \right\} \qquad (2)$$

5.4 Feature Selection

The feature selection areas include text processing of gene expression array analysis, internet data, and combinational chemistry to improve prediction performance providing faster and more cost-effective predictors and better understanding of the underlying process that generated the data. For pattern classification, feature selection plays a very crucial role in the preprocessing step, aim to

deal with the storage space, dimensionality reduction problem and classification time, to improve the understanding of the problem as well as results' interpretation [213, 214].

Feature selection process is useful to provide their necessary mechanism that clean out redundant features to provide some biological interpretation of the incorporated features. In this framework, feature selection in biological data can be employed in two different ways:

- By using Positional Feature Selection (PFS) [215] technique that identifies the best-fitting dependency length based on the discriminative power of each feature.
- Reducing feature set by selecting a subset of the original features that contains most of the essential information, using the same criteria as the PCA and method name is principal feature analysis (PFA)[216].

Positional Feature Selection (PFS): PFS recognizes the best-fitting dependency length by distinguishing each feature. It selects the optimal feature among those describing a specific position by comparing their discriminative power; from a set of positional models of different lengths. It is applicable to solve binary classification problems [215].The central model uses the F-score value as a selection criterion for the best-suitable feature per splice site [217].

Principal Feature Analysis (PFA): It is very much similar with methods such as principal component analysis (PCA) only variant is possible by choosing a subset of the original feature vector that retains the underlying discriminative information using the same optimality criteria as in PCA. Instead of identifying a projection of all features included to the original feature space to a lower dimensional space, PFA utilize the properties of the primary components to select a subset of the original features [216]. PFA consider the mutual information among the selected features. In this case, the source features are the second-order MMs and the outcome is the principal feature subset that competently characterizes the initial group of probabilistic parameters. The extracted components are separately studied for their statistical importance by performing the Wilcoxon rank sum test ($p < 0.05$).In which, it will test for equal medians gives an insight of the differences between positive and negative in-stances of each feature and does not imply any assumption on the distribution of the tested features.

In our proposed model we have selected Principal Feature Analysis (PFA) for feature selection process due to its better sensitivity as compared to Positional Feature Selection (PFS).

5.5 Classification

The fundamentals of Support Vector Machines (SVM) were studied extensively by Vapnik [69, 167, 168, 218]. The formulation of SVM uses the Structural Risk Minimization (SRM) principle [219], which is more superior to the conventional Empirical Risk Minimization (ERM) principle

used with usual neural networks. An SVM build one or multiple hyperplane in a high dimensional space. Better partition can be accomplish by using the hyperplane that has the largest distance to the nearest training data point of any class (functional margin). The basic rule in SVM classifier is that the generalization error gets reduced when the margin is high [70, 167, 220]. Figure 3.5 shows SVM with hyperplane and margin.

SVM uses hypothetical space of linear function in high dimensional feature space trained with a learning algorithm. Using the method of Lagrange multipliers, we can obtain the twin formulation, which is expressed in terms of the variable α_i .To solve optimization problem SVM classification is given by:

$$Maximize\ f(\alpha) = \sum_{I=1}^{N} \alpha_I - \frac{1}{2} \sum_{I=1}^{N} \sum_{J=1}^{N} \alpha_I \alpha_J Y_I Y_J K(X_I, X_J), \qquad (3)$$

$$\text{Subject to} \sum_{I=1}^{N} \alpha_I Y_J = 0,\ 0 \le \alpha_I \le C,\ I = 1,...N$$

In the above equation N is the number of training data, X is input vectors, Y defines class value that can be either -1 or 1 and C is trade off parameter for generalization performance. The dual formulation leads to an expansion of the weight vector in terms of the input examples:

$$w = \sum_{I=1}^{N} Y_I \alpha_I X_I \qquad (4)$$

Different data points of X_I for which $\alpha_I > 0$ are those points that are on the margin or within the margin when a soft margin SVM is used. These are known as support vectors.

Assuming a query DNA segment is D, the trained SVM classifies based on the decision function:

$$o(D) = sign\left[\sum_{I=T} \alpha_I y_I K(X_I, D) \right] \qquad (5)$$

where set of support vectors are represented by T.

For classification purpose we have used Gaussian RBF kernel with width $\sigma = 1$,where σ control the flexibility of the resulting classifier. Therefore, equation (5) becomes:

$$K_{\sigma}^{GaussianRBF}(X, D) = \exp(-\frac{1}{\sigma} \| X - D \|^2) \qquad (6)$$

After expanding, this equation becomes

$$K_{\sigma}^{GaussianRBF}(X, D) = \exp(-\frac{1}{\sigma} \left[\sum_{I=1}^{N} (X_I - D_I)^2 \right] \qquad (7)$$

Where N is the number of dimensions in vectors X and D, correspondingly, I^{th} element in vectors X and D are X_I and D_I. After substituting equation (7) into equation (5),the output O(D) becomes Gaussian kernel with width $\sigma = 2$,

While D is a vector of conditional probabilities of a sequence of length L:

$$D = \left[P(S_2 \mid S_1), P(S_3 \mid S_2), P(S_4 \mid S_3), ..., P(S_L \mid S_{L-1}) \right] \quad (8)$$

Therefore a SVM classifier with the Gaussian kernel function can approximate higher order Markov model.

5.6 Model Design

The splice site identification process is divided into two sub modules; these are donor splice site identification and the acceptor splice site identification. For each module separate models are created. For example, for HS3D donor data-set, one MM2F-SVM model is created and trained with HS3D donor training dataset. To estimate the classification performance of this model, the HS3D donor test dataset is used. In similar manner a separate MM2F-SVM model is trained and tested with HS3D acceptor training and acceptor test dataset. Similarly DGSplice and NN269, donor and acceptor dataset's are trained and tested.

5.7 Model Learning and Comparison

Training of the proposed model was conducted in three stages: the MM2 parameters estimation, feature selection using PFA and the SVM with Gaussian kernel training having width twenty for classification. True and false splice site training sequences are used to create the second order markov model. Depending upon the true and false splice site class label, desired output level is set to +1 and -1.We used MATLAB [221] implementation of the support vector machine.

To validate the usefulness of our proposed MM2F-SVM method and to compare its performance with others, we have selected other popular methods those are closely related to the proposed method. We used another preprocessing scheme that is zero order markov model (MM0) with SVM and compare their preprocessing performance with our proposed model.

5.8 Performance Measures

The proposed hybrid method's classification performance is estimated on the ROC curves, which gives a measure of the tradeoff between the true positive rate TPR and false positive rate FPR. Sensitivity S_n is the percentage of correct prediction of true splice sites and specificity S_p is the percentage of correct prediction of pseudo splice sites. . The sensitivity (S_N) (or TPR) is the

percentage of correct prediction of true sites and specificity (S_P) is the percentage of correct prediction of false sites as defined below:

$$Sensitivity(S_N) = \frac{TP}{TP + FN} \quad (9)$$

$$Specificity(S_P) = \frac{TN}{TN + FP} \quad (10)$$

$$FPR = 1 - S_p = \frac{FP}{FP + TN} \quad (11)$$

$$precision = \frac{TP}{TP + FP} \quad (12)$$

A true positive is a true donor (true acceptor, respectively) site that is also classified as a true donor (true acceptor, respectively) site. A false positive is a false donor (false acceptor, respectively) site that is wrongly predicted as a true donor (true acceptor, respectively) site. A true negative is a false donor (false acceptor, respectively) site that is also classified as a false donor (false acceptor, respectively) site. A false negative is a true donor (true acceptor, respectively) site that is wrongly classified as a false donor (false acceptor, respectively) site given in Table 3.4

Accuracy (ACC) is the proportion of the candidate sites in the test data set that are classified correctly [222], which tells how well the proposed MM2F-SVM system can assign true sites and false sites into the right categories; it was calculated by the following formula:

$$ACC = \frac{TN + TP}{TN + TP + FN + FP} \quad (13)$$

Matthews's correlation coefficient (MCC) is used as a comprehensive classification performance metric incorporating both sensitivity and specificity measures defined by the following formula [222].

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

where MCC ranges from -1 to 1, and completely well trained classifiers are denoted by 1.

5.8.1 ROC Analysis

Receiver operator curve (ROC) analysis is an effective and widely used method of assessing the performance of models [223]. It is a graphical representation of sensitivity and specificity of a classification model. To approximate the best possible FPR and TPR pair, we used the Euclidean metric. Specially, the best sensitivity, specificity tradeoff is defined by the coordinates of each point on the ROC curve with the minimum distance from a perfectly well-trained classifier. When the

ROC is created from the TPR (on the y axis) and FPR (on the x axis) of the model, the closer a curve approaches the (0, 0) point, the more accurate the model (refer to Figure 5.5 to 5.12).

5.8.2 Cross Validation

A twelve fold cross validation (CV) technique is applied to identify the MM2F-SVM splice site prediction accuracy and to compare their performance with other published methods [224]. Here cross validation is performed by splitting the data into twelve independent subsets, in which every subset does not share any repeating sequences. Each model was trained by selecting eleven of the subsets (training data) and tested on twelfth unused subset (test data). We calculate the average of the twelve prediction accuracies as the final prediction performance of the model, because CV is used for estimating the risk of the model.

5.9 Results and Discussion

5.9.1 Selection of the Best Preprocessing Method

For preprocessing method selection, here we have used other method like MM0 and MM2F with SVM classifiers for splice site prediction. We used HS3D donor and acceptor dataset for predictive accuracy comparison of MM0-SVM and MM2F-SVM methods. The ROC analysis of the models MM0-SVM and MM2F-SVM are shown in Figure 5.3 and 5.4.After observing their performance, the MM2F-SVM model is used as main method for splice site identification.

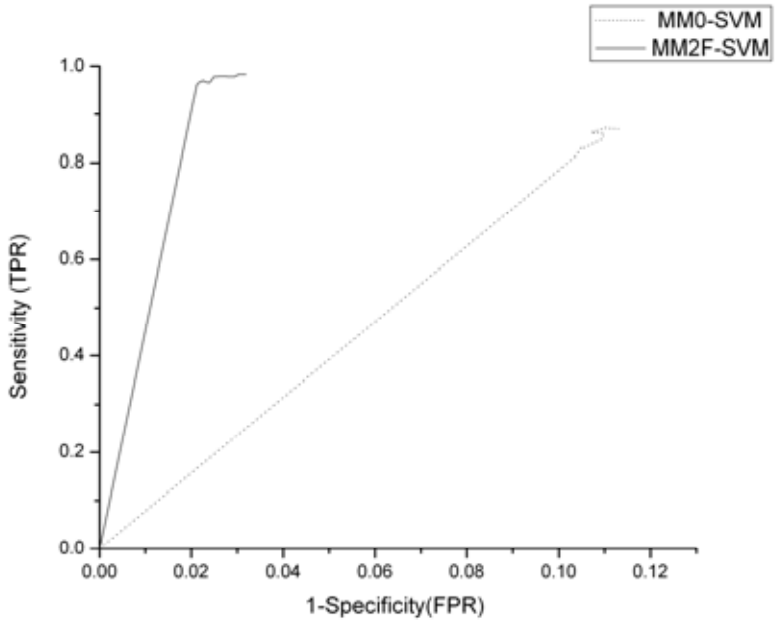

Figure 5.3 ROC curve showing the comparison of performance between MM0-SVM and MM2F-SVM using HS3D donor dataset.

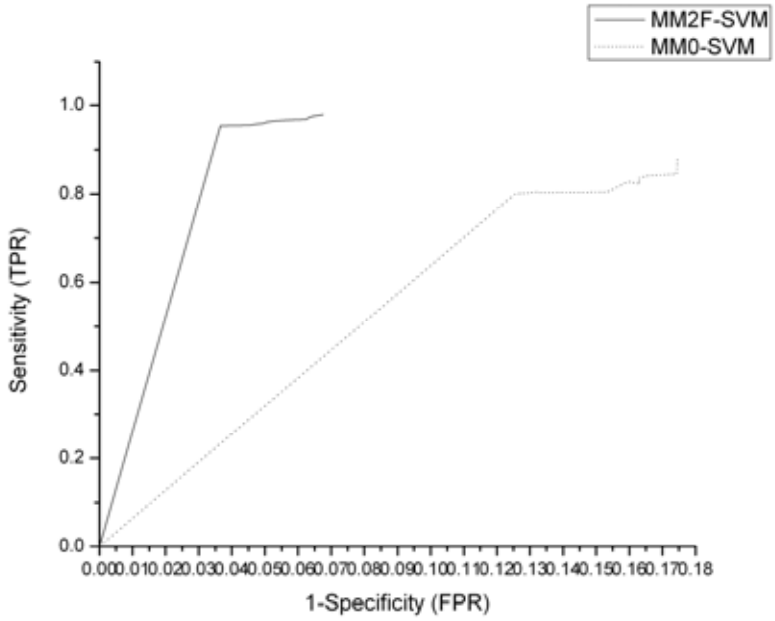

Figure 5.4 ROC curve showing the comparison of performance between MM0-SVM and MM2F-SVM using HS3D acceptor dataset.

5.9.2 Comparison in the Predictive Performance

The results of the 12-fold cross validation of the proposed model in terms of sensitivity (S_n), specificity (S_p), FPR and MCC using HS3D are shown in Tables 5.1 and 5.2.

The comparison of performance between the splice site prediction tools – MM2F-SVM, NNSplice [209] (http://www.fruitfly.org/seq_tools/splice.html) and NetGene2 which is trained on human data (http://genome.cbs.dtu.dk/services/NetGene2/) using HS3D dataset is shown in Figures 5.5 and 5.6. The standard TPR (S_n) and FPR (1-S_p) are employed for this comparison. It is clear from the comparison that MM2F-SVM is the superior model for the prediction of donor and acceptor splice site, and NetGene2 gave the second best performance. The values of S_n and S_p for MM2F-SVM are 98.31% and 97.88% (maximum) for the donor splice site prediction and 97.92% and 96.34% (maximum) for acceptor splice site prediction.

Table 5.1 Performance of Donor MM2F-SVM with Gaussian kernel width 20 for identifying donor (5' splice) sites.

S. No.	No of true donor	No of pseudo donor	TP	FP	TN	FN	Sensitivity (S_n)	Specificity (S_p)	FPR	MCC
1	233	22670	224	480	22190	9	0.96137	0.97882	0.0211	0.5466
2	235	22700	227	490	22210	8	0.96595	0.97841	0.0215	0.5464
3	240	23001	232	501	22500	8	0.96666	0.97821	0.0217	0.5465
4	235	22800	228	515	22285	7	0.97021	0.97741	0.0225	0.5389
5	241	22850	233	532	22318	8	0.96680	0.97671	0.0232	0.5357
6	238	23004	230	553	22451	8	0.96638	0.97596	0.0240	0.5258
7	237	22760	232	567	22193	5	0.97890	0.97508	0.0249	0.5261
8	236	22850	231	589	22261	5	0.97881	0.97422	0.0257	0.5179
9	238	23577	233	613	22964	5	0.97899	0.97400	0.026	0.5121
10	235	22890	230	674	22216	5	0.97872	0.97055	0.0294	0.4912
11	238	22779	234	691	22088	4	0.98319	0.96966	0.0303	0.4907
12	237	22547	233	720	21827	4	0.98312	0.96806	0.0319	0.4820
	Average						**0.97326**	**0.97476**	**0.0252**	**0.5217**

Table 5.2 Performance of Acceptor MM2F-SVM with Gaussian kernel width 20 for identifying acceptor (3' splice) sites.

Sr. No.	No of true acceptor	No of pseudo acceptor	TP	FP	TN	FN	Sensitivity (S_n)	Specificity (S_p)	FPR	MCC
1	240	27500	229	1005	26495	11	0.95416	0.96345	0.0365	0.4121
2	245	27820	234	1259	26561	11	0.95510	0.95474	0.0452	0.3771
3	239	27450	229	1308	26142	10	0.95815	0.95234	0.0476	0.3678
4	250	28000	240	1392	26608	10	0.96	0.95028	0.0497	0.3654
5	239	27780	230	1399	26381	9	0.96234	0.94964	0.0503	0.3584
6	256	28300	247	1487	26813	9	0.9648	0.9474	0.0525	0.3600
7	248	28670	240	1698	26972	8	0.96774	0.94077	0.0592	0.3350
8	253	27600	245	1729	25871	8	0.96837	0.93735	0.0626	0.3348
9	244	27676	237	1745	25931	7	0.97131	0.93694	0.0630	0.3291
10	254	27855	247	1759	26096	7	0.97244	0.93685	0.0631	0.3342
11	249	27650	243	1789	25861	6	0.97590	0.93529	0.0647	0.3297
12	241	27554	236	1861	25693	5	0.97925	0.93245	0.0675	0.3200
	Average						**0.96580**	**0.9448**	**0.0551**	**0.3520**

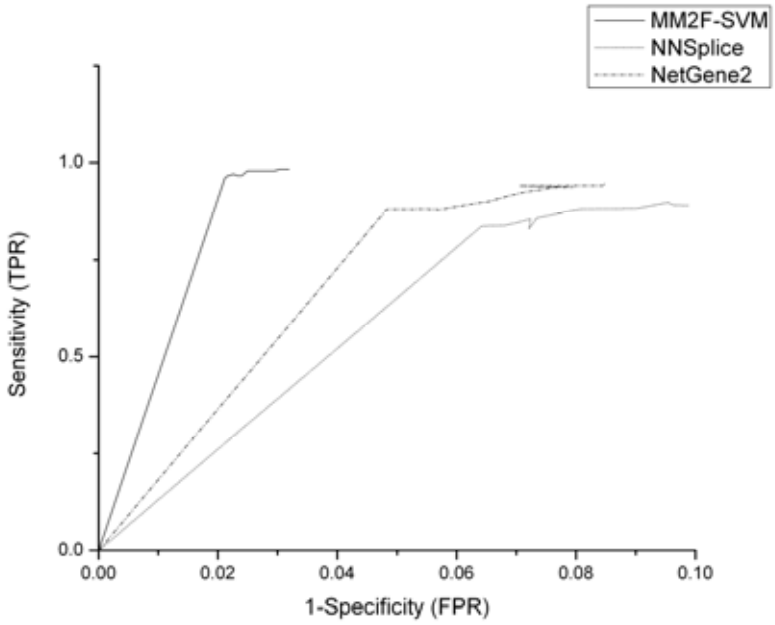

Figure 5.5 ROC curve showing the comparison of performance between NNSplice, NetGene2 and MM2F-SVM using HS3D donor dataset.

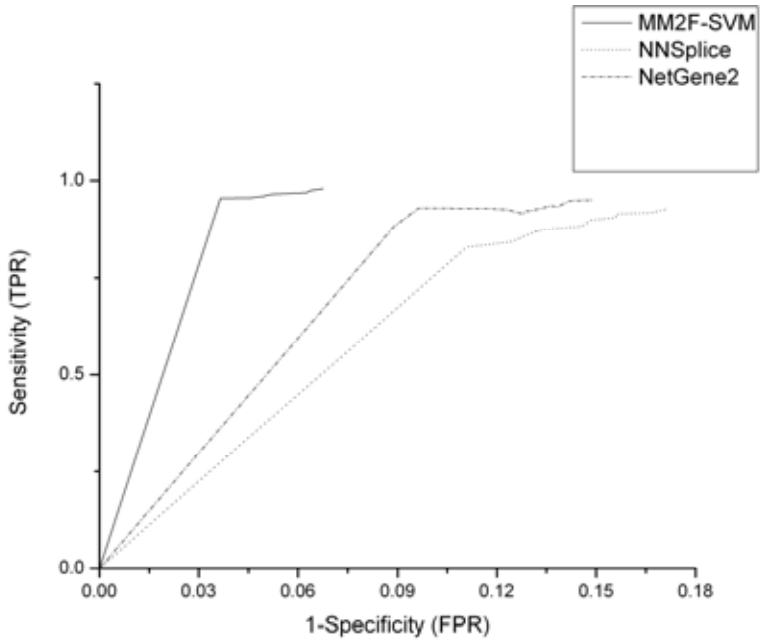

Figure 5.6 ROC curve showing the comparison of performance between NNSplice, NetGene2 and MM2F-SVM using HS3D acceptor dataset.

To verify the prediction accuracies of the MM2F-SVM method we used DGSplicer dataset and compared the performance with MDD method [2] as shown in Figure 5.7 and 5.8, in which MM2F-SVM shows superior performance.

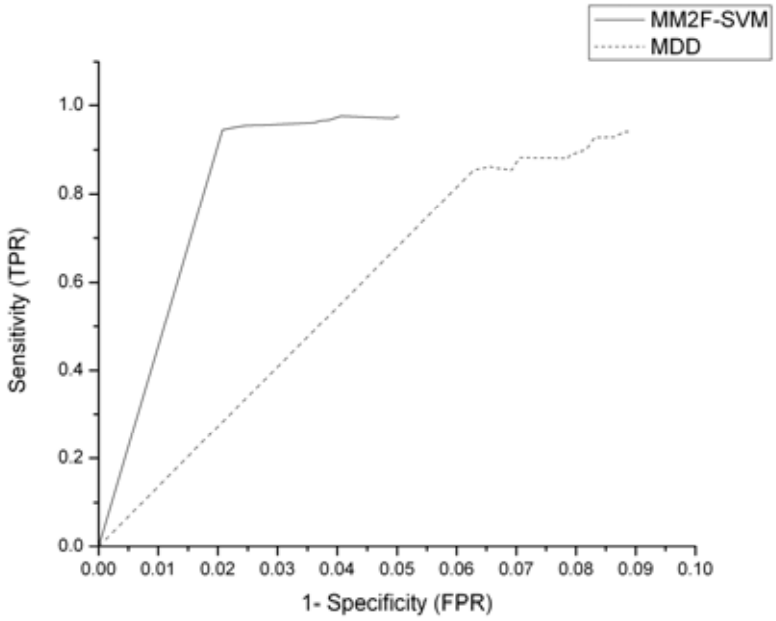

Figure 5.7 ROC curve showing the comparison of performance between MDD and MM2F-SVM using DGSplicer donor dataset.

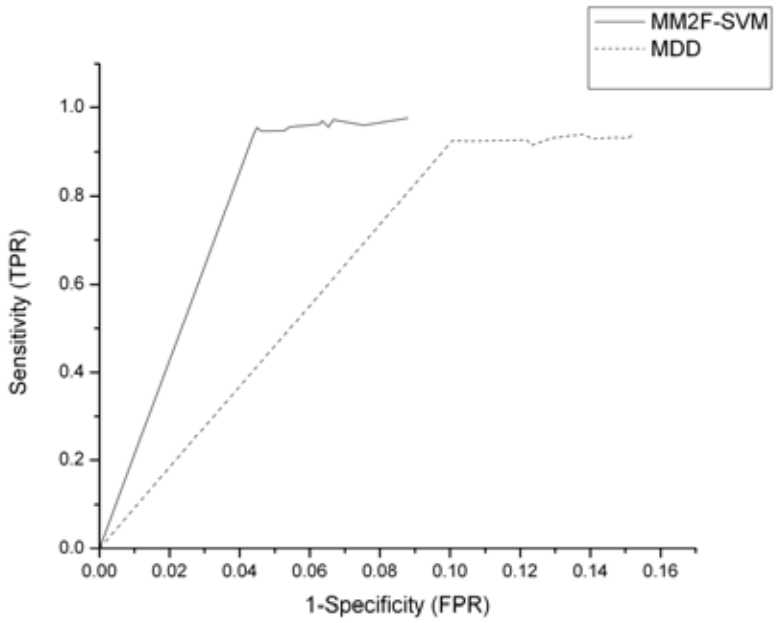

Figure 5.8 ROC curve showing the comparison of performance between MDD and MM2F-SVM using DGSplicer acceptor dataset.

To further verify the prediction accuracies of the MM2F-SVM method we used NN269 dataset and compared the performance with SVM+B method[225] as shown in Figures 5.9 and 5.10, in which MM2F-SVM shows superior performance.

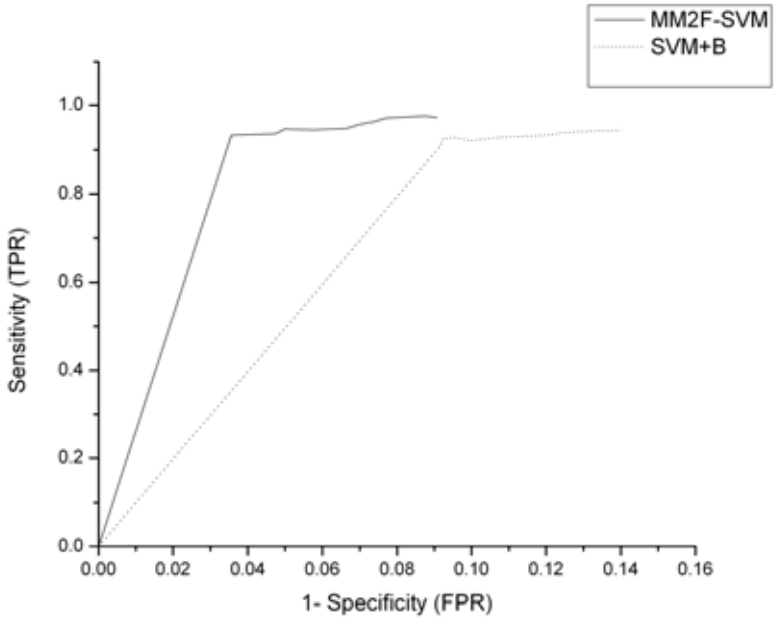

Figure 5.9 ROC curve showing the comparison of performance between SVM+B and MM2F-SVM using NN269 donor dataset.

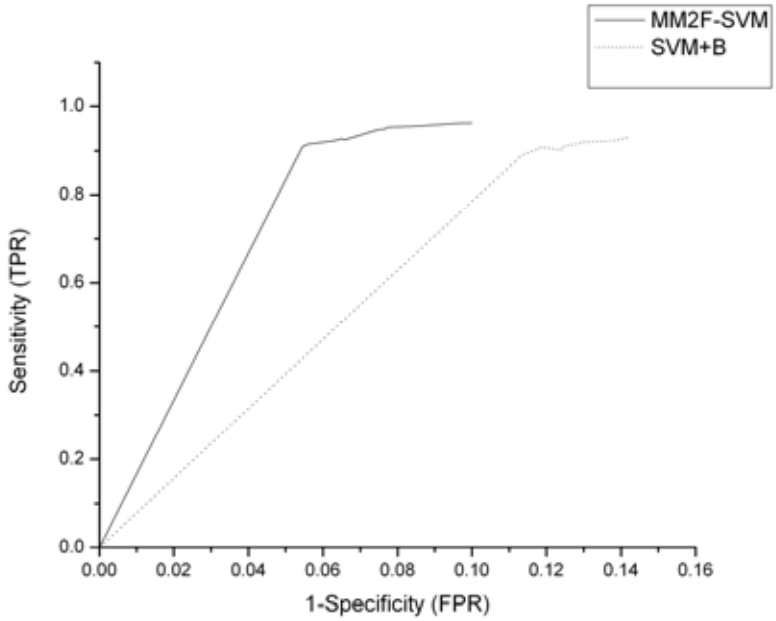

Figure 5.10 ROC curve showing the comparison of performance between SVM+B and MM2F-SVM using NN269 acceptor dataset.

CHAPTER – 6

CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

As the amount of sequenced DNA data is increasing very quickly, therefore, the utilization of computational gene identification methods becomes very essential these days. In this thesis, the core objective was focused on the two techniques for gene identification – conventional and computational intelligence. We have used these techniques/approaches for splice site prediction, which is a vital part of gene prediction itself for identifying donor splice site and acceptor splice site. In the conventional technique, splice site identification is performed with the help of HMM and in the other one, a blend of MM2 (conventional technique) and SVM is used for the purpose.

As described in the fourth chapter, our major accomplishment is developing a modified HMM for identification of splicing junction sites in DNA sequences, in which the implementation of modified HMM is performed. General HMM uses EM algorithm which itself suffer from high false positive rate, therefore, to overcome this snag, we've developed MEM algorithm and introduced in the training phase to increase sensitivity and specificity value. In addition, we have performed 12-fold cross validation experiment, which has proven the method's simplicity and effectiveness. Most importantly, the predictive accuracy has been shown to be significantly higher for the proposed HMM System as compared to the already existing NNSplice and GENIO tools (for splice site prediction), when tested on human splice site dataset. The system is able to correctly predict 95% of the true donor sites and 97% of the false donor sites in the test data set; 95% of the true acceptor sites and 92% of the false acceptor sites in the test data set. Overall, this system is comparatively superior in sensitivity and can correctly detect 97% of the true donor sites and 92% of the true acceptor sites in the standard sequenced data. Hence, this method can be utilized to identify splice sites in the large scale in newly invented genomes.

Another major achievement of our effort(s) is the development of Hybrid Approach using SVM and MM2 in Splice Site Junction Identification, as discussed in the fifth chapter. Here, a new splice site identification model namely Second Order Markov Model Feature-Support Vector Machine (MM2F-SVM) is developed. This proposed method consists of three stages – initial stage, in which a second order Markov Model (MM2) is used, i.e. feature extraction; intermediate, or the second stage in which principal feature analysis (PFA) is done, i.e. feature selection; and the final or the third stage, in which a support vector machine (SVM) with Gaussian kernel is used for final classification of splice sites. This MM2F-SVM model, when compared with the similar other already existing splice site prediction programs, superior performance has been noticed. Hence, it is obvious that this method is simple and effective too, which can be used to identify splice site junction on large scale in sequenced genomics. In this case also, we have used 12-fold cross validation experiment, and it is found that this MM2F-SVM system is able to correctly identify

maximum 98.31% of the true donor sites and 97.88% of the false donor sites in the test dataset; 97.92% of the true acceptor sites and 96.34% of the false acceptor sites in the test data set.

6.2 Recommendations for Future

As already discussed, the frameworks have been proven more efficient and reliable than the conventional or the older techniques, however, the work may be extended to implement the benefits of their increased sensitivity, specificity and accuracy for various splice site prediction work of other kingdom resides under eukaryotic organism like animal, plants and fungi, the fruit fly (*Drosophila melanogaster*), nematode (*Caenorhabditis elegans*) house mouse(*Mus musculus*) and zebrafish (*Danio rerio*) and *Arabidopsis thaliana*. After proper training of the tool, the same can be used as a splice site predictor in the above mentioned organisms. Apart from the specific aim of splice site prediction, other aspects of this work are also important, for example, after incorporating promoter, start codon and stop codon techniques in the existing splice site prediction, this HMM system is capable to identify complete gene structure also. Therefore, this system can be applicable in medical diagnosis, medical treatment and other fields of research. It follows that biological sequences can be treated linguistically with the same techniques used for speech recognition and language processing.

Although the developed MM2F-SVM model, which is following Hybrid Approach using SVM and MM2 in Splice Site Junction Identification, and is trained by three different datasets of Human (Homo Sapiens), HS3D, DGSplicer and NN269; the model may be further modified to implement the benefits of the same to other organisms. The novelty of the developed model is the ability to identify splice site in multiple sequences at the same time. Even though, by using hybrid approach better predictive performance is achieved, however, a lot of time is utilized for preprocessing purpose to remove the redundant sequence data due to unavailability of true splice site as compared to false splice site. So, there is a requirement of more accurate DNA sequence data which contain minimum amount of junk or redundant information. Apart from these, another remarkable feature of this program is its ability to assign a meaningful reliability measure, the exon probability, to each predicted exon, which provides the user a vastly informative guide as to the degree of confidence which should be recognized for each aspect of a prediction. Finally, the prospective use of splice site identification was demonstrated by the finding of splice site in newly sequenced genomes not homologous to any known protein in a published human genomic sequence.

In the existing algorithms, MM2F-SVM is similar in its overall architecture to the already developed MM1-SVM program [226], which uses first order Markov Model (MM1) and Support Vector Machine (SVM) for splice site junction detection. However, both of these models have certain differences, which are:

1. MM1-SVM consists of two stages, where as MM2F-SVM have three stages.
2. In case of previous model, MM1 is used for pre-processing whereas here, MM2 and PFA are used for the same purpose.
3. The proposed model uses SVM with Gaussian kernel, but in case of MM1-SVM, SVM with polynomial kernel is used for the final classification.

Finally, it is worthy to mention here that for all gene prediction methods, the performance depends to a huge extent, on the contemporary biological knowledge, especially at the molecular level of the gene expression. Therefore, it needs great efforts by both experimental and computational biologists to make accurate gene prediction, which can certainly speed up gene finding and knowledge mining.

List of Publications

Publications/Communication in the peer reviewed journals:

1. Srabanti Maji and Deepak Garg, Gene finding using Hidden Markov Model, Journal of Applied Sciences, 12 (15), 1518-1525, 2012.
2. Srabanti Maji and Deepak Garg, Progress in gene prediction: principles and challenges, *Current Bioinformatics,* 8 (2), 226-243, 2013.
3. Srabanti Maji and Deepak Garg, Hidden markov model for splicing junction sites identification in DNA sequences, *Current Bioinformatics* (**Accepted**).
4. Srabanti Maji and Deepak Garg, Hybrid Approach using SVM and MM2 in Splice Site Junction Identification, *Current Bioinformatics* (**Communicated**).

References

[1] J. W. Fickett, "The gene identification problem: An overview for developers," *Computers and Chemistry,* vol. 20, no. 1, pp. 103-118, 1996.

[2] C. Burge, and S. Karlin, "Prediction of complete gene structures in human genomic DNA," *Journal of Molecular Biology,* vol. 268, no. 1, pp. 78-94, 1997.

[3] B. Alberts, D. Bray, J. Lewis *et al.*, *Molecular Biology of the Cell,* 1994.

[4] A. Abdullah, K. Buragga, and M. Ahmad, "A novel optimized approach for gene identification in DNA sequences," *Journal of Applied Sciences,* vol. 11, pp. *806-814.*, 2011.

[5] M. J. Schellenberg, D. B. Ritchie, and A. M. MacMillan, "Pre-mRNA splicing: a complex picture in higher definition," *Trends in Biochemical Sciences,* vol. 33, no. 6, pp. 243-246, 2008.

[6] C. Mathe, M. F. Sagot, T. Schiex *et al.*, "Current methods of gene prediction, their strengths and weaknesses," *Nucleic Acids Research,* vol. 30, no. 19, pp. 4103-4117, 2002.

[7] M. Kumar, and G. P. Raghava, "Prediction of nuclear proteins using SVM and HMM models," *BMC Bioinformatics,* vol. 10, pp. 22, 2009.

[8] U. M. Sanghamitra Bandyopadhyay, and Debadyuti Roy, "Gene identification: classical and computational intelligence approaches," *IEEE Transactions on Systems, Man, and Cybernetics Part-C: Applications and Reviews,* vol. 38, no. 1, pp. 55-68, 2008.

[9] L. D. Stein, "End of the beginning," *Nature,* vol. 431, no. 7011, pp. 915-916, 2004.

[10] B. Mersch, A. Gepperth, S. Suhai *et al.*, "Automatic detection of exonic splicing enhancers (ESEs) using SVMs," *BMC Bioinformatics,* vol. 9, no. 1, pp. 369, 2008.

[11] S. Brenner, "Sequences and consequences," *Philosophical Transactions of the Royal Society B: Biological Sciences,* vol. 365, no. 1537, pp. 207-212, 2010.

[12] Zhumur Ghosh, Bibekan Mallick, "Bioinformatics principles and Applications," *2nd Edition, Oxford University Press, Delhi* 2009.

[13] S.A. Brenner, "Genomics: The end of the beginning," *Science* vol. 287, no. 5461, pp. 2173-2174., 2000.

[14] H. S. Bodiwala, S. Sabde, P. Gupta *et al.*, "Design and synthesis of caffeoyl-anilides as portmanteau inhibitors of HIV-1 integrase and CCR5," *Bioorganic and Medicinal Chemistry,* vol. 19, no. 3, pp. 1256-1263, 2011.

[15] P. Gupta, P. Garg, and N. Roy, "Comparative docking and CoMFA analysis of curcumine derivatives as HIV-1 integrase inhibitors," *Molecular Diversity,* vol. 15, no. 3, pp. 733-750, 2011.

[16] P. Gupta, P. Garg, and N. Roy, "Identification of novel HIV-1 integrase inhibitors using shape-based screening, QSAR, and docking approach," *Chemical Biology and Drug Design,* vol. 79, no. 5, pp. 835-849, 2012.

[17] J. S. Toor, A. Sharma, R. Kumar *et al.*, "Prediction of drug-resistance in HIV-1 subtype C based on protease sequences from ART naive and first-line treatment failures in North India using genotypic and docking analysis," *Antiviral Research,* vol. 92, no. 2, pp. 213-218, 2011.

[18] W. N. van Wieringen, D. Kun, R. Hampel *et al.*, "Survival prediction using gene expression data: A review and comparison," *Computational Statistics and Data Analysis,* vol. 53, no. 5, pp. 1590-1603, 2009.

[19] G. D. Stormo, "Gene-finding approaches for eukaryotes," *Genome Research,* vol. 10, no. 4, pp. 394-397, 2000.

[20] M. Q. Zhang, "Computational prediction of eukaryotic protein-coding genes," *Nature Reviews Genetics,* vol. 3, no. 9, pp. 698-709, 2002.

[21] S. L. Cawley, and L. Pachter, "HMM sampling and applications to gene finding and alternative splicing," *Bioinformatics,* vol. 19, no. suppl. 2, pp. 1136-1141, 2003.

[22] J. M. Claverie, "Computational methods for the identification of genes in vertebrate genomic sequences," *Human Molecular Genetics,* vol. 6, no. 10 Rev. ISS., pp. 1735-1744, 1997.

[23] S. R. Eddy, "A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure," *BMC bioinformatics [electronic resource],* vol. 3, no. 1, pp. 18, 2002.

[24] M. Burset, and R. Guigo, "Evaluation of gene structure prediction programs," *Genomics,* vol. 34, no. 3, pp. 353-367, 1996.

[25] M. S. Gelfand, "Prediction of function in DNA sequence analysis," *Journal of computational biology: a journal of computational molecular cell biology,* vol. 2, no. 1, pp. 87-115, 1995.

[26] D. Haussler, "Computational genefinding", *Trends in Biochemical Sciences, Supplementary Guide to Bioinformatics* vol. 23, no. 1, 1998.

[27] C. A. Fields, and C. A. Soderlund, "GM: A practical tool for automating DNA sequence analysis," *Computer applications in the biosciences: CABIOS,* vol. 6, no. 3, pp. 263-270, 1990.

[28] M. S. Gelfand, "Computer prediction of the exon-intron structure of mammalian pre-mRNAs," *Nucleic Acids Research,* vol. 18, no. 19, pp. 5865-5869, 1990.

[29] P. Flicek, "Gene prediction: Compare and contrast," *Genome Biology,* vol. 8, no. 12, 2007.

[30] J. E. Allen, M. Pertea, and S. L. Salzberg, "Computational gene prediction using multiple sources of evidence," *Genome Research,* vol. 14, no. 1, pp. 142-148, 2004.

[31] M. M. Yin, and J. T. L. Wang, "Effective hidden Markov models for detecting splicing junction sites in DNA sequences," *Information Sciences,* vol. 139, pp. 139-163, 2001.

[32] P. Larranaga, B. Calvo, R. Santana *et al.*, "Machine learning in bioinformatics," *Briefings in Bioinformatics,* vol. 7, no. 1, pp. 86-112, 2006.

[33] H. L. Rajapakse, "Markov encoding for detecting signals in genomic sequences," *IEEE/ACM Trans Comput Biol Bioinform.,* vol. 2, no. 2, pp. 131-42, 2005.

[34] V. Brendel, and J. Kleffe, "Prediction of locally optimal splice sites in plant pre-mRNA with applications to gene identification in Arabidopsis thaliana genomic DNA," *Nucleic Acids Res.,* vol. 26, no. 20, pp. 4748-4757, 1998.

[35] R. Lopez, F. Larsen, and H. Prydz, "Evaluation of the exon predictions of the GRAIL software," *Genomics,* vol. 24, no. 1, pp. 133-136, 1994.

[36] L. R. Rabiner, "Tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE,* vol. 77, no. 2, pp. 257-286, 1989.

[37] A. V. Lukashin, and M. Borodovsky, "GeneMark.hmm: New solutions for gene finding," *Nucleic Acids Research,* vol. 26, no. 4, pp. 1107-1115, 1998.

[38] M. Borodovsky, and J. McIninch, "GENMARK: Parallel gene recognition for both DNA strands," *Computers and Chemistry,* vol. 17, no. 2, pp. 123-133, 1993.

[39] S. Audic, and J. M. Claverie, "Detection of eukaryotic promoters using Markov transition matrices," *Computers & Chemistry,* vol. 21, no. 4, pp. 223-227, 1997.

[40] S. I. Sachem, "A method for identifying splice sites and translational start sites in eukaryotic mRNA," *Computer Applications in the Biosciences,* vol. 13, no. 4, pp. 365-376, 1997.

[41] S. S. Henderson, and K. H. Fasman, "Finding genes in DNA with a Hidden Markov Model," *Journal of Computational Biology,* vol. 4, no. 2, pp. 127-141, 1997.

[42] M. M. Yin and J. T. L. Wang, "GeneScout: A Data Mining System for Predicting Vertebrate Genes in Genomic DNA sequences," *Information Sciences, Special Issue on Soft Computing Data Mining,* vol. 163, no. 1-3, pp. 201-218, 2004.

[43] T. Schiex, A. Moisan, and P. Rouzac, "EuGene: An eukaryotic gene finder that combines several sources of evidence," *Computational Biology: Selected Papers (Lecture Notes in Computer Science),* vol. 2066, pp. 111-125, 2001.

[44] K. L. Howe, T. Chothia, and R. Durbin, "GAZE: A genetic framework for the integration of gene-prediction data by dynamic programming," *Genome Research,* vol. 12, no. 9, pp. 1418-1427, 2002.

[45] M. S. Gelfand, A. A. Mironov, and P. A. Pevzner, "Gene recognition via spliced sequence alignment," *Proceedings of the National Academy of Sciences of the United States of America,* vol. 93, no. 17, pp. 9061-9066, 1996.

[46] I. B. A. Parent, I. Bougie, and M. Bisaillon, "Transcription and mRNA Processing Events: the Importance of Coordination," *Journal of Biological Sciences,* vol. *4,* pp. 624-627, 2004.

[47] L. Vignal, F. D. R. Lisacek, J. l. Quinqueton *et al.,* "A multi-agent system simulating human splice site recognition," *Computers & Chemistry,* vol. 23, no. 3-4, pp. 219-231, 1999.

[48] L. Vignal, Y. d'Aubenton-Carafa, F. Lisacek *et al.,* "Exon prediction in eucaryotic genomes," *Biochimie,* vol. 78, no. 5, pp. 327-334, 1996.

[49] J. Henderson, "Finding genes in DNA with a Hidden Markov Model," *Journal of Computational Biology,* vol. 4, no. 2, pp. 127-141, 1997.

[50] I. Inza, P. Larranaga, R. Etxeberria *et al.,* "Feature subset selection by Bayesian network-based optimization," *Artificial Intelligence,* vol. 123, no. 1-2, pp. 157-184, 2000.

[51] Y. Saeys, S. Degroeve, D. Aeyels *et al.,* "Fast feature selection using a simple estimation of distribution algorithm: A case study on splice site prediction," *Bioinformatics,* vol. 19, no. Suppl. 2, pp. 179-188, 2003.

[52] S. Degroeve, B. de Baets, Y. Van de Peer *et al.,* "Feature subset selection for splice site prediction," *Bioinformatics,* vol. 18, no. suppl 2, pp. S75-S83, October 1, 2002, 2002.

[53] Y. Saeys, S. Degroeve, D. Aeyels *et al.,* "Feature selection for splice site prediction: A new method using EDA-based feature ranking," *BMC Bioinformatics,* vol. 5, no. 1, pp. 64, 2004.

[54] M. P. S. Brown, W. N. Grundy, D. Lin *et al.,* "Knowledge-based analysis of microarray gene expression data by using support vector machines," *Proceedings of the National Academy of Sciences of the United States of America,* vol. 97, no. 1, pp. 262-267, 2000.

[55] R. Staden, "Computer methods to locate signals in nucleic acid sequences," *Nucleic Acids Research,* vol. 12, no. 1 PART2, pp. 505-519, 1984.

[56] M. Q. Zhang, and T. G. Marr, "A weight array method for splicing signal analysis," *Computer Applications in the Biosciences,* vol. 9, no. 5, pp. 499-509, 1993.

[57] A. Y. Kashiwabara, D. C. G. Vieira, A. Machado-Lima *et al.,* "Splice site prediction using stochastic regular grammars," *Genetics and Molecular Research,* vol. 6, no. 1, pp. 105-115, 2007.

[58] C. Burge, "Identification of genes in human genomic DNA," *Ph. D Thesis,* 1997.

[59] X. Zhao, H. Huang, and T. P. Speed, "Finding short DNA motifs using permuted markov models," in *Proceedings of the eighth annual international conference on Research in computational molecular biology*, San Diego, California, USA, 2004.

[60] S. Brunak, J. Engelbrecht, and S. Knudsen, "Prediction of human mRNA donor and acceptor sites from the DNA sequence," *Journal of Molecular Biology*, vol. 220, no. 1, pp. 49-65, 1991.

[61] M. G. Reese, "Application of a time-delay neural network to promoter annotation in the Drosophila melanogaster genome," *Computers & Chemistry*, vol. 26, no. 1, pp. 51-56, 2001.

[62] M. Pertea, X. Lin, and S. L. Salzberg, "GeneSplicer: A new computational method for splice site prediction," *Nucleic Acids Research*, vol. 29, no. 5, pp. 1185-1190, 2001.

[63] A. Baten, B. Chang, S. Halgamuge *et al.*, "Splice site identification using probabilistic parameters and SVM classification," *BMC Bioinformatics*, vol. 7, no. Suppl 5:S15, 2006.

[64] S.-A. Marashi, C. Eslahchi, H. Pezeshk *et al.*, "Impact of RNA structure on the prediction of donor and acceptor splice sites," *BMC Bioinformatics*, vol. 7, pp. 297, 2006.

[65] J. H. Holland, "Adaptation in natural and artificial system," *MIT Press Cambridge, MA, USA*, 1975.

[66] J. Gunnels, P. Cull, and J. L. Holloway, "Genetic algorithms and simulated annealing for gene mapping," *In Proceedings of International Conference on Evolutionary Computation*, vol. 1, pp. 385-390, 1994.

[67] A. Kel, A. Ptitsyn, V. Babenko *et al.*, "A genetic algorithm for designing gene family-specific oligonucleotide sets used for hybridization: the G protein-coupled receptor protein superfamily," *Bioinformatics*, vol. 14, no. 3, pp. 259-270, 1998.

[68] V. G. Levitsky, and A. V. Katokhin, "Recognition of eukaryotic promoters using a genetic algorithm based on iterative discriminant analysis," *In Silico Biology*, vol. 3, no. 1-2, pp. 81-87, 2003.

[69] V. Vapnik, "The nature of statistical learning theory," *Springer*, 1995.

[70] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol. 2, pp. 121-167, 1998.

[71] J. A. K. Suykens, "Support vector machines: A nonlinear modelling and control perspective," *European Journal of Control*, vol. 7, no. 2-3, pp. 311-327, 2001.

[72] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, no. 3, pp. 338-353, 1965.

[73] L. A. Zadeh, "Fuzzy algorithms," *Information and Control*, vol. 12, pp. 94-102, 1968.

[74] B. Roux, and S. Winters-Hilt, "Hybrid MM/SVM structural sensors for stochastic sequential data," *BMC Bioinformatics*, vol. 9, no. Suppl 9:S12, 2008.

[75] Y. Saeys, S. Degroeve, D. Aeyels *et al.*, "Feature selection for splice site prediction: A new method using EDA-based feature ranking," *BMC Bioinformatics*, vol. 5, pp. 64, 2004.

[76] S. Sonnenburg, "New methods for detecting splice junction sites in DNA sequence," *Master's Thesis, Humbold University: Germany*, 2002.

[77] G. Ratsch, S. Sonnenburg, and C. Schafer, "Learning interpretable SVMs for biological sequence classification," *BMC Bioinformatics*, vol. 7, no. Suppl 1: S9, 2006.

[78] Y.-F. Sun, X.-D. Fan, and Y.-D. Li, "Identifying splicing sites in eukaryotic RNA: Support vector machine approach," *Computers in Biology and Medicine*, vol. 33, no. 1, pp. 17-29, 2003.

[79] X. H. F. Zhang, K. A. Heller, I. Hefter *et al.*, "Sequence information for the splicing of human pre-mRNA identified by support vector machine classification," *Genome Research*, vol. 13, no. 12, pp. 2637-2650, 2003.

[80] T. M. Chen, C. C. Lu, and W. H. Li, "Prediction of splice sites with dependency graphs and their expanded bayesian networks," *Bioinformatics*, vol. 21, no. 4, pp. 471-482, 2005.

[81] S. A. Marashi, C. Eslahchi, H. Pezeshk *et al.*, "Impact of RNA structure on the prediction of donor and acceptor splice sites," *BMC Bioinformatics*, vol. 7, pp. 297, 2006.

[82] R. D. Sleator, "An overview of the current status of eukaryote gene prediction strategies," *Gene*, vol. 461, no. 1-2, pp. 1-4, 2010.

[83] D. Cai, A. Delcher, B. Kao *et al.*, "Modeling splice sites with Bayes networks," *Bioinformatics*, vol. 16, no. 2, pp. 152-158, 2000.

[84] M. P. Simmons, H. Ochoterena, and J. V. Freudenstein, "Amino acid vs. nucleotide characters: challenging preconceived notions," *Molecular Phylogenetics and Evolution*, vol. 24, no. 1, pp. 78-90, 2002.

[85] J. E. Bailey, and D. F. Ollis, "Biochemical Engineering Fundamentals," *2nd edition, McGraw-Hill, New York*, 1986.

[86] S. A. Marhon, and S. C. Kremer, "Gene prediction based on DNA spectral analysis: A literature review," *Journal of Computational Biology*, vol. 18, no. 4, pp. 639-676, 2011.

[87] M. Akhtar, E. Ambikairajah, and J. Epps, "Digital signal processing techniques for gene finding in eukaryotes," *Image and Signal Processing, Lecture Notes in Computer Science*, vol. 5099, pp. 144-152, 2008.

[88] W. W. Gibbs, "The unseen genome: beyond DNA," *Scientific American*, vol. 289, no. 6, pp. 106-113, 2003.

[89] T. W. Fox, and A. Carreira, "A digital signal processing method for gene prediction with improved noise suppression," *EURASIP Journal on Advances in Signal Processing*, vol. 2004, pp. 108-114, 2004.

[90] M. L. Shuler, and F. Kargi, "Bioprocessing engineering: basic concepts," *2nd edition, Pearson Education, New Delhi*, 2007.

[91] M. R. Brent, "Predicting full-length transcripts," *Trends in Biotechnology*, vol. 20, no. 7, pp. 273-275, 2002.

[92] S. Datta, A. Asif, and H. Wang, "Prediction of protein coding regions in DNA sequences using Fourier spectral characteristics," *In Proceedings of the IEEE Sixth International Symposium on Multimedia Software Engineering*, pp. 160-163, 2004.

[93] X. Xia, "Bioinformatics and the cell: modern computational approaches in genomics, proteomics, and transcriptomics," *Springer, New York*, 2007.

[94] T. F. Smith, and M. S. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195-197, 1981.

[95] W. R. Pearson, and D. J. Lipman, "Improved tools for biological sequence comparison," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 85, no. 8, pp. 2444-2448, 1988.

[96] S. F. Altschul, W. Gish, W. Miller *et al.*, "Basic local alignment search tool," *Journal of Molecular Biology*, vol. 215, no. 3, pp. 403-410, 1990.

[97] J. W. Fickett, "ORFs and genes: how strong a connection?," *Journal of Computational Biology*, vol. 2, no. 1, pp. 117-123, 1995.

[98] J. W. Fickett, and C. S. Tung, "Assessment of protein coding measures," *Nucleic Acids Research,* vol. 20, no. 24, pp. 6441-6450, 1992.

[99] I. B. Rogozin, and L. Milanesi, "Analysis of donor splice sites in different eukaryotic organisms," *Journal of Molecular Evolution,* vol. 45, no. 1, pp. 50-59, 1997.

[100] J. Kleffe, K. Hermann, W. Vahrson *et al.*, "Logitlinear models for the prediction of splice sites in plant pre-mRNA sequences," *Nucleic Acids Research,* vol. 24, no. 23, pp. 4709-4718, 1996.

[101] S. Salzberg, A. L. Delcher, K. H. Fasman *et al.*, "A decision tree system for finding genes in DNA," *Journal of Computational Biology,* vol. 5, no. 4, pp. 667-680, 1998.

[102] M. A. Roytberg, T. V. Astakhova, and M. S. Gelfand, "Combinatorial approaches to gene recognition," *Computers & Chemistry,* vol. 21, no. 4, pp. 229-235, 1997.

[103] R. Guigo, "Assembling genes from predicted exons in linear time with dynamic programming," *Journal of Computational Biology,* vol. 5, no. 4, pp. 681-702, 1998.

[104] A. J. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory,* vol. 13, no. 2, pp. 260-269, 1967.

[105] R. E. Bellman, "Dynamic programming," *Princeton University Press, Princeton,* 1957.

[106] R. Guigo, S. Knudsen, N. Drake *et al.*, "Prediction of gene structure," *Journal of Molecular Biology,* vol. 226, no. 1, pp. 141-157, 1992.

[107] L. Milanesi, N. A. Kolchanov, I. B. Rogozin *et al.*, "GenView: A computing tool for protein-coding regions prediction in nucleotide sequences," *In Proceedings of the Second International Conference on Bioinformatics, Supercomputing, and Complex Genome Analysis,* pp. 573-588, 1993.

[108] Y. Xu, R. J. Mural, and E. C. Uberbacher, "Constructing gene models from accurately predicted exons: An application of dynamic programming," *Computer Applications in the Biosciences,* vol. 10, no. 6, pp. 613-623, 1994.

[109] A. A. Salamov, and V. V. Solovyev, "Ab-initio gene finding in Drosophila genomic DNA," *Genome Research,* vol. 10, no. 4, pp. 516-522, 2000.

[110] J. S. Chuang, and D. Roth, "Gene recognition based on DAG shortest paths," *Bioinformatics,* vol. 17, no. suppl 1, pp. S56-S64, 2001.

[111] A. E. Tenney, R. H. Brown, C. Vaske *et al.*, "Gene prediction and verification in a compact genome with numerous small introns," *Genome Research,* vol. 14, no. 11, pp. 2330-2335, 2004.

[112] X. Huang, M. D. Adams, H. Zhou *et al.*, "A tool for analyzing and annotating genomic sequences," *Genomics,* vol. 46, no. 1, pp. 37-45, 1997.

[113] I. Korf, P. Flicek, D. Duan *et al.*, "Integrating genomic homology into gene structure prediction," *Bioinformatics,* vol. 17, no. suppl. 1, pp. S140-S148, 2001.

[114] R. F. Yeh, L. P. Lim, and C. B. Burge, "Computational inference of homologous gene structures in the human genome," *Genome Research,* vol. 11, no. 5, pp. 803-816, 2001.

[115] T. Schiex, A. Moisan, and P. Rouze "EuGene: An eukaryotic gene finder that combines several sources of evidence," *Lecture Notes in Computer Science, Springer-Verlag,* vol. 2066, pp. 111-125, 2001.

[116] A. G. Pedersen, and H. Nielsen, "Neural network prediction of translation initiation sites in eukaryotes: perspectives for EST and genome analysis," *In Proceedings of International Conference on Intelligent Systems for Molecular Biology; ISMB,* vol. 5, pp. 226-233, 1997.

[117] S. Maji, and D. Garg, "Progress in gene prediction: principles and challenges," *Current Bioinformatics*, vol. 8, pp. 226-243, 2013.

[118] G. L. Kouemou, "History and theoretical basics of Hidden Markov Models," *Hidden Markov Models: Theory and Applications*, 2011.

[119] T. Koski, "Hidden Markov Models for Bioinformatics," *Kluwer Academic Publishers*, 2002.

[120] E. Birney, "Hidden Markov models in biological sequence analysis," *IBM Journal of Research and Development*, vol. 45, no. 3-4, pp. 449-454, 2001.

[121] K. Karplus, R. Karchin, J. Draper *et al.*, "Combining local-structure, fold-recognition, and new fold methods for protein structure prediction," *Proteins: Structure, Function, and Bioinformatics*, vol. 53, no. S6, pp. 491-496, 2003.

[122] P. Bucher, K. Karplus, N. Moeri *et al.*, "A flexible motif search technique based on generalized profiles," *Computers & Chemistry*, vol. 20, no. 1, pp. 3-23, 1996.

[123] E. L. L. Sonnhammer, S. R. Eddy, E. Birney *et al.*, "Pfam: Multiple sequence alignments and HMM-profiles of protein domains," *Nucleic Acids Research*, vol. 26, no. 1, pp. 320-322, 1998.

[124] V. Di Francesco, P. J. Munson, and J. Garnier, "FORESST: fold recognition from secondary structure predictions of proteins," *Bioinformatics*, vol. 15, no. 2, pp. 131-140, 1999.

[125] D. Kulp, D. Haussler, M. G. Reese *et al.*, "A generalized hidden Markov model for the recognition of human genes in DNA," *In Proceedings of International Conference on Intelligent Systems for Molecular Biology; ISMB*, vol. 4, pp. 134-142, 1996.

[126] A. Krogh, "Two methods for improving performance of an HMM and their application for gene finding," *In Proceedings of International Conference on Intelligent Systems for Molecular Biology; ISMB*, vol. 5, pp. 179-186, 1997.

[127] V. De Fonzo, F. Aluffi-Pentini, and V. Parisi, "Hidden Markov Models in bioinformatics," *Current Bioinformatics*, vol. 2, no. 1, pp. 49-61, 2007.

[128] J. Bilmes, "A gentle tutorial on the EM algorithm and its application to parameter estimation for gaussian mixture and Hidden Markov Models, " U.C. Berkely, TR-97-02, 1997.

[129] S. E. Cawley, A. I. Wirth, and T. P. Speed, "Phat - A gene finding program for Plasmodium falciparum," *Molecular and Biochemical Parasitology*, vol. 118, no. 2, pp. 167-174, 2001.

[130] L. Xiaolin, M. Parizeau, and R. Plamondon, "Training Hidden Markov models with multiple observations-a combinatorial method," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 4, pp. 371-377, 2000.

[131] J. S. Pedersen, and J. Hein, "Gene finding with a Hidden Markov Model of genome structure and evolution," *Bioinformatics*, vol. 19, no. 2, pp. 219-227, 2003.

[132] E. E. Snyder, and G. D. Stormo, "Identification of protein coding regions in genomic DNA," *Journal of Molecular Biology*, vol. 248, no. 1, pp. 1-18, 1995.

[133] M. S. Gelfand, and M. A. Roytberg, "Prediction of the exon-intron structure by a dynamic programming approach," *BioSystems*, vol. 30, no. 1-3, pp. 173-182, 1993.

[134] M. S. Gelfand, T. V. Astakhova, and M. A. Roytberg, "An algorithm for highly specific recognition of protein-coding regions," *Genome Informatics*, vol. 7, pp. 82-87, 1996.

[135] M. S. Gelfand, L. I. Podolsky, T. V. Astakhova, *et al.*, "Recognition of genes in human DNA sequences *" Journal of Computational Biology: a journal of computational molecular cell microbiology,* vol. 3, no. 2, pp. 223-234, 1996.

[136] Y. Xu, J. R. Einstein, R. J. Mural *et al.*, "An improved system for exon recognition and gene modeling in human DNA sequences," *In Proceedings of International Conference on Intelligent Systems for Molecular Biology; ISMB,* vol. 2, pp. 376-384, 1994.

[137] X. W. Chen, G. Anantha, and X. Wang, "An effective structure learning method for constructing gene networks," *Bioinformatics,* vol. 22, no. 11, pp. 1367-1374, 2006.

[138] B. Han, and X. W. Chen, "bNEAT: a Bayesian network method for detecting epistatic interactions in genome-wide association studies," *BMC Genomics,* vol. 12, no. Suppl 2, pp. S9, 2011.

[139] D. Heckerman, D. Geiger, and D. M. Chickering, "Learning Bayesian networks: The combination of knowledge and statistical data," *Machine Learning,* vol. 20, no. 3, pp. 197-243, 1995.

[140] V. Pavlovic, A. Garg, and S. Kasif, "A Bayesian framework for combining gene predictions," *Bioinformatics,* vol. 18, no. 1, pp. 19-27, 2002.

[141] G. B. Fogel, "Computational intelligence approaches for pattern discovery in biological systems," *Briefings in Bioinformatics,* vol. 9, no. 4, pp. 307-316, 2008.

[142] J. Kolodner, "An introduction to case-based reasoning," *Artificial Intelligence Review,* vol. 6, no. 1, pp. 3-34, 1992.

[143] A. Aamodt, and E. Plaza, "Case-based reasoning: foundational issues, methodological variations, and system approaches," *AI Communications,* vol. 7, no. 1, pp. 39-59, 1994.

[144] G. C. Overton, and J. Haas, "Chapter 5 Case-based reasoning driven gene annotation," *New Comprehensive Biochemistry,* vol. 32, pp. 65-86, 1998.

[145] G. C. Overton, and J. Haas, "Chapter 5 Case-based reasoning driven gene annotation," *New Comprehensive Biochemistry,* vol. 32, pp. 65-86, 1998.

[146] E. Costello, and D. C. Wilson, "A case-based approach to gene finding," *In Proceedings of the Fifth International Conference on Case-Based Reasoning Workshop on CBR in the Health Sciences,* pp. 19-28, 2003.

[147] A. Ram, and A. G. Francis, "Multi-plan retrieval and adaptation in an experience-based agent," *In Case-Based Reasoning: Experiences, Lessons, and Future Directions, AAAI Press,* 1996.

[148] C. J. Stoeckert, F. Salas, B. Brunk *et al.*, "EpoDB: a prototype database for the analysis of genes expressed during vertebrate erythropoiesis," *Nucleic Acids Research,* vol. 27, no. 1, pp. 200-203, 1999.

[149] I. Jurisica, and J. Glasgow, "Applications of case-based reasoning in molecular biology," *AI Magazine,* vol. 25, no. 1, pp. 85-95, 2004.

[150] S. Haykin, "Neural networks: a comprehensive foundation," *Second Edition, Pearson Education,* 1998.

[151] P. Werbos, "Beyond regression: new tools for prediction and analysis in the behavioral sciences," *Ph. D. Thesis, Harvard University,* 1974.

[152] P. J. Werbos, "The roots of backpropagation: From ordered derivatives to neural networks and political forecasting," *First Edition, Wiley-Interscience,* 1994.

[153] E. C. Uberbacher, Y. Xu, and R. J. Mural, "Discovering and understanding genes in human DNA sequence using GRAIL," *Methods in Enzymology,* vol. 266, pp. 259-281, 1996.

[154] G. B. Fogel, K. Chellapilla, D. B. Fogel *et al.*, "Identification of coding regions in DNA sequences using evolved neural networks," *Evolutionary Computation in Bioinformatics*, pp. 195-218, 2003.

[155] S. M. Hebsgaard, P. G. Korning, N. Tolstrup *et al.*, "Splice site prediction in Arabidopsis Thaliana Pre-mRNA by combining local and global sequence information," *Nucleic Acids Research*, vol. 24, no. 17, pp. 3439-3452, 1996.

[156] N. Tolstrup, P. Rouze, and S. Brunak, "A branch point consensus from Arabidopsis found by non-circular analysis allows for better prediction of acceptor sites," *Nucleic Acids Research*, vol. 25, no. 15, pp. 3159-3163, 1997.

[157] R. Ranawana, and V. Palade, "A neural network based multi-classifier system for gene identification in DNA sequences," *Neural Computing and Applications*, vol. 14, no. 2, pp. 122-131, 2005.

[158] A. Sherriff, and J. Ott, "Applications of neural networks for gene finding," *Advances in Genetics*, vol. 42, pp. 287-297, 2001.

[159] V. B. Bajic, and S. H. Seah, "Dragon gene start finder: An advanced system for finding approximate locations of the start of gene transcriptional units," *Genome Research*, vol. 13, no. 8, pp. 1923-1929, 2003.

[160] V. B. Bajic, and S. H. Seah, "Dragon gene start finder identifies approximate locations of the 5' ends of genes," *Nucleic Acids Research*, vol. 31, no. 13, pp. 3560-3563, 2003.

[161] M. J. Wood, and J. D. Hirst, "Recent applications of neural networks in bioinformatics," *Biological and Artificial Intelligence Environments*, pp. 91-97, 2005.

[162] L. Breiman, J. H. Friedman, C. J. Stone, *et al.*, "Classification and Regression Trees," 1984.

[163] S. Salzberg, "Locating protein coding region in human DNA using a decision tree algorithm," *Journal of computational biology: a journal of computational molecular cell biology*, no. 2, pp. 473-485, 1995.

[164] S. L. Salzberg, M. Pertea, A. L. Delcher *et al.*, "Interpolated Markov models for eukaryotic gene finding," *Genomics*, vol. 59, no. 1, pp. 24-31, 1999.

[165] M. Mitchell, "Genetic algorithms: An overview," *Complexity*, vol. 1, no. 1, pp. 31-39, 1995.

[166] A. Moore, "http://www.cs.cmu.edu/~awm," 2003.

[167] C. Cortes, and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273-297, 1995.

[168] N. Cristianini, and J. Shawe-Taylor, "An introduction to support vector machines," *Cambridge University Press*, 2000.

[169] H. Drucker, S. Wu, and V. N. Vapnik, "Support vector machines for spam categorization," *IEEE Transactions on Neural Networks*, vol. 10, pp. 1054-1084, 1995.

[170] F. Ojeda, J. A. K. Suykens, and B. De Moor, "Low rank updated LS-SVM classifiers for fast variable selection," *Neural Networks*, vol. 21, no. 2-3, pp. 437-449, 2008.

[171] J. C. Bezdek, and P. F. Castelaz, "Prototype classification and feature selection with fuzzy sets," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, no. 2, pp. 87-92, 1977.

[172] J. M. Keller, and H. Tahani, "Implementation of conjunctive and disjunctive fuzzy logic rules with neural networks," *International Journal of Approximate Reasoning*, vol. 6, no. 2, pp. 221-240, 1992.

[173] J. C. Bezdek, and S. K. Pal, "Fuzzy models for pattern recognition: methods that search for structures in data," *IEEE*, 1992.

[174] H.-J. Zimmermann, “Fuzzy set theory and its applications,” *4th edition, Springer*, 2001.

[175] D. Xu, R. Bondugula, M. Popescu, *et al.*, “Bioinformatics and fuzzy logic,” *IEEE International Conference on Fuzzy Systems*, pp. 817-824, 2006.

[176] L. J. Fogel, A. J. Owens, and M. J. Walsh, “Artificial intelligence through simulated evolution,” *John Wiley & Sons*, 1966.

[177] I. Rechenberg, “Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution, ” *Frommann-Holzboog*, 1973.

[178] J. R. Koza, “Genetic programming: on the programming of computers by means of natural selection,” *MIT Press*, 1992.

[179] R. C. Eberhart, Y. Shi, and J. Kennedy, “Swarm intelligence,” *Morgan Kaufmann*, 2001.

[180] M. Dorigo, and L. M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem,” *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, 1997.

[181] R. Storn, and K. Price, “Differential evolution- a simple and efficient adaptive scheme for global optimization over continuous spaces,” 1995.

[182] R. Storn, “System design by constraint adaptation and differential evolution,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 1, pp. 22-34, 1999.

[183] T. Back, D. B. Fogel, and Z. Michalewicz, "Handbook of evolutionary computation," *IOP Publishing Ltd.*, 1997.

[184] D. B. Fogel, “Evolutionary computation: toward a new philosophy of machine intelligence,” *3rd edition, IEEE Press*, 2005.

[185] G. B. Fogel, and D. W. Corne, “Evolutionary computation in bioinformatics,” *1st Edition, Morgan Kaufmann*, 2002.

[186] D. B. Fogel, “Evolutionary computation: the fossil record,” *Wiley-IEEE Press*, 1998.

[187] A. K. M. A. Baten, S. K. Halgamuge, and B. C. H. Chang, “Fast splice site detection using information content and feature reduction,” *BMC Bioinformatics*, vol. 9, no. suppl. 12:S8, 2008.

[188] M. J. Bishop, and C. J. Rawlings, “Nucleic acid and protein sequence analysis: a practical approach,” *2nd ed., Oxford University Press: USA*, 1995.

[189] M. McElwain, “A critical review of gene prediction software,” *Bioc 218 Final Paper*, 2007.

[190] M. Q. Zhang, "Using MZEF to find internal coding exons," *Current Protocols in Bioinformatics*, 2002.

[191] W. H. Majoros, M. Pertea, C. Antonescu *et al.*, “GlimmerM, Exonomy and Unveil: three ab-initio eukaryotic genefinders,” *Nucleic Acids Research*, vol. 31, no. 13, pp. 3601-3604, 2003.

[192] Z. Wang, Y. Chen, and Y. Li, “A brief review of computational gene prediction methods,” *Genomics Proteomics Bioinformatics*, vol. 2, no. 4, pp. 216-221, 2004.

[193] S. Maji, and D. Garg, “Gene finding using Hidden Markov Model,” *Journal of Applied Sciences*, vol. 12, no. 15, pp. 1518-1525, 2012.

[194] S. Dong, and D. B. Searls, “Gene structure prediction by linguistic methods,” *Genomics*, vol. 23, no. 3, pp. 540-551, 1994.

[195] Y. Ohshima, and Y. Gotoh, “Signals for the selection of a splice site in pre-mRNA: Computer analysis of splice junction sequences and like sequences,” *Journal of Molecular Biology*, vol. 195, no. 2, pp. 247-259, 1987.

[196] T.-M. Chen, C.-C. Lu, and W.-H. Li, “Prediction of splice sites with dependency graphs and their expanded bayesian networks,” *Bioinformatics,* vol. 21, no. 4, pp. 471-482, 2005

[197] S. Wu, and M. R. Green, “Identification of a human protein that recognizes the 3' splice site during the second step of pre-mRNA splicing,” *The EMBO Journal,* vol. 16, no. 14, pp. 4421-32, 1997.

[198] R. Reed, “Initial splice-site recognition and pairing during pre-mRNA splicing,” *Current Opinion in Genetics & Development,* vol. 6, no. 2, pp. 215-220, 1996.

[199] S. Degroeve, Y. Saeys, B. De Baets *et al.*, “SpliceMachine: predicting splice sites from high-dimensional local context representations,” *Bioinformatics,* vol. 21, no. 8, pp. 1332-1338, 2005.

[200] T. L. Bailey, M. E. Baker, and C. P. Elkan, “An artificial intelligence approach to motif discovery in protein sequences: Application to steroid dehydrogenases,” *The Journal of Steroid Biochemistry and Molecular Biology,* vol. 62, no. 1, pp. 29-44, 1997.

[201] G. J. McLachlan, and T. Krishnan, "The EM algorithm and extensions," *Wiley-Interscience: John Wiley & Sons,* 2008.

[202] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *Journal of the Royal Statistical Society, Series B,* vol. 39, no. 1, pp. 1-38, 1977.

[203] T. K. Moon, “The expectation-maximization algorithm,” *Signal Processing Magazine, IEEE,* vol. 13, no. 6, pp. 47-60, 1996.

[204] A. Baten, S. K. Halgamuge, and B. C. H. Chang, “Fast splice site detection using information content and feature reduction,” *BMC Bioinformatics.*, vol. 9, no. 12, pp. s8, 2008.

[205] G. Yeo, and C. B. Burge, “Maximum entropy modeling of short sequence motifs with applications to RNA splicing signals” *J Comput Biol.*, vol. 11, no. (2-3), pp. 377-394, 2004.

[206] L. Jing, H. Hua, and L. Sufang, “Voice Identification Based on HMMs. ,” *Trends in Applied Sciences Research, ,* vol. 1, pp. 79-82, 2006.

[207] T. A. Thanaraj, and F. Clark, “Human GC-AG alternative intron isoforms with weak donor sites show enhanced consensus at acceptor exon positions,” *Nucleic Acids Research,* vol. 29, no. 12, pp. 2581-2593, 2001.

[208] A. Malousi, I. Chouvarda, V. Koutkias *et al.*, “SpliceIT: A hybrid method for splice signal identification based on probabilistic and biological inference,” *Journal of Biomedical Informatics,* vol. 43, no. 2, pp. 208-217.

[209] M. G. Reese, “Improved splice site detection in Genie,” *Journal of Computational Biology,* vol. 4, no. 3, pp. 311-323, 1997.

[210] Q. Zhang, Q. Peng, Q. Zhang *et al.*, “Splice sites prediction of Human genome using length-variable Markov model and feature selection,” *Expert Systems with Applications,* vol. 37, no. 4, pp. 2771-2782, 2010.

[211] A. Shamshad, M. A. Bawadi, W. M. A. Wan Hussin *et al.*, “First and second order Markov chain models for synthetic generation of wind speed time series,” *Energy,* vol. 30, no. 5, pp. 693-708, 2005.

[212] R. Durbin, S. R. Eddy, A. Krogh, G. Mitchison, “Biological sequence analysis: probabilistic models of proteins and nucleic acids," Cambridge University Press, 1998.

[213] I. Guyon, and A. Elisseeff, “An introduction to variable and feature selection,” *Journal of Machine Learning Research,* vol. 3, no. 3, pp. 1157-1182, 2003.

[214] J. Neumann, C. Schnörr and G. Steidl, “Combined SVM-based feature selection and classification,” *Mach Learn* vol. 61(1-3), pp. 129-150, 2005.

[215] A. Malousi, I. Chouvarda, V. Koutkias, S. Kouidou, and N. Maglaveras, “Variable-length positional modeling for biological sequence classification," *Proceedings of the American medical informatics association symposium (AMIA)*, p. 91-5.. 2008

[216] Y. Lu, I. Cohen, X. S. Zhou, T. S. Huang, “Feature selection using principal feature analysis," *Proceedings of the 15th international conference on Multimedia*," New York, pp. 301-304, 2007.

[217] Y. W. Chen, C. J. Lin, “Combining SVMs with various feature selection strategies,” *Feature extraction: foundations and applications. Springer-Verlag*, pp. 315-23, 2006.

[218] H. Drucker, S. Wu, V. N. Vapnik, “Support vector machines for spam categorization,” *IEEE Neural Network*, vol. 10, no. 5, pp. 1048-1054, 1999.

[219] S. S. Haykin, *Neural Networks: Comprehensive Foundation*: Prentice Hall, 1999.

[220] A. Ben-Hur, C. S. Ong, S. Sonnenburg *et al.*, “Support vector machines and kernels for computational biology,” *PLoS computational biology*, vol. 4, no. 10, pp. 100-173, 2008.

[221] S. R. Rivard, J. G. Mailloux, R. Beguenane, H. T. Bui, “Design of high-performance parallelized gene predictors in MATLAB,” *BMC Res Notes*, vol. 5, no. 183, 2012.

[222] B. Matthews, “Comparison of the predicted and observed secondary structure of T4 phage lysozyme,” *Biochim Biophys Acta*, vol. 405, no. 2, pp. 442 - 451, 1975.

[223] G. Y. C. B. Burge, “Maximum Entropy Modeling of Short Sequence Motifs with Applications to RNA Splicing Signals,” *Journal of Computational Biology*, vol. 11(2-3), pp. 377-394, 2004.

[224] S. Arlot, and A. Celisse, “A survey of cross-validation procedures for model selection,” *Statistics Surveys*, vol. 4, pp. 40-79, 2010.

[225] Y. Zhang, C. H. Chu, Y. Chen *et al.*, “Splice site prediction using support vector machines with a Bayes kernel,” *Expert Systems with Applications*, vol. 30, no. 1, pp. 73-81, 2006.

[226] A. K. M. A. Baten, S. K. Halgamuge and J. Li, “Splice site identification using probabilistic parameters and SVM classification,” *BMC Bioinformatics*, vol. 8, pp. 241, 2007.