Master Thesis

A graph kernel approach to the identification and characterization of structured non-coding RNAs using multiple sequence alignment information

Mariam Alshaikh

April 15, 2015

Faculty of Engineering

Department of Computer Science

Chair of Bioinformatics

Prof. Dr. Rolf Backofen

Dr. Fabrizio Costa

Acknowledgments

Acknowledgments text...

Abstract

Bioinformatics supports all different life science fields by providing automated tools and methods. These techniques help handling problem instances in efficient ways. In general, all different life science fields have one shared aim, which is having a better understanding of all living entities. Different fields try to tackle common tasks from various perspectives. Many tools and methods have been proposed that help in answering biological questions for DNA, RNA, or proteins. For instance, some tools can do alignment, or secondary structure prediction on the sequence or structure level. The importance of non-coding RNAs is becoming increasingly evident. However, the discovery of many different kinds of these RNAs requires the availability of complex and sophisticated analysis tools in order to be able to handle this new abundance of data.

Within the context of this thesis, a novel approach was developed that can identify and characterize aligned ncRNAs by using a machine learning method. Multiple Alignment Graph Generator `MAGG` is the presented tool by this work. It is a graph generating tool based on ncRNAs sequence alignments. The principle idea behind `MAGG` is to encode ncRNA alignments as graphs. These graphs can be efficiently evaluated using a machine learning graph kernel based method. Graphs generated by `MAGG` can encode information on the secondary structure prediction of the alignment. In addition, they can encode the evolutionary conservation of sequences and structures. `MAGG` builts four different types of graphs. Each graph can represent various types of information in different graph structures. By applying `MAGG`, the machine learning graph kernel based method `EDeN` can perform functional classification for ncRNA alignments.

The results of `MAGG` combined with `EDeN` achieved a `ROC` value of 70% when testing on different types of graphs. However, some of the tested graphs are able to achieve a `ROC` of 90%. The first results obtained by this method are promising . Fur the future, it seems reasonable to assume that more improvement can be made in order achieve even better predictive performance. Also, an adaption can be done allowing the tool to work one different types of sequences like proteins for instances.

Zusammenfassung

Zusammenfassung text...

Contents

List of Figures

List of Tables

Abbreviations

MAGG	Multiple Alignment Graph Generator
$\mathscr{N}$	Single node graph
$\mathscr{S}$	Stem node graph node
$\mathscr{L}$	List node graph
$\mathscr{U}$	Unattached graph
DNA	Deoxyribonucleic acid
RNA	Ribonucleic acid
ncRNA	non-coding RNA
tRNA	transfer RNA
mRNA	messenger RNA
rRNA	ribosomal RNA
ML	Machine Learning
SVM	Support Vector Machine
CM	Covariance Model
EM	Expectation Maximization
NSPDK	Neighborhood Subgraph Pairwise Distance Kernel
ROC	Receiver Operator Characteristic
snoRNA	small nucleoar RNAs

Chapter 1

Introduction

In this chapter, we discusses the different biological knowledge necessary to understand this work. Followed by an overview of the bioinformatics field and how does it evolved solving biological problems in section 2. Section 3, is devoted to discuss alignment and its types with a summary on currently available techniques. The application of machine learning and its different approaches applicable in bioinformatics are discussed in section 4. The last section, Section 5, we describe and discuss the exact problem covered by this thesis.

1.1 Biological background

Genetics is an important field in biology as well as in bioinformatics. The field studies genes, the primary unit of inheritance in organisms. Today, it is well known that all organisms share a common mechanism of coping DNA and transferring the information encoded on the DNA into RNA and protiens. The field started to evolve, in 1868 when Friedrish Miescher performed an experiment on leukocytes which led to the discovery of DNA, what he called nuclein [5]. Nowadays, DNA refers to deoxyribonucleic acid. Years after, McCarty and MacLeod discovered that DNA is not protein as it was believed at that time(ref. mccarthy32). By 1951 the discovery of the bases of DNA had taken place by Chargaff [6]. Couple of years after, James Watson and Francis Crick described the correct structure of DNA [7]. All DNAs contain a double strand of biological information. Each strand is composed of small molecules called nucleotides. It contains four different nucleotides: Adenine (A), Cytosine (C), Guanine (G), and Thymine (T). In DNA double helix, (A) can form a so called base pair with (T), while (C) can pair with (G). DNA replication is the process where an identical copy from the current double stranded DNA is done creating a set of DNA helix. Watson and Crick notion of DNA helix duplication as it *"is, in effect a pair of templates, each of which is complementary to the other. We imagine that prior to duplication the hydrogen bonds are broken down and the two chains unwind and separate. Each chain then acts as a template for the formation onto itself of new companion chain so that eventually we shell have two pairs of chains, where we only have one before. Moreover, the sequence of the pairs of bases will have been duplicated exactly"* [8]. The statement of Watson and Crick explains the DNA replication. Finally, DNA molecules relay inherited information to RNA which in turn can either code for proteins or be non coding.

Ribonucleic acid known as RNA has been recognized at the same time as it reacts differently than DNA with respect to alkaline. Based on DNA as a template (RNA) is synthesized in a process called transcription. In this process, the RNA polymerase enzyme copies a segment of the DNA and transcribes it into RNA. Therefore, RNA has complementary nucleotides when compared to its template DNA but it replaces Thymine (T) by Uracil (U). Also, unlike DNA, RNA is single stranded and may fold on to itself. The difference between DNA and RNA is illustrated in Figure 1.1. In the 1950s, the discovery of RNA types started. Messenger RNA (mRNA) was the first to be discovered by Crick alongside his Central Dogma of Molecular Biology which states that, *information cannot be transferred from protein to either protein or nucleic acid* [9]. In other words, DNA forms RNA, which can in turn encode proteins. From this statement, the light arise on the functionality of RNA in synthesizing protein. The discovery of ribosomal RNA (rRNA) as it also plays an important role in protein synthesizing, where amino acids are carried to the ribosome to be linked together forming proteins [10]. Transfer RNA (tRNA) is another type of ribonucleic acid that is also important in the protein translation, where it serves as a connector between the nucleic acid of DNA or RNA and amino acid of protein [11].

As briefly hinted at earlier in the text, not all RNAs are translated into protein. Therefore, they are called non-protein-coding RNA (npcRNA) and are as popular as non-coding RNA. These RNA molecules encoded by genes in genome are functional but non-translated into proteins called non-coding RNA (ncRNA). Alanine transfer RNA is the first discovered ncRNA [12], followed by the ribosomal RNA (rRNA) and many more afterwards. It is important to note that these ncRNAs have vital functions even though they are not translated into proteins. Hence, some of them play a significant role in protein synthesis.

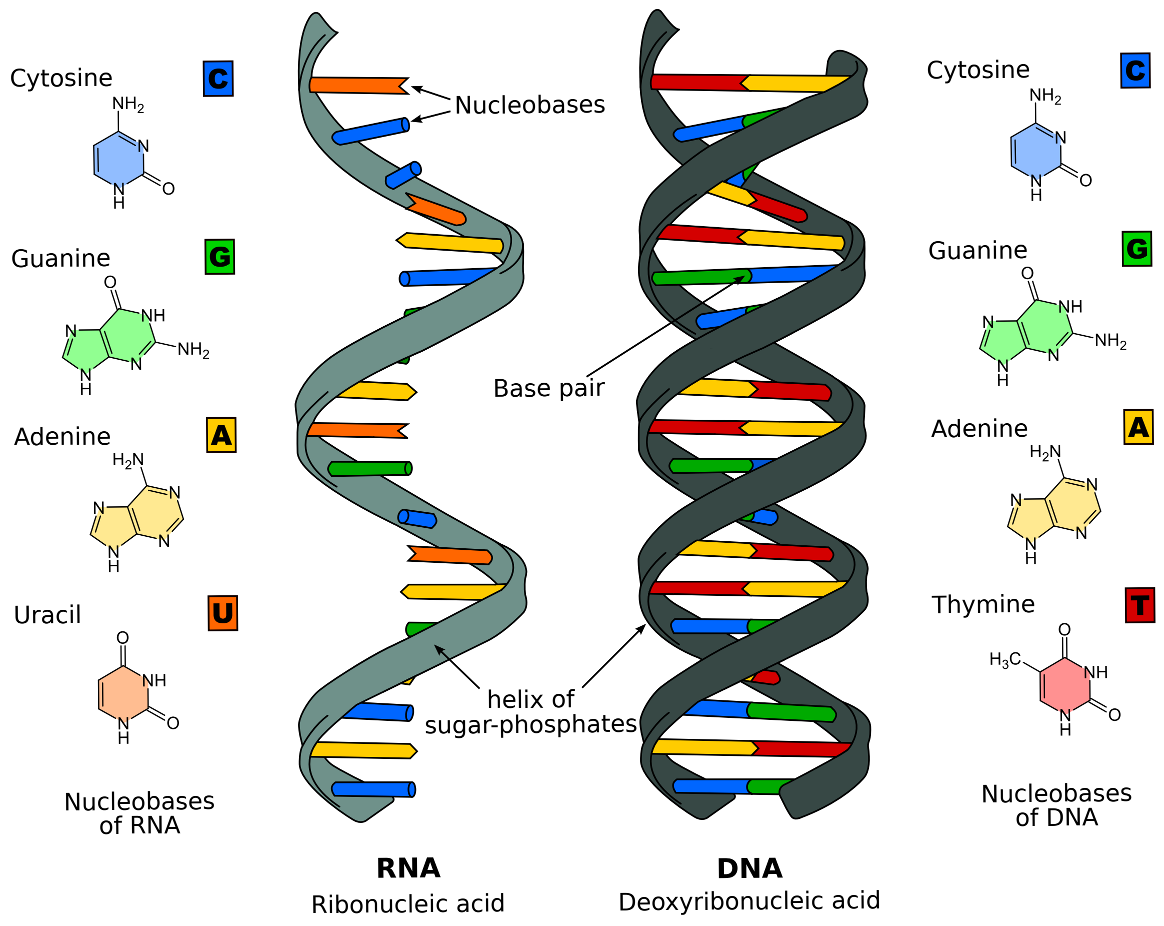


Figure 1.1: Difference between DNA and RNA reproduced from [1]. The figure shows a comparison of the double strand of the DNA (right side) and the single strand RNA (left side). It also shows the chemical structure of each nucleobase.

Prominent examples are, the transfer RNA (tRNA) which aids in translating the triplet code on the mRNA into amino acids during translation. Furthermore, rRNA is a vital component of ribosomes. YRNA is also a non-coding RNA necessary for DNA replication [13]. Furthermore, some of the non-coding RNAs play a big role in RNA processing and gene regulation. Since abnormality in some ncRNA can cause some disease such as, cancer and autism [14]. Over the years, many ncRNA functions have been discovered, and studies are still running in order to validate the functions of the other thousands of newly identified ncRNAs. The *cis*-regulatory elements are regions that function as binding motifs for the transcription factors of near by genes.

Identifying RNA motifs is an impotent way to understand and find the functionality of ncRNAs. RNA motifs are patterns that frequently occur in individual RNA molecules. In other words, the motif of an RNA is a shared pattern occurs repeatedly in every sequence. Such common motifs can be a hint at a shared function. Therefore, the analysis of motifs is a critical matter in the study of RNA or other biological polymers. Several algorithms are available to solve the problem of identifying the motif and their positions in aligned sequences [15–17].

Proteins are important components in the cell, since they have many central functions. Protein biosynthesis can be split into two subprocesses [18], transcription is the first step before the actual protein synthesis begins. In transcription one strand of the DNA double helix is used as a template by the RNA polymerase to synthesize a messenger RNA (mRNA). The coding part of the mRNA sequence can be described as a unit of several nucleotide triplicates called a codons. These codons are importent to translate the information on the mRNA to amino acids during the second step of protein biosynthesis, which is called translation. The translation step starts in the ribosome, where the it binds to the mRNA at the start codon generated in the transcription step. The mRNA is decoded in order to produce a specific amino acid chain, which is also called polypeptide. Then, this polypeptide is going to be carried by the tRNA and be bind to the appropriate codon in mRNA forming a complementary base pair with the tRNA anticodon. When the stop codon is reached, the ribosome releases the created protein.

1.2 Bioinformatics

As we pointed earlier, biology is the main field of life science that aims of studying all living entities, human, animals, plants, and so on. This field has evolved over the years from one field focusing on the biological aspect into more than one field with more complex point of view. Bioinformatics is the most recent field that came into the life science area. This field is interested in developing methods and tools to simulate and visualize biological data and systems with the aim of better understanding them using other life science fields. It involves a mix of biology, mathematics, statistics, engineering and of course computer science. Beside bioinformatics we have biophysics, where the study of the interaction between the different cell systems, for example, the interaction between DNA, RNA ,and protein, is done in quantitatively manner. Biochemistry is another life science field that combines chemistry with biology to get a field interested in studying the chemical process within an organisms [19]. All of these different fields are interested in understanding life and living entities.

The term of "bioinformatics" was used for the first time in the 1970s by Hogeweg and Hesper to identify their research field. As they, Hogeweg and Hesper, have defined bioinformatics as *"the study of informatics process in biotic systems"* [20]. Besides the biophysics and biochemistry, bioinformatics works as a research field to process and analyze huge amount of biological data discovered daily. Bioinformatics methods are very powerful in the current life sciences area, where they are used in the investigation on not only protein data with respect to its design and evolution but also DNA, as well as RNA. Different algorithms and approaches are designed and used in the studies of nucleotides, amino acids, and proteins with regard to their sequence and structure analysis, their alignments, and their secondary structure predictions.

In practice, bioinformatics is used in the identification of genes and nucleotides in addition to nucleic acid and protein sequences. Therefore, bioinformatics has become a very important field in the life science area, since it creates powerful tools to analyze biological data. This field has evolved over the years by turning from theoretical point of view into practical science field. It is an active researching field considering the different biological instances (DNA, RNA, and protein), finding new algorithms and methods or adapting existing ones. Beside the mathematical point of view, artificial intelligence, as a field of computer science, has been involved more and more into the analysis. It started exploring new horizon as neural network model [21]. generating optimization algorithms (ref.3note), which are very important in understanding biological systems as information processing systems. Pattern or data analysis is also a significantly important field in bioinformatics, where an analysis of variation of genetic patterns is considered as a first step, then a comparison between the outcomes models with the so called real data is done.

However, the overwhelming amount of new biological data discovered everyday arises the necessity of having a method that can mange them without losing any kind of information. Therefore, machine learning approaches, also as a computer science field, are very useful in life since. Since, it offers a lot of methods that can be used in analyzing, processing, optimizing complex data [2]. Models generated by these techniques can be used to explore the functionality of the considered data [22]. McNaught and Ananiadou, pointed the importance of text mining techniques in biology and biomedicine in (ref. textmining6note). Gene prediction is also another interesting branch in bioinformatics. In [23, 24] and [25] give an explanation of different approaches used in the identification of regions of genomic DNA as well as RNA and protein that encodes genes.

As mentioned before, all of that has only one aim that is having a better understanding of the biological process using some computational techniques like pattern recognition, machine learning, and visualization.

1.3 Alignment

All biological cell encodes a lot of different information. This information is represented as a sequence. A biological sequence is continuous molecule of nucleic or amino acid. In different words, a sequence is a chain of subunits, such as amino acid in proteins or nucleic acid in DNA or RNA molecules. In order to understand these sequences we need to analyze them to find their similarity. Therefore, searching among a set of sequences in order to find the maximal homology is a significant problem in molecular understanding and analysis. In general, alignment is the process of adjusting parts so that they are in proper relative position in order to find the their similarity. In bioinformatics, alignments measure the similarity or dissimilarity between different sequences or structures in order to understand and predict their function, for formal definition see Chapter 4. These sequences could be from proteins, DNAs, or RNAs. A simple example illustrating the idea of alignment is given in Table 1.2.

$$\begin{array}{cccccc} \mathrm{A} & \mathrm{G} & \mathrm{U} & \mathrm{A} & \mathrm{A} & \mathrm{A} \\ | & | & | & - & - & - \\ \mathrm{A} & \mathrm{G} & \mathrm{U} & \mathrm{U} & \mathrm{G} & \mathrm{G} \end{array}$$

Figure 1.2: [A simple example illustrating the concept of alignment. Here two words are aligned. A pipe indicates matching “word A and B agrees on the same latter” and dash indicates a mismatch.

There are two kinds of alignments: sequence, where the alignment is done based on the nucleotide level, and structure alignment, where the secondary structure of the sequences is taken into account to apply the alignment. Both types can be done globally or locally. In global alignment the similarity measure is done along all sequences [26], i.e, it is end to end alignment, where a consideration of the whole of all sequences is done. However, in local alignment a determination of similar subregions of nucleotides or protein sequences is done [27]. The aim of the local alignment is to find the maximum score for aligning sub-sequences of RNA or DNA instead of the whole sequence. Mostly, the use of local alignment arises when the dissimilarity between the sequences to be aligned are more than their similarity. In this case, the local alignment approach can find the most similar sub-region by comparing all possible different segments of the sequence and try to optimize the similarity measure. Such an algorithm was first proposed by Smith and Waterman in 1981 [27]. The field of sequence alignment started with Needleman and Wunsch [26], where they represent the similarity between sequences in two-dimensional array. They defined the maximum match as *‘the largest number of amino acids of one protein that can be matched with those of another protein while allowing for all possible deletions* [26]. Following this work, several algorithms, like the one of Fitch 1966 and Dayhoff 1969, having the same motivation. Seller [28] has also introduced an algorithm for finding regions of similarity between different sequence regions based on the evolutionary distance. Often more than two sequences need to be aligned. In this case, multiple alignment is used. Thus, many algorithms and tools are developed solving this problem, such as BLAST [29]performing a local alignment, FASTA [30], T-COFFEE [31]and others as a global alignment, and others.

An important benefit of alignment is that it helps in the prediction of protein function. Gene coding is also another advantage of alignment, where a prediction of a particular gene production is done (ref. seq,genome30note). Nevertheless, it is the first step in analyzing any sequences or structure.

1.4 Machine learning

Machine learning as a branch of computer science and artificial intelligence has very powerful techniques that are useful in the field of biology and bioinformatics [32, 33]. As a result of the exponential growth of biological data, cause the problem of handling this data abundance and extracting the useful information. Therefore, there is a need for tools and methods that help solving the problem of classifying this data, help in maintaining them, and then extract useful information. To this end, a simplification of the problem is generated, which helps in obtaining a predictive system. Figure 1.3, illustrates the different machine learning techniques applied in solving bioinformatics problems.

As a result of the increasing number of available genomic sequences discovered, an automatic approach is necessary to handle this data. Firstly, different mechanisms are implemented in order to solve the problem of identification of regulatory elements as in [34]. There a description of an implementation of a genetic algorithm that searches for an optimal combination of transcription factor binding sites (TFBS) in a set of given sequences is presented. Bockhorst [35] used a Bayesian network approach to predict operons in prokaryotic genomes. RNAs are also considered in the identification problem as in (ref. a computational13note). Proteins are very complex since they have a three dimensional structure, hence, we need to predict the structure in order to foretell their function. Therefore, the protein structure prediction is an important computational problem in bioinformatics. Additionally, the number of possible structures for any type of sequence is huge, which makes the problem of prediction even more complex. Thus, optimization techniques are necessary [36]. Secondly, comes the analysis of the data, which is going to be an easier and faster process after the optimization step.

Text mining and information retrieval are highly considered in computational biology. This is the outcome of the growing amount of research done in bioinformatics which also increases the amount of information available to search among. McNaught and Ananiadou [37] have pointed out the importance of text mining techniques in biology and biomedicine. These technologies, text mining and information extraction, have an impor-

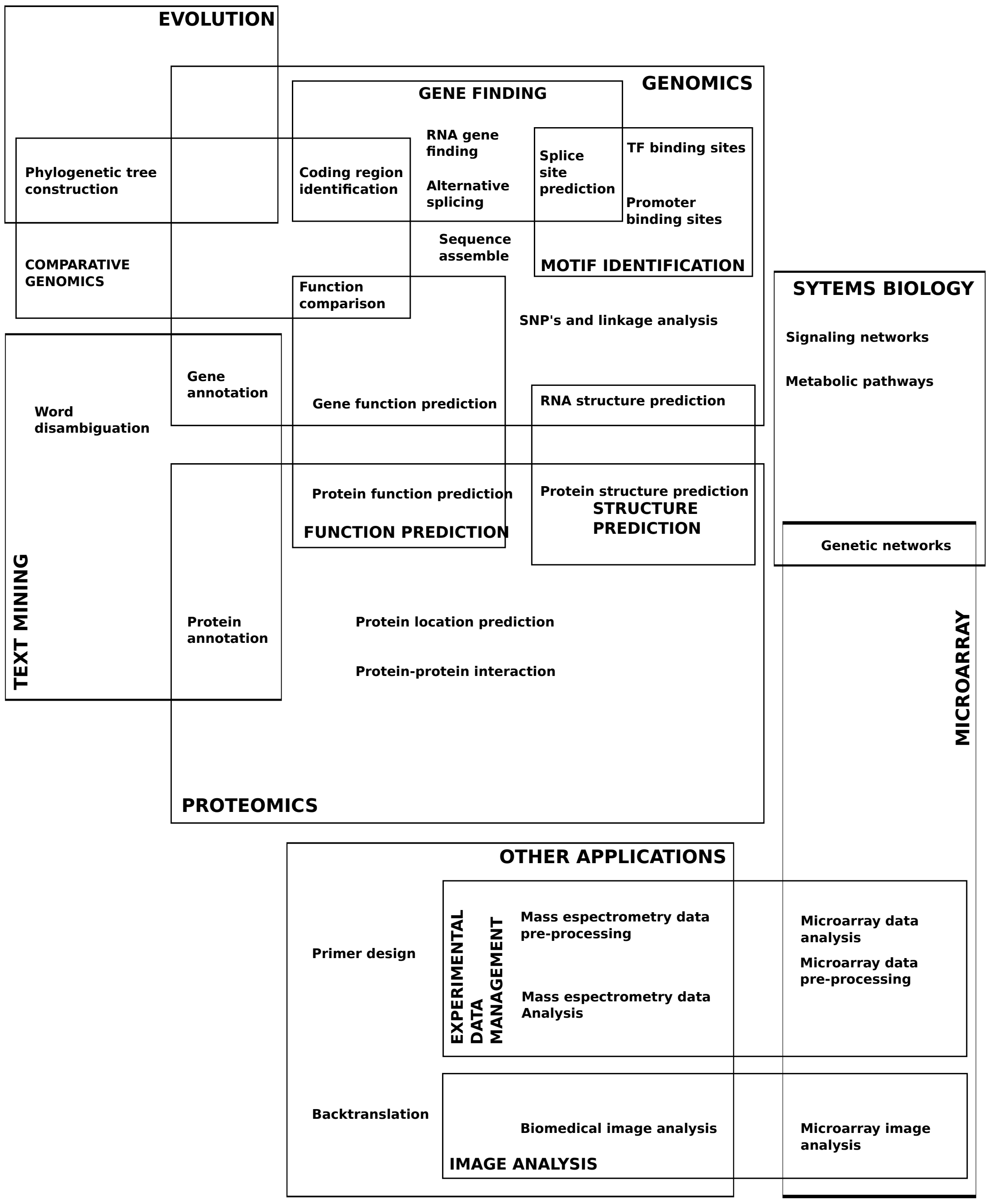


Figure 1.3: Machine learning in Bioinformatics adapted [2]. The scheme is an illustration of the different machine learning methods used in bioinformatics field. The authors have classified biological problems by the application of the different ML methods. Genomics and proteomics indicate the study of nucleotides and proteins respectively.

tant role in integrating relevant information through literature analysis. Another reference indicating to the use and the importance of text mining in bioinformatics is [38].

Classification is a very important process in machine learning and bioinformatics where a categorizing of the data is carried out. In biology, the idea of classification is based on the work of Linnaeus back in the 18th century. He was the first who started classifying species according to shared physical characteristics [39]. His work has developed into the modern classification system. Ernst Mayr defined biological classification as *'The arrangement of entities in hierarchical series of nested classes, in which similar or related classes at one hierarchical level are combined comprehensively into more inclusive classes at the next higher level* [40]. He also defined the class as a: *collection of similar entities* [40]. In the classification problem, a set of data or elements are given which need to be divided into classes that could identify or demonstrate their relationship to each other in a better way.

Most of the classification algorithms can give good results. Yet, finding the correct feature to give to any algorithm is a critical issue. Therefore, the feature selection is an important part of machine learning. The feature subset selection (FSS) can be described as: given a set of candidate features, select the best subset under some learning algorithm [2]. The problem can be viewed as a search problem, trying to find the best solution with the required criteria. Several techniques exist solving this problem. All of them apply different search techniques. Wrapper method, for instance, uses best first search as a base strategy to find an optimal feature subset selection methods [41]. Filter method is also another approach in feature selection. Kohavi R. and John G. in [42] have compared the two different approaches. The problem with the previously mentioned methods is that they can get trapped in local minima, which is solved by other techniques like genetic algorithms [43] for instance.

In bioinformatics, classification trees, where the classification is represented as a tree, are used for protein coding regions in human DNA [44]. In [44], Salzberg has used a decision tree technique in order to provide an accurate protein coding region classification. He concluded that his algorithm is very efficient and provides more accurate classification than older methods like linear discriminants and neural networks [45] and [46] respectivel., since it uses a decision tree which can adapt to work with different sequence lengths. Mathe and others have compared all the different prediction methods available until the year 2002 (ref. mathe24). Dynamic programming and evolutionary algorithms have been used and more precisely, are preferred to be used in the prediction of RNA secondary structure, since they can deal with exponentially large space [47–49].

One other approach used to analyze, and classify DNA, RNA, or proteins is kernel method. Kernel is a support vector machines (SVM) method used to handle high dimensional problems where the liner classification can not be applied. In other words, kernel method allow the liner classifier to work on non-liner problems. The trick is the specification of a function that can be used as a similarity function. Necessity for a method like kernel arises due to the high dimensionality of the input space, which makes the linear classification methods impossible to function. Nevertheless, most of the available algorithms and techniques in machine learning and statics are developed to work in linear space. However, in practice, the real world data requires nonlinear classifiers in order to return a successful prediction. Therefore, kernel method is used to transform this dimensional in to a dimensional can such classifier work on. This transformation is done by the kernel function. The new space introduced by it is called feature space [50]. Using this kernel function any non-liner space can be turned into a liner space which is easier and cheaper to be handled from a computational perspective. Graph kernel, is a type of kernel methods, in which the problem is represented as a graph and the similarity between pairs of graphs can be measured [50, 51]. More details on this topic are given in Chapter 5.

1.5 Problem description

The explosion in the discovery of non-characterized ncRNAs raises the need for efficient approaches that can deal with these data, where the analysis, the identification, the characterization, and the classification is important. Here, bioinformatic methods play a big role. Many strategies have been proposed concerning these problems. Some methods can do the alignment while others take the alignment and predict the structure. However, the classification problem of the functionality of ncRNA is sill not solved automatically and is frequently done in a manual fashion. From the ncRNAs the identification of the *cis*-regulatory regions of special interest, since they play an important role in the regulation of gene transcription [52]. This makes the identification of these regions very important as it can lead to a deeper understanding of gene regulation.

Zasha Weinberg and his colleagues [53–55]strongly focused on the *cis*-regulatory in order to discover and the identify these regions in ncRNA. They identified a big set of ncRNA motifs [53–55]. However, because the lack of ncRNA analysis tools that can characterize them by their functionality, they did the classification manually. Therefore, this thesis is introducing an approach that attempts to carry out such classification automatically.

Precisely, The developed approach is a bioinformatics strategy based on machine learning method that can use the available transcribed ncRNAs and distinguish the functional, that has known function, from the non-functional ones. The approach uses aligned ncRNA sequences that have been generated by CMfinder method. These alignments are processed to build a novel graph that encodes the alignments. A machine learning method known as EDeN [56] is then used for the learning and the prediction step. The method was tested on data provided by Z. Weinberg [54]. The aim is to find the best graph representation that leads to the maximum discriminative power between functional and non-functional RNAs. The experiment was carried out using different types of alignment information represented in a graph structure. The graph encodes information on the secondary structure prediction of the alignment as well as the evolutionary conservation of sequence and structure. We have found that some information, like the conservation for instance, leads to accurate classification. More details about our method and the results discussed in chapter 6 and chapter 7 respectively.

This thesis is divided into nine chapters. The first chapter is an introduction to basic background of the work presented here followed by the thesis problem description. In the second chapter, the related work of the approach is discussed. Preliminaries is chapter three, where some important definitions used in this thesis are given. Alignment definition, and types is discussed in chapter four. Nevertheless, the main alignment method 'CMfinder' used to generate the data used in this thesis is described in details in the same chapter. Chapter five, discusses the machine learning approach and the kernel method used in this work. A theoretical description of the proposed approach is given in the sixth chapter followed by the implementation details. The evaluation of this approach is discussed in chapter seven. Finally, in chapter eight draw the conclusions of this thesis.

Chapter 2

Related Work

A lot of researches are focusing and dealing with the problem of sequence and structure alignment, which is a way of sorting the DNA, RNA or protein sequences in such a way to find the similarity regions. These regions of similarity may lead to identify new functions, structures or evolutionary relationship between these sequences.

The problem of identifying and classifying non-coding RNA is an active matter in both biology and bioinformatics. Since, the recent discoveries of large amount of non-coding RNA this problem became harder and harder. A lot of researches are focusing and dealing with the problem of sequence and structure alignment, which is a way of sorting the DNA, RNA or protein sequences in such a way to find the similarity regions. These regions of similarity may lead to identify new functions, structures or evolutionary relationship between these sequences. Therefore, a big need of having an automatic identification and characterization tools has raised. However, all methods available can do the alignment or-and then do the structure prediction. Then, because there is no available analysis tool, the analysis of these alignments or the predicted structure is done manually, which is insufficient and time consuming. As a consequence, Zasha and his group in [54] has used the CMfinder [57] to do the alignments and the secondary structure prediction but then did the analysis manually to identify the best candidates and characterize their possible function. Therefore, we have integrated this idea of having a tool that can do the identification of functional and non-functional non-coding RNAs. In this work, we have adapted an automatic approach that could do the analysis automatically using a machine learning technique proposed by F. Costa [56].

Yet, some approaches has used stochastic context-free grammars as a way of RNA secondary structure prediction as Kundsen and Hein have done in [58]. Where they use prior information about the RNA structure in order to have a maximum a posteriori (MAP) estimation of the aligned RNA secondary structure, where the alignment is structural alignment.As a result of there approach a single secondary structure for the alignment sequences. However, they have the drawback of that they need an initial alignment and this alignment need to be a good one, otherwise the computation going to stuck on local maxima in the likelihood function they are using for the alignment. Our approach based on CMfinder [57] does not need any initial sequence or structure alignment, since it can start with now alignment at all. Another approach has been proposed by Hofacker [59], where he introduced a tool uses the combination of thermodynamic and phylogenetic information to do the sequence alignment and then predict the structure. However, the method can only predict conserved structure only if a sufficient large number of sequences are given, otherwise purely prediction going to be generated. Hence, this method is not going to work with datasets with low sequence similarity. Moreover, A. Coventry [60] has implemented a tool that identifies conserved sequence alignment RNA motifs by considering reverse-complementary regions. The tool could accurately predict the structure of small alignment mutation. Note, all the previous approaches did the alignment as first step and then did the folding in order to predict. Some other methods has considered the opposite techniques, where they apply the folding first and then find the consensus secondary structure [61] . Nonetheless, this approach is not accurate with single sequence prediction but with the help of dynamic programming some other method had solved this problem as in [62] where Eddy and Dowell have done the folding and the prediction simultaneously. Nevertheless, it is computational expensive. Probability played a role solving this problem by using expectation maximization (EM) algorithm for instance. Covariance model has been applied in order to infer a pretty accurate prediction model. Moreover, this technique could start with no alignment, which is an advantage, but it could not discover the the local motifs. In our approach, CMfinder [57] could start with no alignment and uses the probabilistic model to find the maximum consensus structure. It uses the covariance model (CM) [3] to model the RNA motif and then extend this motif by the expectation maximization (EM) algorithm [63, 64].

Machine learning is a resent computational approach considered in the analysis of the RNA sequences and structures alignments. Hidden Markov model (HMM) has been considered in the investigating in the bio-molecular sequences it also address the problem of analyzing multiple sequence alignments [65]. Neural network and support vector machine are also used in the extraction of shared features in set RNA sequences for the identification of new functional RNA in genomic sequences as what Carter did in [22]. Kernel and graph kernel are a very active techniques in the analysis of biological instance, for instance, the graph kernel is used in the extraction of protein-protein interaction, where the problem is given a binary protein interaction to an extraction system , how would the system distinguish between interaction from the non-interaction. This problem has been addressed by A. Airola and his group in [66, 67] and [68]. Fast neighborhood sub-graph pairwise distance kernel (FNSPDK) proposed by Costa [56] is a new approach than can decompose any problem presented as a graph into sub-graphs or more precisely into all pairs of neighbourhood sub-graphs and compare them in order to compute their similarity. The NSPDK tool has proven its efficiency on the analysis and prediction of different problems. In order to use the NSPDK the problem has to be presented as a graph which allows it to learn and decompose it into small sub-graph, compare them in order to score their similarity. It has gave a good result in the prediction of peptide recognition modules (PRM), that can predict 0.73 area under the precision-recall curve [68].

In this thesis, we have used an already aligned structured RNA from bacteria and archaea provided by Z. Weinberg [54] as experimental data. Then, We have introduced an automatic tool that can build a graph out of these data. As a last step, we have used the machine learning tool `EDeN`introduced by Costa [56] as a comparative tool to measure the similarity of these structures, characterize them in order to identify there functionality.

Chapter 3

Preliminaries

In this chapter we provide some definitions and notations that are necessary for understanding our work.

We use the term *graph* in the following sense.

Definition 3.1 (Graph). *A graph G is a tuple (N, E) where:*

- *N is a set of Nodes.*
- *$E \in V \times V$ is a set of edges connecting two nodes. We denote an edge $(u, v) \in E$ by uv, if there exists only one such edge.*
- *N is refer to by $N(G)$ and to E by $E(G)$.*
- *$\mathcal{L} : N \cup E \rightarrow L$ is a function mapping nodes and edges to labels from a label set L.*
- *A path is a sequence of edges $(n_0, n_1), (n_1, n_2), ..., (n_{n-1}, n_n) \in E$ between two nodes v_0 and v_n, where n is the length of the path. A node v is reachable from a node u if there exists a path between u and v.*
- *The distance $D(u, v)$ between two nodes $u, v \in V$ is the length of a shortest path between u and v or inifinity if there exists no path between them.*
- *G is a rooted graph, if there exists a nodes v such that all other nodes are reachable from v. We call v root node and write G^v, if G is a rooted graph.*

We denote the set of all graphs by $\mathcal{G}$.

We apply a common definition of string concatenation.

Definition 3.2 (**String Concatenation**). *String concatenation is an operation* $concat(str_0, str_1)$*, that takes two strings* $str_0 = a_0 ... a_n$*,* $str_1 = b_0 ... b_m$ *and returns* $a_0 ... a_n b_0 ... b_n$*. We may write* $str_0 str_1$ *or* $str_0 \cdot str_1$ *instead of* $concat(str_0, str_1)$*.*

Furthermore, we talk about graph Isomorphism and refer to the following definition.

Definition 3.3 (**Graph Isomorphism**). *Two graphs* $G_1 = (V_1, E_1)$ *and* $G_2 = (V_2, E_2)$ *are isomorphic if there exists a bijection* $\theta : V_1 \to V_2$ *such that for all vertices* $u, v \in V_1$*, there is an edge* $(u, v) \in E_1$ *iff there is an edge* $\theta(u), \theta(v) \in E_2$*. We donate isomorphic graphs by* $G_1 \simeq G_2$*.*

Isomorphism invariant is a graph property that is identical for two isomorphic graphs.

We also define the relation between any two instance as:

Definition 3.4 (**Relation**). *A relation R is a subset of the cross product of a set of sets* $A_0, A_1, ..., A_n$*:*

$$R \subseteq A_0 \times A_1 \times ... \times A_n$$

The inverse relation R^{-1} *is a subset of* $A_n \times A_{n-1} \times ... \times A_0$*.*

Chapter 4

Alignment

An alignment of any of the nucleotide, RNA or DNA, or protein is simply a comparison of their structures or sequences. It is used to detect the similarity of subregions of these sequences or structures. The similarity determination may leds to predict their functionality or evolutionary relationships. Since, there is different interest of understanding or analyzing these biological instances, different approaches have been developed solving different matters. This chapter describing the different types of alignments and the different approaches available. Then, an elaboration on the method used to create the data used in the evaluation. For simplicity and because the work presented in this thesis is foucsed on RNA, the explanations provided here only for RNA.

The first section gives formal definition of the alignment and its types. Section 2 discussing an overview and definition of RNA folding is given. The main part in this chapter is section 4, where an extensive discussion of the CMfinder approach, since it is the method producing the data of this thesis.

4.1 Alignments and alignments types

This section starts with a formal definition of the alignment then a discussion on the different types of alignment available is given. In order to do the alignment,an adjustment to the sequences has to be done.The edit operation definition 4.1 define the operation that can be applied on the sequences in order to align them.

Definition 4.1 (**Edit Operations**). *Given a finite alphabet Σ with a gap symbol $- \in \Sigma$. An edit operation is a pair $(x, y) \in (\Sigma \cup -) \times (\Sigma \cup -)$.*

A pair (x, y) is called:

- *substitution, if $x \neq -$ and $y \neq -$*
- *insertion, if $x = -$ and $y \neq -$*
- *deletion, if $x \neq -$ and $y = -$*

We write $a \rightarrow_{(x,y)} b$, if b is generated from a by the replacement of x with y (substitution), or by deletion of one x (deletion), or respectively insertion of one y (insertion).

After aligning the different sequences, one needs to find the alignment that has the minimum edit distance. That means, the alignment that applies the minimum number of insertion or deleting. This is defined in Definition 4.2.

Definition 4.2 (**Alignment (general definition)**). *Given two words $x, y \in \Sigma^*$, an alignment of (x, y) consists of two sequences $x^{'}, y^{'} \in (\Sigma \cup -)^*$ such that:*

- $\left|x^{'}\right| = \left|y^{'}\right|$, *i.e. x and y have the same strings length*
- $\forall\ 1 \leq i \leq \left|x^{'}\right| \ :\ \neg(x_i^{'} = - = y_i^{'})$, *i.e. it is not possible to align a gab with a gab*
- $x^{'}\ |_{\Sigma} = x$ *and* $y^{'}\ |_{\Sigma} = y$, *i.e removing the gap from the alignment yields the original sequences.*

Then the alignment distance is: $w(x^{'}, y^{'}) = \sum_{i=1}^{\left|x^{'}\right|} w(x_i^{'}, y_i^{'})$ *The alignment distance of two words $x, y \in \Sigma^*$*
$D_w(x, y) = min\{w(x^{'}, y^{'}) \mid (x^{'}, y^{'}) \text{ is alignment of } (x, y)\}$

Where:

$$w(x, y) = \begin{cases} 1 & \textit{if insertion or deletion} \\ 0 & \textit{match} \end{cases}$$

Definition 4.3 (**Alignment (general definition)**). *Given two words $x, y \in \Sigma^*$, an alignment of (x, y) consists of two sequences $x^{'}, y^{'} \in (\Sigma \cup -)^*$ such that:*

- $\left|x^{'}\right| = \left|y^{'}\right|$, *i.e. x and y have the same strings length*

- $\forall\ 1 \leq i \leq \left|x'\right|\ :\ \neg(x'_i = - = y'_i)$, *i.e. it is not possible to align a gab with a gab*
- $x' \mid_\Sigma = x$ *and* $y' \mid_\Sigma = y$, *i.e removing the gap from the alignment yields the original sequences.*

4.1.1 Sequence alignment

In sequence alignment the focus is set on the primary structure, i.e. the sequence of the RNA. It aim at finding similar sequences or sub sequences which may aid of homology and hence functional prediction. In order to compare two or more sequences, gaps are inserted so that all the sequences, in addition to the resulting alignment, have the same length. A scoring function is needed in order to compute the similarity of the sequences and then get the maximum similarity. Needleman and Wunsch [26] considered the matter of alignments as a dynamic programming issue. There they solve subproblems of the entier problem first, i.e, align sub sequence of the whole sequence, and then recursively solve larger one. For instance, consider the following example:

Example

let A and B be two RNA sequencesthat need to be aligned, where A=GAATTCAGTTA and B=GGATCGA. We have the scoring function as:

$$s(i,j) = \begin{cases} 1 & \text{if } i = j \text{ (match)} \\ 0 & \text{Otherwise} \end{cases}$$

The final alignment is calculated by building a matrix of similarity by comparing every ith position in sequence A and the jth position in the B sequence as shown in Figure **??** and then the final result is given in Figure 4.1.1.

The final sequence alignment is

4.1.2 Structure alignment

Beside the RNA sequence information, the structure plays a very big role in finding the similarity of RNAs. The structure of RNA sequence provides more information which makes the similarity measurement more accurate. In the structural alignment, a consideration of the sequence structure is taken into account. Structure alignment plays a big role in the

$$\begin{array}{ccccccccccccc}
 & & G & A & A & T & T & C & A & G & T & T & A \\
 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
G & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
G & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 \\
A & 0 & 1 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 3 \\
T & 0 & 1 & 2 & 2 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 \\
C & 0 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 4 & 4 & 4 & 4 \\
G & 0 & 1 & 2 & 2 & 3 & 3 & 4 & 4 & 5 & 5 & 5 & 5 \\
A & 0 & 1 & 2 & 2 & 3 & 3 & 4 & 5 & 5 & 5 & 5 & \mathbf{6}
\end{array}$$

Figure 4.1: The alignment matrix for two sequences generated using the Needleman and Wunsch algorithm.

```
G - A A T T C A G T T A
|     |   | |   |     |
A G - A - T C - G - - A
```

Figure 4.2: The final sequence alignment with an alignment score of 6. A pipe indicates matching "word A and B agrees on the same latter" and dash indicates a mismatch.

discovery of the function of ncRNA. Therefore, finding the similarity of these ncRNAs is mostly based on the structure alignments.This is because, most functional RNA molecules exhibit a characteristic secondary structure that is highly conserved in evolution. In [69], David Hoksza and Daniel Svozil have found a method that can search for RNA structure similarity. The developed tool uses a pairwise comparison method based on 3D similarity of the generalized secondary structure units.

4.1.3 Local and global alignment

When the query sequences are more similar to each other the the global alignment is used in order to give a high similarity measure. In the global alignment [26] the similarity measure is calculated along the whole sequence. In different words, it is end to end alignment, where a consideration of the entire of multiple sequences is done. On the other hand, in the local alignment [27] a determination of the similarity is focused on locally similar regions instead of the whole sequence of nucleotides. The aim of the local alignment is to find the maximum score for aligning sub-sequences of RNA. Mostly, local alignment is applied when the dissimilarity between the sequences to be aligned is bigger than their similarity. Here, this approach can find the most similar region by comparing all possible different segments of the sequence and try to maximize the similarity measure.

4.1.4 Multiple alignment

When the alignment considers more than two sequences, then it is called multiple alignment. Multiple alignment can be done on sequence or structure level. Mostly the need for of the multiple alignment arises when the identification of the conserved sequence regions across a large group of sequences is needed. Several algorithms are available applying multiple alignments like BLAST [29], applying multiple-local alignment. On the other hand, FASTA [70] is doing the multiple-global alignment.

4.2 RNA folding

After having the alignment, and in order to predict the function of an RNA, one needs to fold this alignment to predict the structure. Different methods are available that execute the folding for the aligned RNA sequences. For instance, alifold [71] is a method used to predict the consensus secondary structure of a set of aligned sequences. The method does the prediction by using dynamic programming taking the sequence covariation A.2 into consideration.

4.3 CMfinder

Since *cis*-regulatory elements of mRNA and tRNA pointed to diversity of biological functions for non-coding RNAs, such as replication, localization, and translation, very powerful computational models are needed. The available identification techniques required not only fluent of manual work but they also fail when the sequence conservation is too low, because of the poor alignment, or too high owing to the lack of sequence covariation. Nevertheless, some of the available methods do not cooperate with some RNA homology search tools. For instance, fast genome-scale covariance model (CM) [3] search is a probabilistic model that works with aligned as well as unaligned RNA sequences. It describes the secondary structure of these sequences. However, it has the problem that it can not deal with large size of RNA sequences but only between 150-200 nucleotides. The method also needs large number of sequences in order to provide a good prediction model. As another example for homology search we have Rfam database [72]. which is a database of ncRNAs families generated by multiple sequence alignments and covariance models. This database aims to provide an automatic system for analyzing and annotating sequences in order to find homologies to the known structural RNA. Though, the database is acting very well on recognizing the non-coding RNA families, it has some limitation of recognizing some families such as microRNA and small nucleoar RNAs (snoRNA). Nevertheless, both of these approaches, CM and Rfam, did not solve the many problem of identifying and characterizing conserved secondary structure RNAs motif. However, CMfinder [57] is a motif finding approach that considered solving these problems. It is an expectation maximization algorithm that uses the covariance model in order to capture the secondary structure of aligned RNA sequences. It does not suffer from the problem of dealing with large size of RNA sequences, since it uses heuristics for choosing a set of candidate elements as an initial step for the expectation maximization step. This way the problem of scalability has been solved. This method also has the advantage of being applicable on unaligned and unrelated input sequences. In addition, it provides not only a structural alignment but also a statical model that can be used for homologous search. Having a structural alignment model is a big advantage, since; the model can be extended and refined iteratively.

Basically, in order to address the problem of identifying conserved secondary structure motifs, CMfinder works in two steps. Firstly, it uses the covariance model (CM) to model an RNA motif that describe the primary sequences consensus and the secondary structure of an RNA [3]. Secondly, an expectation maximization algorithm is considered to search the motif space. In order to start the process, CMfinder needs to identify the motif location and structure efficiently. Hence, firstly, a candidate selection is done by computing the minimum free energy of every sub sequences for each input sequence [61]. Then, the top ranking candidates are selected based on the minimum free energy, defined as 4.4 of the sequence scaled by its length. In the second step, a comparison of the predicted secondary structure is done. In order to consider the comparison on the structure beside the sequence, the comparison of the candidates are done using covariance model algorithm produced by Hofacker in [61] on the single base or base pair level. To improve the accuracy Yao and his group [57] has aligned two candidate only if their local conserved regions found by BLAST search [29] are compatible. At the end of this phase, an initialization of the alignment is identified based on the covariance model by choosing one candidate from each sequence. This initial alignment is the seed for the second phase where an expectation maximization algorithm is used in order to be able to estimate the model and the motif instances simultaneously.

Definition 4.4 (Minimum Free Energy adapted from [73]). *Given RNA sequence S, compute two possibly different energies for each subsequence s_{ij} such that: for all pairs i, j satisfying $1 \leq i \leq j \leq N$, let $w(i, j)$ and $v(i, j)$ be the two minimum free energy of* *all possible admissible structure formed from s_{ij} in which s_i, s_j are basepair. Then the minimum free energy E is:*

$$E = min\{w(i,j), v(i,j)\}.$$

If s_i, s_j are not basepair then E=∞.

The expectation maximization (EM) algorithm is an unsupervised method that iteratively runs in order to find the maximum likelihood. It works in two essential steps: first, the expectation step (E), where expectation function of the likelihood is estimated which going to be fed to the second step: the maximization step (M). In the maximization step computations of parameters that maximize the likelihood function from the expectation step. These two steps are repeated iteratively until the maximum likelihood is found or until the termination condition is met. MEME [63, 64, 74] is an extended algorithm of the expectation maximization (EM) for identifying motifs in unaligned sequences where little or no information is known about the motifs. This method allows multiple appearance of a motif to occur in any sequence and ignores sequences that do not share it, which is a good advantage for CMfinder for getting the maximum similarity of a motif.

In CMfinder the focus is to find the finite mixture model that gives the highest probability of containing a sequence the tested motif. Based on the first alignment, created by the CM model, an estimation of the finite mixture model is created. Then, using the expectation maximization algorithm, in both E-step and M-step, a refinement of the model parameters is done. Hence, In the expectation step (E-step), a computation of an optimal alignment is done of each candidate. The resulting probability from this step is the probability of that candidate with its suggested structural alignment being a motif instance. Next, in the maximization step (M-step), an update of the CM motif, estimated by the covariance model, and the mixture probability is done. After that, an adjustment to the candidates is considered before the second EM iteration starts, by using the covariance model to scan each sequence and pick the top hits and treat them as new candidates. Iteratively, the EM runs until it find the best alignment. An illustration of this process is shown in Figure 4.3.

Testing the CMfinder on selected 19 families from Rfam, which has a large collection of multiple sequence alignments for non-coding RNAs, they have found that it has significantly outperformed RNAalifold [59, 71], Pfold [58], Foldalign [75], and others. The table in Figure 4.3 shows a summary of comparing CMfinder to some other methods. In addition, it is proven that CMfinder could achieve better results than CM model [3] with

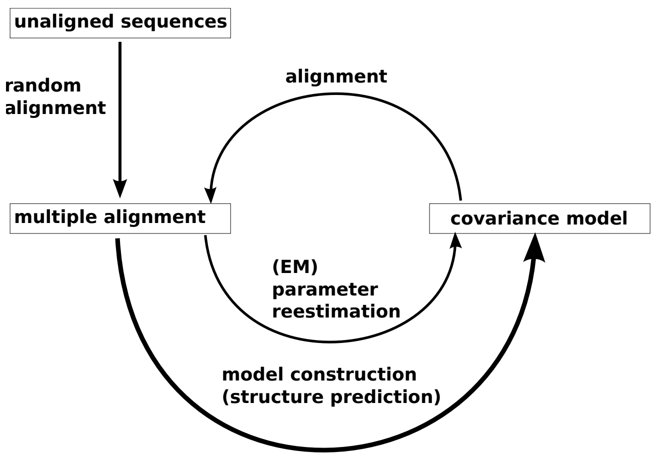


Figure 4.3: A flowchart summarize CMfinder process. Here the process goes in two steps: Firstly, CMfinder uses the covariance model(CM) in order to model a RNA sequence. Secondly, it uses the expectation maximization to search among the motif space in order to maximize the similarity

respect to the base pairs prediction with average of 77%. However, the combination of CMfinder with other available computational tools is used for the discovery of novel non-coding RNA family. For instance, Footprinter [76], which is a comparative genomics tool. An earlier version of it Blanchette was relying on multiple alignment [77], which is not the case with its combination with CMfinder. Since, with CMfinder it can start with no alignment at all. However, with the combination of Footprinter and CMfinder, Footprinter uses the homologous input sequences resulting from CMfinder and identifies the regulatory elements. CMfinder is also combined with RNA genome search tool [70] in order to speed up the homologous search in Rfam database. Another use of CMfinder is in finding novel ncRNAs as in [78].

In general, CMfinder produces a highly accurate model of conserved large *cis*-regluatory RNA motifs in unaligned sequences. It is robust and computationally fast comparing with other techniques. This model can expand and refine the discovered motifs iteratively. Therefore, our evaluation data generated by Z.Weinberg and his group has been alignment using CMfinder [54]. Since, it can produce very high homology model. Plus, it can infer a secondary structure which make it easier for the homologies search. (more details in chapter 7).

ID	Family	Rfam ID	#seqs	%id	length	#hp	CMfinder	CW/Pfold	CW/RNAalifold	Carnac	Foldalign	ComRNA
1	Cobalamin	RF00174	71	49	216	4	0.59	0.05	0.00	X	-	0.00
2	ctRNA_pGA1	RF00236	17	74	83	2	0.91	0.70	0.72	0.00	0.86	0.00
3	Entero_CRE	RF00048	56	81	61	1	0.89	0.74	0.22	0.00	-	0.00
4	Entero_OriR	RF00041	35	77	73	2	0.94	0.75	0.76	0.80	0.52	0.52
5	glmS	RF00234	14	58	188	4	0.83	0.12	0.18	0.00	-	0.13
6	Histone3	RF00032	63	77	26	1	1.00	0.00	0.00	0.00	-	0.00
7	Intron_gpII	RF00029	75	55	92	2	0.80	0.30	0.00	0.00	-	0.00
8	IRE	RF00037	30	68	30	1	0.77	0.22	0.00	0.00	0.38	0.00
9	let-7	RF00027	9	69	84	1	0.87	0.08	0.42	0.00	0.71	0.78
10	lin-4	RF00052	9	69	72	1	0.78	0.51	0.75	0.41	0.65	0.24
11	Lysine	RF00168	48	48	183	4	0.77	0.24	0.00	X	-	0.00
12	mir-10	RF00104	11	66	75	1	0.66	0.59	0.60	0.00	0.48	0.33
13	Purine	RF00167	29	55	103	2	0.91	0.07	0.00	0.00	-	0.27
14	RFN	RF00050	47	66	139	4	0.39	0.68	0.26	0.00	-	0.00
15	Rhino_CRE	RF00220	12	71	86	1	0.88	0.52	0.52	0.69	0.41	0.61
16	s2m	RF00164	23	80	43	1	0.67	0.80	0.45	0.64	0.63	0.29
17	S_box	RF00162	64	66	112	3	0.72	0.11	0.00	0.00	-	0.00
18	SECIS	RF00031	43	43	68	1	0.73	0.00	0.00	0.00	-	0.00
19	Tymo_tRNA-like	RF00233	22	72	86	4	0.81	0.33	0.36	0.30	0.80	0.48
			Average accuracy:				0.79	0.36	0.28	0.17	0.60	0.19
			Average specificity:				0.81	0.42	0.57	0.83	0.60	0.65
			Average sensitivity:				0.77	0.36	0.23	0.13	0.61	0.17

Table 4.1: CMfinder performance comparison with other related tools.

Chapter 5

Machine Learning

In this chapter an introduction to machine learning and the machine learning techniques used in this work. Graph kernel is the essential is the core concept used in this thesis. In the first section, section 1, a small introduction to the machine learning field is given. Followed by the definition of kernel in section 2 . The graph kernel is introduced in details in the third section. Section 3 present the fast neighborhood sub-graph pairwise distance kernel (FNSPDK) as a type of graph kernel, as it is the graph kernel tool used for the evaluation.

5.1 Support Vector Machine

Machine learning is a computational method for learning. It includes supervised, unsupervised, as well as the semi-supervised learning. In supervised learning the algorithm leans from giving example, which is not the case for the unsupervised learning. The semi-supervised algorithm is the case in between the two previous cases.
Support vector machine (SVM) are one of the most powerful and effective methods in machine learning. As SVM can be used in classifications and regression. As a consequence of having kernel, the use of the support vector machine is possible with non-linear classification problems. The next section define kernel and kernel trick.

5.2 Kernel

Recently many kernel based algorithms have been developed in the machine learning area. At first, they were used as a solution to binary classification problems. However, because of the increasing number of the complex data, the need for methods solving those problems in the cheapest, easiest way is needed. In general, kernel based algorithms are a linear methods using the inner products known as kernel function as a transformation from the original non-linear space into a linear space that allows the use of a linear classification algorithm [51] Figure5.2 illustrates this concept. We define kernel as a continuous and symmetric function (κ), that maps two arguments χ, χ' to a real value that measures their similarity.

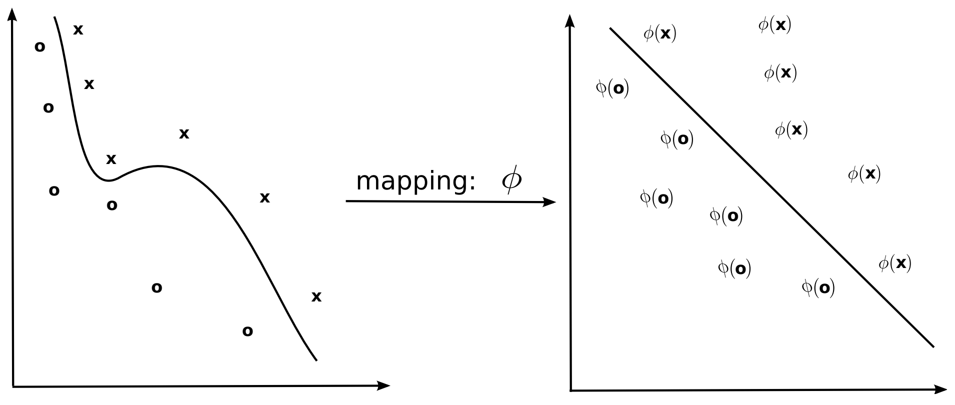


Figure 5.1: Kernel function illustration. Kernel function maps the original data space into feature space, where the linear classifier can work.

5.3 Graph kernel

When we want to measure the similarity between high structural data using kernels, then we need to present these data as a graph. Conveniently, graphs can represent a lot of complex data such as biological sequence data and RNA structures. When dealing with graphs we deal with discrete structure therefore we need to use the convolution kernel introduced by Haussler [79]. Note that all notations and definitions in this section is taken from Costa [56] and Haussler [79].

Before we give the definition of convolution kernel we need to define the kernel.

Definition 5.1 (Kernel). *Let X be a set. A kernel is a function $K : X \times X \to \mathbb{R}$, if K is:*

- *symmetric: $K(x, y) = K(y, x)$ for all $x, y \in X$.*
- *positive-semidefinite: for the matrix $K_{ij} = K(x_i, x_j)$, $n > 1$ and $x_1, ..., x_n \in X$ the sum $\sum\limits_{ij} c_i c_j K_{ij} \geq 0$ holds for all $c_1, ..., c_n \in \mathbb{R}$.*

Kernel method maps the input space into a new space called feature space where the kernel function can be applied.

Definition 5.2 (Feature space). *For a given kernel K over $X \times X$ that can be represented as function $K(x, y) =< \theta(x), \theta(y) >$ for any choice of θ , and X is countable set, then the vector space induced by θ is called feature space.*

For better understanding of the convolution kernel definition, we first give the definition of the relation notation we are using.

Definition 5.3 (Relation $R_{r,d}$). *$R_{r,d}$ is a relation on $\mathcal{G} \times \mathcal{G} \times \mathcal{G}$, where:*

- *$\mathcal{G}$ is the set of all graphs.*
- *$R_{r,d}(A^u, B^v, G)$ is true for some graphs $A^u, B^v, G \in \mathcal{G}$ iff $A^u, B^v \in \{\mathcal{N}_r^v : v \in V(G)\}$.*
- *A^u, B^v are isomorphic to some $\mathcal{N}_r$.*
- *$D(u, v) = d$ in G.*

From all previous definitions we could define the convolution kernel.

Definition 5.4 (Convolution Kernel). *A convolution kernen K is a valid kernel such that:*

$$K(x, y) = \sum_{\substack{x_1,...,x_d \in R^{-1}(x) \\ y_1,...,y_d \in R^{-1}(y)}} \prod K_d(x_d, y_d)$$

where:

- *$x_1, ..., x_d, y_1, ..., y_d \in X$ are parts of $x, y \in X$ such that $x = (x_1, ..., x_d), y = (y_1, ..., y_d)$. We call x, y a composite dividable object.*
- *R is a relation on the set $X_1 \times ... \times X_d \times X$ such that $R(x_1, ..., x_d, x)$ is true iff $x_1, ..., x_d$ are the parts of x.*

5.4 The Neighborhood Subgraph Pairwise Distance Kernel

In this section we define the neighborhood sub-graph pairwise distance kernel (NSPDK) and discuss how dose it works. At the end we show how it helps us in the work of this thesis. All notations and definitions in this section is taken from Costa [56].

The neighborhood sub-graph pairwise distance kernel is an instance of decomposition graph kernel. Following, the definition of the NSPDK is given and then clarify the definition with many definitions.

Definition 5.5 (**Neighborhood Subgraph Pairwise Distance Kernel**). *The neighborhood sub-graph pairwise distance kernel (NSPDK) is defined as:*
$K(G, G') = \sum_r \sum_d \kappa_{r,d}(G, G')$
Where $\kappa_{r,d}$ over $G \times G$ is a convolution kernel on the relation $R_{r,d}$ defined as:

$$\kappa_{r,d}(G, G') = \sum_{\substack{A^v, B^u \in R^{-1}_{r,d}(G) \\ A^{u'}, A^{v'} \in R^{-1}_{r,d}(G')}} \delta(A^v, A^{u'})\delta(B^u, B^{v'})$$

where

$$\delta(x, y) = \begin{cases} 1 & \textit{if } x \simeq y \textit{ (x,y are isomorphic graphs)} \\ 0 & \textit{otherwise} \end{cases}$$

The function $\kappa_{r,d}$ counts how many identical pairs of neighboring graphs of radius r at distance d between two graphs.

Definition 5.6 (**Neighborhood Subgraph**). *The neighborhood $\mathcal{N}_r(v)$ of a vertex $v \in V(G)$ and radius r is the set of vertices at maximal distance r from v:*

$$\mathcal{N}_r(v) = \{u \in V(G) \mid D(u, v) \leq r\}$$

The neighborhood subgraph N_r^v of vertex v and radius r is the graph induced by the neighborhood $\mathcal{N}_r(v)$.

5.4.1 EDeN

`EDeN` is the machine learning tool created by F. Costa [56]. It is used for the analisis of the different graph structures generated by the method of this thesis. `EDeN` works based on the NSPDK concept introduced above. The tool works in two steps:

1. It decomposes the graph into all its subgraphs by defining the relation R defined in 5.3.

2. It counts the number of identical pairs of neighboring subgraphs of radius r and distance d between two graphs.

The last step is done by first using the decomposition kernel on the relation $R_{r,d}$ as defined in 5.5. Then, the counting of the identical subgraph pairs is done by summing up all similar graph on the specified r, d.

Not to forget, that r is the upper bound of the radius and d is the upper bond of the distance defined by the user. Figure 5.2 shows how `EDeN` applies the previously explained steps on a graph.

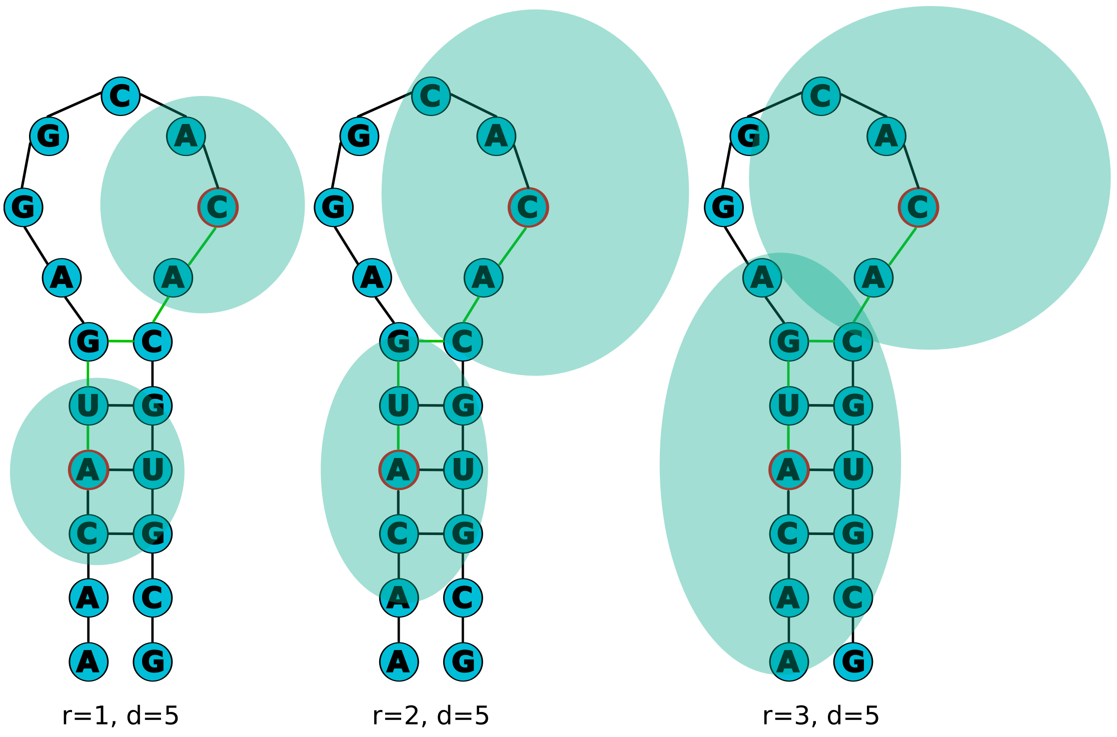


Figure 5.2: An illustration of pairs of neighborhood graphs. The subgraphs are generated for radius r= 1,2,3 and distance d=5. The red circle indicates the root of the two subgraphs, the green area, are considered in the comparison `EDeN` perform.

Chapter 6

Multiple Alignment Graph Generator

This chapter provides the core of the approach of this thesis. The aim of this work is to generate graphs that can explain ncRNA alignments while also considering their structure. These graphs are then used as a seed for a machine learning tool for prediction. In the first section a general description of the approach is provided with explanation of the different types of graphs. The second section introduces a tool called `MAGG` developed by this work, that generats these graphs.

6.1 A novel graph approach

The approach is focusing on building a graph out of ncRNA alignments. The input for this approach are aligned ncRNAs sequences. These alignments were created by CMfinder, which is a multiple sequence alignment method [57]. Each alignment data has multiple information attached to it. These information encode more details about the alignment A.2, for example, the secondary structure prediction of the alignment, the conservation of the aligned column, the strength of the conservation, and the covariation of the alignment base pair. Each information mentioned is used to build different graphs. More details on these information is presented in Appendix A.2.

This work is considering two graph classes $\mathscr{N}$ and $\mathscr{S}$. Figure 6.1 illustrates these classes of graph.. Each containing differently structured graphs. The class $\mathscr{N}$ contains graphs without stem information, while the $\mathscr{S}$ class does include them. Each of these classes are divided in into two subclasses $\mathscr{U}$ and $\mathscr{L}$. $\mathscr{U}$ contains graphs that encode the alignment information in single node, while $\mathscr{L}$ contains those graphs using several nodes to encode

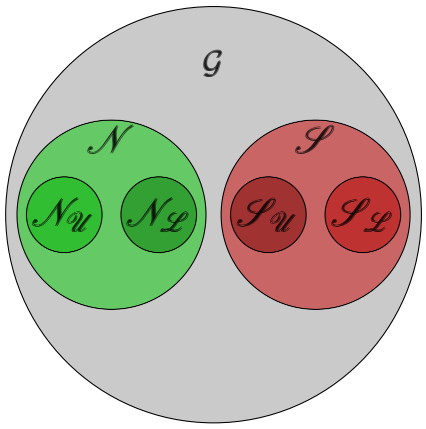

$\mathcal{G}$
$\mathscr{N}$
$\mathscr{S}$
$\mathscr{N}_{\mathscr{U}}$
$\mathscr{N}_{\mathscr{L}}$
$\mathscr{S}_{\mathscr{U}}$
$\mathscr{S}_{\mathscr{L}}$

Figure 6.1: The diagram shows that there are two different classes $\mathscr{N}$ and $\mathscr{S}$. Each of them are again subdivided into $\mathscr{N}_{\mathscr{U}}$,$\mathscr{N}_{\mathscr{L}}$, $\mathscr{S}_{\mathscr{U}}$, and $\mathscr{S}_{\mathscr{L}}$.

the information. The two graph classes can have a single or a concatenation of multiple alignment information as node labeling.

As an example for graphs in $\mathscr{N}$, consider Figure 6.2. The left side of the figure shows an instances of $\mathscr{N}_{\mathscr{U}}$ and the right side an instance of $\mathscr{N}_{\mathscr{L}}$. The graph $\mathscr{N}_{\mathscr{L}}$ differs from $\mathscr{N}_{\mathscr{U}}$ in away that it encodes extra information in individual node (green nodes).

In the second graph class $\mathscr{S}$, new nodes providing additional information about the sequence structure are added. These additional nodes represent stem and non-stem regions. A Stem is a continuous region of base pairs in the ncRNA sequence. Any other secondary structure as loops or hairpins are considered as non-stamp regions. This way, all nodes that encode a stem are connected to an extra node labeled by the letter **S**, where nodes that do not form a stem are connected to a node labeled by **NS**. These nodes are attached to other nodes indicating some information about the stem or non-stem regions. Similarly to the first graph class $\mathscr{N}$, this class contains two subclasses $\mathscr{S}_{\mathscr{U}}$ and $\mathscr{S}_{\mathscr{L}}$. Figure 6.3 illustrates an instance of $\mathscr{S}_{\mathscr{U}}$ on the left side and an instance of $\mathscr{S}_{\mathscr{L}}$ on the right side.

6.2 Processing pipeline

Several steps are considered to process the alignment information in our approach. Two tools are used . Both of them are organized in a tool chain and process a set of files. Figure 6.4 depicts these tools and the chain used in this work. The first tool in the toolchain is called `MAGG`, which is presented in this thesis. As input `MAGG` takes a set of files encoding

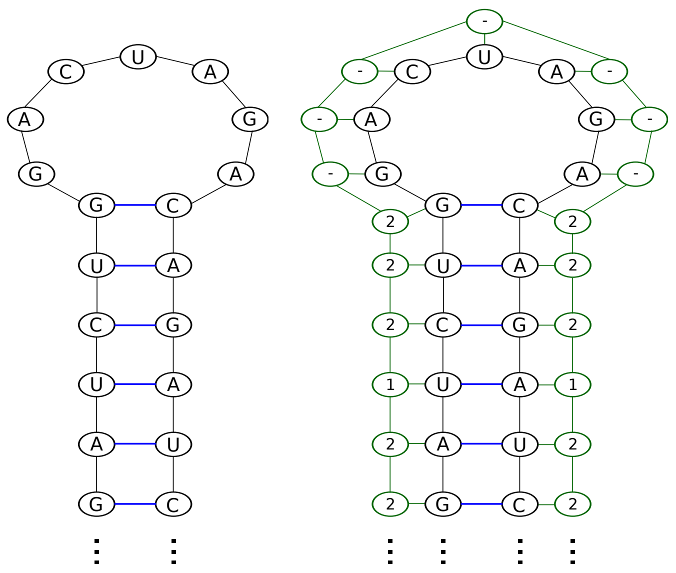


Figure 6.2: The left side shows an instance of a graph from $\mathscr{N}_{\mathscr{U}}$ encoding conservation information in single nodes. The right side is an instance of a graph from $\mathscr{N}_{\mathscr{L}}$ encoding covariation information in extra nodes (green colored).

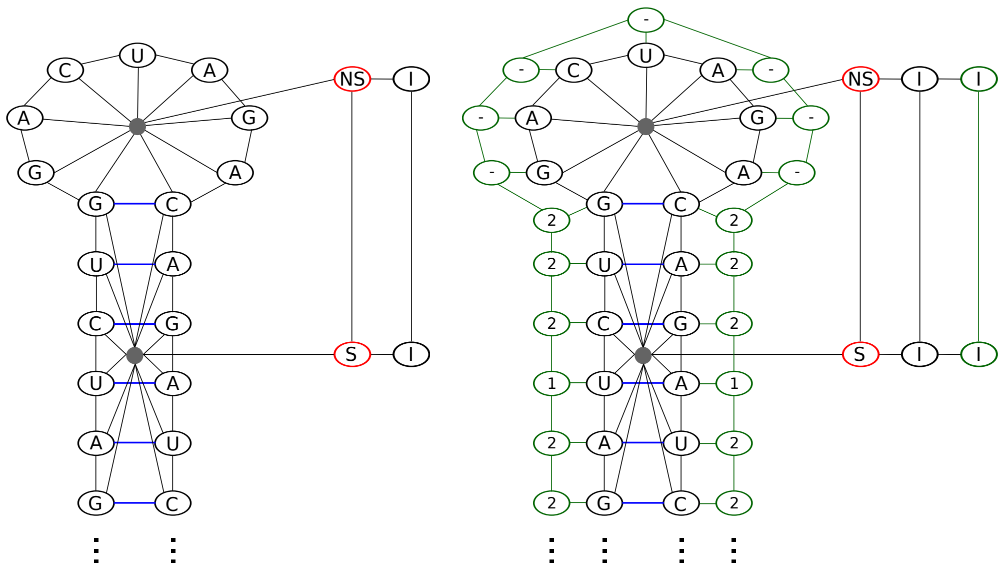


Figure 6.3: The left side shows an instance of a graph from $\mathscr{S}_{\mathscr{U}}$ encoding conservation information in single nodes. The right side is an instance of a graph from $\mathscr{S}_{\mathscr{L}}$ encoding covariation information in extra nodes (green colored). In both graphs, the stem regions are connected to a node S and the non-stem regions to a node NS.

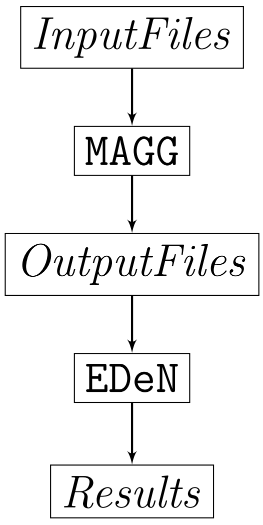


Figure 6.4: The two tools, `MAGG` and `EDeN`, organized in a toolchain. `MAGG` processes a set of files (upper level) encoding ncRNA sequence alingments and generates a set of files encoding our graph structure. `EDeN` processes these files and outputs its results.

ncRNA sequence alignments and generates a set of files each encoding graphs that we described earlier (see Section **??**). Next, these files, which called class files, passed on to `EDeN`. `EDeN` is a machine learning tool that we use for analyzing our graph structure. The next sections provide some insight on these tools and describe how they work.

6.2.1 MAGG

Multiple Alignments Graph Generator referred to it as `MAGG` is the tool developed by this work. This tool reads a set of files encoding ncRNA sequence alignments to generate a set of files each encoding a graph structure (see Section **??**). The set of input files is divided into two subsets. For the structure of these files refer to Appendix A.2. These data will be described in the following Chapter 7. The files returned by `MAGG` are then processed by `EDeN`, that provides analysis results.

The developed tool can be customized to generate differently structured graphs based on a given specification. This specification is called *pattern*. A pattern is a template for a layer, i.e. a subgraph, of the generated graph. That is, the generated graph can be described by a set of subgraphs all satisfying the same pattern. An example will be provided in order to clarify this point. In more detail, a pattern consists of a set of node and edge templates describing how these nodes are connected to each other. A node template is labeled by a composition of labels. The set of available labels is $\{cons, conss, sscons, cov, entropy\}$, were each element represents alignment information, and all sequences have the same length. `MAGG` processes this pattern by generating nodes connected according to the specified node and edge templates, where each node is labeled by the i-th character of an alignment.

Definition 6.1 (**Pattern**). *A pattern is a structure* (S, E, I)*, where* $S \subset 2^N$ *is a subset of nodes* N*,* $E \in N \times N$ *is a set of edges connecting nodes and* $I \in N \times N \cup \{\emptyset\}$ *is a set of inter layer edge templates. An inter edge template (called inter edge) is a template for an edge between two nodes of different layers.*

Each node is labeled by concatenating the i-th character of each element of a subset L' *of labels* $L = \{cons, conss, sscons, cov, entropy\}$*. Formally, for* i *and* L' *the labeling is* $info_0[i] \cdot info_1[i] \cdot \ \ldots \cdot info_n[i]$*, where* $info_0[i] \in L'$*,* $info_j[i]$ *represents the i-th character of* $info_j$ *and* $\cdot$ *the string concatenation operation.*

As an example consider the following pattern p. For simplicity we put the labeling of a node after an equal sign.

$$\begin{aligned}
p &:= (S, E, I) \\
S &:= N_0 = cons, cov,\ N_1 = cov,\ N_2 = ent \\
E &:= (N_0, N_1),\ (N_1, N_2) \\
I &:= (N_0, N_0),\ (N_1, N_2)
\end{aligned}$$

When a node is labeled, then it follows the labeling described in the definition above (Definition 6.1). Hence, a node is labeled by concatenating the i-th characters of each element of a subset of labels. The example pattern defines three nodes N_0, N_1 and N_2. The first node, N_0, is labeled by *cons* and *cov*, i.e. by $cons[i]$ and $cov[i]$. The second node, N_1, is labeled only by *cov* and the last one, N_2, is labeled by *ent*. These nodes are connected as follows: N_0 is connected to N_1 and N_1 to N_2. The `MAGG` tool processes this pattern by generating sub-graphs, or as called layers, which are isomorphic to the graph structure just described. The first layer is generated from the 0-th character of each labeling information, then the second is generated from the 1-th character, and so on. In total the generated graph consists of n layers, where n is the number of a sequence information. Note that all sequence information have the same length. An illustration of these layers is shown in Figure 6.2.1.

Furthermore, these layers are connected by so called inter edges. An inter edge connects two nodes of adjacent layers. In our example N_0 from the currently generated layer is

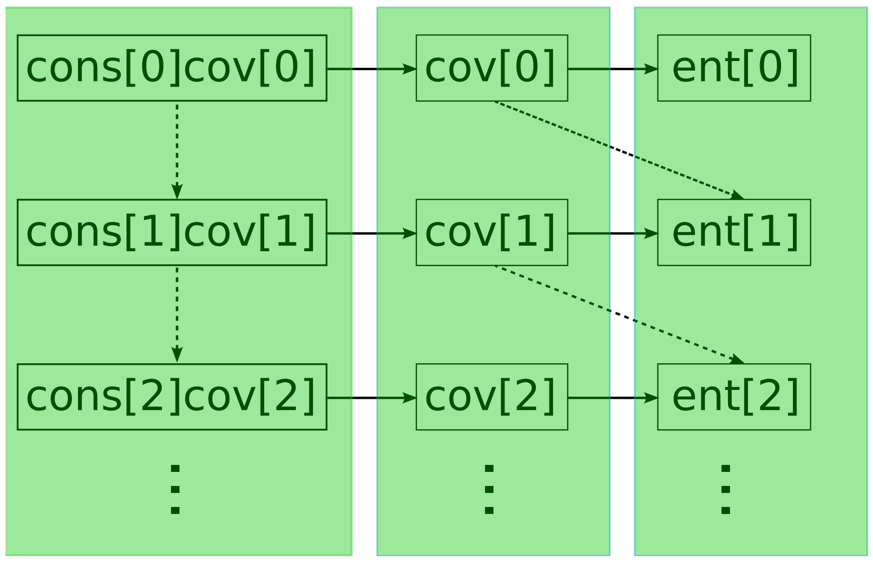


Figure 6.5: This graph is according to pattern p. Solid edges represent instances of edge templates and dashed edges represent instances of inter edge templates. We depict string concatenation without the string concatenation operator $\cdot$, as in $cons[1]cov[1]$ instead of $cons[1] \cdot cov[1]$.

connect to N_0 of the next generated layer. Similarly, N_1 is connected to N_2 of the next generated layer. We depict inter edges in Figure 6.2.1 by dashed edges. The next section covers how such graphs are processed by `EDeN`.

With the help of the described patterns, many types of different graphs can be generate encoding different types of alignment information. The most interesting patterns will be evaluate in Chapter 7.

Chapter 7

Evaluation

This chapter summarizes the evaluation results of the approach in this work. The first section, Section **??**, describes the input data for the evaluation. In section 2, the ROC measurement is defined, since it is the measurement unit used in this work. In the last section, Section 7.3, the results of this method are presented and discussed.

7.1 Data description

The experimental data was assembled by Zasha Weinberg and his group [53–55]. They have generated a pipeline using different methods to generate the alignment and predict the secondary structure but the functionality classification was done manually. Figure 7.1 is a flowchart showing the different methods that were considered by Z. Weinberg in the data assembly. In the coming paragraphs we are going to show every step in details.

Firstly, The Conserved Domain Database (CDD) [80], which is a a database for identifying conserved domains in new sequences, in order to identify the homologous gene sets. For each gene, only collected the 5ùpstream sequences are assembled, since the *cis*-regulatory elements are mostly conserved in this area. At this point 2,946 CDD groups are present in the dataset. In the second step, the Footprinter tool [76] is applied in order to select the sets that host non-coding RNAs. Footprinter is a tool used to discover regulatory elements in a set of homologous sequences, by identifying the conserved motifs in those regions. Thirdly, CMfinder [57] is used. This infer the RNA motifs in each unaligned sequence. This leads to a dataset of 35,975 motifs out from the initial CDD groups. Then, the redundant and poor motifs are removed in the postprocessing step in order to identify the rest of motifs based on the different RNA elements. With this step they reduce the number of unrelated motifs to 1,740 motifs, which helps in the prediction. RaveNnA is used in the fourth step to find more motif instances by scanning the prokaryotic genome database. A refinement is applied on the newly discovered motif members. This refinement and the motif postprocessing steps are repeated in order to get the best result. After this step, they are left with 1,466 motifs grouped in 1,0660 classes. Finally, they have performed gene context analysis and report the top ranking motifs.

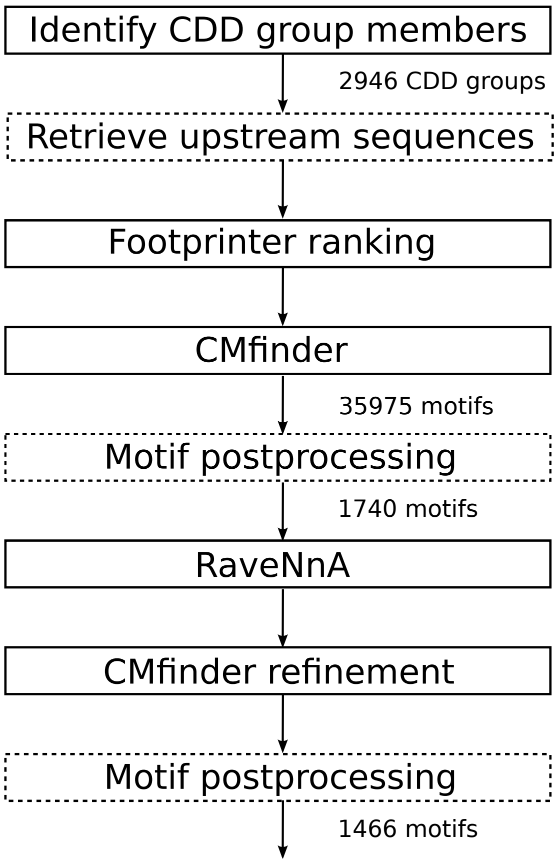


Figure 7.1: It shows the different steps the data goes through and also illustrates the methods used in generating them. The soild boxes indicate intensive computation (approx. run time) specified next to the box. After each step the number of remaining motifs is shown.

As a result of the pipeline 7.1 an identification of promising motifs by searching RNAs is done. These motifs have regions with conserved nucleotide sequences and have evidence of secondary structure. Testing on bacteria and archaea, they have identified 104 candidate structured RNAs, 12 of them were metabolite-binding RNA [54].

Wienberg and his team classified the resulting motifs into two different classes: negative and positive. The negative class contains motifs that have permuted alignments, where they have computed multiple sequence alignment by CLUSTALW of the 100 sequence datasets that have the highest motif scores. Then, they randomly permuted the alignment 50 times preserving the gap pattern. After that, they degap the permuted alignment and apply CMfinder and keep the top ranking motif from each dataset. The positive class contains known non-coding RNAs recognized in Rfam families.

7.2 Receiver Operator Characteristic: Evaluation metrics

The Receiver Operator Characteristic (ROC) curve is used in machine learning in binary decision problems. This curve is mostly used to measure the performance of a learning algorithm on dataset that has positive and negative instances [81]. Formally, for a set of instances I, each instance i is mapped to one class P,N, where P is the positive class and N is the negative class. The *classification model* is a mapping from instances to the predicted class. Let Y and N be the *prediction classes* produced by a model. Given an instance and classifier (classification model) [4], we can see that there are four possible outcomes. If the classifier predicted that this instance belongs to the positive class and it is actually positive, then it known as *true positive* **TP**. However, if the instance is positive but classified as negative then it is *false negative* **FN**. If the instance is negative and classified as negative the it is counted as *true negative* **TN**. Finally, if the instance is negative but classified as positive then it is *false positive* **FP**. Figure 7.2 shows these four cases as it known as confusion matrix.

The true positive rate also known as *recall* is estimated as :

$$tprate = \frac{Positivescorrectlyclassified(TP)}{totalpositives(P)} \tag{7.1}$$

The false positive rate known as *false alarm*

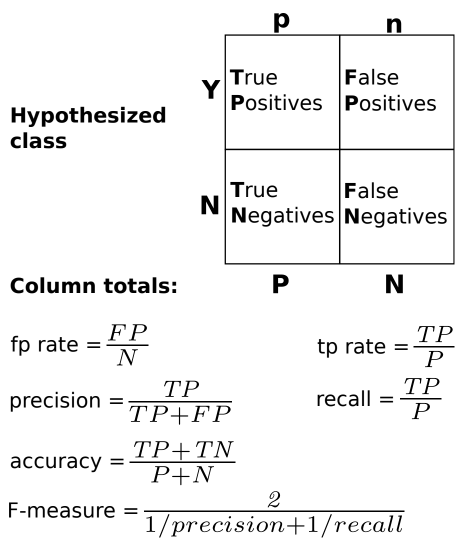


Figure 7.2: The upper part of the figure shows the four different cases of the prediction, where **p** and **n** are the true class and **Y** and **N** are the predicted class. The lower part of the figure gives some measurement calculations.

$$fprate = \frac{negativesincorrectlyclassified(FN)}{totalnegatives(N)} \tag{7.2}$$

The ROC curve is created by plotting the true positive rate (*tp* rate) against the false positive rate (*fp* rate) at a threshold. Figure 7.2 shows the ROC curve drown for five randomly classified instances. In this work, we use the ROC curve as a measurement unit of the classification method used in this work.

7.3 Results

In this section the results of the approach of this work is represented and discussed. The evaluation is executed on a cluster that consists of several computers with different configurations. The pipeline described in chapter 5 is executed on this cluster. The input is the data described in the previous Section 7.1. The pipeline works as follows:

- Different graphs are generate from these positive and negative classes according to a pattern using `MAGG`..
- `EDeN` is applied on the output of `MAGG` delivering the final results.

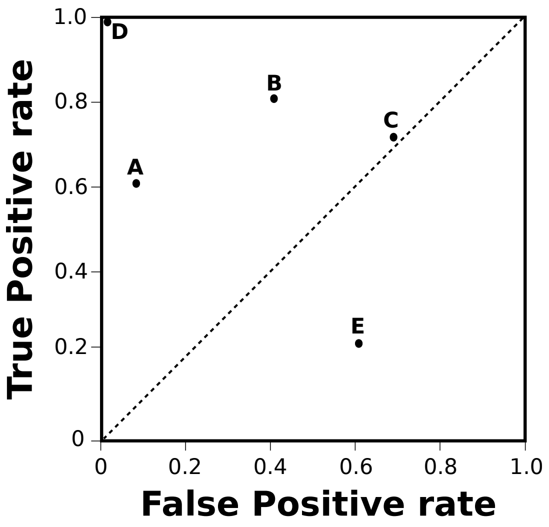


Figure 7.3: A basic ROC graph showing five discrete classifiers.

The EDeN parameters configuration that is used in this work are radius $r = \{0, \ldots, 4\}$ and distances $d = \{0, \ldots, 20\}$. These ranges have been chosen because it allows EDeN to cover all created graphs.

Because of the run time considering all radiuses and distances is very time consuming, only some combinations of radius and distance are considered. after a pre-evaluation, the most promising combinations are chosen.This section covered these combinations and in Appendix A.3 all tested different parameters combinations are provided.
Different patterns are considered creating different types of graphs. The set of graph classes representation used in this work are $\{\mathscr{N}, \mathscr{S}\}$, where $\mathscr{N}$ is the set of graphs without the stem nodes. $\mathscr{S}$ is the set of graphs that recognize the stem regions. Each of them have the representation of $\{\mathscr{U}, \mathscr{L}\}$, as described in Chapter6. The results of this evaluation are based on running EDeN on radius $r = \{2, 4\} \times d = \{12\}$, because these parameters provide the best results. The average and the standard deviation of the ROC measurement is considered . This way, the rate of the true positive (correctly classified) and the false positive (wrongly classified) instances is calculated.

In the following, the results of running EDeN with the selected parameters, radius $r = \{2, 4\}$ and distance $d = 12$, is discussed. When EDeN is running with $r = 2$, then it is creating two subgraphs every two nodes. Each of these subgraphs contain nodes that are in maximum distance 12 from the root node. Then, the comparison is measuring the similarity between these subgraphs according to the method introduced in Chapter 5.

For better presentation, a short way of writing the pattern is used. For example, when the tested pattern is $N_0 = x_0, N_1 = x_1, \ldots, N_n = x_n$, then it stands for:

$$\{N_0 = x_0, N_1 = x_1, \ldots, N_n = x_n\}, \{(N_0, N_1), (N_1, N_2), \ldots, (N_{n-1}, N_n)\}, \{\langle N_0, N_0\rangle\}.$$

The tables are sorted with respect to the highest `ROC` value first. The following two tables, Table 7.1 and Table 7.2, depict the results of running two different types of graphs created by `MAGG` from $\mathscr{N}_{\mathscr{U}}, \mathscr{N}_{\mathscr{L}}$, respectively.

	r = 2, d = 12		**r = 4, d = 12**	
Pattern	**AVG$_0$ ± (Std)**	**AVG$_1$ ± (Std)**	**AVG$_0$ ± (Std)**	**AVG$_1$ ± (Std)**
`N1=cons`	0 .719 ± 0.202	0.871 ± 0.112	0.745 ± 0.190	0.958 ± 0.040
`N1=cov`	.544 ± .248	.538 ± .248	.551 ± .247	.539 ± .248
`N1=cons cov`	.537 ± .248	.510 ± .249	.513 ± .249	.495 ± .249
`N1=cov con`	.535 ± .248	.555 ± .246	.570 ± .245	.573 ± .244
`N1=sscons cons`	.530 ± .249	.581 ± .243	.560 ± .246	.599 ± .240
`N1=sscons cov`	.522 ± .249	.559 ± .246	.562 ± .246	.515 ± .249

Table 7.1: This table shows the final result of the first class of the generated graphs $\mathscr{N}_{\mathscr{U}}$ running on radius $r = 2, 4$ and distance $d = 12$. The first column shows the running pattern, i.e. which information considered when building the graph. The second and the third columns give the result of average and (+/−) standard deviation. of testing on the two positive classes.

	r = 2, d = 12		**r = 4, d = 12**	
Pattern	**AVG$_0$ ± (Std)**	**AVG$_1$ ± (Std)**	**AVG$_0$ ± (Std)**	**AVG$_1$ ± (Std)**
`N1=cons N2=sscons`	.669 ± .221	.706 ± .207	.621 ± .235	.653 ± .226
`N1=cov N2=cons`	.658 ± .224	.676 ± .218	.639 ± .230	.629 ± .233
`N1=cons N2=cov`	.647 ± .228	.690 ± .213	.630 ± .233	.661 ± .223
`N1=sscons N2=cons`	.628 ± .233	.646 ± .228	.631 ± .233	.626 ± .234
`N1=cons N2=cov N3=conss N4=ent`	.603 ± .239	.651 ± .226	.574 ± .244	.607 ± .238
`N1=cons N2=cov N3=conss`	.596 ± .240	.661 ± .223	.609 ± .238	.637 ± .230
`N1=conss N2=cons`	.470 ± .249	.527 ± .249	.553 ± .247	.584 ± .242

Table 7.2: In this table we show the result of the first class of the generated graphs $\mathscr{N}_{\mathscr{L}}$running on $r = 2, 4, d = 12$. The table follows the notation introduced in the previous table 7.1.

Based on Table 7.1 one can observe that, aside of any type of the alignment information, the graph built using conservation of the alignment gives the best `ROC` results of approximately 80%. The rest of information types are close to the norm, i.e. they do not provide enough information to do the classification. When one takes the higher radius

$r = 4$ into account, one can see a major improvement in the results. That is, the `ROC` value of the graph encoding the conservation is improved from 80% to 90% with a small standard deviation of 0.040.
However, comparing Table 7.1 with Table 7.2, one can see a significant improvement in the latter table. Considering the same patterns but encoding the information in individual nodes helps the machine learning tool to do the classification better. For instance, considering the pattern, `N1=cov N2=cons` that encodes the covariance as a first node and the conservation as a second node, results in a `ROC` value of 65% (Table 7.2) in comparison to 53% (Table 7.1). Note that, the pattern resulting in a `ROC` value of 53% encodes the same type of information in as single node instead of two nodes. This hints that dividing the information is more useful in the classification. As one can notice comparing the two different radiuses $r = 2$ and $r = 4$, when the graph encodes different types of information, does not result in a notable improvement. On the contrary, the smaller radius provides better results most of the time.

In the following paragraphs the results from $\mathscr{S}_{\mathscr{U}}$ ans $\mathscr{S}_{\mathscr{L}}$ are discussed. As a reminder, the class $\mathscr{S}$ encodes stem and non-stem regions. This is done by adding a new node describing these regions as discussed in Chapter 6. The following tables, Table 7.3 and Table 7.4, are showing the results of running the same patterns considered above.

	r = 2, d = 12		**r = 4, d = 12**	
Pattern	$\mathbf{AVG_0} \pm$ **(Std)**	$\mathbf{AVG_1} \pm$ **(Std)**	$\mathbf{AVG_0} \pm$ **(Std)**	$\mathbf{AVG_1} \pm$ **(Std)**
`N1=cons -s`	.721 ± .200	.856 ± .122	.752 ± .186	.971 ± .028
`N1=cov -s`	.605 ± .238	.623 ± .234	.583± .243	.597 ± .240
`N1=cons cov -s`	.594 ± .241	.531 ± .248	.553 ± .247	.503 ± .249
`N1=sscons cons -s`	.557 ± .246	.588 ± .242	.577 ± .244	.581 ± .243
`N1=cov cons -s`	.554 ± .246	.599 ± .240	.570 ± .245	.611 ± .237
`N1=cons conss -s`	.551 ± .247	.484 ± .249	.573 ± .245	.571 ± .244
`N1=conss cov -s`	.525 ± .249	.505 ± .249	.582 ± .243	.521 ± .249

Table 7.3: In this table we show the result of the second class of the generated graphs $\{\mathscr{S}_{\mathscr{U}}\}$ running on $r = \{2, 4\}, d = 12$. The table follow the notations introduced in Table 7.1. The $-s$ after the pattern indicates that stem and non-stem regions are recognized.

When the stem regions are added into the graph structure a significant improvement is observed. Naturally, this is because more information is encoding in the graph. Comparing the results of Table 7.1 with those of Table 7.3 a remarkable improvement in the results can be seen. The results of patterns from graphs from$\mathscr{N}_{\mathscr{U}}$ are mostly less than the mean, which is not the case with patterns for graphs from the$\mathscr{S}_{\mathscr{U}}$. For instance, the

	r = 2, d = 12		**r = 4, d = 12**	
Pattern	$\mathbf{AVG_0} \pm$ **(Std)**	$\mathbf{AVG_1} \pm$ **(Std)**	$\mathbf{AVG_0} \pm$ **(Std)**	$\mathbf{AVG_1} \pm$ **(Std)**
`N1=cons N2=cov -s`	$.672 \pm .220$	$.690 \pm .213$	$.654 \pm .226$	$.658 \pm .224$
`N1=cov N2=cons -s`	$.695 \pm .211$	$.669 \pm .221$	$.673 \pm .220$	$.631 \pm .232$
`N1=cons N2=sscons -s`	$.681 \pm .216$	$.701 \pm .209$	$.646 \pm .228$	$.655 \pm .225$
`N1=cons N2=cov N3=conss N4=ent -s`	$.632 \pm .232$	$.627 \pm .233$	$.598 \pm .240$	$.583 \pm .243$
`N1=cons N2=cov N3=conss -s`	$.631 \pm .232$	$.640 \pm .230$	$.620 \pm .235$	$.608 \pm .238$
`N1=conss N2=cons -s`	$.505 \pm .249$	$.567 \pm .245$	$.602 \pm .239$	$.616 \pm .236$

Table 7.4: In this table we show the result of the second class of the generated graphs $\{\mathscr{S}_{\mathscr{L}}\}$ running on $r = \{2, 4\}, d = 12$. The table follow the notations introduced in Table 7.1. The $-s$ after the pattern indicates that stem and non-stem regions are recognized. Therefore, nodes labeled by $(\mathbf{S,NS})$ are generated.

pattern , +`N1=cov`, that encods the covariation gives a `ROC` value of 54% which increased to be $\approx 60\%$ when taking the stem regions into consideration. The same applies when the graph represents multiple types of information. As an example, in Table 7.2 the pattern +`N1=conss N2=cons`, results in a `ROC` value of 47%, which increased to 50%. In general, the graphs from $\mathscr{S}$ is returning better results. This is because, they encode more information about the sequence alignment structure. This way `EDeN` can be more accurate in classification.

Graphs from $\mathscr{S}_{\mathscr{L}}$ including stem regions gives `ROC` value of approximately 70%. The results do not differ much considering radius $r = 2$ and $r = 4$. In some cases the prediction increases from 50% to 60% as the results of the last pattern in Table 7.4. That shows that sometimes the larger radius is better, specially when the graph encodes information in separate nodes. Analogously, $\mathscr{N}_{\mathscr{L}}$ gives better results than the $\mathscr{N}_{\mathscr{U}}$.

		r = 2, d = 12		**r = 4, d = 12**	
Graph class	**Pattern**	$\mathbf{AVG_0} \pm$ **(Std)**	$\mathbf{AVG_1} \pm$ **(Std)**	$\mathbf{AVG_0} \pm$ **(Std)**	$\mathbf{AVG_1} \pm$ **(Std)**
$\mathscr{S}_{\mathscr{U}}$	`N1=cons -s`	$.721 \pm .200$	$.856 \pm .122$	$.752 \pm .186$	$.971 \pm .028$
$\mathscr{N}_{\mathscr{U}}$	`N1=cons`	$.719 \pm .202$	$.871 \pm .112$	$.745 \pm .190$	$.958 \pm .040$
$\mathscr{S}_{\mathscr{L}}$	`N1=cov N2=cons -s`	$.695 \pm .211$	$.669 \pm .221$	$.673 \pm .220$	$.631 \pm .232$
$\mathscr{N}_{\mathscr{L}}$	`N1=cons N2=sscons`	$.669 \pm .221$	$.706 \pm .207$	$.621 \pm .235$	$.653 \pm .226$

Table 7.5: This table shows the best results of each graph class generated by `MAGG` and evaluated by `EDeN` The table follow the notations introduced in Table 7.1. The $-s$ after the pattern indicates that stem and non-stem regions are recognized.

Table 7.5summarizes the best results of Table 7.1- 7.4. It can be observed, the conservation information supports `EDeN` in classification. In general, $\mathscr{S}$ graph class result in higher classification results than those of $\mathscr{N}$. This is again because more information about the alignment structure is provided which.

Chapter 8

Conclusion

Over the last decade, the importance of ncRNAs has become increasingly evident. Therefore, having a tool that can apply machine learning in order to predict the functionality of such RNAs is useful. The main goal of this thesis was to create a computational method that predicts whether a set of aligned ncRNAs is functional or not. These alignments contain conservation and structure information. The method developed in the presented work is called `MAGG`. It is used in combination with a tool based on a graph kernel approach (`EDeN`). `EDeN` is used to characterize graphs. The function of `MAGG` is the construction of different types of graphs. It can generate four graph types: two of them are $\mathscr{N}$ and two are $\mathscr{S}$. Every generated graph can encode different ncRNA alignment information. These aligned ncRNA sequences were generated by the CMfinder method. `EDeN` is the machine learning frame work that is used in this work.

The proposed method returned `ROC` values of 86% with the $\mathscr{S}_{\mathscr{U}}$ graph, 85% with the $\mathscr{N}_{\mathscr{U}}$ graph, 68% with the $\mathscr{S}_{\mathscr{L}}$ graph, and 69% with the $\mathscr{N}_{\mathscr{L}}$ graph 7.5. The first two `ROC` values are returned using the conservation information as node encoding.The third `ROC` value is returned when using the covariance and the conservation as node encoding. The fourth `ROC` value is returned when using the conservation and the secondary structure prediction as node encoding. These results indicate that the most correct classifications can be obtained by using the $\mathscr{S}_{\mathscr{U}}$ and $\mathscr{N}_{\mathscr{U}}$ graphs encoding the conservation information. Notably, the results returned for the $\mathscr{U}$ graph do not markedly improve by including the stem regions. However, given that the function of ncRNAs is often governed by their structure, it appears sensible to assume that encoding additional structural information beyond simple stems (hairpins, bulges, loops) might improve the final classification. At this stage the classification approach can definitely be used as a powerful pre-filtering method when a large amount of alignments is provided. Of course, further work is needed to develop and optimize this method.

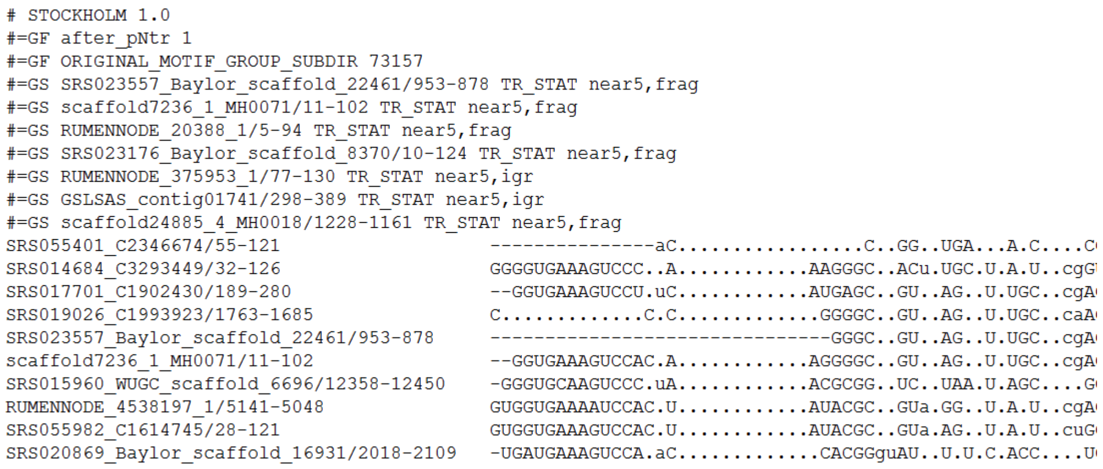
Currently, `MAGG` starts from aligned ncRNA sequences and then builds graphs based on them. To simplify the application, automated alignment of the input sequences could be implemented. Furthermore, the application of the current approach on other types of sequences may be interesting.

Appendix A

Appendix

A.1 Stockholm format

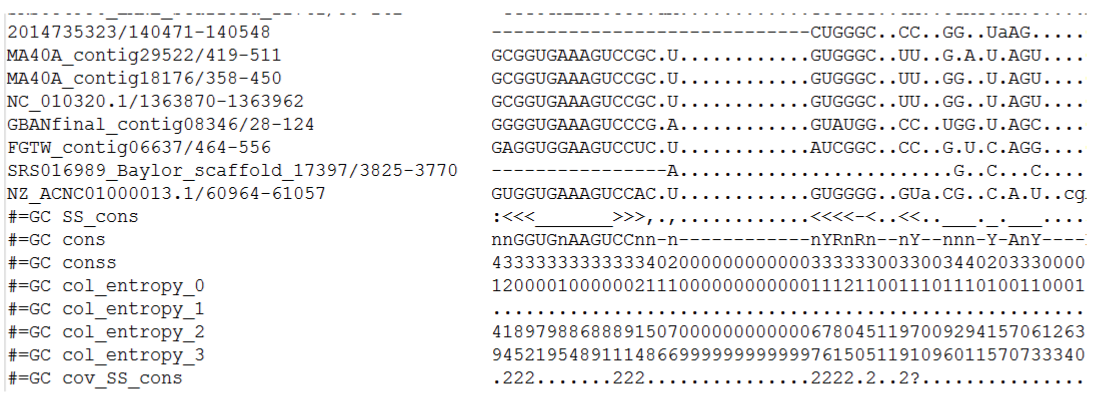
The data files we are using in our experiment are in stockholm format. This format is a multiple sequence alignment format used by Rfam and Pfam for representing RNA sequence alignments. Each file in stockholm format starts with a header that indicates the stockholm identifiers followed by some command and then the sequence alignments, Figure A.1 shows how stockholm file looks like.



```
# STOCKHOLM 1.0
#=GF after_pNtr 1
#=GF ORIGINAL_MOTIF_GROUP_SUBDIR 73157
#=GS SRS023557_Baylor_scaffold_22461/953-878 TR_STAT near5,frag
#=GS scaffold7236_1_MH0071/11-102 TR_STAT near5,frag
#=GS RUMENNODE_20388_1/5-94 TR_STAT near5,frag
#=GS SRS023176_Baylor_scaffold_8370/10-124 TR_STAT near5,frag
#=GS RUMENNODE_375953_1/77-130 TR_STAT near5,igr
#=GS GSLSAS_contig01741/298-389 TR_STAT near5,igr
#=GS scaffold24885_4_MH0018/1228-1161 TR_STAT near5,frag
SRS055401_C2346674/55-121                   ---------------aC.................C..GG..UGA...A.C....C
SRS014684_C3293449/32-126                   GGGGUGAAAGUCCC..A............AAGGGC..ACu.UGC.U.A.U..cgG
SRS017701_C1902430/189-280                  --GGUGAAAGUCCU.uC............AUGAGC..GU..AG..U.UGC..cgA
SRS019026_C1993923/1763-1685                C.............C.C.............GGGGC..GU..AG..U.UGC..caA
SRS023557_Baylor_scaffold_22461/953-878     -------------------------------GGGC..GU..AG..U.UGC..cgA
scaffold7236_1_MH0071/11-102                --GGUGAAAGUCCAC.A............AGGGGC..GU..AG..U.UGC..cgA
SRS015960_WUGC_scaffold_6696/12358-12450    -GGGUGCAAGUCCC.uA............ACGCGG..UC..UAA.U.AGC....G
RUMENNODE_4538197_1/5141-5048               GUGGUGAAAAUCCAC.U............AUACGC..GUa.GG..U.A.U..cgA
SRS055982_C1614745/28-121                   GUGGUGAAAGUCCAC.U............AUACGC..GUa.AG..U.A.U..cuG
SRS020869_Baylor_scaffold_16931/2018-2109   -UGAUGAAAGUCCA.aC.............CACGGguAU..U.U.C.ACC....U
```

Figure A.1: Stockholm alignment file format.

These files are the input into `MAGG`. From all the information encoded in the file we use the conservation of the alignment column, the strength of the column conservation, as well as the entropy information to build our graph. Figure A.2 shows the desired lined in the stockholm file. Our graph model depends on the type of information we wants to build it based on. For each information we create a vertex and then connect these vertices based on the relation between them. The output graph is undirected labeled graph, for definition see chapter 3.



```
2014735323/140471-140548                      -----------------------------CUGGGC..CC..GG..UaAG.....
MA40A_contig29522/419-511                     GCGGUGAAAGUCCGC.U............GUGGGC..UU..G.A.U.AGU....
MA40A_contig18176/358-450                     GCGGUGAAAGUCCGC.U............GUGGGC..UU..GG..U.AGU....
NC_010320.1/1363870-1363962                   GCGGUGAAAGUCCGC.U............GUGGGC..UU..GG..U.AGU....
GBANfinal_contig08346/28-124                  GGGGUGAAAGUCCCG.A............GUAUGG..CC..UGG.U.AGC....
FGTW_contig06637/464-556                      GAGGUGGAAGUCCUC.U............AUCGGC..CC..G.U.C.AGG....
SRS016989_Baylor_scaffold_17397/3825-3770     ----------------A.......................G..C...C....
NZ_ACNC01000013.1/60964-61057                 GUGGUGAAAGUCCAC.U............GUGGGG..GUa.CG..C.A.U..cg
#=GC SS_cons                                  :<<<_______>>>,.,............<<<<-<..<<..___._.___....
#=GC cons                                     nnGGUGnAAGUCCnn-n------------nYRnRn--nY--nnn-Y-AnY----
#=GC conss                                    43333333333333402000000000000333333003300344020333000 0
#=GC col_entropy_0                            1200001000000211100000000000011121100111011101001100 01
#=GC col_entropy_1                            ......................................................
#=GC col_entropy_2                            4189798868889150700000000000067804511970092941570612 63
#=GC col_entropy_3                            9452195489111486699999999999976150511910960115707333 40
#=GC cov_SS_cons                              .222.......222...............2222.2..2?...............
```

Figure A.2: Information needed from the input file.

A.2 Nodes label information

Firstly, the *sscons* indicates the alignment secondary structure prediction. This type of information indicates the base pair relationship by using brackets like <> ()[], and dots indicates unpaired nucleated.

Secondly, the *cons* shows conservation of the alignment column. The conservation information shows the nucleotides that conserved its position during evolution. It has different values as follow:

1. A nucleotide (A,C,G,U) is conserved (this is probably the most interesting case),
2. The column is conserved as a purine/pyrimadine (R=A or G ; or Y=C or U),
3. 'n' indicates that the presence of a nucleotide is conserved but not a specific one.
4. A dash '-' indicates that it's usually a gap.

Thirdly, *conss* indicates the strength of conservation. It's a number whose meaning depends on the symbol in the *cons* line. There is different cases:

- If the *cons* line is a nucleotide or purine/pyrimidine (A,C,G,U,R or Y), then the *conss* number is between $1-3$ where:
  - 1 is ≥ 97 % conserved.
  - 2 is ≥ 90 %.
  - 3 is ≥ 75 %.
- If the *cons* line has 'n', then the *conss* number is between $1-4$ where:
  - 1 means there's a nucleotide there $\geq 97\%$ of the time.
  - 2 means ≥ 90 %.
  - 3 means ≥ 75 %.
  - 4 means ≥ 50 %.
- If the cons symbol is a gap '-', then the *conss* number is always zero.

Fourthly, *cov* indicates the covariation of each base pair. The covariation of the base pair shows how randomly these base pairs are changing together. Here is the measurement for the covariation:

- 2 means there is covariation.
- 1 means only compatible mutations (e.g. A-U and G-U).
- 0 means no changes in the type of base pair.
- ? means that there are too many non-canonical base pairs.

Fifth, *ent* indicates the column entropy. The entropy of a column is the profitability of occurring the nucleotide in the column.

A.3 Result of different parameters

Here is a representation of one of the pattern considered in the result reported in Chapter 7. The following table shows the different raduises and distances tested running `EDeN`. From these tested parameters came the choose of the parameters chosen for running the evaluation. The pattern considered is $N1 = cons - e - s$. For better presentation, the resuting table is divided in to tables.

Class P	**Class N**	**Radius**	**Distance**	**PRE**	**REC**	**PRF**	**ROC**
0	0	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.564 ± 0.001
0	0	4	20	0.774 ± 0.004	0.376 ± 0.000	0.505 ± 0.000	0.704 ± 0.000
0	0	2	10	0.828 ± 0.007	0.115 ± 0.000	0.202 ± 0.000	0.701 ± 0.000
0	0	2	12	0.831 ± 0.008	0.146 ± 0.000	0.248 ± 0.000	0.705 ± 0.000
0	0	2	16	0.829 ± 0.004	0.200 ± 0.000	0.321 ± 0.000	0.692 ± 0.001
0	1	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.561 ± 0.001
0	1	4	20	0.801 ± 0.009	0.376 ± 0.000	0.510 ± 0.000	0.733 ± 0.007
0	1	2	10	0.839 ± 0.009	0.115 ± 0.000	0.202 ± 0.000	0.715 ± 0.002
0	1	2	12	0.849 ± 0.009	0.146 ± 0.000	0.248 ± 0.000	0.718 ± 0.002
0	1	2	16	0.849 ± 0.006	0.200 ± 0.000	0.323 ± 0.000	0.704 ± 0.002
0	2	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.558 ± 0.000
0	2	4	20	0.792 ± 0.009	0.376 ± 0.000	0.509 ± 0.000	0.730 ± 0.007
0	2	2	10	0.828 ± 0.007	0.115 ± 0.000	0.202 ± 0.000	0.710 ± 0.002
0	2	2	12	0.831 ± 0.008	0.146 ± 0.000	0.248 ± 0.000	0.714 ± 0.002
0	2	2	16	0.833 ± 0.005	0.200 ± 0.000	0.322 ± 0.000	0.700 ± 0.002
0	3	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.569 ± 0.000
0	3	4	20	0.791 ± 0.009	0.376 ± 0.000	0.508 ± 0.000	0.729 ± 0.007
0	3	2	10	0.834 ± 0.008	0.115 ± 0.000	0.202 ± 0.000	0.711 ± 0.002
0	3	2	12	0.845 ± 0.009	0.146 ± 0.000	0.248 ± 0.000	0.715 ± 0.002
0	3	2	16	0.840 ± 0.006	0.200 ± 0.000	0.322 ± 0.000	0.701 ± 0.002
0	4	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.564 ± 0.001
0	4	4	20	0.803 ± 0.009	0.376 ± 0.000	0.511 ± 0.000	0.734 ± 0.007
0	4	2	10	0.852 ± 0.007	0.115 ± 0.000	0.203 ± 0.000	0.714 ± 0.002
0	4	2	12	0.853 ± 0.008	0.146 ± 0.000	0.249 ± 0.000	0.719 ± 0.002
0	4	2	16	0.849 ± 0.006	0.200 ± 0.000	0.323 ± 0.000	0.705 ± 0.002
0	5	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.565 ± 0.001
0	5	4	20	0.797 ± 0.010	0.376 ± 0.000	0.510 ± 0.000	0.730 ± 0.007
0	5	2	10	0.848 ± 0.008	0.115 ± 0.000	0.202 ± 0.000	0.712 ± 0.002
0	5	2	12	0.849 ± 0.009	0.146 ± 0.000	0.248 ± 0.000	0.716 ± 0.002
0	5	2	16	0.846 ± 0.006	0.200 ± 0.000	0.323 ± 0.000	0.701 ± 0.002
0	6	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.566 ± 0.001
0	6	4	20	0.817 ± 0.005	0.376 ± 0.000	0.514 ± 0.000	0.741 ± 0.006
0	6	2	10	0.856 ± 0.005	0.115 ± 0.000	0.203 ± 0.000	0.720 ± 0.001
0	6	2	12	0.862 ± 0.006	0.146 ± 0.000	0.249 ± 0.000	0.725 ± 0.001
0	6	2	16	0.860 ± 0.003	0.200 ± 0.000	0.324 ± 0.000	0.713 ± 0.001
0	7	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.557 ± 0.000
0	7	4	20	0.800 ± 0.009	0.376 ± 0.000	0.510 ± 0.000	0.733 ± 0.007
0	7	2	10	0.834 ± 0.008	0.115 ± 0.000	0.202 ± 0.000	0.714 ± 0.002
0	7	2	12	0.841 ± 0.009	0.146 ± 0.000	0.248 ± 0.000	0.717 ± 0.002
0	7	2	16	0.843 ± 0.006	0.200 ± 0.000	0.322 ± 0.000	0.703 ± 0.002
0	8	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.566 ± 0.001
0	8	4	20	0.789 ± 0.009	0.376 ± 0.000	0.508 ± 0.000	0.733 ± 0.007
0	8	2	10	0.828 ± 0.007	0.115 ± 0.000	0.202 ± 0.000	0.716 ± 0.002
0	8	2	12	0.826 ± 0.006	0.146 ± 0.000	0.248 ± 0.000	0.719 ± 0.002
0	8	2	16	0.840 ± 0.006	0.200 ± 0.000	0.322 ± 0.000	0.705 ± 0.002
0	9	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.565 ± 0.001
0	9	4	20	0.798 ± 0.009	0.376 ± 0.000	0.510 ± 0.000	0.734 ± 0.007
0	9	2	10	0.839 ± 0.009	0.115 ± 0.000	0.202 ± 0.000	0.718 ± 0.001
0	9	2	12	0.845 ± 0.009	0.146 ± 0.000	0.248 ± 0.000	0.721 ± 0.002
0	9	2	16	0.837 ± 0.006	0.200 ± 0.000	0.322 ± 0.000	0.707 ± 0.002

Table A.1: Caption

Class P	Class N	Radius	Distance	PRE	REC	PRF	ROC
1	0	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.641 ± 0.001
1	0	4	20	0.883 ± 0.001	0.853 ± 0.000	0.868 ± 0.000	0.942 ± 0.000
1	0	2	10	0.921 ± 0.002	0.292 ± 0.000	0.443 ± 0.000	0.846 ± 0.000
1	0	2	12	0.918 ± 0.002	0.346 ± 0.000	0.502 ± 0.000	0.846 ± 0.000
1	0	2	16	0.911 ± 0.001	0.430 ± 0.000	0.584 ± 0.000	0.831 ± 0.000
1	1	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.639 ± 0.001
1	1	4	20	0.897 ± 0.003	0.853 ± 0.000	0.874 ± 0.000	0.948 ± 0.000
1	1	2	10	0.926 ± 0.002	0.292 ± 0.000	0.444 ± 0.000	0.854 ± 0.000
1	1	2	12	0.927 ± 0.002	0.346 ± 0.000	0.503 ± 0.000	0.854 ± 0.001
1	1	2	16	0.921 ± 0.001	0.430 ± 0.000	0.586 ± 0.000	0.839 ± 0.001
1	2	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.637 ± 0.001
1	2	4	20	0.893 ± 0.003	0.853 ± 0.000	0.872 ± 0.000	0.947 ± 0.000
1	2	2	10	0.921 ± 0.002	0.292 ± 0.000	0.443 ± 0.000	0.851 ± 0.000
1	2	2	12	0.918 ± 0.002	0.346 ± 0.000	0.502 ± 0.000	0.851 ± 0.000
1	2	2	16	0.913 ± 0.001	0.430 ± 0.000	0.585 ± 0.000	0.835 ± 0.001
1	3	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.647 ± 0.000
1	3	4	20	0.892 ± 0.003	0.853 ± 0.000	0.871 ± 0.000	0.947 ± 0.000
1	3	2	10	0.924 ± 0.002	0.292 ± 0.000	0.443 ± 0.000	0.852 ± 0.000
1	3	2	12	0.925 ± 0.002	0.346 ± 0.000	0.503 ± 0.000	0.852 ± 0.001
1	3	2	16	0.917 ± 0.002	0.430 ± 0.000	0.585 ± 0.000	0.836 ± 0.001
1	4	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.643 ± 0.001
1	4	4	20	0.899 ± 0.003	0.853 ± 0.000	0.875 ± 0.000	0.949 ± 0.000
1	4	2	10	0.933 ± 0.001	0.292 ± 0.000	0.445 ± 0.000	0.856 ± 0.000
1	4	2	12	0.929 ± 0.002	0.346 ± 0.000	0.504 ± 0.000	0.856 ± 0.000
1	4	2	16	0.921 ± 0.001	0.430 ± 0.000	0.586 ± 0.000	0.840 ± 0.001
1	5	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.644 ± 0.001
1	5	4	20	0.895 ± 0.003	0.853 ± 0.000	0.873 ± 0.000	0.948 ± 0.000
1	5	2	10	0.931 ± 0.002	0.292 ± 0.000	0.444 ± 0.000	0.853 ± 0.000
1	5	2	12	0.927 ± 0.002	0.346 ± 0.000	0.503 ± 0.000	0.853 ± 0.001
1	5	2	16	0.920 ± 0.002	0.430 ± 0.000	0.586 ± 0.000	0.837 ± 0.001
1	6	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.644 ± 0.001
1	6	4	20	0.908 ± 0.001	0.853 ± 0.000	0.879 ± 0.000	0.952 ± 0.000
1	6	2	10	0.936 ± 0.001	0.292 ± 0.000	0.445 ± 0.000	0.859 ± 0.000
1	6	2	12	0.934 ± 0.001	0.346 ± 0.000	0.505 ± 0.000	0.859 ± 0.000
1	6	2	16	0.929 ± 0.000	0.430 ± 0.000	0.588 ± 0.000	0.845 ± 0.000
1	7	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.634 ± 0.000
1	7	4	20	0.897 ± 0.003	0.853 ± 0.000	0.874 ± 0.000	0.948 ± 0.000
1	7	2	10	0.924 ± 0.002	0.292 ± 0.000	0.443 ± 0.000	0.854 ± 0.000
1	7	2	12	0.923 ± 0.002	0.346 ± 0.000	0.503 ± 0.000	0.854 ± 0.001
1	7	2	16	0.918 ± 0.002	0.430 ± 0.000	0.586 ± 0.000	0.838 ± 0.001
1	8	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.644 ± 0.001
1	8	4	20	0.891 ± 0.003	0.853 ± 0.000	0.871 ± 0.000	0.947 ± 0.000
1	8	2	10	0.921 ± 0.002	0.292 ± 0.000	0.443 ± 0.000	0.854 ± 0.000
1	8	2	12	0.916 ± 0.001	0.346 ± 0.000	0.502 ± 0.000	0.853 ± 0.001
1	8	2	16	0.917 ± 0.002	0.430 ± 0.000	0.585 ± 0.000	0.838 ± 0.001
1	9	0	0	0.000 ± 0.000	0.000 ± 0.000	0.000 ± 0.000	0.643 ± 0.001
1	9	4	20	0.896 ± 0.003	0.853 ± 0.000	0.873 ± 0.000	0.948 ± 0.000
1	9	2	10	0.926 ± 0.002	0.292 ± 0.000	0.444 ± 0.000	0.857 ± 0.000
1	9	2	12	0.925 ± 0.002	0.346 ± 0.000	0.503 ± 0.000	0.856 ± 0.000
1	9	2	16	0.915 ± 0.001	0.430 ± 0.000	0.585 ± 0.000	0.840 ± 0.001

Table A.2: Caption

Bibliography

[1] Difference DNA and RNA. `http://commons.wikimedia.org/wiki/File:Difference_DNA_RNA-EN.svg`, April 2015.

[2] Pedro Larrañaga, Borja Calvo, Roberto Santana, Concha Bielza, Josu Galdiano, Iñaki Inza, José Antonio Lozano, Rubén Armañanzas, Guzmán Santafé, Aritz Pérez Martínez, and Victor Robles. Machine learning in bioinformatics. *Briefings in Bioinformatics*, 7(1):86–112, 2006.

[3] Sean R. Eddy and Richard Durbin. RNA sequence analysis using covariance models. *Nucleic Acids Research*, 22(11):2079–2088, 1994.

[4] Tom Fawcett. ROC graphs: Notes and practical considerations for researchers. Technical report, HP Laboratories, MS 1143, 1501 Page Mill Road, Palo Alto, CA 94304, 2004.

[5] Ralf Dahm. Friedrich miescher and the discovery of DNA. *Developmental biology*, 278(2):274–288, 2005.

[6] Erwin Chargaff. Structure and function of nucleic acids as cell constituents. In *Federation proceedings*, volume 10(3), page 654, 1951.

[7] Watson JD and Crick FC. Molecular structure of nucleic acids: A structure for deoxyribose nucleic acid. *JAMA*, 269(15):1966–1967, 1993.

[8] James D. Watson and Francis H. C. Crick. Genetical implications of the structure of deoxyribonucleic acid. *JAMA*, 269(15):1967–1969, 1993.

[9] Francis H. C. Crick. Central dogma of molecular biology. *Nature*, 227(5258):561–563, August 1970.

[10] Harry F Noller, Vernita Hoffarth, and Ludwika Zimniak. Unusual resistance of peptidyl transferase to protein extraction procedures. *Science*, 256(5062):1416–1419, 1992.

[11] Francis H. C. Crick. The origin of the genetic code. *Journal of Molecular Biology*, 38(3):367 – 379, 1968.

[12] Robert W. Holley, Jean Apgar, George A. Everett, James T. Madison, Mark Marquisee, Susan H. Merrill, John Robert Penswick, and Ada Zamir. Structure of a ribonucleic acid. *Science*, 147(3664):1462–1465, 1965.

[13] Christo P. Christov, Timothy J. Gardiner, Dávid Szüts, and Torsten Krude. Functional requirement of noncoding Y RNAs for human chromosomal DNA replication. *Molecular and Cellular Biology*, 26(18):6993–7004, 2006.

[14] Mark N. Ziats and Owen M. Rennert. Aberrant expression of long noncoding RNAs in autistic brain. *Journal of Molecular Neuroscience*, 49(3):589–593, 2013.

[15] Isabel López de Silanes, Ming Zhan, Ashish Lal, XiaolingYang, and Myriam Gorospe. Identification of a target RNA motif for RNA-binding protein HuR. *Proceedings of the National Academy of Sciences of the United States of America*, 101(9):2987–2992, 2004.

[16] Valentina Boeva, Didier Surdez, Noëlle Guillon, Franck Tirode, Anthony P. Fejes Olivier Delattre, and Emmanuel Barillot. De novo motif identification improves the accuracy of predicting transcription factor binding sites in ChIP-Seq data analysis. *Nucleic Acids Research*, 38(11):e126, 2010.

[17] Philip Machanick and Timothy L. Bailey. MEME-ChIP: motif analysis of large DNA datasets. *Bioinformatics*, 27(12):1696–1697, 2011.

[18] Francis H. C. Crick. Ideas on protein synthesis. *Self Published*, pages 1–2, 1956.

[19] Lois N. Magner. *A history of the life sciences.* CRC Press, New York, 3 edition, 2002.

[20] Paulien Hogeweg. The roots of bioinformatics in theoretical biology. *PLoS Computational Biology*, 7(3), 2011.

[21] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, DTIC Document, 1961.

[22] Richard J. Carter, Inna Dubchak, and Stephen R. Holbrook. A computational approach to identify genes for functional RNAs in genomic sequences. *Nucleic Acids Research*, 29(19):3928–3938, 2001.

[23] Ian Korf. Gene finding in novel genomes. *BMC Bioinformatics*, 5:59, 2004.

[24] David R. Kelley, Bo Liu1 Arthur L. Delcher, Mihai Pop, and Steven L. Salzberg. Gene prediction with glimmer for metagenomic sequences augmented by classification and clustering. *Nucleic Acids Research*, 40(1), 2012.

[25] G. Schweikert, J. Behr, A. Zien, G. Zeller, CS. Ong, S. Sonnenburg, and G. Rätsch. mGene.web: a web service for accurate computational gene finding. *Nucleic Acids Research*, 37:W312–6, 2009.

[26] Saul B. Needleman and Christian D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, March 1970.

[27] T.F. Smith and M.S. Waterman. Identification of common molecular subsequences. *Journal of Molecular Biology*, 147(1):195 – 197, 1981.

[28] Peter H. Sellers. The theory and computation of evolutionary distances: Pattern recognition. *J. Algorithms*, 1(4):359–373, 1980.

[29] Stephen F. Altschul, Warren Gish, Webb Miller, Eugene W. Myers, and David J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, October 1990.

[30] David J. Lipman and William R. Pearson. Rapid and sensitive protein similarity searches. *Science*, 4693(227):1435–1441, 1985.

[31] Cédric Notredame, Desmond G. Higgins, and Jaap Heringa. T-Coffee: A novel method for fast and accurate multiple sequence alignment. *Journal of molecular biology*, 302(1):205–217, 2000.

[32] Supawan Prompramote, Yan Chen, and Yi-PingPhoebe Chen. Machine learning in bioinformatics. In Yi-PingPhoebe Chen, editor, *Bioinformatics Technologies*, pages 117–153. Springer Berlin Heidelberg, 2005.

[33] Iñaki Inza, Borja Calvo, Rubén Arma nanzas, Endika Bengoetxea, Pedro Larra naga, and José A. Lozano. Machine learning: An indispensable tool in bioinformatics. *Bioinformatics methods in clinical research*, 593, March 2009.

[34] Stein Aerts, Peter Van Loo, Yves Moreau, and Bart De Moor. A genetic algorithm for the detection of new cis-regulatory modules in sets of coregulated genes. *Bioinformatics*, 20(12):1974–1976, 2004.

[35] Joseph Bockhorst, Mark Craven, David Page, Jude W. Shavlik, and Jeremy D. Glasner. A bayesian network approach to operon prediction. *Bioinformatics*, 19(10):1227–1235, 2003.

[36] Julia Handl, Douglas B. Kell, and Joshua D. Knowles. Multiobjective optimization in bioinformatics and computational biology. *IEEE/ACM Trans. Comput. Biology Bioinform.*, 4(2):279–292, 2007.

[37] Sophia Ananiadou and John Mcnaught. *Text Mining for Biology And Biomedicine.* Artech House, Inc., Norwood, MA, USA, 2005.

[38] Martin Krallinger, Ramon Alonso-Allende Erhardt, and Alfonso Valencia. Text-mining approaches in molecular biology and biomedicine. *Drug discovery today*, 10(6):439–445, 2005.

[39] Michel Laurin. The subjective nature of linnaean categories and its impact in evolutionary biology and biodiversity studies. *Contributions to Zoology*, 79(4):131–146, 2010.

[40] Ernst Mayr and W. J. Bock. Classifications and other ordering systems. *Journal of Zoological Systematics and Evolutionary Research*, 40(4):169–194, 2002.

[41] George H. John, Ron Kohavi, and Karl Pfleger. Irrelevant features and the subset selection problem. In Cohen and Hirsh [88], pages 121–129.

[42] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, 1997.

[43] Ludmila Kuncheva. Genetic algorithm for feature selection for parallel classifiers. *Inf. Process. Lett.*, 46(4):163–168, 1993.

[44] Steven Salzberg. Locating protein coding regions in human DNA using a decision tree algorithm. *Journal of Computational Biology*, 2(3):473–485, 1995.

[45] James W. Fickett. Recognition of protein coding regions in DNA sequences. *Nucleic Acids Research*, 10(17):5303–5318, 1982.

[46] Farber Robert, Alan Lapedes, and Karl Sirotkin. Determination of eukaryotic protein coding regions using neural networks and information theory. *Journal of molecular biology*, 226(2):471–479, 1992.

[47] Gary B. Fogel, V. William Porto, Dana G. Weekes, David B. Fogel, Richard H. Griffey, John A. McNeil, Elena Lesnik, David J. Ecker, and Rangarajan Sampath. Discovery of RNA structural elements using evolutionary computation. *Nucleic Acids Research*, 30(23):5310–5317, 2002.

[48] Michael Zuker. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Research*, 31(13):3406–3415, 2003.

[49] Ting Chen, Ming-Yang Kao, Matthew Tepel, John Rush, and George M. Church. A dynamic programming approach to de novo peptide sequencing via tandem mass spectrometry. *Journal of Computational Biology*, 8(3):325–337, 2001.

[50] Bernhard Scholkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA, USA, 2001.

[51] F. Perez-Cruz and O. Bousquet. Kernel methods and their potential use in signal processing. *IEEE Signal Processing Magazine*, 21(3):57–65, May 2004.

[52] David S. Latchman. Transcription factors: an overview. *The international journal of biochemistry & cell biology*, 29(12):1305–1312, 1997.

[53] Zasha Weinberg, Jeffrey E. Barrick, Zizhen Yao, Adam Roth, Jane N. Kim, Jeremy Gore, Joy Xin Wang, Elaine R. Lee, Kirsten F. Block, Narasimhan Sudarsan, Shane Neph, Martin Tompa, Walter L. Ruzzo, and Ronald R. Breaker. Identification of 22 candidate structured RNAs in bacteria using the CMfinder comparative genomics pipeline. *Nucleic Acids Research*, 35:4809–4819, 2007.

[54] Zasha Weinberg, Joy Wang, Jarrod Bogue, Jingying Yang, Keith Corbino, Ryan Moy, and Ronald Breaker. Comparative genomics reveals 104 candidate structured RNAs from bacteria, archaea, and their metagenomes. *Genome Biology*, 11(3):R31, 2010.

[55] Zizhen Yao, Jeffrey Barrick, Zasha Weinberg, Shane Neph, Ronald R. Breaker, Martin Tompa, and Walter L. Ruzzo. A computational pipeline for high- throughput discovery of cis-regulatory noncoding RNA in prokaryotes. *PLoS Computational Biology*, 3(7), 2007.

[56] Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In Fürnkranz and Joachims [90], pages 255–262.

[57] Zizhen Yao, Zasha Weinberg, and Walter L. Ruzzo. CMfinder - a covariance model based RNA motif finding algorithm. *Bioinformatics*, 22(4):445–452, 2006.

[58] Bjarne Knudsen and Jotun Hein. RNA secondary structure prediction using stochastic context-free grammars and evolutionary history. *Bioinformatics*, 15(6):446–454, 1999.

[59] Ivo L. Hofacker. The vienna RNA secondary structure server. *Nucleic Acids Research*, 31(13):3429–3431, 2003.

[60] Alex Coventry, Daniel J. Kleitman, and Bonnie Berger. MSARI: multiple sequence alignments for statistical detection of RNA secondary structure. *Proceedings of the National Academy of Sciences of the United States of America*, 101(33):12102–12107, August 2004.

[61] Ivo L. Hofacker, Walter Fontana, Peter F. Stadler, Sebastian L. Bonhoeffer, Manfred Tacker, and Peter Schuster. Fast folding and comparison of RNA secondary structures. *Monatsh. Chem.*, 125:167–188, 1994.

[62] Robin D. Dowell and Sean R. Eddy. Evaluation of several lightweight stochastic context-free grammars for RNA secondary structure prediction. *BMC Bioinformatics*, 5:71, 2004.

[63] Timothy L. Bailey and Charles Elkan. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning*, 21(1-2):51–80, 1995.

[64] Timothy L Bailey and Charles Elkan. The value of prior knowledge in discovering motifs with MEME. In *Ismb*, volume 3, pages 21–29, 1995.

[65] Byung-Jun Yoon. Hidden markov models and their applications in biological sequence analysis. *Current Genomics*, 10(6):402–415, 2009.

[66] Antti Airola, Sampo Pyysalo, Jari Björne, Tapio Pahikkala, Filip Ginter, and Tapio Pahikkala. A graph kernel for protein-protein interaction extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing*, BioNLP '08, pages 1–9, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics.

[67] Karsten M. Borgwardt, Cheng Soon Ong, Stefan Schönauer, S. V. N. Vishwanathan, Alexander J. Smola, and Hans-Peter Kriegel. Protein function prediction via graph kernels. In *Proceedings Thirteenth International Conference on Intelligent Systems for Molecular Biology 2005, Detroit, MI, USA, 25-29 June 2005* [92], pages 47–56.

[68] Kousik Kundu, Fabrizio Costa, and Rolf Backofen. A graph kernel approach for alignment-free domain-peptide interaction prediction with an application to human SH3 domains. *Bioinformatics*, 29(13):335–343, 2013.

[69] David Hoksza and Daniel Svozil. Efficient rna pairwise structure comparison by setter method. *Bioinformatics*, 28(14):1858–1864, 2012.

[70] Zasha Weinberg and Walter L. Ruzzo. Faster genome annotation of non-coding RNA families without loss of accuracy. In Bourne and Gusfield [87], pages 243–251.

[71] Ivo L. Hofacker, Martin Fekete, and Peter F. Stadler. Secondary structure prediction for aligned RNA sequences. *Journal of Molecular Biology*, 319(5):1059–1066, 2002.

[72] Sam Griffiths-Jones, Alex Bateman, Mhairi Marshall, Ajay Khanna, and Sean R. Eddy. Rfam: an RNA family database. *Nucleic Acids Research*, 31(1):439–441, 2003.

[73] Michael Zuker and Patrick Stiegler. Optimal computer folding of large rna sequences using thermodynamics and auxiliary information. *Nucleic acids research*, 9(1):133–148, 1981.

[74] Timothy L. Bailey and Charles Elkan. The value of prior knowledge in discovering motifs with MEME. In Rawlings et al. [91], pages 21–29.

[75] Jakob Hull Havgaard, Rune B. Lyngsø, Gary D. Stormo, and Jan Gorodkin. Pairwise local structural alignment of RNA sequences with sequence similarity less than 40%. *Bioinformatics*, 21(9):1815–1824, 2005.

[76] Mathieu Blanchette and Martin Tompa. Footprinter: a program designed for phylogenetic footprinting. *Nucleic Acids Research*, 31(13):3840–3842, 2003.

[77] Mathieu Blanchette, Benno Schwikowski, and Martin Tompa. Algorithms for phylogenetic footprinting. *Journal of Computational Biology*, 9(2):211–223, 2002.

[78] Jan Gorodkin and Ivo L. Hofacker. From structure prediction to genomic screens for novel non-coding RNAs. *PLoS Computational Biology*, 7(8), 2011.

[79] David Haussler. Convolution kernels on discrete structures. Technical report, UC Santa Cruz, UCS-CRL-99-10, 1999.

[80] Aron Marchler-Bauer, John B. Anderson, Praveen F. Cherukuri, Carol DeWeese-Scott, Lewis Y. Geer, Marc Gwadz, Siqian He, David I. Hurwitz, John D. Jackson, Zhaoxi Ke, Christopher J. Lanczycki, Cynthia A. Liebert, Chunlei Liu, Fu Lu, Gabriele H. Marchler, Mikhail Mullokandov, Benjamin A. Shoemaker, Vahan Simonyan, James S. Song, Paul A. Thiessen, Roxanne A. Yamashita, Jodie J. Yin, Dachuan Zhang, and Stephen H. Bryant. CDD: a conserved domain database for protein classification. *Nucleic Acids Research*, 33(Database-Issue):192–196, 2005.

[81] Jesse Davis and Mark Goadrich. The relationship between precision-recall and ROC curves. In Cohen and Moore [89], pages 233–240.

[82] Felix Autenrieth, Barry Isralewitz, Zaida Luthey-schulten, Anurag Sethi, and Taras Pogorelov. Bioinformatics and sequence alignment. In *University of Illinois at Urbana-Champaign Luthey-Schulten Group Beckman Institute for Advanced Science and Technology Theoretical and Computational Biophysics Group San Francisco Workshop*, 2005.

[83] O. T. Avery, C. M. MacLeod, and M. McCarty. Studies on the chemical nature of the substance inducing transformation of pneumococcal types. induction of transformation by a desoxyribonucleic acid fraction isolated from pneumococcus type iii. *Molecular medicine (Cambridge, Mass.)*, 1(4):344–365, May 1994.

[84] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13:281–305, 2012.

[85] C. Mathé, M. F. Sagot, T. L. Bailey, and P. Rouzé. Current methods of gene prediction, their strengths and weaknesses. *Nucleic acids research*, 30(19):4103–4117, October 2002.

[86] Pavel Pudil, Jana Novovicová, and Josef Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(10):1119–1125, 1994.

[87] Philip E. Bourne and Dan Gusfield, editors. *Proceedings of the Eighth Annual International Conference on Computational Molecular Biology, 2004, San Diego, California, USA, March 27-31, 2004*. ACM, 2004.

[88] William W. Cohen and Haym Hirsh, editors. *Machine Learning, Proceedings of the Eleventh International Conference, Rutgers University, New Brunswick, NJ, USA, July 10-13, 1994*. Morgan Kaufmann, 1994.

[89] William W. Cohen and Andrew Moore, editors. *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*. ACM, 2006.

[90] Johannes Fürnkranz and Thorsten Joachims, editors. *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel.* Omnipress, 2010.

[91] Christopher J. Rawlings, Dominic A. Clark, Russ B. Altman, Lawrence Hunter, Thomas Lengauer, and Shoshana J. Wodak, editors. *Proceedings of the Third International Conference on Intelligent Systems for Molecular Biology, Cambridge, United Kingdom, July 16-19, 1995*. AAAI, 1995.

[92] *Proceedings Thirteenth International Conference on Intelligent Systems for Molecular Biology 2005, Detroit, MI, USA, 25-29 June 2005*, 2005.