REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : IVA06/M2/2022

Mémoire

Présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Image et Vie Artificielle (IVA)**

Classification of non-coding RNAs using Deep Learning

Par :

ABDELKRIM ACHOUAK

Soutenu le 26/06/2020 devant le jury composé de :

Zerari Abd El Mouméne	MCB	Président
Belounnar Saliha	MAA	Rapporteur
Naidji Ilyes	MAB	Examinateur

Année universitaire 2021-2022

الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة محمد خيضر بسكرة

تصريح شرفي

(خاص بالالتزام بقواعد النزاهة العلمية لإنجاز بحث)

أنا الممضي أسفله،

السيد (ة): أشواق عبد الكريم الصفة: **طالب**

الحامل لبطاقة التعريف الوطنية رقم:2070255853 والصادرة بتاريخ: 04/10/2021

المسجل بكلية: العلوم الدقيقة و علوم الطبيعة و الحياة

قسم: الإعـــــــــــــــــــــــــلام الآلـــــــي

والمكلف بإنجاز مذكرة تخرج في الماستر عنوانها:

Classification of non-coding RNAs using Deep Learning

أصرح بشرفي أني ألتزم بمراعاة المعايير العلمية والمنهجية ومعايير الأخلاقيات المهنية والنزاهة الأكاديمية المطلوبة في إنجاز البحث المذكور أعلاه.

التاريخ: 2022/06/22

توقيع المعني:

(وَمَا أُوتِيتُمْ مِنَ الْعِلْمِ إِلّا قَلِيلا)

[الإسراء - 85]

Dedications

To my parents, they always support me and believe in my weird ideas,
to my little stars, Jazia and Chaker,
to the lovely partner in this journey, my husband, and his family,
to all the Google communities, you were my second family,
to Hafsa and Islam.

Acknowledgments

First of all, Alhamdulillah for being here and living these moments,

Then, I would like to express my sincere thanks and appreciation to my supervisor Mrs. Belounnar Saliha for her trust and for following up on this work from the idea to the realization.

In addition, I thank all the staff and academic crew of the Computer Science department of Mohamed khider University of Biskra, for their infinite efforts,

Finally, I can't forget Houda and Wafa, they always make me proud of my little achievements, my rays of sunshine.

Abstract

Non-coding RNAs (ncRNAs), which function directly as RNAs without translating into proteins, play diverse and crucial roles in biological and medical processes, such as disease detection and drug development. Thus, here we present a comparative analysis between three projects that uses deep learning approaches for the classification of the ncRNAs, and then we try to implement the most performant model of them. Our model is based on a supervised learning concept with a convolutional neural networks (CNN) architecture, which classify our data into Thirteen (13) ncRNA families using sequences to extract features.

Keywords: Bioinformatics, ncRNA, CNN, Classification, deep learning.

Résumé

Les ARN non codants (ARNnc), qui fonctionnent directement comme des ARN sans se traduire en protéines, jouent des rôles divers et cruciaux dans les processus biologiques et médicaux, tels que la détection des maladies et le développement de médicaments. Ainsi, nous présentons ici une analyse comparative entre trois projets qui utilisent des approches d'apprentissage profond pour la classification des ARNnc, et nous essayons ensuite d'implémenter le modèle le plus performant entre eux. Notre modèle est basé sur un concept d'apprentissage supervisé avec une architecture de réseaux de neurones convolutifs (CNN), qui classifie nos données en treize (13) familles d'ARNnc en utilisant des séquences pour extraire les caractéristiques.

Mots-clés: Bioinformatique, Les ARNs non-codants, CNN, Classification, Apprentissage profond.

List of content

List of Figures

List of tables

General Introduction

With the development era of biological research, non-coding RNAs (ncRNAs) have received increasing attention in the fields of biology and medicine. They play important roles in biological processes such as transcription and translation. Classification of ncRNAs is critical for our understanding of disease mechanisms and therapeutic design. Many ncRNA classification methods have been developed, some of which use machine learning and deep learning as well. So, in this research, we will present a comparative study of existing similar methods and try to implement the most performant method between them.

This research is divided into four chapters, the first one is about a general overview of the molecular biology basics and an introduction to the bioinformatics field, next in the second chapter we covered the principal approaches of deep learning and we introduced the concept of non-coding RNAs classification and cited some of its methods, also we mentioned Three of the similar works to our project, and compare them with presenting a brief analyzes about them according to some metrics, then in the third chapter, we pass to the practical part, so we present the general architecture of our model and defined the basic used methods and approaches, finally, in the last chapter we cite the used development tools, then realize the results of the implementation and discuss them.

The main objective of this project is to construct an effective Deep learning-based model for the classification of non-coding RNAs according to their sequences.

Chapter 1 Molecular Biology and Bioinformatics

1. Introduction

Bioinformatics became today one of the trending fields in the world, and it is considered a combination between biology, computer science, and mathematics, it is interesting to study the molecular organisms of the cell by applying computer science tools and methods, and as we are from an informatics background we have to understand some of the biological bases and get familiar with bioinformatics aspects and terms, in order to go deeper in this research.

So this chapter is divided into two parts, the first one is about molecular biology and its main approaches (Nucleotides, Amino acids, DNA, RNA, Central Dogma), and the second part is a brief explanation of the bioinformatics discipline and some of its aspects (History and definition, Issues studies through bioinformatics, some of its application domains, its main limitations).

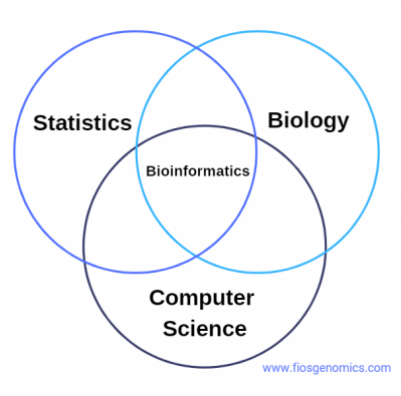


Figure 1.1 Bioinformatics field position

2.Molecular Biology

Molecular biology is the field of biology that studies the composition, structure, and interactions of cellular molecules, such as nucleic acids and proteins, that carry out the biological processes essential for the cell's functions and maintenance [1].

2.1. Molecules

When we review the biological hierarchy, we conclude that they are the elements that exist within the cell and participate in organic processes, these molecules may be big ones such as proteins, sugars, lipids, nucleic acids, or as small as amino acids, nucleotides, metabolites.

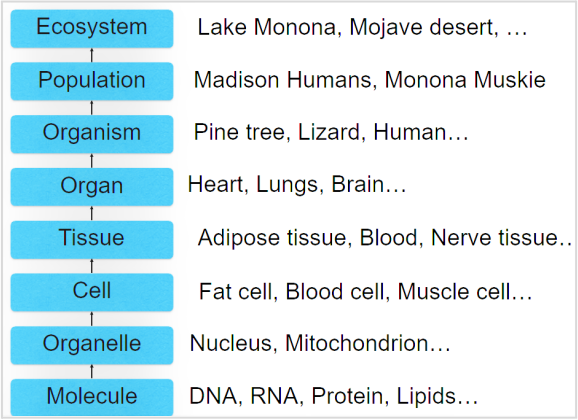


Figure 1.2 Levels of the Biological hierarchy

Molecular biology is generally concerned with all the different processes that affect molecules, whether they are building or demolition processes. It is also concerned with studying the structures and roles of nucleic acids, and these are either DNA or RNA.

2.2. Nucleotides

They are the basic building blocks of nucleic acids, or as they are known the building blocks of DNA and RNA.
In terms of structure, it contains 3 parts:

2.2.1. Nitrogen Base

Which is known as a heterocyclic compound and this means that they are heterogeneous rings or they are organic compounds containing a ring holding atoms of two or more different chemical elements.

There are two types of Nitrogenous bases:

- **First type:** Pyrimidine is a Hexagonal ring of Carbon-containing two Nitrogen atoms.

- **Second type:** Purine consists of two rings, hexagonal and pentagonal, and four Nitrogen atoms instead of two.

From these formulas we have five subtypes (Three for Pyrimidine and Two for Purine):

- Cytosine, Uracil, and Thymine for Pyrimidine, the main difference between them is in the side links.

- As for purine, it has two types, adenine, and guanine, Also the difference between them is in the side links.

In terms of presence, adenine, guanine, and cytosine are present in both DNA and RNA, while uracil is found only in RNA, and thymine is found only in DNA and some rare secondary types of RNA.

2.2.2. Phosphate Group

We find it in the nucleotides in one of the following forms:

- Mono-Phosphate (one Phosphate molecule) as AMP.
- Di-Phosphate (two Phosphate molecules) such as ADP.
- Or Tri-Phosphate (three Phosphate molecules) as ATP.

As we know in chemistry, the more phosphate molecules within a compound, the more energy that compound contains.
The form of the presence of nucleotides (Monophosphate, Diphosphate, or Triphosphate) varies according to their function in the body, for example, in the case of Polymerase Nucleotides, we find it in the form of a Monophosphate.

2.2.3. Sugar molecule

We find it in two forms:

- **Ribose:** It is a pentagonal ring containing four carbon atoms and the fifth atom attached to it, in addition to an oxygen atom. The numbering of these atoms is as follows: 1', 2', 3', 4', 5', and the reason for putting signs above the numbers of carbon atoms in the sugar ring is to distinguish them from the numbering of carbon atoms in the rings of nitrogen bases. And this is its chemical formula:

The other type of sugar found in nucleic acids is **Deoxyribose.**

Now that we have seen the molecules that make up the nucleotide, it remains for us to collect them together.

We first start by linking the sugar with the Nitrogenous base, producing a compound called nucleoside, then the names of the bases change to become as follows: Adenosine, Guanosine, Cytidine, Uridine, Thymidine.

Now, that we form the Nucleosides, we combine them with the Phosphates to produce the final compound, which is called a Nucleotide.

2.3. Amino acid

Proteins are considered a polymer of monomers called amino acids, which are the building blocks of the molecule. Only 20 amino acids are involved in protein synthesis, even though more than 300 amino acids have been identified [2]. Therefore, the sequence of amino acids in a specific order is what constitutes the structure of the protein, and it is what determines its specific function. We can define the chemical formula of an amino acid as follows:

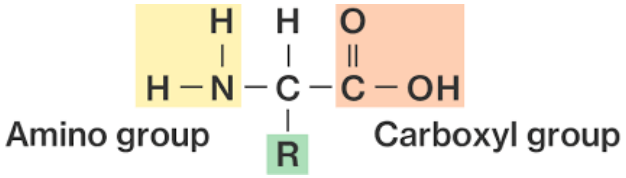


Figure 1.3 Amino Acids chemical structure

What gives us the different types of amino acids is the difference in the R side chain, which is often used in their classification.

2.4. DNA

DNA is the cell's library in which information is stored in its sequence of nucleotides[3]. DNA preserves the identity and characteristics of organisms and protects them from any relative change through generations, and we find it in both prokaryotes and eukaryotes. DNA is involved in two basic processes in the life of the cell:

The first one is regulating cellular expression, which is the transcription and translation of the genetic information carried by the DNA to eventually give directions for the synthesis of the protein. Hence, it controls each of the types, sequences, and quantities of protein produced by the cell, the DNA exercises control through the so-called central dogma.

The second process is the Transmission of genetic information: The DNA undergoes a process of self-replication, which allows it to make copies of itself when the cell regenerates and gives these copies to new cells so that they can inherit every characteristic of the parent cell.

DNA is considered a double-stranded component, it consists of two chains of nucleotides that are opposite, one of the chains starts from 3' towards 5' and the opposite starts from 5' towards 3', these two chains are linked through hydrogen bonds between nitrogenous bases, this linkage is not randomly, but according to the base-pairing rules, where Adenosine is always linked with Thymidine by two hydrogen bonds, while three hydrogen bonds link Guanosine and Cytosine, and the number of hydrogen bonds affects the strength of the bonding of the two chains to each other, the more the number of Guanosine and Cytosine bases the more the bond of the two chains is stronger and resistant to disintegration.

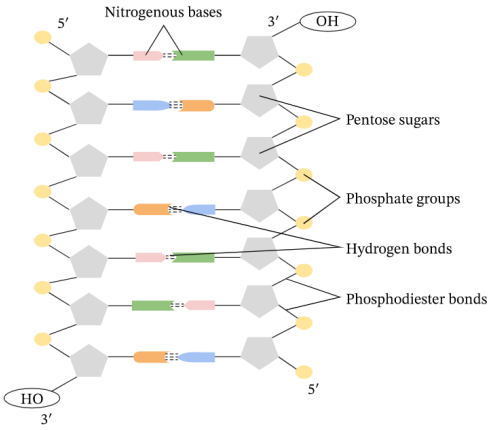


Figure 1.4 DNA Opposite structure

DNA turns on its axis to form a helical structure, as shown in the Figure:

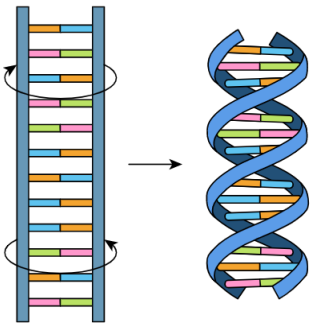


Figure 1.5 DNA Helical structure

DNA is not present in the cell directly in this form, but it exists in more complex forms, and the reason is that every cell in the body contains an amount of DNA about two meters long.

We observe that the double helix first wraps around protein compounds called Histones to form a structure called the nucleosome, after that, the nucleosomes form structures called chromatin, and these structures continue to complex to form the final shape of the chromosome.

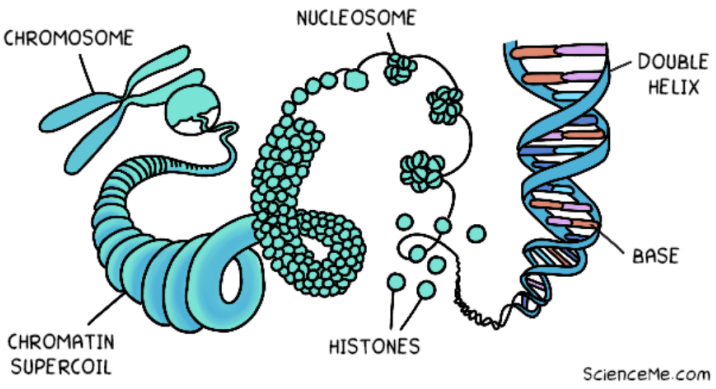


Figure 1.6 Complex structure levels of DNA

2.5. RNA

RNA is a linear polymer of four different nucleotides. Each nucleotide is composed of three parts: a five-carbon sugar known as ribose; a phosphate group; and one of four bases attached to each ribose, that is, adenine (A), cytosine (C), guanine (G), or uracil (U).

The combination of base and sugar constitutes a nucleoside. The structure of RNA is a repeating chain of ribose and phosphate moieties, with one of the four bases attached to each ribose. The structure and function of the RNA vary depending on its sequence and length. [4]

When comparing the structure of DNA and RNA, we find some differences, including the number of strands, wherein DNA we have a double-stranded helix, while in RNA we have one strand, the second difference is in the type of sugar, in DNA we find deoxyribose sugar, but in RNA we find ribose sugar, the third difference in the type of nucleotides wherein RNA we find uracil (U) instead of Thymine (T), and this affects the properties of the compound, as we find that RNA is less stable than DNA, which affects the functional aspect, as DNA is the fixed copy carrying the genetic characteristics of the cell, while RNA serves a specific function that the cell needs for a temporary period.

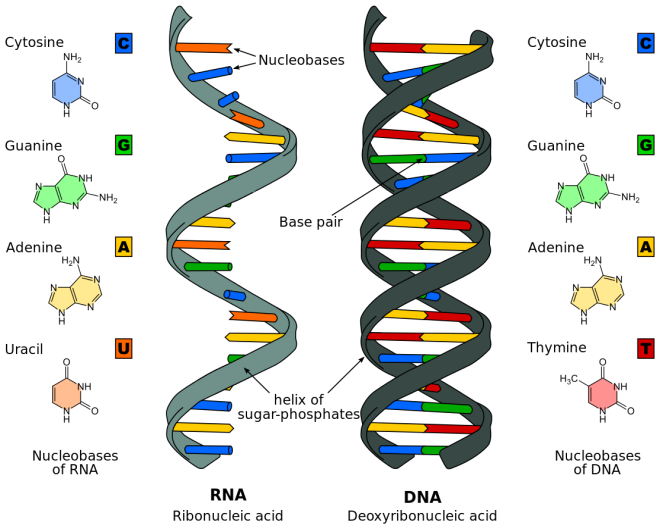


Figure 1.7 Difference between DNA and RNA

2.5.1. Classes of RNA

RNA has initially two main classes, Coding-RNAs and Non-Coding-RNAs, the coding-RNA consists of messenger RNA (mRNA), and Non-Coding RNA is divided into two types, housekeeping non-coding-RNA, and regulatory non-coding RNA, housekeeping-RNA contains transfer-RNA (tRNA) and ribosomal-RNA (rRNA) in it, regulatory RNA divided into long non-coding RNA (lncRNA) and small non-coding-RNA (sRNA), this last one is further classified into five different types that are: micro RNA (miRNA), small nucleolar RNA (snoRNA), small interfering RNA (siRNA), small nuclear

RNA (snRNA) and PIWI-interacting RNA (piRNA). We can summarize all of that in the Figure below:

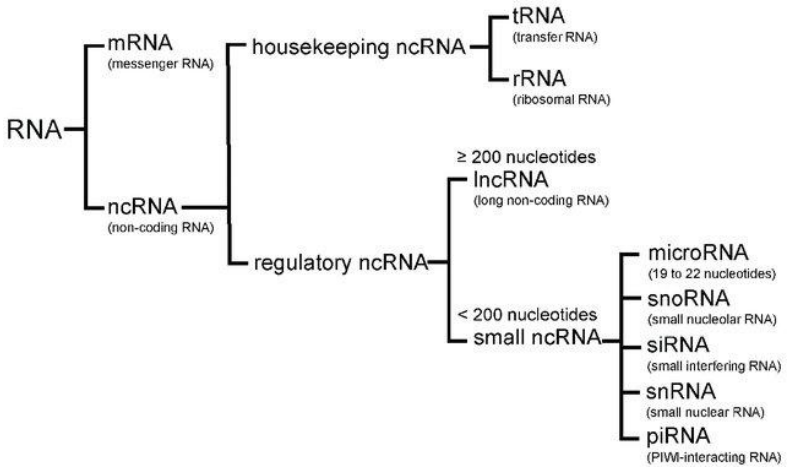


Figure 1.8 Major RNA types

2.5.2. Coding and non-coding RNAs

Coding RNAs

Coding RNAs generally refer to mRNA that encodes protein to act as various components including enzymes, cell structures, and signal transductors. [5]

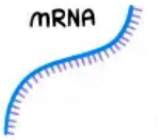


Figure 1.9 mRNA structure

Non-Coding RNAs

A non-coding RNA (ncRNA) is a kind of RNA that is not converted into protein, however, it is involved in many biological processes, diseases, and cancers. Numerous ncRNAs have been identified and classified with high throughput sequencing technology. Hence, accurate ncRNAs class prediction is important and necessary for further study of their functions. Several computation techniques have been employed to predict the class of ncRNAs. Recent classification methods used the secondary structure as their primary input. However, the computational tools of

RNA secondary structure are not accurate enough, which affects the final performance of ncRNAs predictors. [6]

Non-coding RNAs including tRNA, rRNA, lncRNA, and sRNA with their five sub-classes miRNA, snoRNA, piRNA, snRNA, and siRNA.

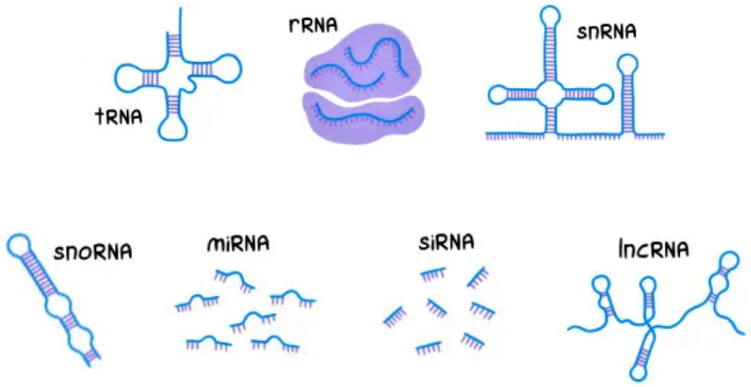


Figure 1.10 Several ncRNA structures

2.6. Genetic information: from gene to protein

2.6.1. Genome

A genome is the complete set of genetic information in an organism. It provides all of the information the organism requires to function. In living organisms, the genome is stored in long molecules of DNA called chromosomes. Small sections of DNA, called genes, code for the RNA and protein molecules required by the organism. In eukaryotes, each cell's genome is contained within a membrane-bound structure called the nucleus. Prokaryotes, which contain no inner membranes, store their genome in a region of the cytoplasm called the nucleoid. The full range of RNA molecules expressed by a genome is known as its transcriptome, and the full assortment of proteins produced by the genome is called its proteome.

There are 23 pairs of chromosomes in the human genome. Between 1990 and 2003, all twenty-three pairs were fully sequenced through an international research undertaking known as the Human Genome Project. The study and analysis of genomes is called genomics. [7]

2.6.2. Chromosome

Chromosomes are large subcellular structures, visible in the light microscope, that are found in the nuclei of most eukaryotic cells. Each chromosome consists of a single very long DNA molecule that has been compacted approximately 10,000-fold by interactions with proteins, such that the resulting chromosome structure fits within a typical eukaryotic nucleus of only 10 microns in diameter. Several levels of the structural organization are involved in the formation of chromosomes. Most chromosomal DNA is wrapped in left-handed superhelical turns around protein 'spools', called histone octamers, to form nucleosomes. Arrays of these nucleosomes, or 'beads on a string', are further compacted into solenoidal structures, called 30 nm chromatin fibers. [8]

2.6.3. Gene

A gene is a basic functional and structural unit of genetic information. In molecular genetics, a gene is a segment of DNA that is transcribed and contains regulatory and coding parts. According to the genetic information that they carry, we can divide genes into three groups: structural genes, regulatory genes, and genes for the RNA molecules, except mRNA. [9]

2.6.4. Protein

A protein is a naturally occurring, extremely complex substance that consists of amino acid residues joined by peptide bonds. Proteins are present in all living organisms and include many essential biological compounds such as enzymes, hormones, and antibodies. [10]

2.6.5. Central Dogma

The journey from gene to protein is complex and tightly controlled within each cell. It consists of two major steps: transcription and translation. Together, transcription and translation are known as gene expression. [11]

Transcription

During the process of transcription, the information stored in a gene's DNA is passed to a similar molecule called RNA (ribonucleic acid) in the cell nucleus. Both RNA and DNA are made up of a chain of building blocks called nucleotides, but they have slightly different chemical properties. The type of RNA that contains the information for making a protein is called messenger RNA (mRNA) because it

carries the information, or message, from the DNA out of the nucleus into the cytoplasm.

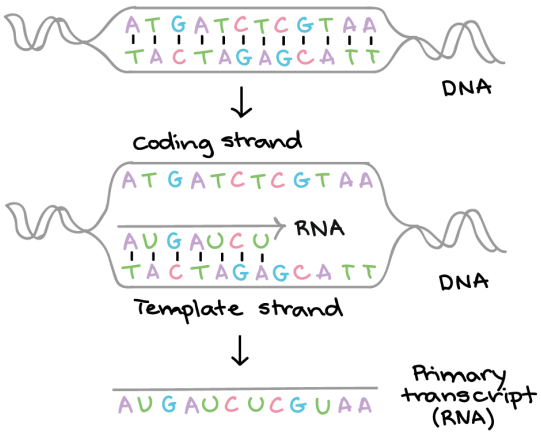


Figure 1.11 Transcription phase process

Translation

Translation, the second step in getting from a gene to a protein, takes place in the cytoplasm. The mRNA interacts with a specialized complex called a ribosome, which "reads" the sequence of mRNA nucleotides. Each sequence of three nucleotides, called a codon, usually codes for one particular amino acid. (Amino acids are the building blocks of proteins.) A type of RNA called transfer RNA (tRNA) assembles the protein, one amino acid at a time. Protein assembly continues until the ribosome encounters a “stop” codon (a sequence of three nucleotides that does not code for an amino acid).

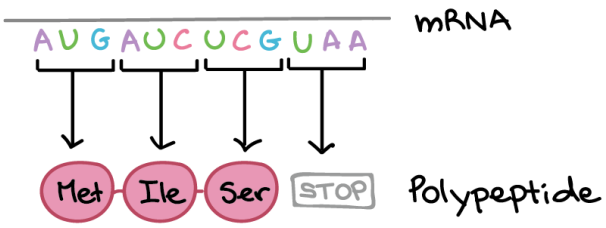


Figure 1.12 Translation phase process

The flow of information from DNA to RNA to proteins is one of the fundamental principles of molecular biology. It is so important that it is sometimes called the “central dogma.” [11]

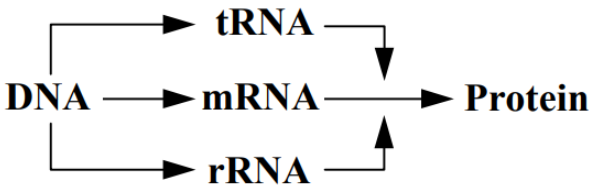


Figure 1.13 Central Dogma process

rRNA: ribosomal RNA that is required for building ribosomes, which are structures necessary for protein synthesis.
tRNA: transfer RNA that serves to transfer individual amino acid molecules from the general cytoplasm to their appropriate location in a growing polypeptide during protein synthesis.
mRNA: messenger RNA that carries the specific instructions for building a specific protein. Both rRNA and tRNA are generic groups of molecules in that all types of rRNA and all types of tRNA are involved in the synthesis of every type of protein. However, mRNA is specific in that a different type of mRNA is required for every different type of protein. [12]

3.Bioinformatics

3.1. History

It is easy for today's students and researchers to believe that modern bioinformatics emerged recently to assist next-generation sequencing data analysis. However, the very beginnings of bioinformatics occurred more than 50 years ago, when desktop computers were still a hypothesis and DNA could not yet be sequenced.

The foundations of bioinformatics were laid in the early 1960s with the application of computational methods to protein sequence analysis (notably, de novo sequence assembly, biological sequence databases, and substitution models). Later on, DNA analysis also emerged due to parallel advances in molecular biology methods, which allowed easier manipulation of DNA, as well as its sequencing, and computer science, which saw the rise of increasingly miniaturized and more powerful computers, as well as novel software better suited to handle bioinformatics tasks. In the 1990s through the 2000s, major improvements in sequencing technology, along with reduced costs, gave rise to an exponential increase of data. The arrival of 'Big Data' has laid out new challenges in terms of data mining and management, calling for more expertise from computer science in the field.

Coupled with an ever-increasing amount of bioinformatics tools, biological Big Data had (and continues to have) profound implications on the predictive power and reproducibility of bioinformatics results. To overcome this issue, universities are now fully integrating this discipline into the curriculum of biology students. Recent sub-disciplines such as synthetic biology, systems biology, and whole-cell modeling have emerged from the ever-increasing complementarity between computer science and biology. [13]

Figure 1.14 The first automated DNA sequencer was the AB370A, introduced in 1986 by Applied Biosystems

Figure 1.15 The NextSeq 2000 Sequencing System, The latest NGS system from Illumina

3.2. What is Bioinformatics?

Bioinformatics is a multidisciplinary field of science that encapsulates biology, chemistry, physics, statistics, and computer science as base domains to solve complex

biological phenomena. Bioinformatics is one of the magnanimously growing scientific fields that are very flexible.

The main purpose of bioinformatics is to store, analyze, and retrieve essential information about organisms that can in turn help understand the dynamics of such organisms. [14]

Today, new high throughput technologies bring unprecedented opportunities for life science research, they allow us to get new data never seen before, to study new questions impossible to study before, and to discover new phenomena unimaginable before. Now we can sequence not only one person's genome but also many different people's genomes.

This Figure shows the number of nucleotides sequenced and stored in the GenBank database over the year.

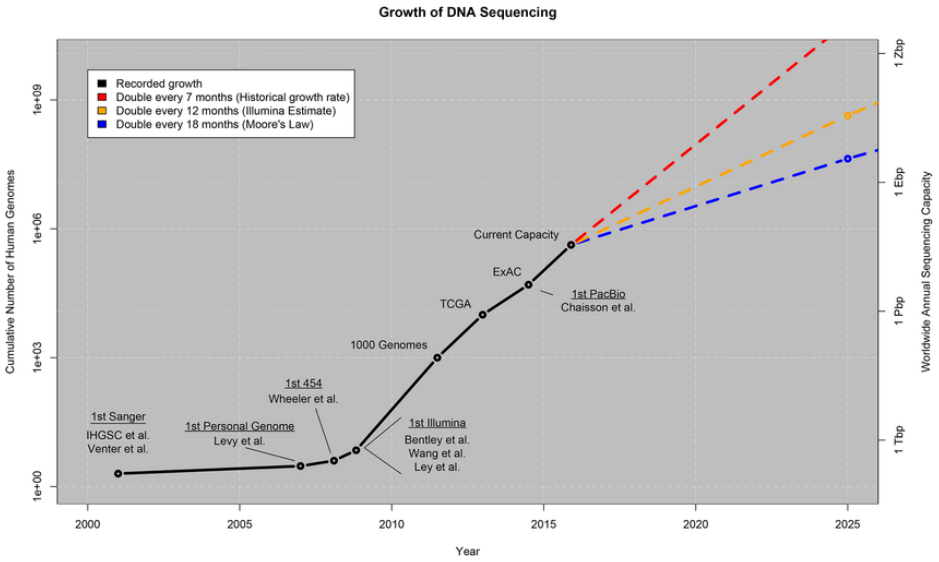


Figure 1.16 Growth of DNA sequencing

The number of nucleotides has been increasing exponentially, and at the same time the sequencing cost has been decreasing every year, the reason is that Next-Generation sequencing (NGS) technologies have been developed and used more and more widely in all areas of life sciences.

3.3. Different types of problems studied in bioinformatics

- Development of new algorithms and statistics with which to assess relationships among members of large data sets.
- Analyses and interpretation of various types of sequences, domains, and structures.

- Development and implementation of tools that enable efficient access and management of different types of information. [15]

3.4.Application Domains of Bioinformatics

Bioinformatics can contribute to many research, development, and even industrial fields, including

3.4.1. Molecular medicine

The completion of the human genome, means that we can search for the genes directly associated with different diseases and begin to understand the molecular basis of these diseases more clearly. This new knowledge of the molecular mechanisms of disease will enable better treatments, cures and even preventative tests to be developed.

3.4.2. Gene therapy

In the not too distant future, the potential for using genes themselves to treat disease may become a reality. Gene therapy is the approach used to treat, cure or even prevent disease by changing the expression of a person's genes. Currently, this field is in its infantile stage with clinical trials for many different types of cancer and other diseases ongoing.

3.4.3. Drug development

Currently, all drugs on the market target only about 500 proteins. With an improved understanding of disease mechanisms and using computational tools to identify and validate new drug targets, more specific medicines that act on the cause, not merely the symptoms, of the disease, can be developed. These highly specific drugs promise to have fewer side effects than many of today's medicines. [16]

3.5. Limitations of bioinformatics

- Bioinformatic predictions aren’t the formal proofs of any concepts, they can’t replace the experimental research of actually testing the theory.
- The quality of bioinformatic predictions depends upon the quality of the data and the algorithms being used.

- The Informatics-Based approach allows the development of drugs in less time and with low cost and higher potency, with few side effects as compared to traditional trial and error approaches in drug development.
- The sequenced data often contains errors, if the sequences are wrong then the annotations are incorrect, so the further research is misleading.

Chapter 2 Classification of ncRNAs

Introduction

Deep learning is one of the most tech approaches used in the field of bioinformatics, due to its effectiveness in working on large genomic data, and also because of its precision of analysis results. In order to well understand the theoretical approaches of this project, we firstly define in this chapter deep learning and some of its concepts, then we mention the added value of using deep learning in bioinformatics, in the second part we cite some of the ncRNA classification methods, then, in the last part, we make a comparative analysis between three of similar projects to our project, by giving a brief description of them, compare them according to four criteria (Dataset, classification approach, DL model, and accuracy), and finally discuss the differences between them.

1.Deep Learning overview

1.1. What is deep learning?

Deep learning is a type of machine learning, which is a subset of artificial intelligence, and we can define it as a neural network-based method with multiple hidden layers and is inspired by the representation of biological neural networks. Deep learning is considered among the best paradigms of machine learning approaches for prediction and classification. In the past decade, deep learning has been successfully applied to real-world and research problems such as image recognition, speech recognition, and language translation. In the recent past, deep learning has been applied in bioinformatics and medicine, particularly to RNA–seq data, and has shown improvements over previous results [17].

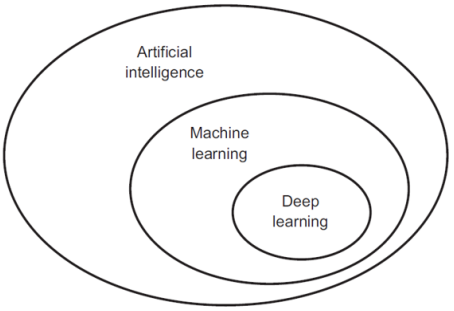


Figure 2.1 Relation between AI, ML, and DL

1.1.1. Basic Structure of Neural Network

A neural network is a class of information processing modules, frequently utilized in machine learning. Within a multi-layer context, the basic building units, namely neurons, are connected to each other among the adjacent layers via internal links, but the neurons belonging to the same layer have no connection. [18]

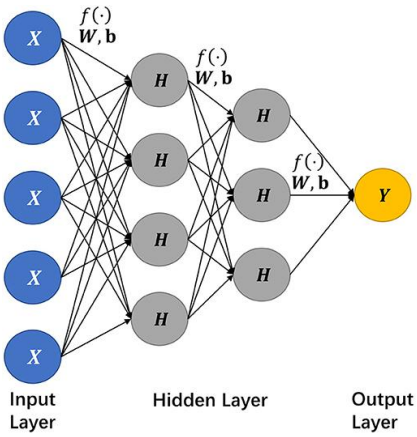


Figure 2.2 The network structure of a deep learning model

1.2.Machine learning approaches

Generally, we can count three learning approaches used in machine learning, supervised learning, unsupervised learning, and reinforcement learning, defined as follow:

1.2.1.Supervised learning

Supervised learning is where you know the input variables and the output variable and you use an algorithm to learn the mapping function between them.

The goal is to approximate the mapping function so well that when we have new input data, we can predict the output variables for that data.

It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

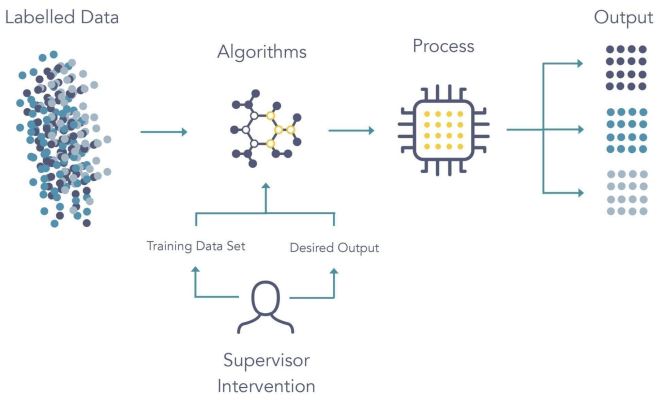


Figure 2.3 Supervised learning pipeline

There exist two types of problems that can be solved with Supervised learning **Classification** and **Regression**

a. Classification

This task is used when we need a limited set of outcomes. It generally provides predicted output values, which may be True or False. It performs in the way of two types such **Binomial** and **Multi-Class**. [19] For example, we can find whether an RNA sequence is coding.

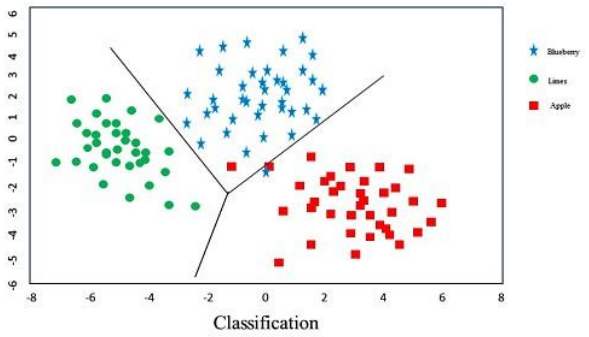


Figure 2.4 Example of Classification

Classification supported Algorithms

- Logistic regression
- KNN (k-Nearest Neighbor)
- Naive-Base
- Discriminant Analysis
- SVM (support vector machine)

- Decision tree
- Neural network

Applications

- Image classification
- Email spam detection

b. Regression

It can help us to predict (expect continuous values) and explains objects based on a given set of numerical and categorical data. For example, we can predict disease causes based on protein attributes.

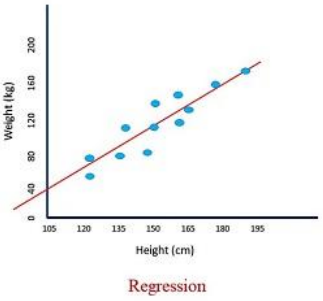


Figure 2.5 Example of regression

Regression supported algorithms

- Simple linear regression
- Multiple linear regression
- Polynomial linear regression
- Decision tree
- Radom forest

Applications

- Risk assessment
- Score prediction
- Market forecasting

1.2.2. Unsupervised learning

Unsupervised learning uses machine learning algorithms to analyze and cluster unlabeled datasets. These algorithms discover hidden patterns or data groupings without the need for human intervention. Its ability to discover similarities and differences in information makes it the ideal solution for exploratory data analysis, cross-selling strategies, customer segmentation, and image recognition.

Unsupervised learning models are utilized for three main tasks-clustering, association, and dimensionality reduction. [20]

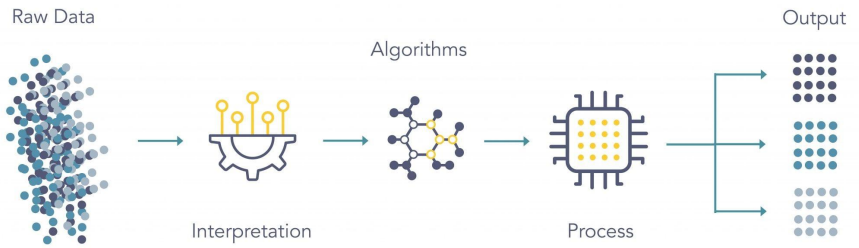


Figure 2.6 Unsupervised learning pipeline

1.2.3. Reinforcement learning

Reinforcement-learning differs from supervised learning in the way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience. [21]

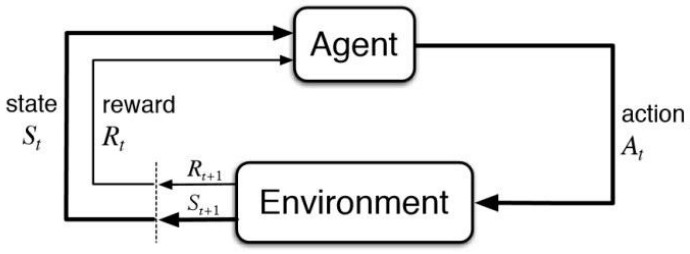


Figure 2.7 Reinforcement learning pipeline

1.3. Deep Neural Network models

Recently, deep neural network models have emerged as powerful machine learning and artificial intelligence tool. A deep neural network (DNN) is an artificial neural network (ANN) with multiple layers between the input and output layers, we can find three main types of DNN models as follow:

1.3.1. Multi-Layer Perceptrons (MLP)

Multilayer perceptrons (MLPs) are a class of feedforward artificial neural networks (ANNs). An MLP model is the most basic deep neural network, consisting of a series of fully connected layers. Today, MLP machine learning methods can be used to overcome the high computational power requirements of modern deep learning architectures.

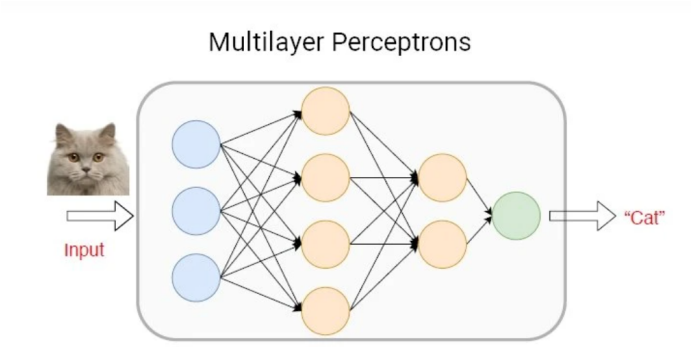


Figure 2.8 Concept of Multilayer Perceptrons

1.3.2. Convolutional Neural Networks (CNN)

In contrast to the fully connected layers in MLP, in a CNN model, one or more convolutional layers extract simple features from the input by performing convolution operations. Each layer is a set of non-linear functions that are weighted sums at different coordinates of a spatially close subset of the output of the previous layer, allowing weights to be reused.

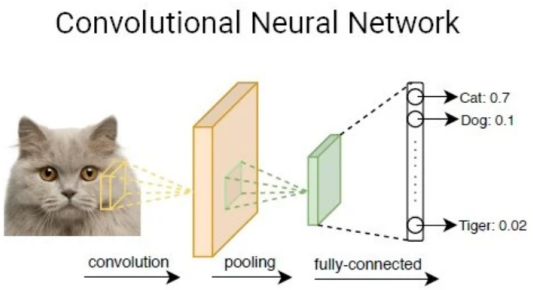


Figure 2.9 Concept of a Convolution Neural Network

1.3.3. Recurrent Neural Networks (RNN)

Recurrent Neural Networks (RNNs) is another class of artificial neural networks that use sequential data injection. RNNs were developed to solve time series problems with sequential input data.

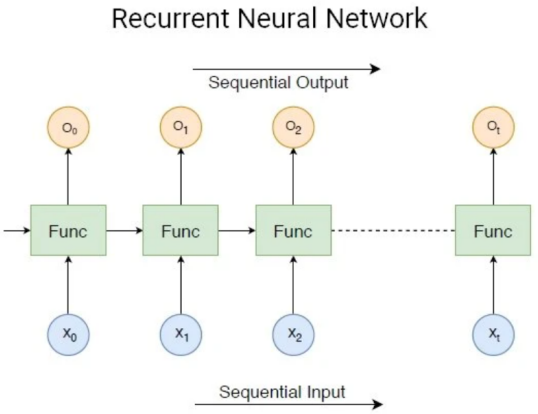


Figure 2.10 Concept of a Recurrent Neural Network

1.4. Machine learning in Bioinformatics

In the era of big data, the transformation of large quantities of data into valuable knowledge has become increasingly important in various domains, and bioinformatics is no exception. Significant amounts of biomedical data, including omics, have been accumulated, and the resulting potential for applications in biological and healthcare research has caught the attention of both industry and academia. For instance, IBM developed Watson for Oncology, a platform analyzing patients' medical information and assisting clinicians with treatment options. In addition, Google DeepMind, having achieved great success with AlphaGo, recently launched DeepMind Health to develop effective healthcare technologies.

To extract knowledge from big data in bioinformatics, deep learning has been a widely used and successful methodology. Deep learning methods use training data to uncover underlying patterns, build models, and make predictions based on the best fit model. Indeed, some well-known algorithms have been applied in genomics, proteomics, systems biology, and numerous other domains. [22]

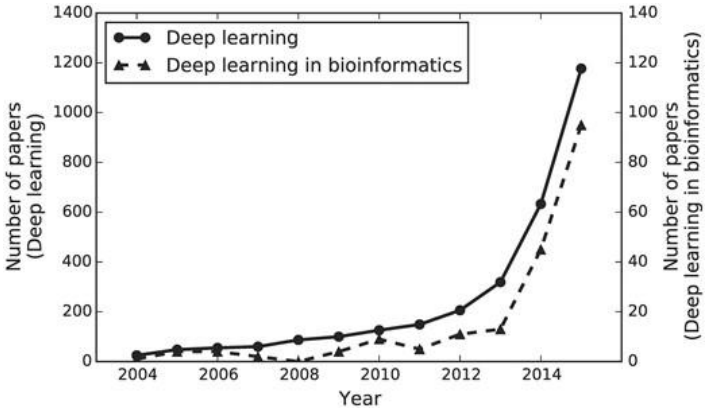


Figure 2.11 Approximate number of published deep learning articles by the year 2004-2014

2. ncRNA classification methods

The classification of the great majority of ncRNAs relies on the empirical attributes originally used to detect them. This reflects their short history relative to protein-coding genes and provides a convenient basis by which to classify these uncharacterized RNA species.

Here, we will mention three of the classification methods.

2.1. Classification based on transcript length

The length estimate of ncRNAs serves as the most commonly used attribute for their classification. Typically, a threshold of 200 bases separates long from short ncRNAs. Often, our knowledge is limited to sequence reads mapped to a 'region of transcription', and even with improvements in NGS read length, this will probably continue for the foreseeable future.

2.2. Classification based on association with annotated protein-coding genes

This commonly used attribute serves as the foundation of the GENCODE classification of ncRNAs. It underlies the logical challenge of overlapping non-coding and coding transcripts at a given locus - called "transcriptional forests" by the FANTOM consortium.

2.3. Classification based on function

ncRNAs can participate in a plethora of different cellular processes: chromatin remodeling, regulation of transcription and translation, RNA stability, scaffolding, and

innate immunity just to name a few. We discuss here only examples of functions used for classification.

3. Comparative analysis

3.1. Similar works

3.1.1. nRC Classification tool:

This tool is under a research paper titled 'nRC tool - non-coding RNA Classifier based on structural features '. It is defined as a novel method for classifying ncRNA sequences belonging to different classes that uses the structural features extracted from the ncRNA secondary structure, rather than the primary structure since it has been demonstrated that the structure of ncRNAs can provide relevant information about their biological functions and therefore their class type. Moreover, It adopted a supervised classification algorithm implementing a deep learning (DL) architecture based on convolutional neural networks (CNNs). [23]

Proposed method: It classifies ncRNA sequences by exploiting a set of discriminative substructures extracted from RNA secondary structures. Starting from a dataset composed of ncRNA Fasta sequences belonging to different non-coding classes, It first predicts the secondary structure of each sequence. Then, It identifies as features all the discriminative frequent sub-structures extracted from predicted ncRNA secondary structures. Finally, a supervised classification algorithm is trained using as input an ncRNA sequence vs. sub-structures boolean matrix.

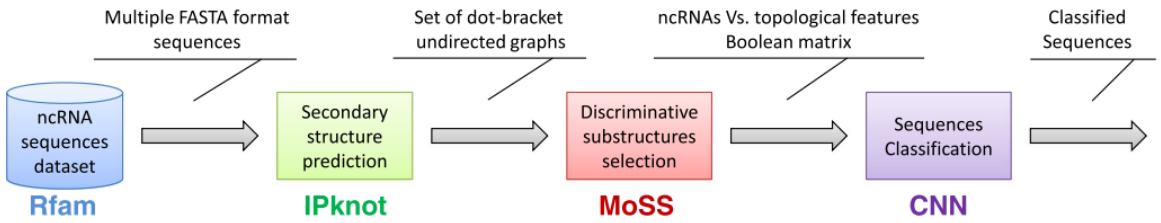


Figure 2.12 Pipeline of the nRC classification tool

3.1.2.ncRDense Classification tool:

The title of the related paper to this project is 'ncRDense: A novel computational approach for classification of non-coding RNA family by deep learning '. This project is consist of a novel deep learning-based architecture, to effectively classify and distinguish ncRNA families.

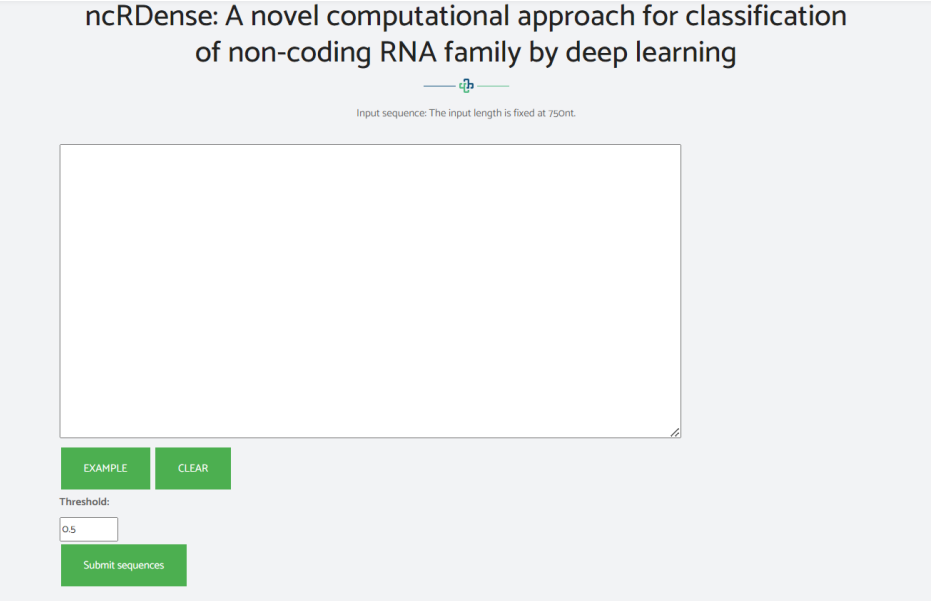


Figure 2.13 ncRDense web server screenshot

Proposed method: The main architecture of ncRDense consists of several blocks of densely connected convolutional layers with fully connected layers at the end for the classification. A block in ncRDense is composed of 1-dimensional convolutional layers where only the first convolutional layer is connected to the subsequent convolutional layers. Moreover, to identify the family to which a given ncRNA belongs, ncRDense utilizes ncRNA feature extraction as an extra information source.

The proposed model has an improved ability to distinguish between classes that belong to the same sub-category and share similar properties. The use of the combination of ncRNA sequence and secondary structure leads to the best scores on traditional evaluation metrics, including accuracy, recall, precision, F1-score, and Matthews correlation coefficient (MCC). Finally, they built a freely accessible webserver for ncRNA prediction based on the proposed model. [24]

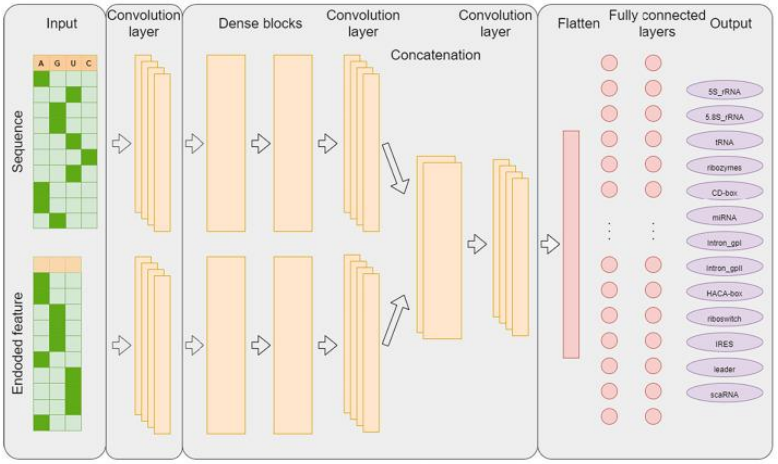


Figure 2.14 ncRDense model architecture

3.1.3.RNAcon Classification tool:

This project is demonstrated in the research paper titled 'Prediction and classification of ncRNAs using structural information ', In this study, they initially develop prediction tools to discriminate coding or non-coding transcripts and thereafter classify ncRNAs into respective classes. In comparison to the existing methods that employed multiple features, this SVM-based method by using a single feature (tri-nucleotide composition), achieved an MCC of 0.98. Also, they use graph properties of predicted ncRNA structures to classify the transcripts into 18 different non-coding RNA classes. They developed classification models using a variety of algorithms (BayeNet, NaiveBayes, MultilayerPerceptron, IBk, libSVM, SMO, and RandomForest).

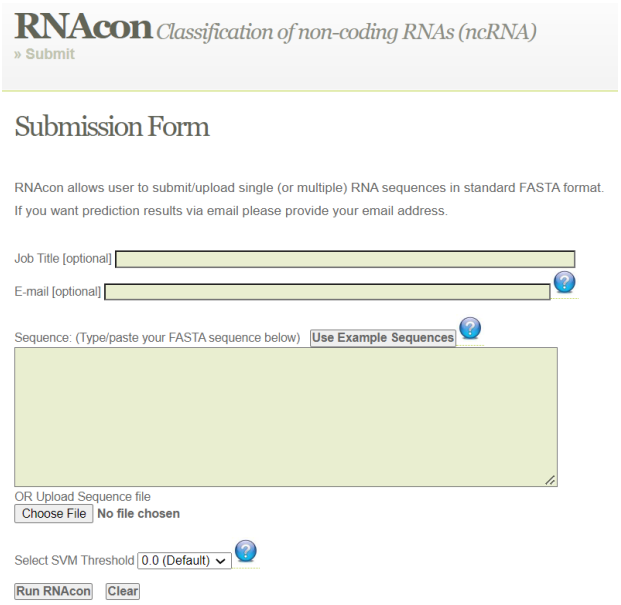


Figure 2.15 RNAcon web server screenshot

Proposed method: In this study, a systematic attempt has been made to predict and classify ncRNAs. The SVM-based TNC approach discriminated between non-coding and coding RNAs efficiently. Furthermore, the graph properties-based approach classifies different ncRNA classes using the RandomForest classifier. Analysis showed that the length of RNAs has a negative correlation with the prediction sensitivity for classifying noncoding RNAs. Comparatively, RNAcon performed well than other gene-calling programs and Rfam-based covariance models. All these prediction models have been implemented in the form of a web server. [25]

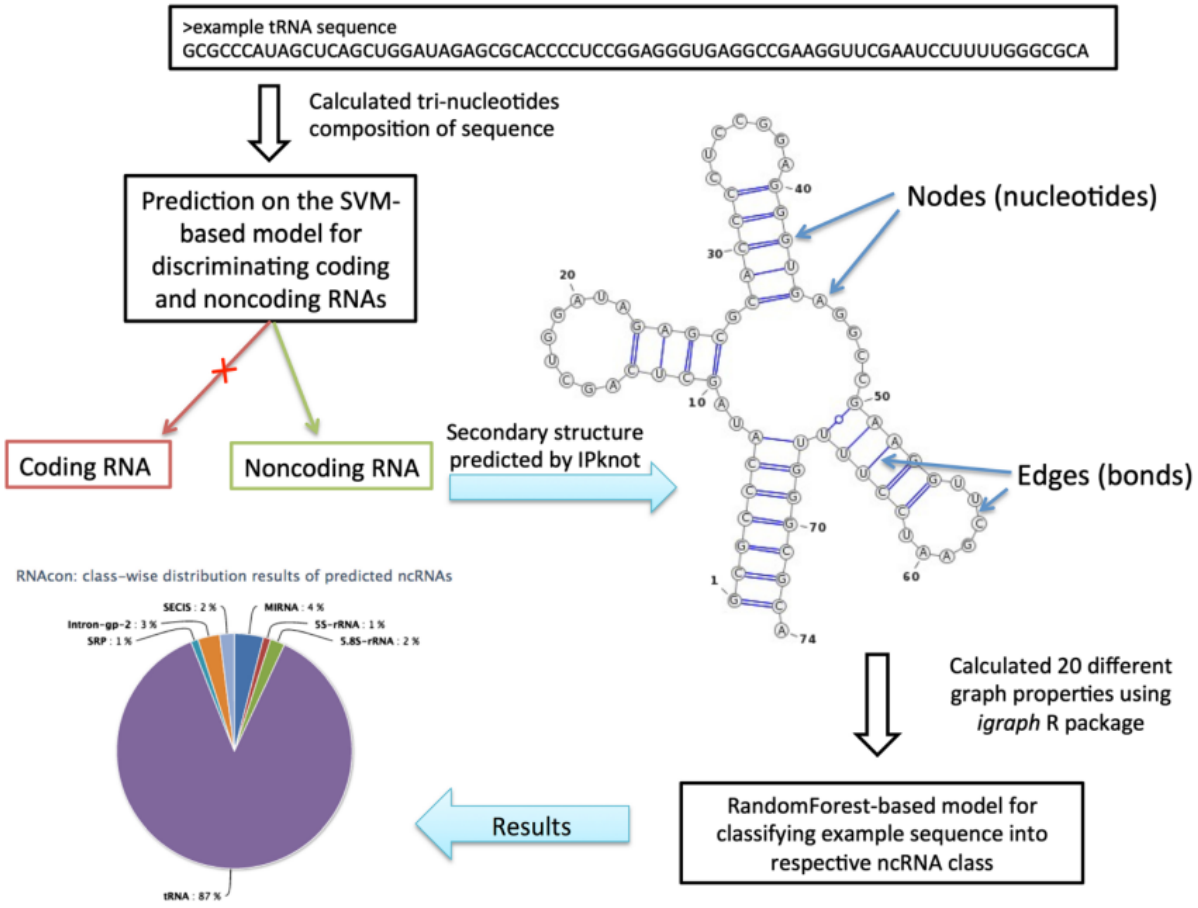


Figure 2.16 An overview of the RNAcon with an example sequence

3.2. Comparison

We can summarize the main differences between the three previous works in the table below:

	Dataset sources	Classification method	Model Approach	Accuracy score
nRC	Rfam repository	Primary/ Secondary structure features	Convolutional Neural Networks (CNNs) model	74%
ncRDense	Rfam repository	-Secondary structure, -Electron/ion interaction pseudopotentials (EIIPs), -Enhanced nucleic acid composition, and nucleotide chemical property (NCPs)	Convolutional Neural Networks (CNNs) model	82.32%
RNAcon	-CONC datasets -Rfam -RefSeq database	Tri-nucleotide composition	-SVM-based model -Graph-based approach	43%

table 2.1 Comparison between the mentioned similar works

3.3. Discussion

We observed that all the previous projects have used the Rfam repository to import sequences data, because it is a rich and verified source and updated periodically, except in the RNAcon they used other repositories, for the classification method, all of them have used structure-based approach, whereas, for the architecture used in the model, both of the nRC and the ncRDense projects have used convolutional neural networks (CNN)

architectures, except the RNAcon project which used several algorithms and architectures to compare between them and then it chooses an SVM architecture with graph-based approach, finally, for the accuracy score we observe that the ncRDense has the best score between them with 82.32% as we see in the graph below:

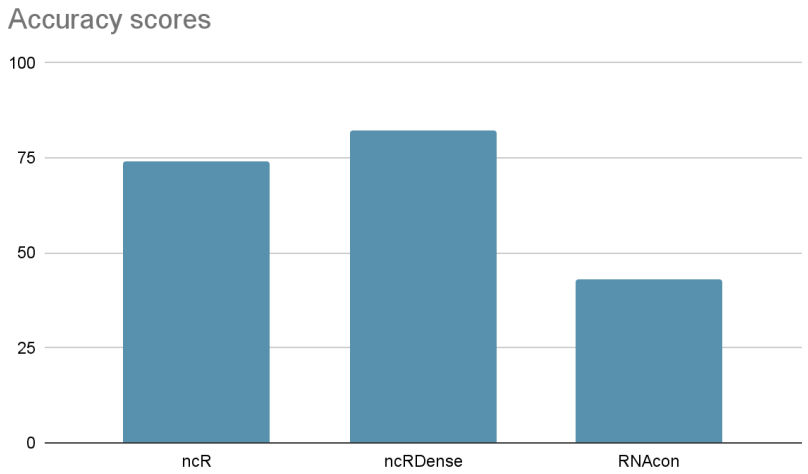


Figure 2.17 Comparison between the accuracy scores of the mentioned similar works

Chapter 3 Project Design and structure

1. Overview and objectives

The effective classification and identification of ncRNAs is a challenging and vital task in biology and bioinformatics because of their enormous impact on many biological processes.

As we have seen in the comparison between similar projects in the previous chapter, the project ncRDense is the best performant project, so, here we will try to implement a project with an approximate model and method, also we will try to optimize the evaluation metrics scores like accuracy and sensitivity in order to give effective classification results that will be used in biological and medical issues.

2. Project pipeline

2.1. Model Structure

The main function of this project is to train a deep learning model based on the Convolutional Neural Networks (CNN) approach. The Figure below explained the general conception of the model we will train, and we will the detailed architecture in the next chapter.

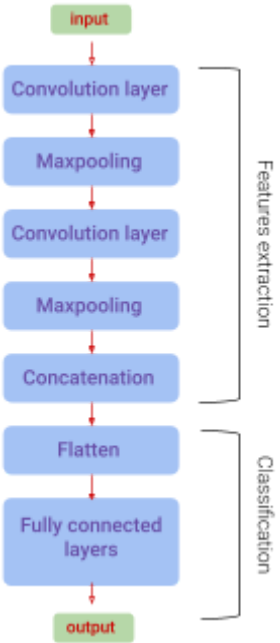


Figure 3.1 project general pipeline concept

- **Inputs:** Consist of ncRNA sequences in FASTA format.

- **Convolutional layers:** They apply a convolution operation to the input, passing the result to the next layer.

- **Concatenation:** A concatenation layer takes inputs layers and concatenates them along a specified dimension. in this case, concatenate 2D layers to 1d layers.

- **Flatten:** Flattening is used to convert all the resultant arrays from pooled feature maps into a single long continuous linear vector. The flattened matrix is fed as input to the fully connected layer to classify the data.

- **Fully connected layers:** are those layers where all the inputs from one layer are connected to every activation unit of the next layer. The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened and then fed into the fully connected layer.

- **Output:** It is the final phase of the model architecture and it is consist of the ncRNA classes.

3. Data preparation

Basically, our Datasets will be in FASTA format which is a text-based format for representing either nucleotide sequences or peptide sequences, in which base pairs or amino acids are represented using single-letter codes. A sequence in FASTA format begins with a single-line description, followed by lines of sequence data. The main purpose to use this format is due to its simplicity and sensitivity.

Figure 3.2 A sample of FASTA File

3.1. Rfam repository

The source of our Datasets will be the Rfam repository, which is a collection of RNA sequence families of structural RNAs including non-coding RNA genes as well as cis-regulatory elements. Each family is represented by a multiple sequence alignment and a covariance model (CM). You can reach it here [26].

Figure 3.3 Rfam repository logo

In the phase of data preparation, we have three steps to do:

Step 1: Import the data

Here we will download the FASTA files we need to train our model from the Rfam repository.

Step 2: Cleaning the data

In this step, we will:

- Check the data files and information corresponding to our subject (non-coding RNAs).
- Handle the missing or damaged data, for a clean model and accurate results.
- Reformate the data to uniform values (in our case we will change the RNA sequences to a One-Hot encoding and fix the length to 250).

Step 3: Splitting the data

Here we will divide the dataset into two sets: a **training set** and a **testing set**. 90% for training, and 10% for testing (validation).

```
>AT1G09780|1|training
GTGGAGTAGAAGAATTGAGAGCCTTATCAG
TTTTTGAAGAGAGGGCTGAAACTCTCTAGT
TATCTTTTGTTGCTTTTCTAATAATAAGAG
TTTACACACAG
>AT1G31812|0|testing
TCCTCATCTGCAGTAACTTTATCTTAAGCA
TCAAAATAACATTGCATAAGACTTGTTCTT
GCTCTTGTGTTTCTATCATATTTAAGCTAT
CTACTTTGTGA
```

Figure 3.4 An example of split data

4. Model training

4.1. Load Datasets

Datasets can be loaded from local files stored on your computer, and also from remote files. in our model, we will use the NumPy library to load data.

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. [27]

Figure 3.5 NumPy Library logo

4.2. ConFigure model

Any model has a wide range of configurable options, including which features are used, how data is selected, a wide variety of algorithm-specific learning settings, potential pre or post-processing, verification methods, etc.

4.3. Create model

Here we define all the model parameters such as max-pooling, activation, flatten functions, and so on,..., in order to get a robust model.

4.3.1. Max Pooling

Calculate the maximum value for each patch of the feature map, It adds a small amount of translation invariance - meaning translating the matrix by a small amount does not significantly affect the values of most pooled outputs.

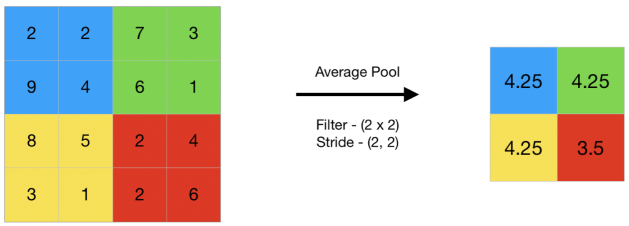


Figure 3.6 Maximum Pooling example

4.3.2. Activation

An activation function in a neural network defines how the weighted sum of the input is transformed into an output from a node or nodes in a layer of the network.

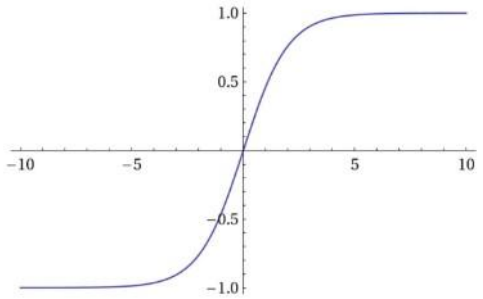


Figure 3.7 Softmax Activation function graph

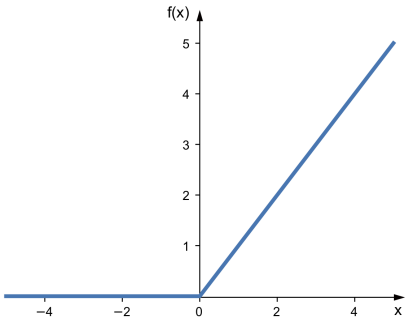


Figure 3.8 ReLU Activation function graph

4.3.3. Flatten

Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector.

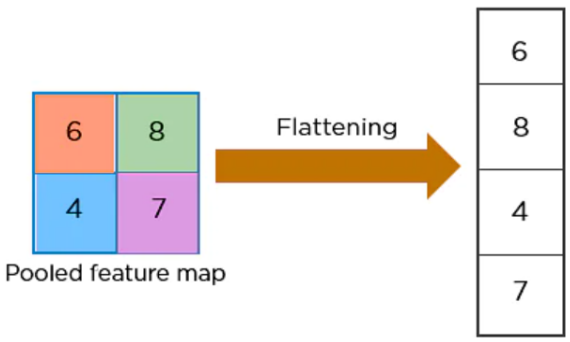


Figure 3.9 Flatten function process

4.3.4. DenseNet

A DenseNet is a type of convolutional neural network that use dense connections between layers, through Dense Blocks, where we connect all layers (with matching feature-map sizes) directly with each other.

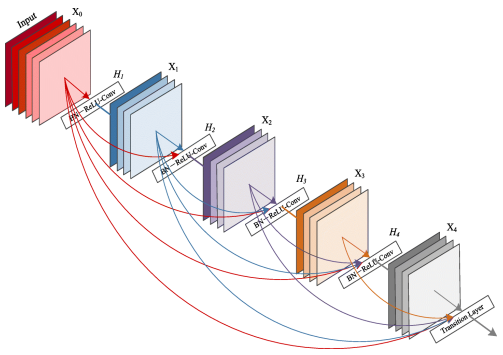


Figure 3.10 DenseNet Architecture

4.4. Compile model

The compilation is performed using one single method call called compile(), The compile method requires several parameters, the loss parameter, the metrics parameter, and an optimizer for training the network.

4.4.1. Loss function:

The loss function is the function that computes the distance between the current output of the algorithm and the expected output. It's a method to evaluate how your algorithm models the data. It can be categorized into two groups. One for classification (discrete values, 0,1,2...) and the other for regression (continuous values). [28].

There are a lot of Loss function types, in our case, we use the **Cross-entropy** function to train our model because it leads to better generalization models and faster training.

Range of values for this class of Loss function:

- **0.00:** Perfect probabilities
- < **0.02:** Great probabilities
- < **0.05:** In a good way
- < **0.20:** Great

- > **0.30:** Not great
- **1.00:** Hell
- > **2.00** Something is not working

4.4.2. Metrics parameter

A metric is a function that is used to judge the performance of your model. Metric functions are similar to loss functions, except that the results from evaluating a metric are not used when training the model. Note that you may use any loss function as a metric. [29]

To evaluate our model, we will use the **Accuracy** metric, which is a class of metrics that calculates how often predictions equal labels, an Accuracy measure of anything between 70%-90% is not only ideal, it's realistic.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Where TP, TN, FP, and FN are True Positives, True Negative, False Positives, and False Negatives respectively.

4.4.3. Optimizer

An optimizer is a function that modifies the attributes of the neural network, such as weights and learning rate. Thus, it helps in reducing the overall loss and improving the accuracy [30].

Here, we use the **Adam** optimizer because it has the best accuracy in enhancing the CNN ability in classification.

Chapter 4 Implementation and Results

1. Implementation

1.1. Tools and environments

1.1.1. Python

Python is an interpreted, cross-paradigm, cross-platform programming language. It promotes structured, functional, and object-oriented imperative programming.

Used version: 3.10.0

Figure 4.1 Python language logo

1.1.2. Google Colab

Google Colab or Colaboratory is a cloud service, offered by Google, based on Jupyter Notebook and intended for training and research in machine learning. This platform allows the training of Machine Learning models directly in the cloud. Without therefore having to install anything on our computer except a browser.

Figure 4.2 Google Colab environment logo

1.1.3. Tensorflow

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML and developers quickly build and deploy ML-powered applications.

Used version: 2.9.1

Figure 4.3 Tensorflow Platform logo

1.1.4. Keras

Keras is a high-level neural network API, written in Python and interfaceable with TensorFlow, CNTK, and Theano. It was developed with the objective of allowing rapid experimentation. Being able to go from idea to result with the shortest possible delay is the key to effective research.

Used version: 2.9.0

Figure 4.4 Keras API logo

1.1.5. Pandas

pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open-source data analysis/manipulation tool available in any language. It is already well on its way towards this goal.

Used version: 1.4.2

Figure 4.5 Pandas package logo

1.1.6. NumPy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.

Used version: 1.22.4

Figure 4.6 Numpy library logo

1.1.7. Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

Used version: 3.5.0

Figure 4.7 Matplotlib library logo

1.2. Dataset

The dataset was obtained and collected from the publicly available **Rfam** repository. This dataset contains 6320 samples in 13 well-balanced known ncRNA classes. The 13 classes are miRNA, 5S rRNA, 5.8S rRNA, ribozymes, CD-box, HACA-box, scaRNA, tRNA, Intron gpI, Intron gpII, IRES, leader, and riboswitch. Each class has 50 samples in the training dataset except IRES, which has 32 samples. The data is in Fasta format including class name and the RNA sequence itself. The testing set includes 632 ncRNA sequences that were kept out of the training process in order to test the performance of the model. The testing set includes 50 sequences belonging to each class.

Figure 4.8 Example of the benchmark dataset

2. Results

2.1. OneHot encoding example test

One hot encoding is a method of converting data to prepare it for an algorithm and get a better prediction, in our case it is employed to extract features from ncRNA sequences. With one-hot, we convert each categorical value into a new categorical column and assign a binary value of 1 or 0 to those columns. We used one-hot encoding to convert ncRNA sequences into a mapped binary representation.

We used this type of encoding as follows:

A —> `[[1],[0],[0],[0]]`

U —> `[[0],[0],[0],[1]]`

C —> `[[0],[1],[0],[0]]`

G —> `[[0],[0],[1],[0]]`

We test this method on the example sequence `'AACAUUCCG'`

```
[[1], [0], [0], [0]], [[1], [0], [0], [0]], [[0], [1], [0], [0]],
[[1], [0], [0], [0]], [[0], [0], [0], [1]], [[0], [0], [0], [1]],
[[0], [0], [0], [1]], [[0], [1], [0], [0]], [[0], [1], [0], [0]]
```

Figure 4.9 One-Hot encoding test result

2.2. Identification of Classes test

Class to ID and ID to Class:

```
{'5_8S_rRNA': 0, '5S_rRNA': 1,
{0: '5_8S_rRNA', 1: '5S_rRNA',
```

Figure 4.10 Identification of ncRNA's classes example

We identify the classes to facilitate reading data before training the model and then make the reverse function to clarify the classification results in the end.

2.3. Reading data files test

Here, we read the data files information from the dataset directory

```
Read Test_0
total data : 632
data count in each class : {'5_8S_rRNA': 50,
input  shape : (632, 250, 4, 1)
output shape : (632, 13)
================================
Read Train_0
total data : 5688
data count in each class : {'5_8S_rRNA': 450,
input  shape : (5688, 250, 4, 1)
output shape : (5688, 13)
================================
Read Test_1
total data : 632
data count in each class : {'5_8S_rRNA': 50,
input  shape : (632, 250, 4, 1)
output shape : (632, 13)
================================
Read Train_1
total data : 5688
```

Figure 4.11 Reading data example

2.4. Save npy data to folder numpy_data test

We load data using ReadData (test and train data), and then save it as .npy data as follows:

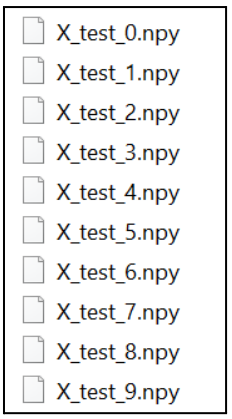


Figure 4.12 Data in npy format

2.5. Check shape of each dataset test

We check the shape of datasets in order to confirm that all the values of classes is well ordered and identified, here is an example of checking a file's dataset shape:

```
fileID = 0
Train File         : (5688, 750, 4, 1) (5688, 13)
Test  File         : (632, 750, 4, 1) (632, 13)
Test  classID = 00 : (50, 750, 4, 1) (50, 13)
Test  classID = 01 : (50, 750, 4, 1) (50, 13)
Test  classID = 02 : (50, 750, 4, 1) (50, 13)
Test  classID = 03 : (50, 750, 4, 1) (50, 13)
Test  classID = 04 : (50, 750, 4, 1) (50, 13)
Test  classID = 05 : (50, 750, 4, 1) (50, 13)
Test  classID = 06 : (32, 750, 4, 1) (32, 13)
Test  classID = 07 : (50, 750, 4, 1) (50, 13)
Test  classID = 08 : (50, 750, 4, 1) (50, 13)
Test  classID = 09 : (50, 750, 4, 1) (50, 13)
Test  classID = 10 : (50, 750, 4, 1) (50, 13)
Test  classID = 11 : (50, 750, 4, 1) (50, 13)
Test  classID = 12 : (50, 750, 4, 1) (50, 13)
```

Figure 4.13 Checking dataset example

2.6. Model configuration and creation

2.6.1. Model architecture

The model consists of two parts. The first part is used to extract the features from ncRNA sequences, while the second part is used to classify the ncRNA sequences.

First part: It functions as a feature extractor. To do this, it performs template matching by applying convolution filtering operations. The first layer filters the inputs with several convolution kernels and returns "feature maps", which are then normalized (with an activation function) and/or resized.

In our proposed model we repeat this phase two times (as we can see in the model configuration summary), and it is composed of:

- A first convolutional layer **Conv2D** with 32 filters of size 4x3, with a RuLU activation function.
- A MaxPooling layer **max_pooling2D** with a pool of size 2x1, which resizes the feature maps issued from the Conv2D layer.
- A dropout layer with a rate of 0.25 to prevent overfitting.
- A second convolutional layer **Conv2D_1** with 64 filters of size 3x1, with a RuLU activation function.
- A MaxPooling layer **max_pooling2D_1** with a pool of size 2x1, which resizes the feature maps issued from the Conv2D layer.
- A dropout layer with a rate of 0.25 to prevent overfitting.

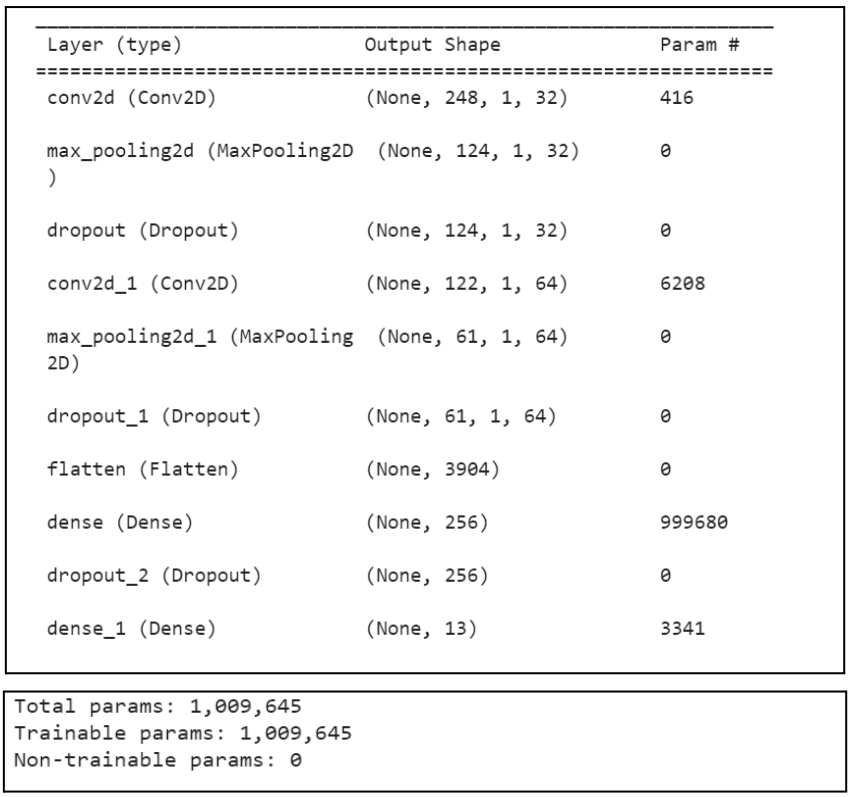
Finally, the values of the last feature maps are concatenated into a vector. This vector defines the output of the first block and the input of the second.

Second part: The input vector values are transformed to return a new vector to the output. This last vector contains as many elements as there are classes: element i represents the probability that the sequence belongs to class i. Each element is therefore between 0 and 1, and the sum of all is worth 1. These probabilities are calculated by the last layer of this block which uses a softmax function as an activation function.

In our model, this part is composed of:

- A **flatten** layer.
- A fully connected layer **dense** with 256 nodes, and a RelU activation function.
- A **dropout** layer with a rate of 0.25 to prevent overfitting.
- A final fully connected layer **dense_1** with 13 nodes (which is the number of the ncRNA classes), and a SoftMax activation function to make predictions.

In this model, we used **Adam** optimizer with a rate of 0.001. Categorical **cross-entropy** was employed for the loss function. The **batch-size** was set to 250. The number of **epochs** was set to 25.



Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 248, 1, 32)	416
max_pooling2d (MaxPooling2D)	(None, 124, 1, 32)	0
dropout (Dropout)	(None, 124, 1, 32)	0
conv2d_1 (Conv2D)	(None, 122, 1, 64)	6208
max_pooling2d_1 (MaxPooling2D)	(None, 61, 1, 64)	0
dropout_1 (Dropout)	(None, 61, 1, 64)	0
flatten (Flatten)	(None, 3904)	0
dense (Dense)	(None, 256)	999680
dropout_2 (Dropout)	(None, 256)	0
dense_1 (Dense)	(None, 13)	3341

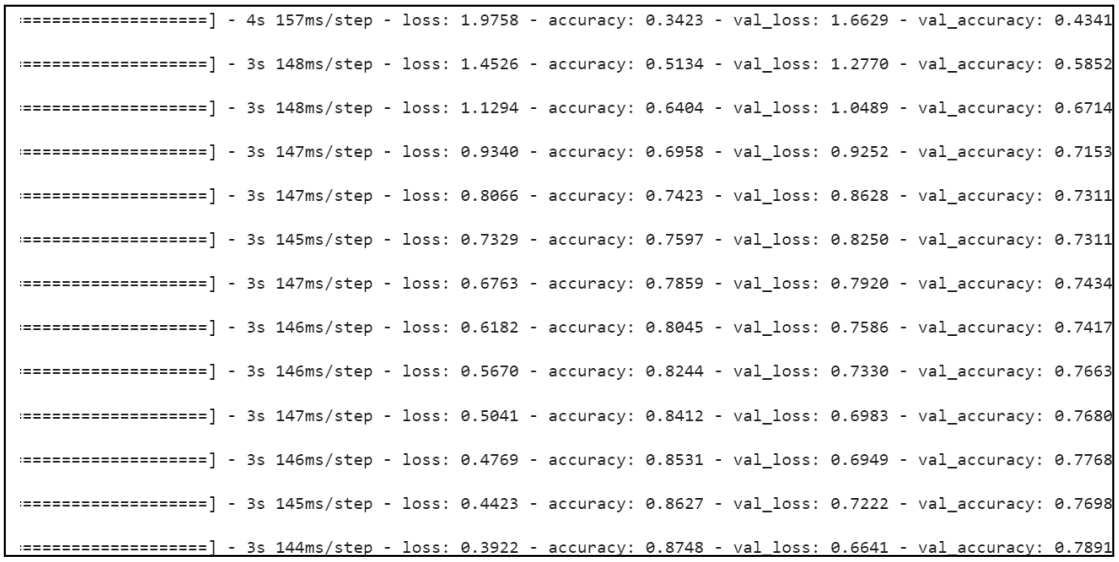
```
Total params: 1,009,645
Trainable params: 1,009,645
Non-trainable params: 0
```

Figure 4.14 Model architecture summary

In this section, we made several trials with different parameter values to establish the configuration that gave the best performances in terms of evaluation criteria. Then we compile it and we will see the outputs in the next section.

2.7. Model compilation test

Here is the output of the compilation section:



```
==========] - 4s 157ms/step - loss: 1.9758 - accuracy: 0.3423 - val_loss: 1.6629 - val_accuracy: 0.4341
==========] - 3s 148ms/step - loss: 1.4526 - accuracy: 0.5134 - val_loss: 1.2770 - val_accuracy: 0.5852
==========] - 3s 148ms/step - loss: 1.1294 - accuracy: 0.6404 - val_loss: 1.0489 - val_accuracy: 0.6714
==========] - 3s 147ms/step - loss: 0.9340 - accuracy: 0.6958 - val_loss: 0.9252 - val_accuracy: 0.7153
==========] - 3s 147ms/step - loss: 0.8066 - accuracy: 0.7423 - val_loss: 0.8628 - val_accuracy: 0.7311
==========] - 3s 145ms/step - loss: 0.7329 - accuracy: 0.7597 - val_loss: 0.8250 - val_accuracy: 0.7311
==========] - 3s 147ms/step - loss: 0.6763 - accuracy: 0.7859 - val_loss: 0.7920 - val_accuracy: 0.7434
==========] - 3s 146ms/step - loss: 0.6182 - accuracy: 0.8045 - val_loss: 0.7586 - val_accuracy: 0.7417
==========] - 3s 146ms/step - loss: 0.5670 - accuracy: 0.8244 - val_loss: 0.7330 - val_accuracy: 0.7663
==========] - 3s 147ms/step - loss: 0.5041 - accuracy: 0.8412 - val_loss: 0.6983 - val_accuracy: 0.7680
==========] - 3s 146ms/step - loss: 0.4769 - accuracy: 0.8531 - val_loss: 0.6949 - val_accuracy: 0.7768
==========] - 3s 145ms/step - loss: 0.4423 - accuracy: 0.8627 - val_loss: 0.7222 - val_accuracy: 0.7698
==========] - 3s 144ms/step - loss: 0.3922 - accuracy: 0.8748 - val_loss: 0.6641 - val_accuracy: 0.7891
```

Figure 4.15 Model compilation test

Here we will train the model in two steps, the first one is to test it on an example data file, then test it by training Ten models of it to extract the best features that we can use lately:

- Train the model on all the test files, **'Test_0.fasta'** as an example:

```
Test loss: 0.6208557486534119 / Test accuracy: 0.8148733973503113
```

Figure 4.16 Results of training the model on the file 'test_0.fasta'

- Training of 10 models (Ten-fold cross-validation):

```
Epoch 11/25
21/21 [==============================] - 3s 150ms/step - loss: 0.4715 - accuracy: 0.8549 - val_loss: 0.6699 - val_accuracy: 0.7838
Epoch 12/25
21/21 [==============================] - 3s 149ms/step - loss: 0.4441 - accuracy: 0.8566 - val_loss: 0.6452 - val_accuracy: 0.7926
Epoch 13/25
21/21 [==============================] - 3s 150ms/step - loss: 0.4114 - accuracy: 0.8720 - val_loss: 0.6209 - val_accuracy: 0.7996
Epoch 14/25
21/21 [==============================] - 3s 150ms/step - loss: 0.3741 - accuracy: 0.8828 - val_loss: 0.6518 - val_accuracy: 0.7944
Epoch 15/25
21/21 [==============================] - 3s 149ms/step - loss: 0.3510 - accuracy: 0.8877 - val_loss: 0.6141 - val_accuracy: 0.7944
Epoch 16/25
21/21 [==============================] - 3s 150ms/step - loss: 0.3146 - accuracy: 0.8988 - val_loss: 0.6186 - val_accuracy: 0.8084
Epoch 17/25
21/21 [==============================] - 3s 149ms/step - loss: 0.2787 - accuracy: 0.9131 - val_loss: 0.6195 - val_accuracy: 0.7979
Epoch 18/25
21/21 [==============================] - 3s 150ms/step - loss: 0.2632 - accuracy: 0.9156 - val_loss: 0.6070 - val_accuracy: 0.7996
Epoch 19/25
21/21 [==============================] - 3s 149ms/step - loss: 0.2465 - accuracy: 0.9258 - val_loss: 0.6287 - val_accuracy: 0.7926
Epoch 20/25
21/21 [==============================] - 3s 149ms/step - loss: 0.2282 - accuracy: 0.9244 - val_loss: 0.6291 - val_accuracy: 0.8137
Epoch 21/25
21/21 [==============================] - 3s 149ms/step - loss: 0.2093 - accuracy: 0.9316 - val_loss: 0.6004 - val_accuracy: 0.8120
Epoch 22/25
```

Figure 4.17 Results of training 10 models

2.7. Evaluation

We will evaluate this model in two scenarios:

2.7.1. The first scenario: Evaluation of each test file

In the Figure below, we present the evaluation of all the test data-set, according to their loss and accuracy scores, we observe then, that all the files give performant values (>70% of the accuracy, and <70% of the loss).

```
FileID: 0 / Test loss: 0.6359888911247253 / Test accuracy: 0.8354430198669434
FileID: 1 / Test loss: 0.5678181052207947 / Test accuracy: 0.8575949072837830
FileID: 2 / Test loss: 0.6423696279525757 / Test accuracy: 0.8370253443717957
FileID: 3 / Test loss: 0.5495265126228333 / Test accuracy: 0.8528481125831604
FileID: 4 / Test loss: 0.6555993556976318 / Test accuracy: 0.8354430198669434
FileID: 5 / Test loss: 0.5136015415191650 / Test accuracy: 0.8354430198669434
FileID: 6 / Test loss: 0.5462594032287598 / Test accuracy: 0.8370253443717957
FileID: 7 / Test loss: 0.6593993902206421 / Test accuracy: 0.8085442781448364
FileID: 8 / Test loss: 0.5476348400115967 / Test accuracy: 0.8496835231781006
FileID: 9 / Test loss: 0.5344201922416687 / Test accuracy: 0.8275316357612610
```

Figure 4.18 Evaluation of each test file scores

2.7.2. The Second scenario: Evaluation of each class from each file

Here, we go more deeply, by evaluating each ncRNA class of each test file from the test data-set.

The Figure below shows two examples of the files **'Test.0'** and **'Test.2'**, and we see that we get such satisfactory results:

```
FileID: 0 / ClassID: 00 / Test loss: 0.2756442129611969 / Test accuracy: 0.9200000166893005
FileID: 0 / ClassID: 01 / Test loss: 0.2975552380084991 / Test accuracy: 0.9200000166893005
FileID: 0 / ClassID: 02 / Test loss: 0.4806312322616577 / Test accuracy: 0.8600000143051147
FileID: 0 / ClassID: 03 / Test loss: 0.9168565273284912 / Test accuracy: 0.7799999713897705
FileID: 0 / ClassID: 04 / Test loss: 0.2757062017917633 / Test accuracy: 0.9200000166893005
```

```
FileID: 2 / ClassID: 02 / Test loss: 1.0309625864028931 / Test accuracy: 0.7200000286102295
FileID: 2 / ClassID: 03 / Test loss: 1.2705874443054199 / Test accuracy: 0.6800000071525574
FileID: 2 / ClassID: 04 / Test loss: 0.2624850273132324 / Test accuracy: 0.9200000166893005
FileID: 2 / ClassID: 05 / Test loss: 0.0678117126226425 / Test accuracy: 0.9800000190734863
FileID: 2 / ClassID: 06 / Test loss: 1.0767688751220703 / Test accuracy: 0.7187500000000000
```

Figure 4.19 Evaluation of each class from each file example

2.8. Average accuracy analysis

- The average accuracy of the model in general:

Model t1
0.837658

table 4.20 Average accuracy of the model

This Figure represents the average accuracy of the model, and as we observe, our model gives an accuracy score extremely equal to the ncRDense project that we talk about in the last chapter, which means more than 83 correct predictions out of 100 total examples.

- The average accuracies of each class (family):

Family	Model t1
5_8S_rRNA	0.952000
5S_rRNA	0.952000
CD-box	0.905333
HACA-box	0.859000
Intron_gpI	0.863600
Intron_gpII	0.885667
IRES	0.851554
leader	0.858859
miRNA	0.836319
riboswitch	0.836087
ribozyme	0.831534
scaRNA	0.822906
tRNA	0.832375

table 4.2 Average accuracies of each ncRNA family

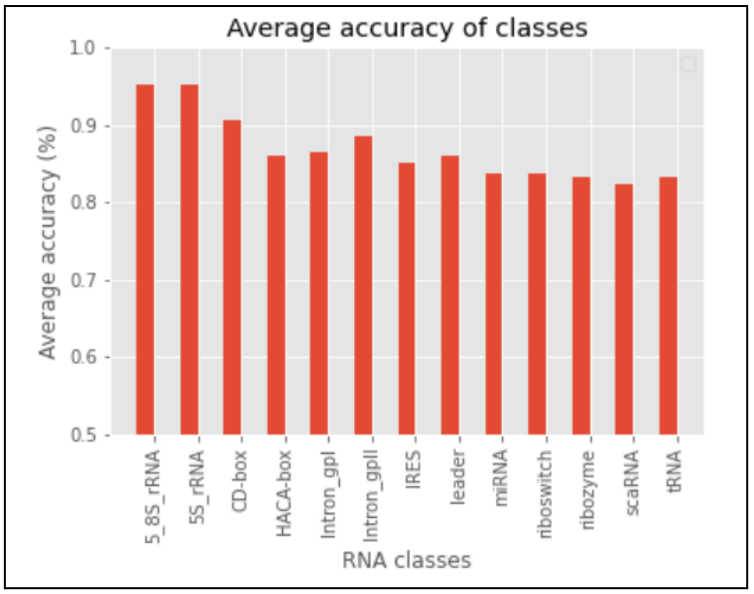


Figure 4.20 Average accuracies of each ncRNA family plot

In this Figure, we note that all the classes have a good performance, especially the '5_85_rRNA' and '5S_rRNA' families.

3. Conclusion

As we saw in this work, we covered the basics of ncRNA classification using deep learning, and we compared some of the works that realize this objective, then we tried to construct a supervised deep-learning-based model with a convolutional neural networks architecture, this architecture is based on extracting structural features of the ncRNAs sequences, and uses One-Hot feature encoding method, we tested our approach for the classification of 13 different ncRNA classes and we obtained classification scores, using the most common statistical measures, in particular, we reach an accuracy score of about 84%.

The limitation of this project is the poor trusted information about non-coding RNAs classes, also, the difficulties of backing up the missing data from the datasets.

For future work, we are working to train a classification model with much more ncRNA sequences, that also belong to some other well-studied ncRNA classes, in addition, we aim at creating a publicly available web service for the classification of unlabelled non-coding RNA sequences.

Bibliography

[1] Nature Portfolio. (2018, 04 13). *Molecular biology*. Nature. https://www.nature.com/subjects/molecular-biology

[2] Kamble, C., R. Chavan, R., & Kamble, V. (2022, 01 20). A Review on Amino Acids. *RRJoDDD*, 8.

[3] Schleif, R. F. (1993). Genetics and Molecular Biology. Johns Hopkins University Press.

[4] Gray, M. W. (2014, 08 24). Ribonucleic acid (RNA). *AccessScience*.

[5] Li, J., & Liu, C. (2019, 05 22). Coding or Noncoding, the Converging Concepts of RNAs. *Frontiers in Genetics*, 10.

[6] Tuvshinbayar, C., Dae, L., Hilal, T., & Kil, C. (2020, 08 01). ncRDeep: Non-coding RNA classification with convolutional neural network. (Computational Biology and Chemistry).

[7] Nature Education. (n.d.). *Genome*. Scitable by Nature Education. https://www.nature.com/scitable/definition/genome-43/

[8] Benbow R. M. (1992). Chromosome structures. *Science progress*, *76*(301-302 Pt 3-4), 425–450.

[9] Böhmer, D., Repiská, V., & Danišovič, Ľ. (2010). *INTRODUCTION TO MEDICAL AND MOLECULAR BIOLOGY*. Asklepios Bratislava.

[10] Haurowitz, F. and Koshland, . Daniel E. (2020, December 1). protein. Encyclopedia Britannica. https://www.britannica.com/science/protein

[11] *How do genes direct the production of proteins?* (2021, March 26). MedlinePlus. Retrieved June 22, 2022, from https://medlineplus.gov/genetics/understanding/howgeneswork/makingprotein/

[12] Robbins, R. J. (1995). *Molecular Biology Fundamentals*. Johns Hopkins University.

[13] Gauthier, J., Vincent, A. T., Steve J., S. J., & Derome, N. (2018). A brief history of bioinformatics. *Briefings in Bioinformatics*.

[14] Qazi, S., & Raza, K. (2021). Translational bioinformatics in healthcare: past, present, and future. In *Advances in ubiquitous sensing applications for healthcare* (p. 12). Nilanjan Dey.

[15] Purna Chandra, S., & Prafulla, B. (2017, 07). Bioinformatics: Applications and Issues. *CSI Communications*.

[16] *Application of Bioinformatics in various Fields*. (n.d.). Biotech. Retrieved June 22, 2022, from http://biotech.fyicenter.com/resource/Application_of_Bioinformatics_invarious_Fields.html

[17] Noorul, A., Annette, M., & Yi-Ping Phoebe, C. (2019, 05 01). Evaluation of deep learning in non-coding RNA. *Nature Machine Intelligence*.

[18] Binhua, T., Zixiang, P., Kang, Y., & Asif, K. (2019, 03 26). Recent Advances of Deep Learning in Bioinformatics and Computational Biology. *Frontiers in Genetics*, 10.

[19] Rangaraj, E. (2019, September 28). *A Quick Overview Of Machine Learning Tasks*. C# Corner. Retrieved June 23, 2022, from https://www.c-sharpcorner.com/article/a-quick-overview-of-machine-learning-tasks/

[20] *What is Unsupervised Learning?* (2020, September 21). IBM. Retrieved June 22, 2022, from https://www.ibm.com/cloud/learn/unsupervised-learning

[21] GeeksforGeeks. (2022, 06 03). *Reinforcement learning*. GeeksforGeeks. https://www.geeksforgeeks.org/what-is-reinforcement-learning/

[22] Min, S., Lee, B., & Yoon, S. (2017, 10 05). Deep learning in bioinformatics. *Briefings in Bioinformatics*.

[23] Fiannaca, A., La Rosa, M., La Paglia, L., Rizzo, R., & Urso, A. (2017). nRC: non-coding RNA Classifier based on structural features. *BioData Mining*.

[24] Chantsalnyam, T., Siraj, A., & Tayara, H. (2021, 10 05). ncRDense: A novel computational approach for classification of non-coding RNA family by deep learning. *Genomics*, 8.

[25] Panwar, B., Arora, A., & Raghava, G. P. (2014). Prediction and classification of ncRNAs using structural information. *BMC Genomics*, 13.

[26] Rfam. (n.d.). *Rfam Database*. http://www.docs.rfam.org/

[27] NumPy. (n.d.). *NumPy Library*. https://numpy.org/

[28] Pere, C. (2020, 01 17). What are Loss Functions? *Medium*.

[29] (n.d.). Keras: the Python deep learning API. Retrieved June 22, 2022, from https://keras.io/

[30] Gupta, A. (2021, October 7). A Comprehensive Guide on Deep Learning Optimizers. *Analytics Vidhya*. https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/