CHAPTER 5

HOW NEURAL NETWORKS FIND PROMOTERS USING RECOGNITION OF MICRO-STRUCTURAL PROMOTER COMPONENTS

Vladimir B. Bajić

Institute for Infocomm Research
bajicv@i2r.a-star.edu.sg

Ivan V. Bajić

University of Miami
ivan_bajic@ieee.org

Finding regulatory components in genomic DNA by computational methods is an attractive and complex research field. Currently, one of the important targets is finding protein coding genes in uncharacterized DNA. One of the significant aspects of gene recognition is the determination of locations of specific regulatory regions—promoters—that usually occupy the position at the beginning of a gene. Promoters are responsible for the initiation of the DNA transcription process. Current computer methods for promoter recognition are still either insufficiently accurate or insufficiently sensitive. We discuss in this chapter some of the general frameworks and conceptual issues related to the use of artificial neural networks (ANNs) for promoter recognition. The scenario discussed relates to the case when the main promoter finding algorithms rely on the recognition of specific components contained in the promoter regions of eukaryotes. Some specific technical solutions are also presented and their recognition performances on an independent test set are compared with those of the non-ANN based promoter recognition programs.

ORGANIZATION.

Section 1. We first discuss the motivation for finding promoters of genes in uncharacterized DNA sequences. Then we provide some background material on the use of ANN for this purpose.

Section 2. Next we describe some of the characteristic motifs of eukaryotic promoters, as well as problems associated with them. This gives us an understanding of the challenges in arriving at a general model for eukaryotic promoter recognition.

Section 3. A few attempts have been made in eukaryotic promoter recognition that use information based on the more common promoter region motifs—such as their position weight matrices—and their relative distances. We mention some of them next. We also review the study of Fickett and Hatzigeorgiou,[249] which reveals the high level of false positives in some of these programs. The role that enhancers may have played in false positive recognition of promoters is then discussed.

Section 4. Some details of some of the principles used in designing ANNs for recognizing eukaryotic promoters are presented in this and the next sections. In particular, we begin with discussing representations of nucleotides for ANN processing.

Section 5. Then we describe the two main forms of structural decomposition of the promoter recognition problem by ANN, *viz.* parallel vs. cascade composition of feature detectors. The promoter recognition part of GRAIL[549] and promoter 2.0[439] are then used to illustrate these two forms. We also discuss the construction of the underlying feature detectors.

Section 6. After that, we introduce time-delay neural networks, which are used in the NNPP program[702, 703, 705] for promoter recognition. We also discuss the issue of pruning ANN connections in this context.

Section 7. Finally, we close the chapter with a more extensive discussion and survey on the performance of promoter recognition programs.

1. Motivation and Background

Advances in genetics, molecular biology, and computer science have opened up possibilities for a different approach to research in biology—the computational discovery of knowledge from biological sequence data. Numerous methods aimed at different facets of this goal are synergized in a new scientific discipline named "bioinformatics". This new field has the potential to unveil new biological knowledge on a scale and at a price unimaginable 2-3 decades ago. We present here an overview of capabilities of the artificial neural network (ANN) paradigm to computationally predict, in uncharacterised long stretches of DNA, special and important regulatory regions of DNA called promoters. For references on ANNs, see for example Bose and Liang,[96] Caudill,[135] Chen,[143] Fausett,[241] Hercht-Nielsen,[338] Hertz *et al.*,[350] Kung,[465] and Rumelhart and McClelland.[730]

This chapter focuses on some of the basic principles that can be used in constructing promoter prediction tools relying on ANNs that make use of information about specific short characteristic components—the so-called "motifs"—of eukaryotic promoters' micro-structure. These motifs include subregions such as the TATA-box,[75, 116, 173, 624, 661] the CCAAT-box,[75, 223] Inr,[392, 422, 423, 640, 661, 774, 776, 872] the GC-box,[116] and numerous other DNA sites that bind a particular class of proteins known as transcription factors (TFs).[476, 878] In this approach the ANN system attempts to recognize the presence of some of these motifs and bases its final prediction on the evidence of such presences. This is conceptually different from the macro-structural approach, where the recognition of the promoter region is based primarily on the recognition of features of larger sections of the promoter and neighboring regions,[46–49, 51, 190, 321, 383, 674, 753] such as CpG-islands.[81, 180, 181, 271, 474]

Although the techniques to be presented here relate to the recognition of a particular group of regulatory regions of DNA—the promoter regions—the techniques discussed are far more general and are not limited to promoter recognition only. They could well be used in the large-scale search for other important sections of DNA and specific genomic signals, such as enhancers, exons, splice sites, *etc*.

1.1. *Problem Framework*

Probably the most fascinating aspects of bioinformatics is the computational investigation, discovery, and prediction of biological functions of different parts of DNA/RNA and protein sequences. One of the important practical goals of bioinformatics is in reducing the need for laboratory experiments, as these are expensive and time consuming.

The worldwide effort aimed at sequencing the human genome—the so-called Human Genome Project[132, 171, 467, 635, 859, 873, 874]—is virtually finished, although it was initially planned for finalization by the year 2004. [544] To illustrate the quantity of information contained in the human genome, note that it contains approximately 3 billion bp[893] and within it about 35,000–65,000 genes, while the number of regulatory regions and other functional parts in the human genome still remain to a large extent unclear.

An open question is our ability to computationally locate all important functional parts of the human genome, as well as to computationally infer the biological functions of such segments of genetic information. Recognition of constituent functional components of DNA, and consequently annotation of genes within a genome, depend on the availability of suitable models of such components. This relies on our understanding of the functionality of DNA/RNA and related cell products. This problem is very difficult and complex, as our current knowledge and understanding of DNA/RNA functioning is not complete.

Hence, one of the general problems that has to be solved is the accurate complete annotation of different genomes. This involves the identification and location of genes, as well as their associated promoters. Currently, the dominant interest is in finding genes that code for proteins, and the recognition of related promoters is a part of the problem. There are many techniques aimed at recognizing genes or some of their main features in long stretches of DNA—see Burge *et al.*,[121–123] Claverie,[166] Dong and Searls,[210] Eckman *et al.*,[217] Fickett *et al.*,[247, 248]

Gelfand *et al.*,[282,283] Gish and States,[290] Guigo *et al.*,[305,306] Hatzigeorgiou *et al.*,[333,334] Hayes and Borodovsky,[336] Henderson *et al.*,[341] Hutchinson and Hayden,[378] Jiang and Jacob,[397] Kleffe *et al.*,[435] Krogh,[453,454] Lukashin and Borodovsky,[529] Mathe *et al.*,[546] Milanesi and Rogozin,[566] Murakami and Takagi,[587] Quandt *et al.*,[690] Roytberg *et al.*,[727] Salzberg *et al.*,[738,739,741] Snyder and Stormo,[781] Solovyev *et al.*,[784,785] Sze *et al.*,[811,812] Tiwari *et al.*,[831] Uberbacher and Mural,[848] Ureta-Vidal *et al.*,[851] Xu *et al.*,[912−914] and Zhang.[930]

Some of these techniques utilize ANNs as parts of the solution at different levels of the problem and for different features and purposes—for example, see Brunak *et al.*,[114] Hatzigeorgiou *et al.*,[329,331−334] Hayes and Borodovsky,[336] Lapedes *et al.*,[473] Rampone,[697] Snyder and Stormo,[781] and Uberbacher *et al.*[848,914] However, numerous problems remain unsolved and, so far, the overall results of different types of predictions are not yet satisfactory.[127,167,304,334] Within the problems that still await successful solutions is an accurate recognition of promoters, which remains one of the crucial components of the complete gene recognition problem.

1.2. *Promoter Recognition*

There are several main reasons why we are interested in searching for promoters in genomic DNA.[249,656,681] For example:

- Promoters have a regulatory role for a gene. Thus, recognizing and locating promoter regions in genomic DNA is an important part of DNA annotation.
- Finding the promoter determines more precisely where the transcription start site (TSS) is located.
- We may have an interest in looking for specific types of genes and consequently for locating specific promoters characteristic for such genes.

The problem of promoter recognition is not a simple one and it has many facets, such as:

- Determination of the promoter region, without any attempt to find out what such regions contain.
- Determination of the location of different binding sites for numerous TFs that participate in the initiation of the transcription process.
- Determination of the TSS, which is an important reference point in the context of transcription initiation.
- Determination of the functional classes of promoters, *etc.*

Thus, many techniques exist for promoter recognition and location. More details on such methods for prokaryotic organisms can be found in Alexandrov and Mironov,[19] Demeler and Zhou,[196] Grob and Stuber,[302] Hirst and Sternberg,[356] Horton and Kanehisa,[365] Lukashin *et al.*,[528] Mulligan and McClure,[584] Nakata *et al.*,[605] O'Neil,[638] Reese,[702] Rosenblueth *et al.*,[719] Staden,[795] *etc*.

The techniques for the recognition of eukaryotic promoters are much less efficient. The eukaryotic promoters are far more complex and possess very individual micro-structures that are specialized for different conditions of gene expression. It is thus much more difficult to devise a general promoter recognition method for eukaryotes. As in the case of prokaryotic promoter recognition, different techniques have been used to deal with the recognition of eukaryotic promoters of different classes—see Audic and Claverie,[38] Bajic *et al.*,[43,44,46–49,51] Bucher,[116] Chen *et al.*,[150] Claverie and Sauvaget,[168] Davuluri *et al.*,[190] Down and Hubbard,[211], Fickett and Hatzigeorgiou,[249] Frech and Werner,[262] Hannenhali and Levy,[321] Hatzigeorgiou *et al.*,[330] Hutchinson,[377] Ioshikhes and Zhang,[383] Kondrakhin *et al.*,[446] Mache and Levi,[535] Matis *et al.*,[549] Milanesi *et al.*,[565] Ohler *et al.*,[626–628,630] Ponger and Mouchiroud,[674] Prestridge,[679,681] Quandt *et al.*,[690,691] Reese *et al.*,[703,705] Scherf *et al.*,[753] Solovyev and Salamov,[784] Staden,[796] and Zhang.[931] Recent evaluation studies[49,249] of some publicly available computer programs have revealed that computer tools for eukaryotic promoter recognition are not yet mature.

1.3. *ANN-Based Promoter Recognition*

Some of the techniques mentioned are based on the use of artificial neural networks. Results on the recognition of promoters by ANNs in prokaryotes can be found in Demeler and Zhou,[196] Hirst and Sternberg,[356] Horton and Kanehisa,[365] Lukashin *et al.*,[528] Reese;[702] and those for eukaryotic organisms in Bajic,[42] Bajic *et al.*,[43,44,46–49,51] Hatzigeorgiou *et al.*,[330] Knudsen,[439] Mache and Levi,[535] Matis *et al.*,[549] Milanesi *et al.*,[565] Ohler *et al.*,[627] and Reese *et al.*[702,703,705]

Results of several programs based on ANNs are available for comparison: NNPP;[702,703,705] Promoter2.0,[439] which is an improved version of the program built in GeneID package;[306] SPANN;[42,43] SPANN2;[44] McPromoter;[627] Dragon Promoter Finder;[48,49,51] and Dragon Gene Start Finder.[46,47] The programs mentioned operate on different principles and use different input information. For example, some of ANN-based programs are designed to recognize specific characteristic subregions of eukaryotic promoters, as in the case of NNPP,[705] Promoter2.0,[439] the promoter finding part of the GRAIL program,[549] the program of Hatzigeorgiou *et al.*,[330] and that of Wang *et al.* in Chapter 6 of this book. On the other hand, SPANN,[42, 43] SPANN2,[44] Dragon Promoter Finder,[48, 49, 51] Dragon Gene Start Finder,[46, 47] and McPromoter[627] use primarily integral information about the promoter region. The scores achieved by some of these programs are shown later and indicate that ANN-based methods for eukaryotic promoter recognition rate favorably with regard to non-ANN based programs.

2. Characteristic Motifs of Eukaryotic Promoters

Promoters are those parts of genomic DNA that are intimately related to the initiation of the so-called transcription process. The starting point of transcription—*i.e.*, the TSS—is generally contained within the promoter region and located close to, or at, its 3' end. A promoter in eukaryotes can be defined somewhat loosely as a portion of the DNA sequence around the transcription initiation site.[677] Eukaryotic promoters may contain different subregions—sometimes also called components or elements—such as TATA-box, CCAAT-box, Inr, GC-box, DPE, together with other different TF binding sites.

The problem with these subregions in eukaryotic promoters is that they vary considerably from promoter to promoter. They may appear in different combinations. Their relative locations with respect to the TSS are different for different promoters. Furthermore, not all of these specific subregions need to exist in a particular promoter. The high complexity of eukaryotic organisms is a consequence of high specialization of their genes, so that promoters in eukaryotes are adjusted to different conditions of gene expression, for example, in different tissues or in different cell types.

Thus, the variability of internal eukaryotic promoter structures can be large. Consequently, the characteristics of the eukaryotic promoter are rather individual for the promoter, than common for a larger promoter group.[216, 402, 476, 557, 570, 621, 681, 774, 799, 872, 878, 887] For this reason it is not easy to precisely define a promoter structure in eukaryotic organisms. This is also one of the reasons why at this moment there is no adequate computer tool to accurately detect different types of promoters in a large-scale search through DNA databases.

The simplistic version of the process of the initiation of transcription implies a possible model for eukaryotic promoters: It should have a number of binding sites. However,

- there is a large number of TFs—see TRANSFAC database details in Matys *et al.*;[553]
- TF binding sites—for one of their databases see Ghosh[287]—for different promoters may be at different relative distances from the TSS;[216, 402, 570, 887]

- for functional promoters the order of TF binding sites may be important;
- for different promoters, not all of the TF binding sites need to be present;[886] and
- the composition of TF binding sites for a particular promoter is essentially specific and not shared by a majority of other promoters.

It is thus very difficult to make a general computer tool for promoter recognition that uses information based on TF binding sites and their composition within the eukaryotic promoters.[43, 656]

There are three types of RNA polymerase molecules in eukaryotes that bind to promoter regions. Our specific interest is in RNA Polymerase II and their corresponding promoters—*viz.* Pol II promoters—whose associated genes provide codes for proteins. Many eukaryotic Pol II promoters have some specific subregions that possess reasonably high consensus. A typical example is the TATA-box. The TATA-box is a short region rich with thymine (T) and adenine (A) and located about –25 bp to –30 bp upstream of the TSS. But there are also other frequently present components like the CCAAT-box, Inr, DPE, *etc*.

Arriving at a general model for an eukaryotic promoter is difficult. Nevertheless, suitable models can be derived for specific classes of promoters. For example, the mammalian muscle-actin-specific promoters are modelled reasonably well for extremely accurate prediction.[261] Such modelling of specific narrow groups of promoters makes a lot of sense in a search for specific genes. The point is, however, that computer tools for the general annotation of DNA are aimed at the large-scale scanning and searching of DNA databases so as to recognize and locate as many different promoters as possible, and not to make large numbers of false recognitions. Obviously, it is difficult to make such tools based on highly specific structures of very narrow types of promoters.

3. Motif-Based Search for Promoters

The problems of initiation and control of transcription processes in eukaryotes have a major importance in the biochemistry of cells[656] and are the subject of intensive research.[125, 138, 249, 406, 448, 621, 685, 717, 775, 799, 860, 878] The accurate prediction of promoter location, including that of the TSS, can significantly help in locating genes. We have indicated previously that there are a number of components within the eukaryotic promoter region that may possibly serve as a basis for promoter recognition. Such methods have to take into account the great variability of the internal eukaryotic promoter structure,[216, 402, 476, 557, 570, 621, 681, 774, 799, 872, 887] which contributes to promoter complexity,[926] and to the complexity of the computational promoter recognition.

A reasonable approach in devising techniques for promoter recognition is to identify those patterns that are common to very large groups of promoters, and then to search for such patterns and their mutual distances. Such algorithms would reflect the biochemical background of the transcription process, and, in principle, should result in the least number of false recognition. Unfortunately, constructing such algorithms depends crucially on the detailed knowledge of the biochemistry of promoter's activity which is not yet fully available. In prokaryotic promoters very common patterns exist—for example, the –10 and –35 regions have reasonably high consensus and a very consistent distance between them. It is thus not surprising that a number of techniques have been developed to deal with prokaryotic promoter recognition.[19, 196, 302, 356, 365, 528, 584, 605, 638, 702, 719, 795] Some of these techniques are based on the use of ANNs.[196, 356, 365, 528, 702]

Due to the high structural complexity and the absence of a greater number of strong common motifs in eukaryotic promoters, the existent techniques aimed at computational recognition of eukaryotic promoters are much less accurate. The previously mentioned very individual micro-structure of eukaryotic promoters that are specialized for different conditions of gene expression complicates enormously the development of adequate techniques for promoter recognition. However, there are certain motifs within eukaryotic promoters that are present in larger promoter groups. Many of the eukaryotic promoter recognition programs base their algorithms on searching for some of these motifs, frequently by using some additional information such as the relative distances between the motifs.

A number of attempts have been made in this promoter recognition task that are aimed at utilizing information from some of these more common promoter region components. About 30% of eukaryotic Pol II promoters contain TATA-like motifs. About 50% of vertebrate promoters contain CCAAT-box motifs. Inr is also a very common subregion in eukaryotic promoters. It is found that combination of some of these specific subregions are crucial in the determination of the correct TSS location—such as the combination of TATA-box and Inr.[621] Also, the position weight matrices (PWMs) for the TATA-box, the CCAAT-box, the GC-box, and the cap site have been determined in Bucher.[116] In spite of the fact that the consensus of the TATA-box is not very strong, the PWM of the TATA-box from Bucher[116] appears to be a very useful tool for recognition of a larger group of eukaryotic Pol II promoters. This weight matrix is normally used in combination with the other methods[679] due to the fact that when it is used alone it produces a large number of false recognition of the order of 1 false recognition per 100–120 bp on non-promoter sequences.[679, 682]

It is possible to use only one target motif in the attempt to recognize promoters that contain such a motif and to achieve relatively good success—see Chapter 6. However, most methods that aim at recognizing eukaryotic promoters do not base their algorithms on locating only one of many possible micro-structural promoter components. They rather look for the existence of a suitable combination of such elements which is then assessed and used in the prediction task. For example, in Prestridge,[679] the prediction of eukaryotic Pol II promoters is based on the prediction of the TF binding sites and then combined with an assessment of the PWM score for the TATA-box. The TF binding sites that are used are those corresponding to the TF database from Ghosh.[287] The method is based on the assumption that the distributions of the TF binding sites in promoter and non-promoter regions are different. The resulting program, Promoter Scan, can predict both TATA-containing and TATA-less promoters and has shown a reduced level of false recognition compared with the other promoter-finding programs.[249] The last version of Promoter Scan[681] has an improved and extended functionality compared with the original version.

A sort of an extension of the method used initially for developing Promoter Scan has been made in TSSG and TSSW programs.[784] These programs are extended by the addition of a linear discriminant function that values (1) the TATA-box score; (2) the sequence composition about the TSS—*viz.* triplet preferences in the TSS region; (3) hexamer preferences in the three upstream regions—*viz.* [–300, –201], [–200, –101], [–100, –1]; and (4) potential TF binding sites. The programs use different TF databases.

Also, as in the case of Promoter Scan, a part of the AutoGene program—the program FunSiteP, which is responsible for finding promoters[446]—contains an algorithm devised on the assumption of different distributions of TF binding sites in the promoter regions and in non-promoter sequences. The database source for FunSiteP is based on a collection of binding sites from Faisst and Meyer.[237]

The other group of methods that explicitly use eukaryotic promoter micro-structure components—at least as a part of the algorithm—exploit the modelling and generalization capabilities of ANNs.[44, 46–49, 51, 330, 439, 535, 549, 565, 627, 702–707] Some of them utilize, in one way or other, the fact that combinations of some of the specific subregions—such as the combination of the TATA-box and Inr—helps in the determination of the TSS location. Certain programs also use explicit information on relative distances between such specific subregions. In the next sections we see in more details some of the principles that may be used in designing ANN systems for eukaryotic promoter recognition.

3.1. *Evaluation Study by Fickett and Hatzigeorgiou*

A recent evaluation study[249] of publicly available computer programs has indicated different degrees of success of these programs and revealed that tools for promoter recognitions do require a lot of additional development. On the specific evaluation set used,[249] where only the ability of programs to locate the TSS is considered, the rate of success is in the range of 13%–54% of true positive predictions (TP), while false positive predictions (FP) are in the range of $1/5520$ bp in the best case and up to $1/460$ bp in the worst case.

TP predictions are correct predictions of the TSS location within the prespecified bounds arround the actual TSS location. FP predictions are those reported as predicted TSS locations at positions out of the above mentioned bounds. The interpretation of the FP score of, say, $1/200$ bp means the promoter recognition system produces on an average 1 FP prediction of promoters every 200 bp.

The general observation is that the level of TP predictions is directly correlated to the level of FP predictions. So, the best program in correct positive predictions—NNPP, which is based on neural networks—produces 54% TP predictions, and FP at the level of $1/460$ bp. On the other hand, the Promoter Scan program,[679] which is not neural network based, produces a score of 13% TP predictions, but achieved the lowest proportion of FP at the level of $1/5560$ bp.

It should be indicated that TP and FP as measures of success in prediction programs are not very convenient for comparison of prediction programs that produce different TPs and FPs. Thus, to be able to make a reasonable comparison of different programs on a unified basis, a more convenient measure of success scores from Bajic[50] is used later. It shows a rational ranking of promoter prediction programs and corroborates our statement that ANN-based programs for promoter prediction exhibit comparable or better performance to non-ANN promoter prediction programs.

3.2. *Enhancers May Contribute to False Recognition*

Closely associated with promoter regions in eukaryotes is another class of transcriptional regulatory domains in DNA—the enhancers. Enhancers cooperate with promoters in the initiation of transcription. They are located at various distances from the TSS and sometimes may be several thousands nucleotides away. They can also be located either upstream or downstream of the TSS.

As in promoters, enhancers also contain different TF binding sites. Thus, as pointed out in Fickett and Hatzigeorgiou,[249] one of the reasons for having a high level of FPs produced by programs for finding eukaryotic promoters can be that most of these techniques are mainly based on finding specific TF binding sites within the promoter region, or in the assessment of the density of TF binding sites in the promoter and non-promoter sequences. On this basis it seems that enhancers could frequently be recognized as promoters. Thus, it would be of interest to develop methods to discriminate between the promoter and enhancer regions in uncharacterized DNA, and in this way to contribute to the reduction of the FP scores of some of promoter recognition programs.

4. ANNs and Promoter Components

4.1. *Description of Promoter Recognition Problem*

The best way to understand a possible role of neural networks in promoter prediction in eukaryotes is to examine the essence of the problem that needs to be solved by neural networks. Let us assume that there may exist n specific subregions R_j, $j = 1, 2, ..., n$, some of which we may use for promoter recognition. Let R_j^k denote the region R_j in the k-th promoter. We use the superscript to indicate that the form—the actual composition, length, and relative position with respect to the TSS—of the subregion R_j in the k-th promoter may be, and usually is, different from the form of the same region in another, say, i-th promoter. Let s_j^k and e_j^k denote the starting position and the ending position of the region R_j^k in the k-th promoter. These positions are counted from the 5' end of the DNA string and represent relative distances from an adopted reference position. Thus, $s_j^k < e_j^k$.

Let us also assume that the starting points of these subregions are at distances d_j^k, $j = 1, 2, ..., n$, from the TSS. The values of d_j^k are taken from the set

$$\mathbb{Z}_{\perp} = \mathbb{Z} \cup \{\perp\}$$

where $\mathbb{Z}$ is the set of integers. Note that, due to the possible absence of a subregion R_j^k from the k-th promoter, it may be that d_j^k cannot be defined. This is the reason for having the special symbol $\perp$ in the definition of $\mathbb{Z}_{\perp}$. Note also that we use negative values of d_j^k for the locations of R_j^k upstream of the TSS, we use positive values of d_j^k for the downstream locations. Thus the sign of d_j^k determines only the direction from the TSS.

In an uncharacterized genomic sequence we do not know the location of the TSS—it has yet to be determined. Thus, we cannot use the information of distances d_j^k in the search process, even though we can do this during the training process as it may be assumed that in the training set the TSS and subregions of interest are known. In fact, it makes sense to develop specialized ANN systems aimed at searching for specific promoter components, where these systems use information on distances d_j^k directly. This may be the case in tasks when the location of the TSS is known, but when there is not enough information about the promoter structure.[681] However, we mainly consider here the problem of uncharacterized genomic sequences.

The direct use of distances d_j^k can be circumvented if we use relative distances between the subregions. We assume that in one promoter two functional subregions R_i^k and R_j^k do not overlap. Thus

$$D_{ij}^k = \begin{cases} s_j^k - e_i^k - 1, \text{ if } s_j^k > e_i^k \\ \\ s_i^k - e_j^k - 1, \text{ if } s_i^k > e_j^k \end{cases}$$

denotes the mutual distance of subregions R_i^k and R_j^k in the k-th promoter. This distance does not include the ending point e_i^k (respectively, e_j^k) of the first subregion R_i^k (respectively, R_j^k)—counted in the direction from 5' toward the 3' end—nor the starting point s_j^k (respectively, s_i^k) of the subsequent subregion R_j^k (respectively, R_i^k). Alternatively, one can express these distances by means of subregion length c_i^k as

$$D_{ij}^k = \begin{cases} s_j^k - s_i^k - c_i^k, \text{ if } s_j^k \geq \left(s_i^k + c_i^k\right) \\ \\ s_i^k - s_j^k - c_j^k, \text{ if } s_i^k \geq \left(s_j^k + c_j^k\right) \end{cases}$$

Further, the characteristics of the subregion R_j that we are trying to identify—and by which we attempt to recognize its presence in the k-th promoter—may be varied. So, let us assume that we are interested in identifying a feature F_j of the subregion R_j in all promoters under investigation. That is, we try to identify the feature F_j for all R_j^k, $k = 1, 2, ..., n_p$, where n_p is the number of promoters in the group we analyze. The feature F_j may be expressed, say, as a set of probabilities of specific motifs appearing at appropriate positions relative to the reference indicator, or it may be given in the form of a suitably defined discrepancy function $discrep(R_j^k, R_j^{ref})$—for example, the distance of R_j^k from R_j^{ref} in a suitable space—where R_j^{ref} may represent a prespecified motif, consensus sequence, *etc*. Although we can consider several features associated with a subregion, for simplicity we restrict our consideration to only one such feature. We need to highlight the fact that for virtually any of the specific subregions there is no unique subregion description. As an example, many different compositions of nucleotides may represent the same subregion although the subregion is characterized by a strong consensus signature.

The order of subregions R_j in a specific promoter may be of importance for the functionality of the promoter. Thus the ordering of subregions R_j^k is also a candidate as an input parameter for assessment.

nucleotide	code
A	1000
C	0100
G	0010
T	0001

Fig. 1. Binary code that can be used to represent four nucleotides in DNA.

An additional problem that is encountered in dealing with the locations of subregions is related to the fact that domains of location for two functional subregions R_i and R_j in a set of sequences may overlap, although in a particular promoter they are separate. The overlapping of possible locations comes from considering a group of promoters containing the same subregions. A typical example is in the case of the Inr and TATA-box in eukaryotic Pol II promoters, where Inr can be located within the –14 bp to +11 bp region relative to the TSS, while the TATA-box can be located in the –40 bp to –11 bp domain. Thus, a number of constraints of this type may be of importance when formulating input information to the neural network system.

So, we can summarize the problems that appear:

- generally, the model describing a subregion and its selected feature(s) does describe a set of more or less similar sequences, but does not determine them uniquely;
- not all subregions need to exist in a particular promoter;
- relative distances of subregions from the TSS are variable—which implies that the relative mutual distances of subregions are variable too;
- order of subregions may be of importance; and
- possible overlapping of subregion domains can occur.

4.2. *Representation of Nucleotides for Network Processing*

The DNA sequence is a sequence of 4 different nucleotides denoted in the customary way as “A”, “T”, “C”, and “G”. If no biological or physical properties of these nucleotides are taken into account, then a suitable representation of these nucleotides may be by the code given in Figure 1.

This code has the same Hamming distance between any two coding vectors for the A, C, G, and T nucleotides, which is considered desirable so as not to contribute to biased learning. This code representation has been used in a number of ANN applications to DNA and protein analysis—for example, see Brunak *et al.*,[114] Demeler and Zhou,[196] Farber,[240] Knudsen,[439] O'Neill,[637] Qian and Sejnowski,[687] and Reese and Eeckman.[705]

However, it should be mentioned at this point that this is not the only possible numerical representation of nucleotides. If we want to incorporate some of the physical properties that characterize nucleotides and base our further analysis on such a representation, then, for example, the electron-ion interaction potential (EIIP),[857, 858] may also be used with success in promoter recognition algorithms.[42–49, 51]

It is difficult to determine at this stage of computational genomics research which coding system is more effective. One can argue that the A, T, C, and G nucleotides generally form two groups with specific chemical characteristics—purines (A, G) and pyrimidines (C, T)[547]—so that their numerical representation for the purpose of computer analysis should reflect such similarities in order to more adequately mimic the real-world situation. Consequently, it seems that it is not the right approach to use the binary coding as mentioned above. Also, the ordering of nucleotides is crucial in determining the function of a particular section of DNA. Since the biochemical functions of DNA segments depend on that order—for example, in a particular context several successive purine nucleotides can have different biochemical properties than if their positions are occupied by pyrimidine nucleotides—it seems more logical to use a coding system that reflects physical or biological properties of the nucleotides. Thus we favor numerical coding of nucleotides via physical characteristics that they may have, over the essentially artificial allocation of binary numerical representation, such as the binary coding presented above.

5. Structural Decomposition

The problem of eukaryotic promoter recognition allows several possible structural decomposition forms. We comment on two of such decomposition structures. The first one basically uses parallel composition of feature detectors (PCFD), while the other uses cascade composition of feature detectors (CCFD). Both structures comprise hierarchical systems.

5.1. *Parallel Composition of Feature Detectors*

It is possible to build a neural network system so that, on the first hierarchical level, ANNs attempt to recognize the individual features F_j either as independent

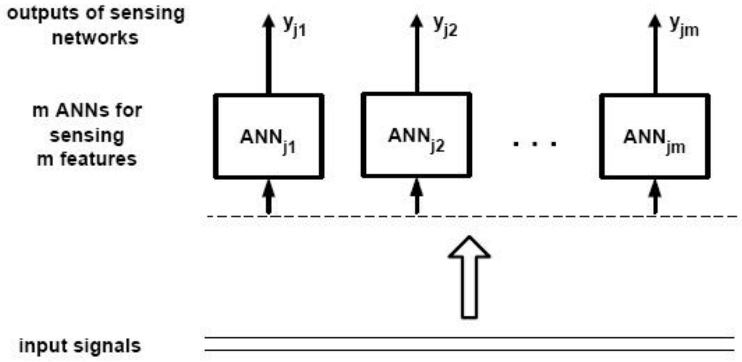

Fig. 2. Conceptual structure of the first hierarchical level of an ANN system aimed at feature recognition of promoter subregions.

features, or in specific combinations—see Figure 2. Naturally, the independent recognition of individual features is far simpler and more practical.

One possible way to realize this first hierarchical level of ANNs is depicted in Figure 3. Let us assume that there are m subregions that we intend to identify, R_j, $j \in \{j_1, j_2, ..., j_m\}$; that each subregion R_j is characterized by a feature F_j; and that we have m ANNs, ANN_j, $j \in \{j_1, j_2, ..., j_m\}$, for the independent recognition of features F_j, $j \in \{j_1, j_2, ..., j_m\}$. We assume that the neural network ANN_j for the recognition of feature F_j requires information gathered by reading data through a data window w_j that slides along the DNA strand from its 5' end towards its 3' end. Then the process of supplying the ANN system with input information is equivalent to independently sliding m required windows w_j along the DNA sequence and feeding each feature-sensing ANN with information from the appropriate window. The network ANN_j then produces output signal y_j.

Depending on the choice of neural networks, these output signals may be continuous or discrete. If they are continuous, then usually their amplitudes are related to the sensing accuracy—the larger the amplitude, the higher the certainty that the subregion is identified, *i.e.*, the greater the chance that the searched-for feature is detected. If the networks for feature sensing are designed to function like classifiers,[96, 135, 143, 241, 350, 465, 513, 730] then they perform the classification of input patterns into appropriate output categories. For example, they may have outputs at 1 to correspond to the “identified” subregion or feature, *vs.* 2 to correspond to the case when the subregion or feature is not identified at the given position. Depending on the problem, the classifier networks may have to learn

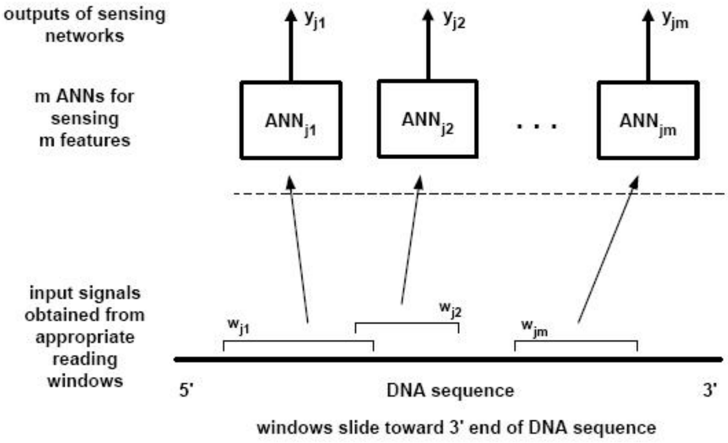
outputs of sensing networks
y_j1
y_j2
y_jm
m ANNs for sensing m features
ANN_j1
ANN_j2
. . .
ANN_jm
input signals obtained from appropriate reading windows
w_j1
w_j2
w_jm
5'
DNA sequence
3'
windows slide toward 3' end of DNA sequence

Fig. 3. Input signals for networks on the first hierarchical level are obtained from the appropriate reading windows w_{j_1}, w_{j_2}, ..., w_{j_m}.

very complicated multidimensional decision boundaries so as to be able to conduct classification properly. Generally, ANN classifiers are capable of learning such complicated decision surfaces, but these capabilities and final success depend on the type of network, as well as the training procedure and the training set.[465]

In our case it is convenient to deal with 2 output categories for each feature-sensing networks, but this need not always be the case. Some of the typical types of ANNs that produce discrete output signals convenient for this category are probabilistic neural networks (PNNs).[788, 789, 791] These networks perform input pattern classification with a decision boundary that approaches asymptotically the Bayesian optimal decision surface. That is, they asymptotically converge to the Bayesian classifier. Another useful type of networks, whose decision boundary also approximates the theoretical Bayesian decision surface, contains the learning vector quantization (LVQ) networks,[442, 443] which may be considered as a variant of self-organizing-map (SOM) networks[440, 442, 444] adjusted for supervised learning. Also useful are the radial basis function based classifiers,[591, 726] or the multilayered perceptrons (MLPs).[885] MLPs are capable of approximating arbitrary discrimination surfaces,[513] although their approximations of these surfaces have some topological constraints.[289] This only means that one may need a very large MLP with a sufficient number of hidden layers in order to sufficiently well approximate the smooth decision boundary. The size of such an MLP may lead to problems of training. The situation can be relaxed somewhat if a nonlinear preprocessing of MLP inputs is made.[884]

For practical purposes it is more convenient to have networks with continuous outputs at this level, such as the radial basis function networks (RBFNs),[53, 109, 152, 412, 576, 577, 667, 670, 676] the generalized regression networks (GRNN),[790, 791] or some of the many other forms of ANNs that produce continuous output signals.[96, 135, 143, 241, 350, 465, 730] As an example, MLPs can be associated with the probabilities of detecting a time-evolving feature[97] so as to produce continuous output signals and can be used in the context of promoter subregion sensing.

It is important to note that the networks used on this level can produce a large number of FPs. This is intimately related to the problem discussed in Trifonov,[835] where it has been shown on some "hard-to-believe" examples—to use the terminology from Trifonov[835]—that sequences quite different from consensus sequences functionally perform better than those consensus sequences themselves. This is one of the general problems related to what consensus sequences represent and how "characteristic" they are for the pattern that is searched for. The same difficulty appears with the usage of the PWM that characterizes a specific subregion. This highlights the real problem of what is a suitable definition of similarity between the signature of the pattern R_j^{ref} (the template pattern) we look for and the potential candidate sequence R_j^k that is tested in the search process. In other words, the problem is how to express mathematically the discrepancy function $discrep(R_j^k, R_j^{ref})$ in the most effective way.

On the higher hierarchical level, a neural network system can be built to assess the identified combination of features in association with their mutual distances and their ordering, as depicted in Figure 4. This problem belongs to the class of the so-called multi-sensor fusion/integration problems,[712, 814, 924] but their implementation is complicated by the necessity to cater at this level for spatial/temporal patterns and their translation invariance. If the information on the relative mutual distance between the sensed feature at the lower hierarchical level is not presented to the higher hierarchical level network, then the network on the higher hierarchical level has to have the capability to learn and recognize the spatial/temporal events so as to be able to assess simultaneously the feature signals y_i obtained from the networks at the lower level and their spatial/temporal differences, so that it can estimate the overall combination and produce the system output signal y_0. Different classes of ANNs can be used for this purpose, but dynamic (recurrent) networks seem to be best suited for this task.[229, 866, 867, 870] If, however, information about mutual relative distances of the sensed features is contained in the input

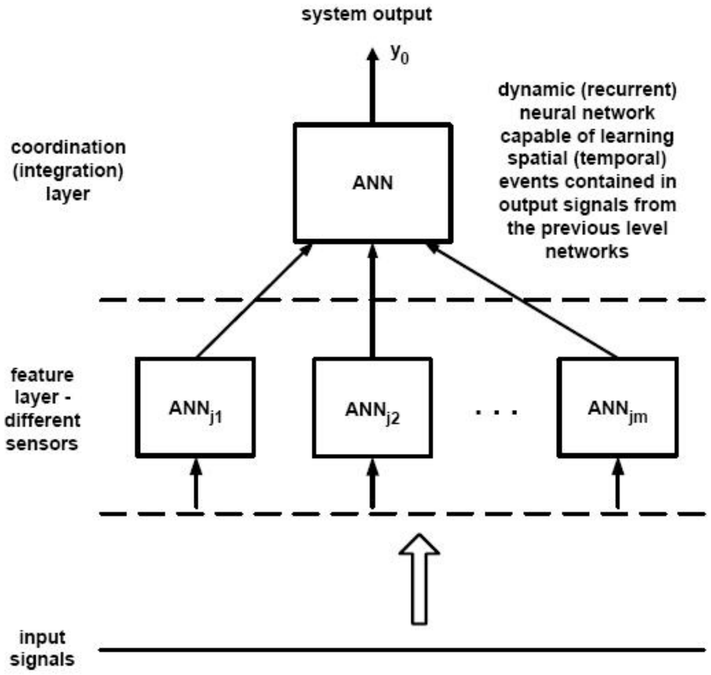
system output
y_0
dynamic (recurrent) neural network capable of learning spatial (temporal) events contained in output signals from the previous level networks
coordination (integration) layer
ANN
feature layer - different sensors
ANN_{j1}
ANN_{j2}
. . .
ANN_{jm}
input signals

Fig. 4. One possible general ANN structure for eukaryotic promoter recognition.

signals to the ANN on the coordination level, then such coordination ANN need not be recurrent.

In principle, suitable and sufficiently complex structure of neural network systems can allow a non-hierarchical approach. However, the problems of training such networks may be considerable, either from the viewpoint of the time required for the training, or from the viewpoint of network parameter convergence to reasonably good values, or both. For these reasons, in the case of recognition based on sensing different promoter subregion features, it is pragmatic to apply a hierarchical approach in designing neural network systems.[376, 482, 588, 712] With this basic idea in mind, one can build several different hierarchical structures of ANNs to suit eukaryotic promoter recognition. The structures presented in Figure 4 and Figure 5 show two of many possibilities.

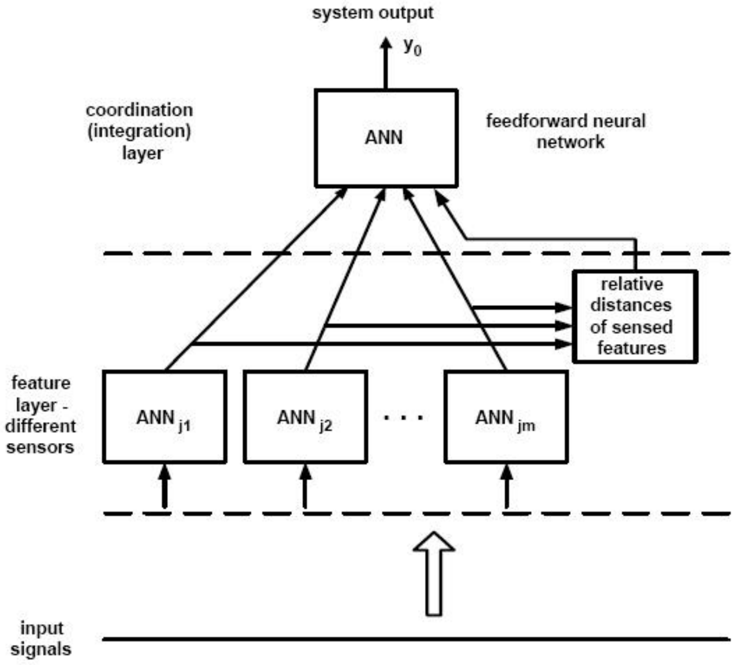

Fig. 5. Another structure of ANN-based system which uses also distances between the signals for promoter recognition.

5.2. *First- and Second-Level ANNs*

At the first level we have the "feature sensing" networks, *viz.* those that are trained to sense the presence of specific subregions of promoters on the basis of selected subregion features. These networks can be any static neural networks.[96, 135, 143, 241, 350, 465, 730] Although some solutions[535, 702−707] utilize time-delay neural networks (TDNN) proposed in Waibel *et al.*,[866] it is not necessary to have dynamic networks on this level. Moreover, dynamic ANNs may not be more accurate as feature sensors. Anyway, whatever choice of static neural networks is made for dealing with information processing at this level, the general problem to be further handled is explained by an example of two feature-sensing networks in what follows.

Let us assume that the system is based on the recognition of subregions R_1 and R_2, whose identified features are F_1 and F_2, respectively. Let the first layer neural networks ANN_1 and ANN_2, serve the purpose of identifying features F_1

and F_2, respectively, and produce continuous output signals y_1 and y_2. These output signals are assumed to be confined to the interval $[0, 1]$, where values close to 1 denote a strong signal—a high certainty that at the given position of the reading window the feature is detected. For lower values of the signal, this certainty reduces; with values close to 0 the chances that at the given window position the feature exists are slim.

In many simpler cases the system has to make the choice regarding the levels of output signals at which, and above which, the features are considered detected. These are frequently determined by the cut-off (threshold) value for each output signal. Note that there may be different cut-off values for each of the output signals, although this is not always necessary. In our case depicted in Figure 6 we consider two different cut-off values, one for each output signal.

Note also that the concept of cut-off values is not necessary, although it simplifies the problem to an extent. One can essentially use output values of the first layer networks and leave decisions about whether the features are detected or not to the higher-level coordination ANN. In this case the input signals of the coordination ANN may have a large number of components corresponding to signals sensed at different positions of the data window. Another possibility to avoid cut-off values is to use only the maximum value of a signal from the output of one sensor within the examined data window, and then to consider the maximum value of the output signal of another sensor, and the relative distance between them. This however is constrained by a serious conceptual problem, to be shown later.

Figure 6 shows a possible situation with the measured signals. According to Figure 6, any position from x_1 up to x_2, including these two positions, determines a valid candidate for the predicted existence of F_1. Analogously, the existence of F_2 is predicted on positions starting with x_3 and ending at x_4. One can raise the legitimate question: Why are there so many predicted possible locations for a “detected” feature? The answer is that in the feature domain the overlapping of characteristics of the correct and wrong subregions is huge which leads to a great number of wrong guesses of the ANNs.

The relevant combinations of positions from $[x_1, x_2]$ and from $[x_3, x_4]$ are subject to some constraints on distance $dist(R_1, R_2)$ between the subregions R_1 and R_2. Generally, for many subregions in a group of promoters we know some of the constraints regarding their locations. These constraints on $dist(R_1, R_2)$ can be used to define the logic to handle signals y_1 and y_2 in the context of their mutual distance. Obviously, the minimal and the maximal spatial distances between R_1 and R_2 for the used cut-off values, as determined by the ANNs, are given by

$$\min\left[dist(R_1, R_2)\right] = x_3 - x_2 + 1,$$

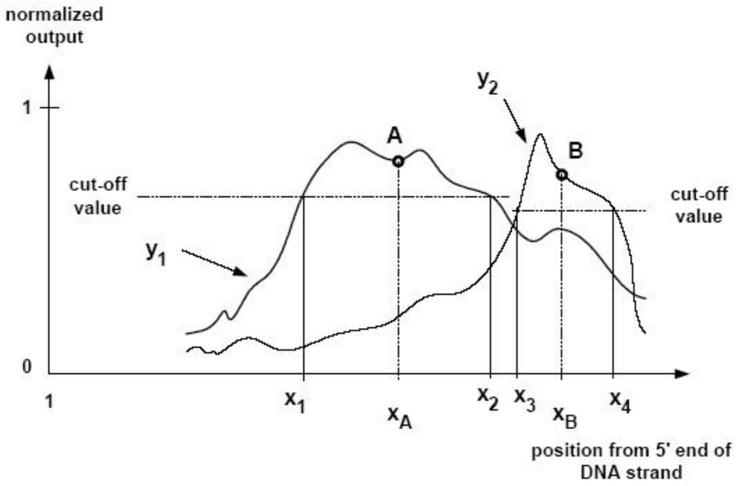

Fig. 6. Presentation of possible output signals y_1 and y_2 from sensor networks at the lower hierarchical level, their cut-off values, and reference positions x_1, x_2, x_3, and x_4 on the basis of which the spatial range (or time interval) for occurring features is to be determined.

$$\max\left[dist(R_1, R_2)\right] = x_4 - x_1 + 1.$$

If the general constraint on $dist(R_1, R_2)$ is given by $dist(R_1, R_2) \in [\alpha_1, \alpha_2]$, then for an arbitrary pair of points A and B, where A belongs to the candidate region $[x_1, x_2]$ and B to the candidate region $[x_3, x_4]$, only some combination of signal value pairs $(y_A(x_A), y_B(x_B))$ are allowed. Those that are allowed are determined by the condition

$$(x_B - x_A) \in [\alpha_1, \alpha_2]$$

Notice the ordering of features F_1 and F_2 contained in the condition above. The logic block operating at the input of the ANN at the higher hierarchical level should handle this problem, or the solution can be implemented via the ANN design.

In addition, it is not always the case that the strongest signals in the detected intervals $[x_1, x_2]$ and $[x_3, x_4]$ are the correct or the best candidates that characterizes the correct detection of features F_1 and F_2. This is a consequence of the same reasons highlighted in the discussion in Trifonov[835] and mentioned in the preceding sections. This is one of the most serious criticisms of solutions that use the maximum output values of the feature sensing ANNs to “recognize” the presence of the sensed feature. The decision of what are the best or successful combinations of the obtained values of output signals and their mutual distance in space or time has to be determined by training ANN on the coordination level. It is suitable that the coordination network be dynamic in order to be capable of learning the most successful combinations of points A and B—*i.e.*, to learn their mutual distance—in combination with the values $y_A(x_A)$ and $y_B(x_B)$, and possibly their ordering. The training of the coordination network and feature-sensing networks is to be done in a supervised manner.

5.3. *Cascade Composition of Feature Detectors*

To explain this scenario, denote by X_j the input signal for each of the feature detection networks ANN_j, $j \in \{j_1, j_2, ..., j_m\}$. These input signals are assumed to be composite signal vectors containing information on:

- the output Y_j of the feature detection networks from lower hierarchical levels;
- information Z_j on the basic raw DNA sequence level; and
- possibly other information—represented by vector G_j—acquired either during the training process, or from the biological constraints related to the data used.

Output Y_j of ANN_j is obtained by postprocessing the raw output y_j of ANN_j. This structure is depicted in Figure 7. Note that at each of the hierarchical levels a particular information can be used or not. Thus,

$$X_j = \langle \lambda_{j-1} \times Y_{j-1}, \lambda_{j-2} \times Y_{j-2}, \ldots, \lambda_1 \times Y_1, \lambda_{j,z} \times Z_j, \lambda_{j,g} \times G_j \rangle$$

Switches λ_i, $i = 1, ..., j-1$, and $\lambda_{j,z}, \lambda_{j,g}$, have value of 1, if the respective information they are associated with is used at the jth hierarchical level, or they have value of 0 otherwise. Training of feature detector networks is done successively in a partial hierarchy, where the network at the level j is trained by having all lower level networks included in the structure. The training process can take many forms. Such cascade hierarchical structures are expected to have good filtering characteristics, but it cannot be concluded that they are advantageous over the PCFD structures.

5.4. *Structures Based on Multilayer Perceptrons*

MLPs can be used to detect individual promoter components, as well as to combine accurately such evidence into higher hierarchical ANNs so as to provide the final prediction of promoter existence. We present two such solutions that are parts of larger packages aimed at gene recognition. The first one a part of the GRAIL package.[549] The other one named Promoter2.0[439] is an improved version of the

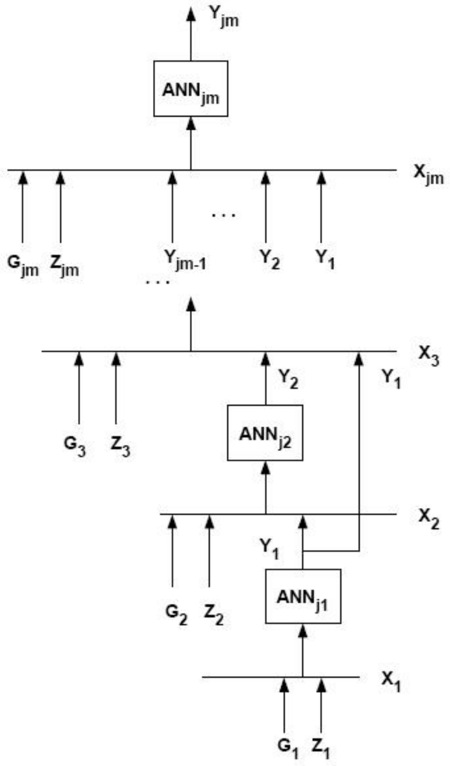

Fig. 7. The cascade structure of feature detectors.

promoter prediction part of the GeneID package.[306] Both of these solutions rely on the sensors for different promoter components and relative distances between them.

The solution presented in Matis *et al.*[549] for the GRAIL package is an example of a parallel composition of feature dectectors (PCFD) structure—*i.e.*, a conventional feedforward structure—which is in principle a two-level hierarchical structure. The MLP at the second hierarchical level receives information from different sensors related to promoter components, such as the Inr, TATA-box, GC-box, CCAAT-box, as well as from the translation start site, and the constraints on the relative distances between them. Such information is nonlinearly processed and the final prediction is produced. The sensors for different components can be based on ANNs, but this need not necessarily be the case. It is emphasized that this solution uses information about the presence of the translation start site, which is not part of promoter region.

The Promoter2.0 program[439] is an example of a cascade composition of feature detectors (CCFD) structure—*i.e.*, a cascade feedforward structure—and presents a different hierarchical solution that uses essentially four hierarchical levels for the final prediction of promoter presence. Based on the explanation provided in Knudsen,[439] the functioning of the system is roughly as follows. ANNs at each of the four levels consist of only one hidden neuron and one output neuron. In each of the ANNs, both neurons receive, as part of the input signals, information about the DNA composition in a binary form from a window of 6 nucleotides. In addition, they also receive in a floating point form the input signals representing the maximum outputs of the other networks on lower hierarchical levels multiplied by a separation function that relates to the normalized distances between the sensed components. The four promoter components that the ANNs are trained for are TATA-box, cap-site, CCAAT-box, and GC-box. Training of this network structure is done by a simplified genetic algorithm. The scanning of the DNA sequences is done in larger data windows of 200–300 nucleotides, within which a smaller window of 6 nucleotides slides, and the results are recorded for each of the four ANNs. After the recording of the maximum output values and the respective positions of the smaller window within the larger data window is finished for all four ANNs, the hierarchical structure produces the prediction of the presence of a promoter in the examined larger data window.

6. Time-Delay Neural Networks

A popular choice of dynamic networks that can inherently comprise the structure presented in Figure 4 is the time-delay neural networks (TDNN). Due to their special architecture, TDNNs are capable of learning to classify features that are invariant regarding their spatial (or temporal) translation. The basic unit (neuron) of a TDNN is a time-delay (TD) neuron, illustrated in Figure 8. These networks are initially used in the problem of phoneme recognition [866, 867] and in word recognition.[470] There are some similarities in the problems of phoneme recognition and in the recognition of promoters, where the latter is based on detection of the micro-promoter components. The features that need to be detected, if they are found in a promoter, may be on different mutual spatial locations. TDNN provides a convenient mechanism to make the neural network system insensitive to mutual distances of relevant events.

A TD neuron, depicted in Figure 8, has s inputs u_i, $i = 1, 2, ..., s$, and each of these inputs is also delayed by a maximum of N time units. Each of the inputs and its delayed version has its own weight $w_{i,j}$, $i = 1, 2, ..., s$, $j = 0, 1, ..., N$.

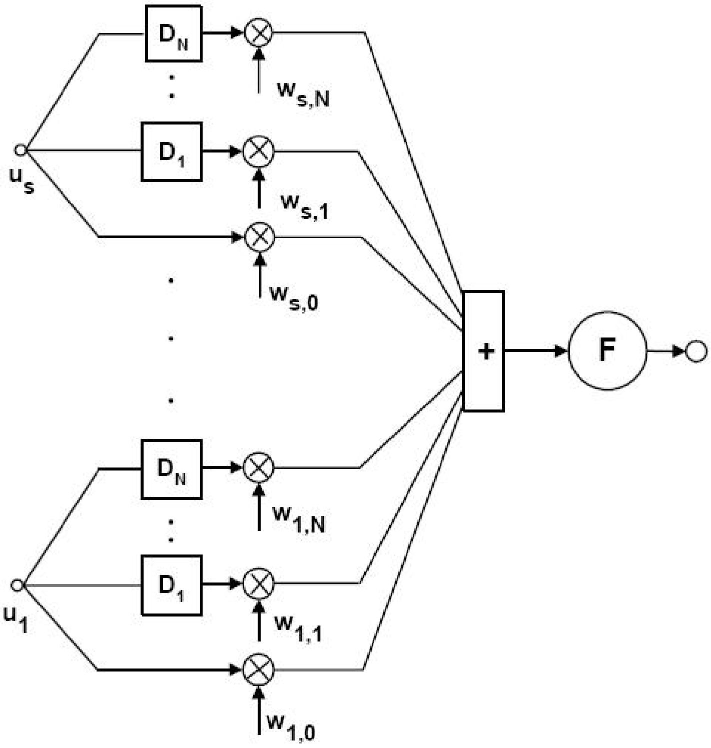

Fig. 8. The TD neuron in this figure has s inputs and each of these inputs is delayed a number of time (position) units (in our case N units). The weighted sum of all delayed inputs and the non-delayed inputs is passed through the nonlinear transfer element F.

This means that there are $s \times (N + 1)$ weighted signals that are summed so as to make the input to the nonlinear transfer element characterized by function F. This transfer element is sigmoid or threshold. Thus, using the notations $u_{\,i}(k)$ to mean the value of input $u_{\,i}$ at instance k and $y(k)$ to mean the value of output y at instance k, the value of the TD neuron output y at instant k is given by

$$y(k) = F\left(\sum_{i=1}^{s}\sum_{j=0}^{N} w_{i,j} \times u_i(k - j)\right)$$

Without aiming to be rigorous, we explain in what follows some of the main aspects of information processing in TD neurons and TDNNs. Particular applications may have many variations of these basic ideas. Let us assume that the input data are of the form of 1-D signals, either scalar or vector. The inputs that the TDNN receives are collections of frames of this data, and these frame collections are supplied to the TDNN in a sequential manner. However, due to a TD neuron input-delaying functionality, this process is equivalent to a sort of inherent rearranging of the 1-D input signals to the 2-D events; see Figure 9. Let us assume that a feature F_0 that we are looking for is contained in a subsequence contained in a data frame of length s which may be located at different relative positions with regard to some reference point, and that the variation of this position is limited to the data window of length w. In other words, the data window of length w contains the subsequence characterized by F_0 and comprises all of its possible positions relative to the reference point. We mention here that the reference point may be a position of another feature F_1 detected by a feature-sensing ANN.

The data window slides along the DNA string and the data it contains at each position of the window can be considered rearranged, so that all window data make—after the rearrangement—a 2-D form. The maximum number of N delays of individual inputs to the TD neuron determines the size of the so-called receptive field of the TD neuron. This size is $N+1$. The way that receptive fields are formed is explained in Figure 9. Data from s consecutive positions in a DNA string serve as inputs to the TD neurons. The consecutive $N + 1$ frames correspond to the TD neuron receptive field—*i.e.*, at moment k, the TD neuron reads data from a collection of $N + 1$ data frames that belong to its receptive field. As depicted in Figure 9, the 2-D form of the window data has $q = w - s + 1$ frames of data, where data in the first frame corresponds to $U(k) = \langle u_1, u_2, ..., u_s \rangle$, in the second frame to $U(k-1) = \langle u_2, u_3, ..., u_{s+1} \rangle$, and so on. Here, the counting of positions starts at the rightmost position of a frame on the DNA string. In order to be able to capture a feature irrespective of its relative position within the window, a layer of TD neurons should have at least $q - N + 1$ TD neurons. This however can be altered and the task of grasping spatial or temporal invariance can be transferred to a higher-level layer of TD neurons.

One of the most crucial ingredients of the translation-invariance of TDNNs is that weights related to different frame positions are copies of the set of weights that correspond to the first set of the receptive field frames. The learning procedure for the TDNN can be based on back-propagation.[728,729] Different learning procedures can be found in Waibel *et al.*.[867] The TDNN is exposed to a sequence of learning patterns so as to be able to learn invariance in the space or time translation of the relevant patterns. In the training process, all the weights are treated as independent—*i.e.*, they can change individually for one learning iteration step—but after that the related weights are averaged and these values are allocated to the relevant connections.[866,867] The process is repeated through a great number

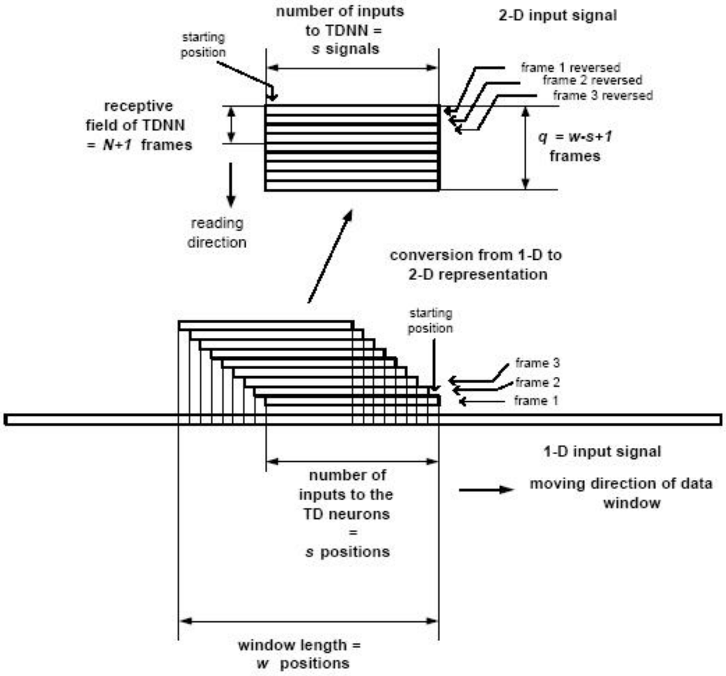

Fig. 9. An equivalent conversion of a 1-D form of input signal to a 2-D form for processing by TDNN. A window of length w positions slides along the input signal. The extracted window part of the signal is rearranged so that it becomes a collection of $q = w - s + 1$ data frames of length s, where s is the number of input signals to the TD neuron.

of learning iteration steps until the network performance function achieves the desired value.

The training phase of the TDNNs can be long and tedious even for shorter time patterns.[866, 867] However, it may achieve a very good classification performance. Owing to the great number of training parameters of TDNNs it is not possible to use many of the efficient training procedures based on second-order methods, such as Levenberg-Marquardt, *etc*. In addition, the backpropagation learning suffers the problem of determination of the proper learning rate. This problem can be circumvented by using adaptive optimized learning rates.[766–768]

6.1. *Multistate TDNN*

TDNNs may be expanded into the so-called multistate (MS) TDNN. MS-TDNN have additional hierarchical layers that use dynamic time warping[736] to enhance the recognition capability of the network and to enable recognition of specific ordered sequences of features independently of their relative locations. These networks are used for promoter recognition tasks in Mache *et al.*[535]

6.2. *Pruning ANN Connections*

A TDNN may have a very large number of weights that need to be adjusted during the training process. This may imply a prohibitively long training time. Another problem with the large number of weights is that, in principle, not all of them contribute significantly to the final network output. Thus, some kind of rationalization is advisable. The third aspect that leads to the desirability for network connection pruning is that, in general, the simplest network that fits the training data should have good generalization ability. These three aspects of ANNs with large numbers of internal connections lead to the requirement of ANN pruning.[701,824,937] The network pruning is intimately related to the problem of overfitting and to the determination of the optimal network size for the intended task.

There are many algorithms that may be used in pruning the networks. Probably the best known are the so-called optimal brain damage (OBD)[479] and optimal brain surgeon (OBS)[325] algorithms. Some of the pruning methods are based on the determination of elements of the network that are not crucial for the network's successful operation by calculating the sensitivity of the error function with regards to the change of these elements.[415,479,581,762] There are other approaches, such as the penalty-term methods that result in weight decays, where the error function is modified so that the training algorithm drives to zero the nonessential weights;[140,385,395,622,669,876,877] interactive methods;[771,772] local and distributed bottleneck methods;[459,460] genetic algorithm based pruning;[883] *etc*.

The problem to be solved by network pruning is how to simplify the network structure by eliminating a number of internal connections so that the pruned network remains good and achieves, after retraining, improved performance. Essentially, the large initial network size is gradually reduced during the training by eliminating parts that do not contribute significantly to the overall network performance. After elimination of the parts with insignificant contributions, the network is retrained; and the process is repeated until a sufficiently good network performance is achieved. The pruning of the TDNN in the NNPP program for promoter recognition during the training process appears to be crucial for the performance of NNPP.[703–706]

7. Comments on Performance of ANN-Based Programs for Eukaryotic Promoter Prediction

In this section we comment on the reported results of some ANN-based programs for eukaryotic promoter recognition that use recognition of individual promoter components as a part of the solution, *viz.* the NNPP program,[705] the Promoter2.0 program,[439] and the SPANN2 program.[44]

The NNPP program is based on the TDNN architecture that belongs to PCFD structures, similar to that presented in Figure 9. It contains two feature-sensing TDNNs that are used to react in the presence of two promoter subregions: TATA-box and Inr. There is also a coordination TDNN that processes outputs of the feature-sensing networks and produces the final output. The feature-sensing TDNNs are trained independently, and all three networks are pruned during the training process.[705] Promoter2.0 is based on a CCFD hierarchical structure as commented on before. The SPANN2 program is based on preprocessing transformation and clustering of input data. The data in each cluster are processed by a structure similar to PCFD one. In total, 11 ANNs are used in the SPANN2 system. Note that SPANN2 system combines the assessment of promoter region and the signal of the presence of the TATA motif.

In order to obtain some reasonable assessment of the capabilities of ANN-based systems for eukaryotic promoter recognition, we use the results obtained in an evaluation study of nine programs for the prediction of eukaryotic promoters, as presented in Fickett and Hatzigeorgiou.[249] These programs are listed in Figure 10 and denoted by program numbers 1 to 9. In addition, we use the results of three other programs that make strand-specific searches and whose results are reported after the study[249] on the same data set. These three programs are indicated in Figure 10 as IMC[626] (program 10), SPANN[42, 43] (program 11), and SPANN2[44] (program 12). The original result achieved by program 3 is replaced by a new result reported in Knudsen,[439] as it scores better. For details on programs 1 to 9 see Fickett and Hatzigeorgiou[249] and references therein. Since different measures of prediction accuracy are available, which produce different rankings of the achieved scores of prediction, we use the average score measure (ASM) as proposed in Bajic[50] to obtain a balanced overall ranking of different promoter prediction programs.

Without entering into the details of the evaluation test, we mention only that the data set from Fickett and Hatzigeorgiou[249] contains 18 sequences of a total length of 33120 bp and 24 TSS locations. A prediction is counted as correct if it is within –200 nucleotides upstream of the real TSS location and +100 nucleotides downstream of the TSS location. For obtaining relative ranking of the above men-

Program	program No.	TP	FP
Audic[38]	1	5	33
Autogene[446]	2	7	51
Promoter2.0	3	10	43
NNPP	4	13	72
PromoterFind[377]	5	7	29
PromoterScan	6	3	6
TATA[116]	7	6	47
TSSG[784]	8	7	25
TSSW[784]	9	10	42
IMC	10	12	39
SPANN	11	12	44
SPANN2	12	8	16

Fig. 10. List of programs whose performance is compared.

tioned 12 programs, we use the ASM as a balanced method for reconciling different ranking results produced by different measures of prediction success. Eleven different measures of prediction success have been used in Bajic[50] to produce the final ASM based ranking and the results are given in Figure 11. The ranking of performances is given in ascending order, so that the best overall performance got the rank position 1.

The ASM, which is a balanced measure, ranks the NNPP program[705] which has best absolute TP score, only at position 9 in the total ranking due to a very large number of FPs. On the other hand, the Promoter Scan program of Prestridge[679]—this program is not based on ANNs—although achieving the least absolute TP score, is ranked much better at 4th overall position. Another evaluated ANN-based program, Promoter2.0,[439] ranks overall at 5th position. For illustration purpose only we observe that the other two ANN-based programs, SPANN[42, 43] and SPANN2,[44] which use very different mechanism of promoter recognition, rank well in the overall system at positions 3 and 1, respectively.

This comparison indicates that ANN-based systems exhibit prediction performances comparable to other non-ANN based algorithms. However, since the content of the data set from Fickett and Hatzigeorgiou[249] is not very representative of eukaryotic promoter sequences, no conclusions about the absolute values of the

Rank Position	Program No.
1	12
2	10
3	11
4	6
5	8
6	9
7	3
8	5
9	4
10	1
11	2
12	7

Fig. 11. Ranking prediction performances of different programs based on ASM which uses 11 different performance measures.

compared programs should be made based on the comparison results in Figure 11.

The comparison analysis of Fickett and Hatzigeorgiou[249] is by now a bit outdated. A new generation of ANN-based programs[46–49, 51, 627] has been developed and evaluated on the complete human chromosomes 21 and 22. The results are presented in Figure 12 together with the results of two other non-ANN based systems, Eponine[211] and FirstEF.[190] Figure 12 gives ranking based on the ASM, as well as on the correlation coefficient (CC). Again, we find that some of the ANN-based promoter prediction programs have superior preformance compared to other non-ANN based systems. On the other hand, we also find that some of the ANN-based programs are considerably inferior to the non-ANN based solutions. This suggests that the real problem is not in the selection of the technology on which promoter predictions are based, but rather on the information used to account for the presence of promoters.

In summary, we have presented in this chapter some of the basic ideas that may be used, or which are already being used, in building ANN systems for the recognition of eukaryotic promoters in the uncharacterized DNA strings. These ANNs are trained to recognize individual micro-structural components—*i.e.*, specific motifs—of the promoter region. The results achieved by ANN-based pro-

Program	TP	FP	Rank by ASM	Rank by CC
Dragon Gene Start Finder[46]	198	52	1	1
Dragon Promoter Finder[48]	250	868	5	5
Eponine[211]	128	19	2	3
FirstEF[190]	234	300	4	4
McPromoter[627]	162	65	3	2
NNPP2[703]	249	3539	6	6
Promoter2.0[439]	135	1869	7	7

Fig. 12. Comparison of several promoter prediction programs on human chromosomes 21 and 22. The total length of sequences is 68,666,480 bp and there are in total 272 experimentally determined TSS.

grams are comparable with those of programs that do not use ANNs. However, the performance of promoter recognition programs are not satisfactory yet. The problem of eukaryotic promoter recognition represents, and remains, a great challenge in the general field of pattern recognition.