Progress in Gene Prediction: Principles and Challenges

Srabanti Maji and Deepak Garg*

Department of Computer Science and Engineering, Thapar University, Patiala-147004, India

Abstract: Bioinformatics is a promising and innovative research field in 21st century. Automatic gene prediction has been an actively researched field of bioinformatics. Despite a high number of techniques specifically dedicated to bioinformatics problems as well as many successful applications, we are in the beginning of a process to massively integrate the aspects and experiences in the different core subjects such as biology, medicine, computer science, engineering, chemistry, physics, and mathematics. Presently, a large number of gene identification tools are based on computational intelligence approaches. Here, we have discussed the existing conventional as well as computational methods to identify gene(s) and various gene predictors are compared. The paper includes some drawbacks of the presently available methods and also, the probable guidelines for future directions are discussed.

Keywords: Bioinformatics, gene identification, DNA, content sensor, splice site, dynamic programming, neural networks, SVM.

1. INTRODUCTION

In the field of bioinformatics, gene identification from large DNA sequence is known to be a significant setback. The human genome project was completed in April 2003, the exact number of genes encoded by the human genome is still unknown [1, 2]. Hence, 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 [3]. 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 [4]. Consequently, 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 [2, 5, 6]. The methods for gene-finding which are being used now a day are more precise and reliable as well than the earlier tactics. The advances in gene finding through dynamic programming, decision trees and Hidden Markov Model (HMM), are also studied [7]. The available gene prediction programs and methods are also reported and summarized [8-10]. The existing methods for gene prediction are also studied and compared [10, 11]. A comprehensive review of prediction methods for functional sites, protein coding genes, tRNA etc. is also reported [12]. A summary of a few techniques based on computational gene identification tools is also reported [13]. Catherine provided a review of the existing approaches of gene identification in eukaryotic organisms, their advantages as well as the limitations [14-16]. 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 [17]. 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 [18]. Other approaches consisting of multiple evidence types can be found in the Eu-Gene [19] and GAZE [20] 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.

In this paper, 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. In most of the previous reviews on this topic, the drawbacks of the classical methods are not described. We have highlighted the recent developments in gene identification tools, especially those based on computational intelligence techniques like Neural Networks and Genetic Algorithms.

2. BACKGROUND AND LACUNAE

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

*Address correspondence to this author at the Department of Computer Science and Engineering, Thapar University, Patiala-147004, India;
Tel: +91-175-2393007, +91-9815599654; Fax: +91-175-2393005;
E-mail: dgarg@thapar.edu

1574-8936/13 $58.00+.00

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 [21].

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 the basic monomer building block units in the nucleic acids [22]. 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 [23]. Intergenic regions constitute approx. 95% of the genomic strand and protein coding genes constitute 2% of the total DNA [24, 25]. 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 [24, 26]. Eukaryotic gene structure is shown in Fig. (**1**).

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 [27]. All these processes are depicted in Fig. (**2**). 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 [6].

2.1. Branches and Uses of Bioinformatics

As shown in Fig. (**3**) [1], there are three basic and closely related branches of bioinformatics, namely –

1. Genomics,
2. Transcriptomics, and
3. Proteomics

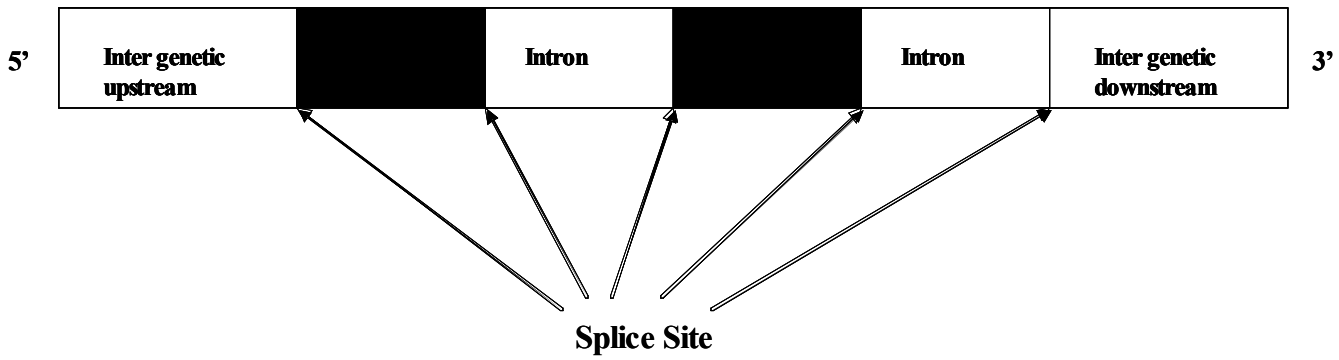

Fig. (1). Eukaryotic gene structure.

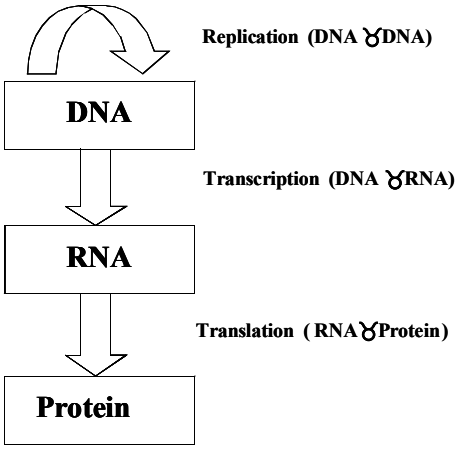

Fig. (2). Central dogma of biology.

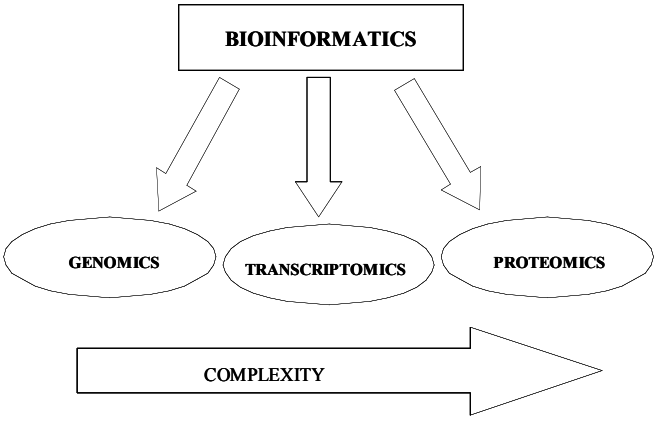

Fig. (3). Major branches of bioinformatics.

2.1.1. Genomics

It play an important role in modern biological research. This involves broad analysis of nucleic acids through molecular biological techniques, before the data are ready for processing by computers. 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 [1].

2.1.2. Transcriptomics

Transcriptomics is the study of transcriptome that depicts the expression level of genes, frequently using techniques capable of sampling tens of thousands of different mRNA molecules at a time (e.g. DNA microarrays). Transcriptome is the set of all mRNA molecules (or transcripts) in one or a population of biological cells for a given set of environmental circumstances.

2.1.3. Proteomics

Proteomics is the field, to study the structure and functions of the proteins. Proteomics involves the sequencing of amino acids in a protein, determining its three-dimensional structure and relating it to the function of the protein. It signifies the initial attempt to identify a major subclass of cellular components, the proteins and their interactions. The word "proteome" is a mixture of "protein" and "genome", and was credited by with the Marc Wilkins in 1994. It is often specifically used for protein purification and mass spectrometry [27].

This field of science has many applications and research areas where it can be applied. Bioinformatics is majorly being used in drug development, molecular medicine, preventative medicine, gene therapy, microbial genome applications, sequence analysis etc.

Overview of various techniques for protein-coding gene prediction is the main focus of our article. A completely different class of algorithms is used for non-coding RNA gene finding which is outside the purview of this article.

2.2. Gene Identification Problem

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 [28]. The identification of protein coding genes is noticeably influenced by the knowledge of other significant features of the sequence, the complexity of considering the automatic annotation problem as a logically integrated process has caused the gene identification problem usually to be considered independently of most other sequence analysis.

Eukaryotic gene regulation is 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.

Computational gene prediction problem can be defined as:

We provide a DNA sequence as input

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

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

Accurate labeling of each element in S as belongs to a coding region (exon), intergenic region or non-coding region (intron) [3].

The basic characteristic of a eukaryotic gene is the organization of its structure into introns and exons. In general, exons can be separated into four types – 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 [6]. 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 [6]. 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 [29].

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. Moreover, some of the codons do not code for any amino acid, they just signal some special event. For example, start and stop codons signal the beginning and end of translation. In vertebrates, the internal exons are very small (typically about 140 nucleotides long), while the introns are much larger (some being more than 100 kb long) [6]. 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 [30]. These and several other issues make the task of identifying genes extremely challenging.

3. 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 Fig. (**4**) [14].

3.1. 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.

3.1.1. Content Sensors

Content sensors are frequencies of various kinds, e.g. nucleotide, dinucleotide and trinucleotide frequencies, e.g. in

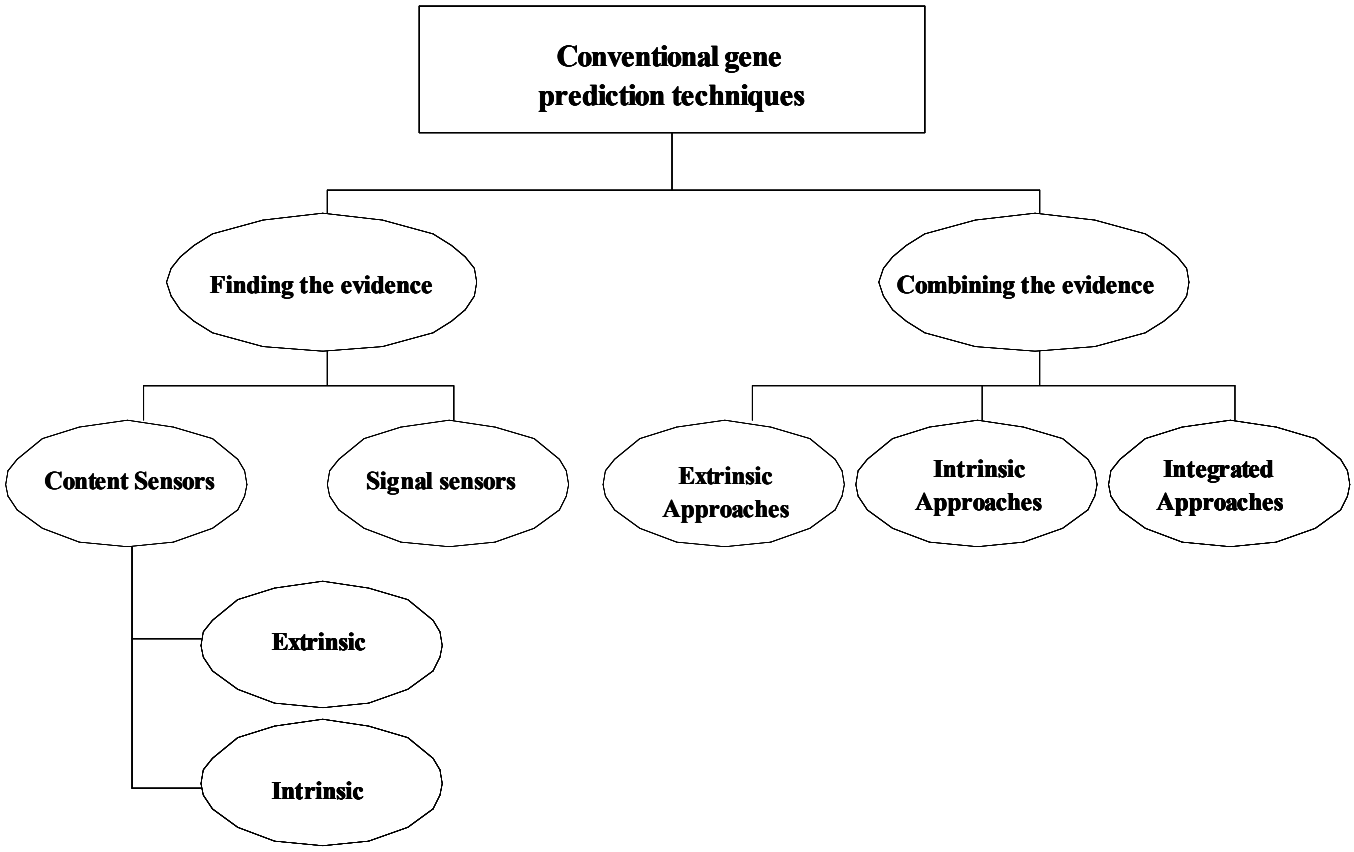

Fig. (4). Gene prediction using conventional techniques.

exons and introns, triplet frequencies or nucleotide frequencies at three different positions of the triplet are usually dissimilar. Content sensor does not include site specific information [31]. Content sensors try to classify a DNA region into different types, e.g. coding versus non-coding [14]. These can be of two types – extrinsic content sensors and intrinsic content sensors.

3.1.1.1. 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 [32], FASTA [33] and BLAST [34] 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.

3.1.1.2. 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 [35]. 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 [36].

3.1.2. 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 [14]. 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 [37] and a logitlinear model based approach (SplicePredictor) [38]. 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 perposition. 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 [39] and later used by Salzberg [40], who applied it in the VEIL [41] and MORGAN [42] software. A modified WAM is also used in Genscan [43] 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 **1**.

3.2. 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 [14]. 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 [15] and of Gelfand [16] 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.

3.2.1. 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 [44] 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 [45]. 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 [46], which produces a most likely gene structure and is known to be a specific instance of the older Bellman shortest path algorithm [47]. This approach is adopted in the programs viz. GeneId [48], GenView [49], GAP III [50], FGENES [51] and DAGGER [52]. They are presented in Table **2**.

Table 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.

Table 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.

3.2.2. Extrinsic/Homology Approaches

Several programs based on similarity searches have emerged during the last decade lead by Gelfand *et al*. with PROCRUSTES [53]. 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. Summarily, 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**.

3.2.3. 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 [54]. Considering this approach, Gene Structure Assembly (GSA) program evolved by combining AAT [55] and Genscan whose results are better than those obtained from these programs used

Table 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.

separately. In this regard, TWINSCAN [56] is an important program worked upon. GenomeScan [57] 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 [58] program, which is a combination of NetGene2 and SplicePredictor for splice site prediction, NetStart [59] 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. In recent times, a method is proposed [18] 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.

4. 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.

4.1. 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 [60]. 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 $X = \text{A, T, G or C}$. By using a Markov model, one can then simply compute the probability of the sequence generated according to this model [61]. HMM are more successful because they can naturally accommodate uneven length models of sequence regions because maximum biological data has variable length properties [62, 63]. They are used for motif finding [64], multiple sequence alignment [65] and identification of protein structure [66]. As shown in Fig. (**5**), 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 [43], Genie [67] and HMM Gene [68].

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:

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

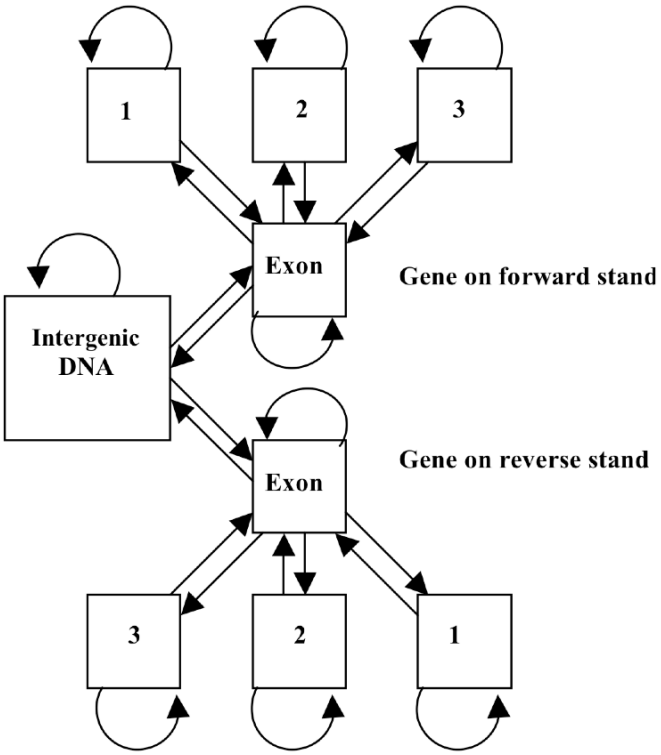

Fig. (5). Hidden Markov Models for Gene in DNA sequence. The HMM is divided into 2 parts: genes on forward or reverse stand of DNA sequence. Every gene model contains a central exon state which has an emission of nucleotides tuned to recognize protein coding regions. Introns are interrupting exons, three 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.

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 [69]. 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 [69].

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 [70]. The Viterbi and Expectation Maximization (EM) algorithms are used for computing with HMM during its training and testing [46, 71]. 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 [72] 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 [73]. 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 [74].

4.2. Dynamic Program 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 [75] 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 [76], 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 [77]. GREAT [78] is a program to identify protein-coding segments in the DNA sequence. The GenView system [49] 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 [79], GeneParser [75], FGENESH [51], GAP III [50] and recent versions of GeneId [48] also use dynamic programming approach. A brief review of the dynamic programming in gene finding is provided [9].

4.3. 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 [80, 81]. Over the last decade, the Bayesian network has become a popular representation for encoding uncertain expert knowledge in expert systems [82]. An example of a Bayesian network is shown in Fig. (**6**). 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.

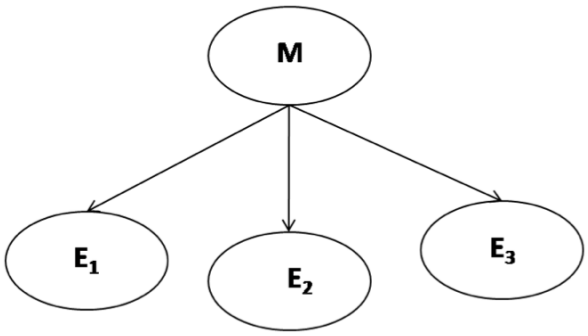

Fig. (6). An Example of a Bayesian Network.

In this example, one such measure is probability $P(E_1 \mid M)$. This model defines a specific factorization of the joint probability distribution function over the variables in the network. Hence, Fig. (**5**) defines $P(M, E_1, E_2, E_3)$ as

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

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 [82]. A Bayesian network framework for combining gene predictions from multiple systems is given [83], where the approach adopted is that of combining the advice of multiple experts.

5. 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 [84]. 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.

5.1. Case Based Reasoning (CBR)

Case-based reasoning (CBR) is the process of solving new problem based on the solution of past similar problem. In other words, in CBR, a reasoned remembers a previous situation that is similar to the current one and uses those experiences to solve a new problem [85]. 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 [86] –

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 [87]. 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 [88] 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 [89] 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 [90]. Three measures for classifying an individual nucleotide was 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 [91] 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 [92].

5.2. 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 [93]. 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.

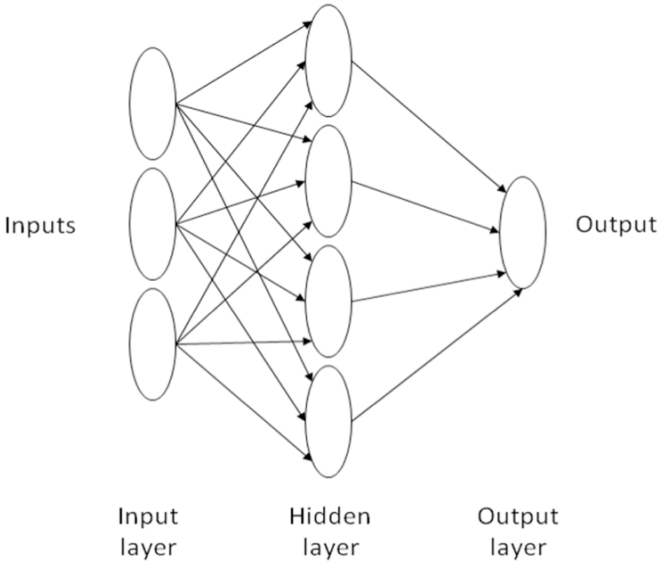

Fig. (7). A typical artificial neural network representation consists of input layer, hidden layer and output layer.

Multilayer perceptrons, also known as feedforward 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-organising 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 Fig.

(**7**). 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 [94, 95]. 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 [96]. Fickett and Tung [36] 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 [97]. 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 [98].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 [99].

For the analysis of Drosophilia melanogaster genome a time delay architecture which is based on feed forward neural network has applied [100]. The E. Coli gene prediction by locating the promoters of genes are performed by neural network based multi-classifier system [101].There are similar other applications of neural network for gene identification [102].The gene identification tool Dragon Gene Start Finder (DGSF) [103, 104] 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 [105].

5.3. 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 [106]. They accurately differentiate between coding and non-coding DNA for sequences ranging from 54 to 162 base pairs in length [107]. 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 [42] 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 [108] 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.

5.4. Genetic Algorithms

Genetic Algorithms (GAs) are computer programs that impersonate the processes of biological evolution to solve the problems and to model evolutionary systems. First proposed by J. H. Holland, the genetic algorithm applies the principles of natural evolution to finding an optimal solution to an optimization problem [109]. 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 [110, 111] 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.

A comparison of GA with Simulated Annealing (SA) is done by Gunnels *et al*. [110] and found that the GA based method rapidly converges to a good solution as it is able to take the benefit from the extra information to create superior local maps which are useful in constructing good global maps. Using a GA, Alexander *et al*. [112] 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 [113] 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. A comparative study between the performance of evolved ANN and GRAIL, in terms of correlation coefficient (CC), sensitivity, specificity, fraction of exactly predicted exons (including start and stop codons) and fraction of predicted exons that overlap with actual exons.

5.5. Support Vector Machine

Support Vector Machines (SVMs) are the set of related supervised learning methods used for classification and regression [114]. They 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 [115]. 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. The SVMs have been developed by Vapnik [114] 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 [116] to traditional Empirical Risk Minimization (ERM) principle, used by conventional neural networks. SRM minimizes an upper bound on the expected risk, where as ERM minimizes the error on the training data. It is this difference which equips SVM with a greater ability to generalize, which is the goal in statistical learning. SVMs were developed to solve the classification problem, but recently they have been extended to solve regression problems [117]. 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 [118-120]. 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 [116]. Fig. (**8**) shows support vector machine with hyperplane and margin.

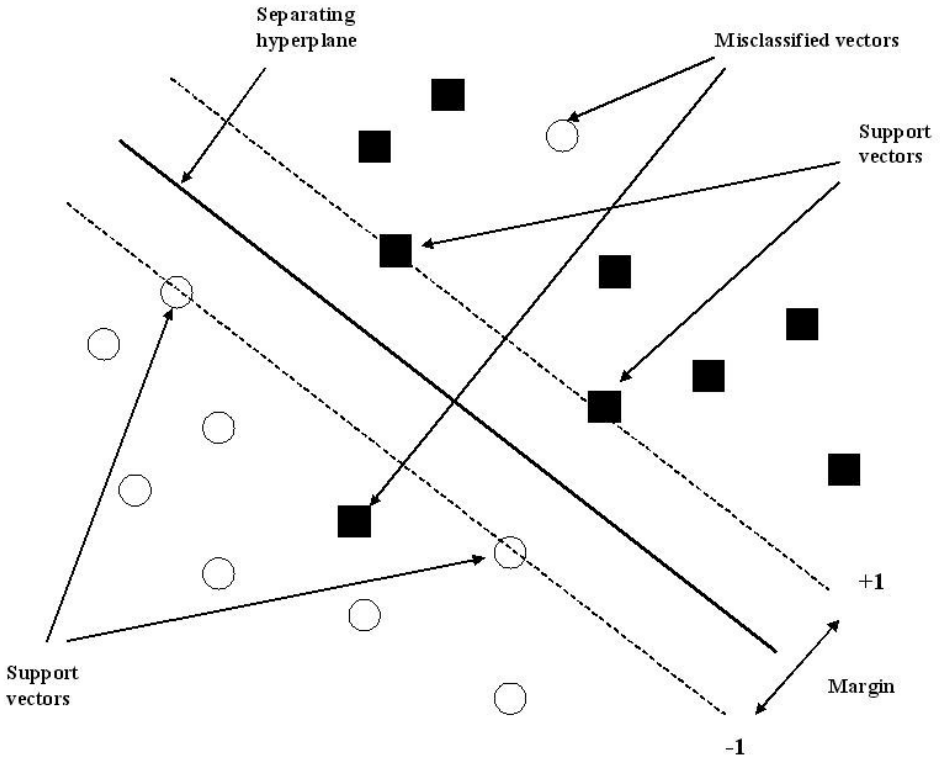

Fig. (8). Support Vector Machine (SVM) with hyperplane and margin.

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 [121].

5.6. Fuzzy System

The concepts of fuzzy sets and fuzzy logic were introduced in the 1960s by Lotfi A. Zadeh [122, 123] 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. In contrast with the traditional logic theory where binary sets have two-valued logic, true or fals, 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 [124] and, used to manage uncertainty in rule-based representations, and rule conflict resolution [125] 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 [126-128].

5.7. Evolutionary Computation

Traditionally, there are many subdivisions of evolutionary computation or evolutionary programming (EP) [129] and evolution strategies (ES) [130]. 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) [131] which represents individuals as tree structures of mathematical expressions, particle swarm optimization (PSO) [132] 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) [133] 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) [134, 135] 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 [136, 137] 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 Fig. (**9**).

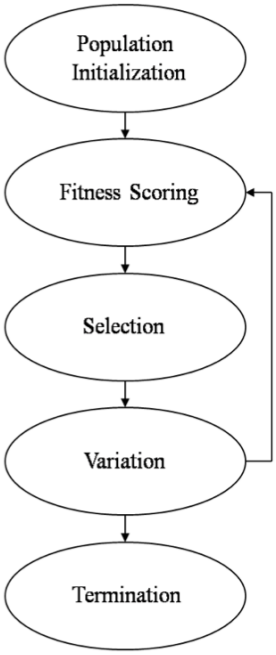

Fig. (9). A standard flowchart for evolutionary algorithms.

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 [138, 139].

5.8. 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 [9, 11]. 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) [140] as shown in Table **4**.

Table 4. Definition of TP, TN, FP and FN

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

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 [96]. 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 [141].

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 [142] 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%) [143]. GENSCAN was found a little bit better than MZEF in terms of wrong exons and missed exons scores. William *et al*. [144] 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 [9, 142].

6. CONCLUSIONS

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 exist 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.

CONFLICT OF INTEREST

The authors confirm that this article content has no conflict of interest.

ACKNOWLEDGEMENT

We thankfully acknowledge the information and the comments of the reviewers which helped in improving the manuscript's quality.

REFERENCES

[1] Ghosh Z, Mallick B. Bioinformatics Principles and Applications. 2nd ed. Oxford University Press: Delhi 2009.
[2] van Wieringen WN, Kun D, Hampel R, Boulesteix AL. Survival prediction using gene expression data: A review and comparison. Comput Stat Data Anal 2009; 53(5): 1590-1603.
[3] Bandyopadhyay S, Maulik U, Roy D. Gene identification: classical and computational intelligence approaches. IEEE Trans Syst Man Cybern Part C Appl Rev 2008; 38(1): 55-68.
[4] Stein LD. End of the beginning. Nature 2004; 431(7011): 915-916.
[5] Stormo GD. Gene-finding approaches for eukaryotes. Genome Res 2000; 10(4): 394-397.
[6] Zhang MQ. Computational prediction of eukaryotic protein-coding genes. Nat Rev Genet 2002; 3(9): 698-709.
[7] Cawley SL, Pachter L. HMM sampling and applications to gene finding and alternative splicing. Bioinformatics 2003; 19 (Suppl 2): ii36-ii41.
[8] Kumar M, Raghava GP. Prediction of nuclear proteins using SVM and HMM models. BMC Bioinformatics 2009; 10: 22.
[9] Claverie JM. Computational methods for the identification of genes in vertebrate genomic sequences. Hum Mol Gen 1997; 6(10): 1735-1744.
[10] Eddy SR. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure. BMC Bioinformatics 2002; 3:18.
[11] Burset M, Guigo R. Evaluation of gene structure prediction programs. Genomics 1996; 34(3): 353-367.
[12] Gelfand MS. Prediction of function in DNA sequence analysis. J Comput Biol J Comput Mol Cell Biol 1995; 2(1): 87-115.
[13] Haussler D. Computational genefinding. TIBS, Suppl Guide Bioinform 1998; 23(1): 12-15.
[14] Mathe C, Sagot MF, Schiex T, Rouze P. Current methods of gene prediction, their strengths and weaknesses. Nucleic Acids Res 2002; 30(19): 4103-4117.
[15] Fields CA, Soderlund CA. Gm: a practical tool for automating DNA sequence analysis. Comput Appl Biosci 1990; 6: 263-270.
[16] Gelfand MS. Computer prediction of the exon-intron structure of mammalian pre-mRNAs. Nucleic Acids Res 1990; 18: 5865-5869.
[17] Flicek P. Gene prediction: Compare and CONTRAST. Genome Biol 2007; 8(12):233.
[18] Allen JE, Pertea M, Salzberg SL. Computational gene prediction using multiple sources of evidence. Genome Res 2004; 14(1): 142-148.
[19] Schiex T, Moisan A, Rouze P. EuGène: An eucaryotic gene finder that combines several sources of evidence. Lect Notes Comput Sc 2001; 2066: 111–125.

[20] Howe KL, Chothia T, Durbin R. GAZE: A generic framework for the integration of gene-prediction data by dynamic programming. Genome Res 2002; 12: 1418–1427.

[21] Simmons MP, Ochoterena H, Freudenstein JV. Amino acid *vs* nucleotide characters: challenging preconceived notions. Mol Phylogenet and Evol 2002; 24(1): 78–90.

[22] Bailey JE, Ollis DF. Biochemical Engineering Fundamentals. 2nd ed. McGraw-Hill: New York 1986.

[23] Marhon SA, Kremer SC. Gene prediction based on DNA spectral analysis:A Literature Review. J Comput Biol 2011; 18(4): 639–676.

[24] Akhtar M, Ambikairajah E, Epps J, *et al.* Digital signal processing techniques for gene finding in eukaryotes. Lect Notes Comput Sc 2008; 5099: 144–152.

[25] Gibbs WW. The unseen genome: beyond DNA. Sci Am 2003; 289(6): 108–113.

[26] Fox T, Carreira A. A digital signal processing method for gene prediction with improved noise suppression. Eurasip J Adv Sig Pr 2004; 1: 108–114.

[27] Shuler ML, Kargi F. Bioprocess Engineering: Basic Concepts. 2nd ed. Pearson Education: New Delhi 2007.

[28] Fickett JW. The gene identification problem: An overview for developers. Comput Biol Chem 1996; 20(1): 103-l18.

[29] Brent MR. Predicting full-length transcripts. Trends Biotechnol 2002; 20(7): 273–275.

[30] Datta S, Asif A, Wang H. Prediction of protein coding regions in DNA sequences using Fourier spectral characteristics. Proc IEEE Sixth Intl Symp Multimedia Software Eng 2004; 160–163.

[31] Xia X. Bioinformatics and the cell: Modern Computational Approaches in Genomics, Proteomics and Transcriptomics. Springer: New York 2007.

[32] Smith TF, Waterman MS. Identification of common molecular subsequences. J Mol Biol 1981; 147(1): 195-197.

[33] Pearson WR, Lipman DJ. Improved tools for biological sequence comparison. Proc Natl Acad Sci USA 1988; 85(8): 2444-2448.

[34] Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ. Basic local alignment search tool. J Mol Biol 1990; 215(3): 403-410.

[35] Fickett JW. ORFs and genes: how strong a connection? J Comput Biol 1995; 2(1): 117-123.

[36] Fickett JW, Tung CS. Assessment of protein coding measures. Nucleic Acids Res 1992; 20(24): 6441-6450.

[37] Rogozin I, Milanesi L. Analysis of donor splice sites in different eukaryotic organisms. J Mol Evol 1997; 45(1): 50-59.

[38] Kleffe J, Hermann K, Vahrson W, Wittig B, Brendel V. Logitlinear models for the prediction of splice sites in plant pre-mRNA sequences. Nucleic Acids Res 1996; 24(23): 4709-4718.

[39] Zhang MQ, Marr TG. A weight array method for splicing signal analysis. Comput Appl Biosci 1993; 9(5): 499-509.

[40] Sachem SI. A method for identifying splice sites and translational start sites in eukaryotic mRNA. Comput Appl Biosci 1997; 13(4): 365-376.

[41] Henderson J. Finding genes in DNA with a Hidden Markov Model. J Comput Biol 1997; 4(2): 127-141.

[42] Salzberg S, Delcher AL, Fasman KH, Henderson J. A decision tree system for finding genes in DNA. J Comput Biol 1998; 5(4): 667-680.

[43] Burge C, Karlin S. Prediction of complete gene structures in human genomic DNA. J Mol Biol 1997; 268(1): 78-94.

[44] Roytberg MA, Astakhova TV, Gelfand MS. Combinatorial approaches to gene recognition. Comput Biol Chem 1997; 21(4): 229-235.

[45] Guigo R. Assembling genes from predicted exons in linear time with dynamic programming. J Comput Biol 1998; 5(4): 681-702.

[46] Viterbi AJ. Error bounds for convolutional codes and an asymptotically optimum decoding algorithm. IEEE T Inform Theo 1967; 13(2): 260-269.

[47] Bellman RE. Dynamic Programming. Princeton University Press: Princeton New Jersey 1957.

[48] Guigo R, Knudsen S, Drake N, Smith T. Prediction of gene structure. J Mol Biol 1992; 226(1): 141-157.

[49] Milanesi L, Kolchanov NA, Rogozin IB, *et al.* GenView: A computing tool for protein-coding regions prediction in nucleotide sequences. Proceedings of the 2nd International Conference on Bioinformatics, Supercomputing, and Complex Genome Analysis 1993: 573-588.

[50] Xu Y, Mural RJ, Uberbacher EC. Constructing gene models from accurately predicted exons: An application of dynamic programming. Comput Appl Biosci 1994; 10(6): 613-623.

[51] Salamov AA, Solovyev VV. Ab initio gene finding in Drosophila genomic DNA. Genome Res 2000; 10(4): 516-522.

[52] Chuang JS, Roth D. Gene recognition based on DAG shortest paths. Bioinformatics 2001; 17(suppl 1): S56-S64.

[53] Gelfand MS, Mironov AA, Pevzner PA. Gene recognition *via* spliced sequence alignment. Proc Natl Acad Sci USA 1996; 93(17): 9061-9066.

[54] Tenney AE, Brown RH, Vaske C, *et al*. Gene prediction and verification in a compact genome with numerous small introns. Genome Res 2004; 14(11): 2330-2335.

[55] Huang X, Adams MD, Zhou H, Kerlavage AR. A tool for analyzing and annotating genomic sequences. Genomics 1997; 46(1): 37-45.

[56] Korf I, Flicek P, Duan D, Brent MR. Integrating genomic homology into gene structure prediction. Bioinformatics 2001; 17(Suppl. 1): S140-S148.

[57] Yeh RF, Lim LP, Burge CB. Computational inference of homologous gene structures in the human genome. Genome Res 2001; 11(5): 803-816.

[58] Schiex T, Moisan A, Duret L, Rouze P. EuGene: An eukaryotic gene finder that combines several sources of evidence. Lect Notes Comput Sc, Springer-Verlag 2000: 2066.

[59] Pedersen AG, Nielsen H. Neural network prediction of translation initiation sites in eukaryotes: perspectives for EST and genome analysis. Proc Intl Conf Intell Syst Mol Biol 1997; 5: 226-233

[60] Kouemou GL. History and Theoretical basic of Hidden Markov Models In Hidden Markov Models, Theory and Applications. Przemyslaw Dymarski (Ed) 2011.

[61] Koski T. Hidden Markov Models for Bioinformatics. Kluwer Academic Publishers 2002.

[62] Birney E. Hidden Markov Models in biological sequence analysis. IBM J Res Dev 2001; 45(3-4): 449-454

[63] Karplus K, Karchin R, Draper J, *et al*. Combining local-structure, fold-recognition, and new fold methods for protein structure prediction. Proteins 2003; 53(S6): 491-496.

[64] Bucher P, Karplus K, Moeri N, Hofmann K. A flexible motif search technique based on generalized profiles. Comput Biol Chem 1996; 20(1): 3-23.

[65] Sonnhammer ELL, Eddy SR, Birney E, Bateman A, Durbin R. Pfam: Multiple sequence alignments and HMM-profiles of protein domains. Nucleic Acids Res 1998, 26(1): 320-322.

[66] Di Francesco V, Munson PJ, Garnier J. FORESST: fold recognition from secondary structure predictions of proteins. Bioinformatics 1999; 15(2): 131-140.

[67] Kulp D, Haussler D, Reese MG, Eeckman FH. A generalized hidden Markov model for the recognition of human genes in DNA. Proc Intl Conf Intell Syst Mol Biol 1996; 4: 134-142.

[68] Krogh A. Two methods for improving performance of an HMM and their application for gene finding. Proc Intl Conf Intell Syst Mol Biol 1997; 5: 179-186

[69] De Fonzo V, Aluffi-Pentini F, Parisi V. Hidden Markov Models in Bioinformatics. Curr Bioinform 2007; 2(1): 49-61.

[70] Bilmes J. A gentle tutorial on the EM Algorithm and its application to parameter estimation for Gaussian mixture and Hidden Markov Models 1997.

[71] Cawley SE, Wirth AI, Speed TP. Phat - A gene finding program for Plasmodium falciparum. Mol Biochem Parasit 2001; 118(2): 167-174.

[72] Rabiner LR, Tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 1989; 77(2): 257-286.

[73] Xiaolin L, Parizeau M, Plamondon R. Training hidden Markov models with multiple observations-a combinatorial method. IEEE T Pattern Anal 2000; 22(4): 371-377.

[74] Pedersen JS, Hein J. Gene finding with a hidden Markov model of genome structure and evolution. Bioinformatics 2003; 19(2): 219-227.

[75] Snyder EE, Stormo GD. Identification of protein coding regions in genomic DNA. J Mol Biol 1995; 248(1): 1-18.

[76] Gelfand MS, Roytberg MA. Prediction of the exon-intron structure by a dynamic programming approach. Biosystems 1993; 30(1-3): 173-182.

[77] Gelfand MS, Astakhova TV, and Roytberg MA. An algorithm for highly specific recognition of protein-coding regions. Genome Inform 1996; 7: 82-87.

[78] Gelfand MS, Podolsky LI, Astakhova TV, Roytberg MA. Recognition of genes in human DNA sequences. J Comput Biol 1996; 3(2): 223-234.

[79] Xu Y, Einstein JR, Mural RJ, Shah M, Uberbacher EC. An improved system for exon recognition and gene modeling in human DNA sequences. Proc Second Intl Conf ISMB 1994; 2: 376-384.

[80] Chen XW, Anantha G, Wang X. An effective structure learning method for constructing gene networks. Bioinformatics 2006; 22(11): 1367-1374.

[81] Han B, Chen XW. bNEAT: a Bayesian network method for detecting epistatic interactions in genome-wide association studies. BMC Genomics 12(Suppl 2): S9.

[82] Heckerman D, Geiger D, Chickering DM. Learning Bayesian networks: The combination of knowledge and statistical data. Mach Learn 1995; 20(3): 197-243.

[83] Pavlovic V, Garg A, Kasif S. A Bayesian framework for combining gene predictions. Bioinformatics 2002; 18(1): 19-27.

[84] Fogel GB. Computational intelligence approaches for pattern discovery in biological systems. Briefings Bioinforms 2008; 9(4): 307-316

[85] Kolodner JL. An Introduction to Case-Based Reasoning. Artif Intell Rev 1992; 6(1): 3-34.

[86] Aamodt A, Plaza E. Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Commun 1994; 7(1): 39-59.

[87] Overton GC, Haas J, Salzberg SL, Searls DB, Simon K. Chapter 5 Case-based reasoning driven gene annotation. New Compr Biochem Elsevier 1998; 32: 65-86.

[88] Overton CG, Haas J. Chapter 5 Case-based reasoning gene annotation. Comput Methods Mol Biol Elsevier Sci 1998.

[89] Costello E, Wilson DC. A case-based approach to gene finding. 5th Intl Conf Case-Based Reason 2003: 19-28.

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

[91] Stoeckert CJ, Salas F, Brunk B, Overton GC. EpoDB: a prototype database for the analysis of genes expressed during vertebrate erythropoiesis. Nucleic Acids Res 1999; 27(1): 200-203.

[92] Jurisica I, Glasgow J. Applications of case-based reasoning in molecular biology. AI Mag 2004; 25(1): 85-95.

[93] Haykin S. Neural Networks: A Comprehensive Foundation 2nd ed. Prentice Hall, 1998.

[94] Werbos P. Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences. Harvard University 1974.

[95] Werbos P. The Roots of Backpropagation: From Ordered Derivatives to Neural Networks and Political Forecasting 1st ed. Wiley-Interscience 1994.

[96] Uberbacher EC, Xu Y, Mural RJ. Discovering and understanding genes in human DNA sequence using GRAIL. Method Enzymol 1996; 266: 259-281.

[97] Fogel GB, Chellapilla K, Fogel DB, Gary BF, David WC. Identification of Coding Regions in DNA Sequences Using Evolved Neural Networks. Evol Comput Bioinform 2003; 1-8: 195-218.

[98] Hebsgaard SM, Korning PG, Tolstrup N, Engelbrecht J, Rouze P, Brunak S. Splice Site Prediction in Arabidopsis Thaliana Pre-mRNA by Combining Local and Global Sequence Information. Nucleic Acids Res 1996; 24(17): 3439-3452.

[99] Tolstrup N, Rouze P, Brunak S. A branch point consensus from Arabidopsis found by non-circular analysis allows for better prediction of acceptor sites. Nucleic Acids Res 1997; 25(15): 3159-3163.

[100] Reese MG. Application of a time-delay neural network to promoter annotation in the Drosophila melanogaster genome. Comput Biol Chem 2001; 26(1): 51-56.

[101] Ranawana R, Palade V. A neural network based multi-classifier system for gene identification in DNA sequences. Neural Comput Appl 2005; 14(2): 122-131.

[102] Sherriff A, Ott J. 20 Applications of neural networks for gene finding. Adv Genet 2001; 42: 287-297.

[103] Bajic VB, Seah SH. Dragon gene start finder: An advanced system for finding approximate locations of the start of gene transcriptional units. Genome Res 2003; 13(8): 1923-1929.

[104] Bajic VB, Seah SH. Dragon Gene Start Finder identifies approximate locations of the 5' ends of genes. Nucleic Acids Res 2003; 31(13): 3560-3563.

[105] Wood MJ, Hirst JD. Recent applications of neural networks in bioinformatics. Biological and Artificial Intelligence Environments, Springer 2005: 91-97.

[106] Breiman L, Friedman JH, Olshen RA, Stone CJ. Classification and Regression Trees 1984.

[107] Salzberg S. Locating protein coding region in human DNA using a decision tree algorithm. J Comput Biol 1995; 2(3): 473-485.

[108] Salzberg SL, Pertea M, Delcher AL, Gardner MJ, Tettelin H. Interpolated Markov models for eukaryotic gene finding. Genomics 1999; 59(1): 24-31.

[109] Holland JH. Adaptation in Natural and Artificial System. MIT Press Cambridge, MA, USA 1975.

[110] Gunnels JA, Cull P, Holloway J. Genetic Algorithms and Simulated Annealing for Gene Mapping. Proceedings of 1st IEEE C Evol Comput 1994: 385-390.

[111] Mitchell M. Genetic Algorithms: An Overview. Complexity 1995; 1(1): 31-39.

[112] Kel A, Ptitsyn A, Babenko V, Meier-Ewert S, Lehrach H. A genetic algorithm for designing gene family-specific oligonucleotide sets used for hybridization: the G protein-coupled receptor protein superfamily. Bioinformatics 1998; 14(3): 259-270.

[113] Victor GL, Alexey VK. Recognition of Eukaryotic Promoters Using a Genetic Algorithm Based on Iterative Discriminant Analysis. In Silico Biol 2003; 3(1): 81-87.

[114] Vapnik V. The Nature of Statistical Learning Theory. Springer: New York 1995.

[115] Mitchell TM, Moore AW. Machine Learning, 10-701 and 15-781, School of Computer Science, Carnegie Mellon University, Pennsylvania, USA. http://www.cs.cmu.edu/~awm/15781/2003/ (Accessed June 17, 2012)

[116] Burges CJC. A tutorial on support vector machines for pattern recognition. Data Min Knowl Disc 1998; 2(2): 121-167.

[117] Suykens JAK. Support vector machines: A nonlinear modelling and control perspective. Eur J Control 2001; 7(2-3): 311-327.

[118] Cortes C, Vapnik V. Support-vector networks. Mach Learn 1995; 20(3): 273-297.

[119] Cristianini N, Shawe-Taylor J. An introduction to support vector machine and kernel based learning methods. Cambridge University press, Cambridge 2000.

[120] Drucker H, Wu D, Vapnik V: Support vector machines for spam categorization. IEEE Trans Neural Netw 1995; 10: 1054-1084.

[121] Ojeda F, Suykens JAK, De Moor B. Low rank updated LS-SVM classifiers for fast variable selection. Neural Networks 2008; 21(2-3): 437-449.

[122] Zadeh LA. Fuzzy sets. Inform Control 1965; 8: 338–353.

[123] Zadeh LA. Fuzzy algorithms. Inform Control 1968; 12: 94–102.

[124] Bezdek JC, Castelaz PF. Prototype classification and feature selection with fuzzy sets. IEEE Trans Syst Man Cybern 1977; 7:87–92.

[125] Keller JM, Tahani H. Implementation of conjunctive and disjunctive fuzzy logic rules with neural networks. Int J Approx Reason 1992; 6: 221–240.

[126] Bezdek JC, Pal SK. Fuzzy Models for Pattern Recognition: Methods that search for structures in data. IEEE Press 1992.

[127] Zimmerman H-J. Fuzzy Set Theory and its Applications 4th ed. Springer 2001.

[128] Xu D, Bondugula R, Popescu M, Keller J. Bioinformatics and fuzzy logic. Fuzz-IEEE 2006: 817–824.

[129] Fogel LJ, Owens A, Walsh MJ. Artificial Intelligence Through Simulated Evolution. John Wiley & Sons 1966.

[130] Rechenberg I. Evolutionsstrategie: Optimerung technischer Systeme nach Prinzipien der biologischen Evolution. Fromman-Holzboog 1973.

[131] Koza JR. Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press 1992.

[132] Eberhart RC, Shi Y, Kennedy J. Swarm Intelligence. San Francisco, CA: Morgan Kaufmann 2001.

[133] Dorigo M, Gambardella LM. Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE T Evolut Comput 1997; 1(1): 53–66.

[134] Storn R, Price K. Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces. Technical Report TR-95-012, ICSI, March 1995.

[135] Storn R. System design by constraint adaptation and differential evolution. IEEE T Evolut Comput 1999; 3(1): 22–34.
[136] Back T, Fogel DB, Michalewicz Z. Handbook of Evolutionary Computation. Oxford University Press 1997.
[137] Fogel DB. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence 3rd ed. IEEE Press 2005.
[138] Fogel GB, Corne DW. Evolutionary Computation in Bioinformatics 1st ed. San Francisco, CA: Morgan Kaufmann 2002.
[139] Fogel DB. Evolutionary Computation: The Fossil Record. IEEE Press 1998.
[140] Baten A, Halgamuge SK, Chang BCH. Fast splice site detection using information content and feature reduction. BMC Bioinformatics 2008; 9 (Suppl 12).
[141] Snyder EE, Stormo GD. Nucleic Acid and Protein Sequence Analysis: A Practical Approach 2nd ed. Oxford University Press: USA 1995.
[142] McElwain M. A Critical Review of Gene Prediction Software. Bioc 218 Final Paper 2007.
[143] Zhang MQ. Using MZEF to Find Internal Coding Exons. Current Protocols in Bioinformatics. John Wiley & Sons 2003; 4: 1-18.
[144] Majoros WH, Pertea M, Antonescu C, Salzberg SL, Glimmer M. Exonomy and Unveil: three ab initio eukaryotic genefinders. Nucleic Acids Res 2003; 31(13): 3601-3604.

Received: October 22, 2011 Revised: April 30, 2012 Accepted: May 1, 2012