mirBoT: A mircoRNA Sequence Prediction Tool From Targeted RNA Sequence Segment based on CNN And LSTMs Stacked in Seq2seq Architecture

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTER OF TECHNOLOGY
IN
BIOINFORMATICS

Submitted by:

Rajkumar Chakraborty

2K16/BIO/03

Under the Supervision of

Dr. YASHA HASIJA

DEPARTMENT OF BIOTECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

JUNE, 2018

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, Rajkumar Chakraborty, Roll No. 2K16/BIo/03 student of M.Tech (Bioinformatics), hereby declare that the project dissertation titled "**mirBoT: A mircoRNA Sequence Prediction Tool From Targeted RNA Sequence Segment based on CNN And LSTMs Stacked in Seq2seq Architecture**" which is submitted by me to the Department of Biotechnology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi **RAJKUMAR CHAKRABORTY**

Date:

DEPARTMENT OF BIOTECHNOLOGY
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CERTIFICATE

I hereby certify that the project dissertation titled "**mirBoT: A mircoRNA Sequence Prediction Tool From Targeted RNA Sequence Segment based on CNN And LSTMs Stacked in Seq2seq Architecture**" which is submitted by Rajkumar Chakraborty, Roll No. 2K16/BIo/03, Department of Biotechnology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology in Bioinformtics, is a record of a project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for ant Degree or Diploma to this University or elsewhere.

Date:
Place: Delhi

(Dr. YASHA HASIJA)
SUPERVISOR
Assistant Professor
Department of Biotechnology

(Prof. Jai Gopal Sharma)
Head of Department
Department of Biotechnology
Delhi Technological University

Delhi Technological University

ABSTRACT

MicroRNAs (miRNAs) are considered as very important cellular constituents that control gene expression at the post-transcriptional level and have fascinated much scientific attention. These small non-coding RNAs play important roles by binding to their target genes and are also known to be associated with various diseases. Computational methods that predict miRNA target sites generally use one or more characteristics such as sequence complementation, thermodynamic stability, evolutionary conservation among species and accessibility. In recent years, deep recurrent neural networks (RNNs) have allowed researchers to tackle a variety of machine learning problems in the domain of natural language processing. Less work has been done with RNNs on what is perhaps the most natural language: the genome, a sequence of four letters (A, C, G, T). We downloaded 19,000 experimentally validated miRNA-target pairs from TarBaseV8, the corresponding mRNA sequences were collected from the ensemble genome browser , and the miRNA sequences from the miRBase. And a model based on RNN, LSTM and seq2seq architecture was used for the prediction of miRNA sequence. And also an important feature, surface-area assecibility at binding site of miRNA at the targated mRNA was also taken into account. After training for 100 epochs, we achieved an accuracy of 0.8 with Validation Loss = 0.0887. We verified our model using experimentally validated data from miDerma, a manually curated database of miRNAs associated with Dermatological Disorders. our model was able to predict on average 72% of microRNAs for each genes from the list of 200 randomly selected genes associated with dermatological disorders. We belive that the successful prediction miRNA may help the scientific community in the fields of therapeutics, biomarker selection etc

ACKNOWLEDGEMENT

I am most thankful to my family for constantly encouraging me and giving me time and unconditional support while pursuing this research.

I wish to express my deep sense of gratitude and indebtedness to Dr. Yasha Hasija, Assistant Professor, Department of Biotechnology, DTU, Delhi; for introducing the present topic and for her inspiring guidance, constructive and valuable suggestion throughout this work. I am heartily thankful for her guidance and support during my project. Her able knowledge and expert supervision with unswerving patience fathered my work at every stage, for without her warm affection and encouragement, the fulfilment of the task would have been very difficult. I would also like to extend my heartfelt gratitude towards all the members of the Department of Biotechnology for gratuitously helping me in the successful completion of the project. I am genuinely appreciative of all my friends for their suggestions and moral support during my work

I also convey my heartfelt gratitude to all the research scholars of the Genome Informatics Lab for their valuable suggestions and helpful discussions throughout the course of this research work.

RAJKUMAR CHAKRABORTY

(2K16/BIO/03)

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

DNA	Deoxyribonucleic acid
RNA	Ribonucleic acid
A	Adenine
T	Thymine
G	Guanine
C	Cytosine
miRNA	Mirco RNA
mRNA	Messenger RNA
Pri-miRNA	Primary Micro RNA
ANN	Artifitial Neural Networks
CNN	Convolutional Neural Networks
RNN	Recurrent Neural Networks
LSTM	Long Short Term Memory
GRU	Gated Recurrent Unit
RISC	RNA-induced silencing complex
UTR	Un-Translated Region
DGCR8	DiGeorge syndrome chromosomal [or critical] region 8
TRBP	TAR RNA binding protein
NGS	Next Genaration Dequenceing
IP	Immunoprecipitation
HITS-CLIP	High-throughput sequencing of RNA isolated by crosslinking immunoprecipitation
CPU	Central Prossecing Unit
GPU	Graphics Processing Unit

CHAPTER 1 INTRODUCTION

The human genome encodes for over 2200 microRNAs (microRNAs), which are mostly 28bp long non-coding RNA molecules which plays an important role in regulating gene expression post-transcriptionally. As one microRNA can target multiple gene transcripts, microRNAs are known to involve in major mechanism to regulate gene expression and mRNA translation. Computationaly anticipating microRNA targets is a basic fundamental step in finding microRNA-mRNA target association for lab approval. The current methods for microRNA target predictions incorporate a scope of various computational methodologies, from the demonstrating of physical association algorithms to the application of machine learning algorithms.

Also it has been seen that neural networks are pretty successful for generating models of predicting biological functions, sequences, classification and other task. In ANNs particularly CNNs and RNNs draws the attention of scientific community for their robustness in extracting feature and generation sequences. It is now a well known fact that CNNs are used of extracting features from datasets and are highly useful in image classification and image annotation. Today we cannot imagine any image classification model without using CNNs. on the other hand we are having RNNs which are very comfortable with sequential data whether they are time series data or sequence of string. RNNs are mostly used in natural language processing where they can be trained for classification of phrases, generating new phrases etc. Next we have seq2seq architecture which are used in chat-bots, they are consist of special arrangement of RNNs or LSTMs which can be trained for text generation can be used as chat bots, machine level translation or image annotation.

So in this work we particularly tried to bring the natural language processing models to process core language of nature i.e. A, T, G, C. Here we tried to train a chat bot model to generate microRNA sequence from mRNA sequence and named it miR-Bot while preserving the biological notions. We download 22,600 experimentally validated microRNA-target pairs from TarBaseV8, The corresponding mRNA sequences are collected from the ensemble , and the microRNA sequences from the miRBase. Then a model base on CNN, LSTM and seq2seq

architecture is trained on these datasets for prediction of microRNA sequences based on mRNA sequences. During training our model acrive an accuracy of 80%. After training the model we used RNAplfold from RNA-Vienna Package to find site accessibility of mRNAs. Site accessibility is a measure of the ease with which a microRNA can locate and hybridize with an mRNA target. mRNA assumes a secondary structure which can interfere with a microRNA's ability to bind to a target site. MicroRNA:mRNA hybridization involves a two-step process in which a microRNA binds first to a short accessible region of the mRNA. The mRNA secondary structure then unfolds as the microRNA completes binding to a target. Therefore, to assess the likelihood that an mRNA is the target of a microRNA, the predicted amount of energy required to make a site accessible to a microRNA should be evaluated. Finally we validated our model using experimentally verified microRNA and RNA pairs involved in skin diseases we were retrieved from our in house developed database miDerma. Here our model was able to predict on average 72% of microRNA from mRNA in each cases correctly. Hence we propose “mirBoT: A MircoRNA sequence prediction tool from RNA sequence base on CNNs, LSTMs and seq2seq architecture.

CHAPTER 2 REVIEW OF LITERATURE

2.1 MircoRna

2.1.1 Discovery and function of microRNAs

The first microRNA was revealed over 30 years ago in the nematode Caenorhabditis elegans as the developmental regulator lin-4[1]. Initially believed to be a typical protein coding gene, till Ruvkun and Ambros labs made the surprising discovery that lin-4 did not code for a protein but instead encode a 22 nucleotide regulatory RNA[2], [3]. They confirmed that the lin-4 RNA can base pair with the mRNA of another gene in the C. elegans developmental network, lin-14, and regulate the production of the LIN-14 protein [3]. Discovery of this microRNA would have had little significance other than C. elegans research community, if the second microRNA, let-7, not been discovered[4]. let-7 is conserved in many organisms, including humans, signifying that this class of small regulatory RNAs has a more general role in biology[5]. The next spectacular development, occurred around the same time, by the discovery of the RNAi pathway; specifically, the ~ 21 nucleotide RNA's role in the silencing machiner[6]. These two pathways have since been shown to be different arms of the same gene silencing pathway[7]. Later, many thousands of microRNAs have been revealed in many organisms, and there are currently 2588 annotated microRNAs in the human genome[8]. Since each microRNA can regulate the expression of hundreds of target mRNA, the microRNA pathway as a whole is a critical machinery for gene expression control[9].

2.1.2 The microRNA biogenesis pathway

During the biosynthesis of all microRNAs they undergo a series of steps that convert the primary microRNA transcript into the active, ~ 22 nucleotide mature microRNA . The mature microRNA with the RNA induced silencing complex (RISC) target mRNAs, leading to translational repression and target mRNA degradation. In this section we will cover the microRNA biogenesis pathway that is followed by most microRNA families for the maturation. For ease we will limit

our focus on the mammalian enzyme mechanisms. Although there are several exceptions from this canonical pathway have been described for unique microRNA families those are not cover under this topic.

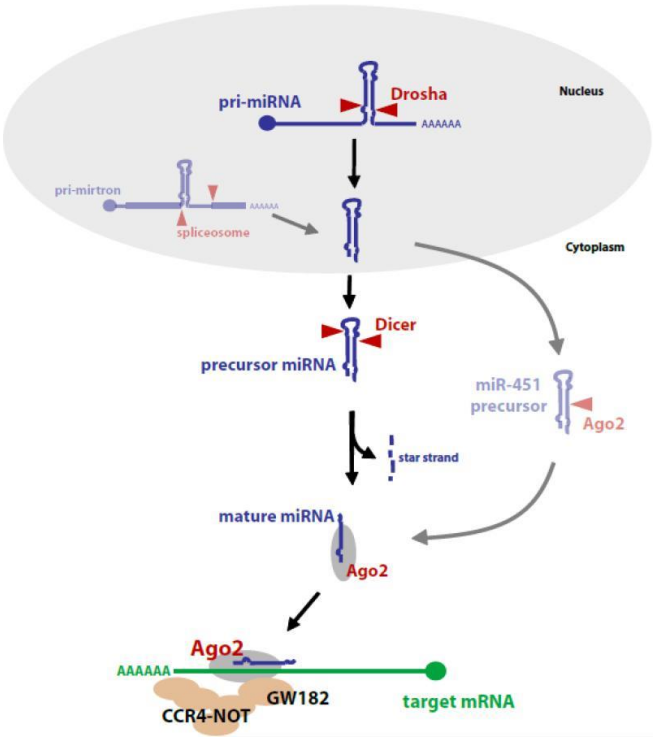


Figure 1: The microRNA biogenesis pathway

2.1.2.1 Transcription

microRNA genes are present in all parts of the genome [10]. Many microRNA genes are noncoding and whose only transcriptional product is the microRNA. In other cases the microRNA is located within an intron or untranslated region (UTR) of a protein coding gene. Typically the defining feature of all microRNA genes is the stem– loop precursor RNA structure, with one (or sometimes both) strands of the stem the source of the mature microRNA [9]. In some cases like of MCM7 there are three clustered of microRNA stem–loops, each leads to a distinct mature microRNA with a exclusive targeted set[11]. microRNAs are not necessary to present within coding exons, as deletion of the microRNA would lead to loss of the protein coding transcript.

Transcription of the microRNA coding gene is done by RNA polymerase II results leads to the pri-microRNA, or primary microRNA transcript [12]. The primary-microRNA is typically spliced, capped, and polyadenylated like a protein coding mRNAs [13]. Although most microRNA promoters have not been studied in detail, the few examples characterize as similar in structure to protein coding gene promoters, including control histone marks and elements [14].

2.1.2.2 Processing by Drosha and Dicer

The primary-microRNA goes through processing steps by two endonuclease before it becomes a mature, active microRNA [15]. The first processing(cleavage) step occurs during transcription of the primary-microRNA by the enzyme Drosha [16]. The RNA binding protein DGCR8 is required for cleavage of the primary-microRNA and associated with Drosha [17]. After cleavage by Drosha, the stem–loop precursor from flanking primary-microRNA transcript sequences releases. The precursor is carried out of the nucleus in a Ran-GTPase dependent manner by Exportin5 [18]. The second processing step occurs in the cytoplasm. The 2nd endonuclease Dicer cleaves the loop region of the precursor releasing the mature microRNA [19]. Like Drosha, Dicer is also accompanied with an RNA binding protein, TRBP [20]. The result of the reaction carried by Dicer, is a duplex RNA of approximately 21 nucleotides length. one strand of the duplex is loaded into RISC as a mature microRNA. The other strand typically degraded. An asterisk is appended to the microRNA name to designate this degraded strand, e.g., miR-125*. However, in some cases both strands of the duplex may be loaded into RISC at similar frequencies. In this case, the strand from the 5′ end of the stem–loop is termed “5p” and the 3′ strand the “3p”. While RISC loading may prefer integration of one strand, current next generation sequencing (NGS) efforts have revealed a small fraction of star strand loaded for essentially all microRNA families [21]. So, some microRNAs show different strand usage depending on cell type or biological state [22]. For these reasons, 5p/3p naming schemes are being widely used rather than the arbitrary mature/star nomenclature.

The specific nucleotide areas of Drosha and Dicer cleavage are generally barely defined, prompting full grown microRNAs with unmistakably defined terminal closures. Some

microRNAs, be that as it may, have heterogenous cleavage destinations, prompting numerous "isomiRs" of the develop microRNA[23]. This heterogeneity could prompt differential target constraint and particular natural exercises. IsomiR inclination can change by cell write and isomiR exchanging has been recognized in infection[24]. The real mechanism for isomiR regulation is unclear.

2.1.2.3 RISC loading and target repression

A definitive destiny of the microRNA is to be consolidated into RISC (or miRISC). The correct arrangement of this protein complex isn't clear however contains the fundamental protein Argonaute, of which four relatives have been identified in humans (Ago1– 4) [25]. After Dicer cleavage, the microRNA duplex is stacked into RISC and the star strand is evacuated by some of a few conceivable actions. on the off chance that the microRNA duplex has complementarity in the central region, the star strand can be cleaved and removed by Ago2 and further degrade by the nuclease complex C3Po [26]. This is the system for RISC stacking for the related siRNA pathway. Most microRNA duplexes, in any case, need central complementarity and subsequently can't take part in star strand cleavage. These microRNA duplexes depend on strand loosening up, and a few helicases have been portrayed to have this activity [27].

Argonaute specifically ties with the develop microRNA and looks for target mRNAs that have complementarity to the microRNA. Specifically, nucleotides 2– 7 of the microRNA, named the "seed" region, are essential for target binding[28]. The 3′ end of the microRNA also has a role in target recognition, and complimentary matched targets have been found [29]. If complementarity found in the seed region of the microRNA (nucleotides 9– 11) at that point the mRNA target can be degrade by means of the endonuclease movement of Ago2 [30]. Most microRNA target binding sites in human, do not have this match and are not straightforwardly degraded by Ago2. So, Argonaute is enlisted to a complex containing GW182 (TNRC6A/B/C) inside cytoplasmic P bodies where translational constraint occurs. The CCR4-NoT deadenylase complex is attached to RISC and this encourages evacuation of the poly(A) tail and inevitable debasement of the mRNA target[31].

2.1.3 microRNA target identification

Identifying target sets for microRNAs is essential for a various reasons. For scientists, finding the target set of a microRNA is critical to uncover its its critical role in biology. For biologists creating microRNA therapeutics, validated target binding sites give the best biomarker(s) for assurance of the efficacy of a microRNA enhencer or inhibitor. The identification of microRNA targets can be taken after three general methodologies: bioinformatic target prediction, biochemical isolation of microRNA/mRNA complexes, and transcriptomic/ proteomic analysis. The three methodologies will be briefly condensed.

2.1.3.1 Bioinformatic target prediction

As it has been state that microRNAs base match with target mRNAs utilizing standard Watson–Crick rules, this reality should make bioinformatic target identification reliable. But, the most vital determinant of target restricting is the seed arrangement of the microRNA which is just 6 nucleotides long. This will prompt an extraordinary number of competitor targets, a significant number of which are false negatives. In this way, all bioinformatic target expectation calculations utilize extra factors to enhance exactness[32].

Since the best described targets are in mRNA 3′ UTRs, numerous calculations restrain target sites to this area. Different components are utilized, including sequence conservation,flanking sequence determinants, flanking arrangement determinants, compensatory matching outside the seed area and target site accessibility. Current methodologies have additionally utilized machine learning calculations that consolidate approved targets sets as learning sets[32]. Various tools have been developed, with TargetScan, miRanda, and PicTar maybe the most prominent[33]. When all is said in done, bioinformatic approaches are a decent beginning stage for microRNA investigation and numerous exploration labs make utilization of them [34].

2.1.3.2 Biochemical target identification

A few related methodologies have utilized physical relationship of microRNA/RISC edifices with target mRNAs to separate and recognize targets. These depend on immunoprecipitation of RISC utilizing anti-Argonaute antibodies, with or without earlier RNA crosslinking, and definition of bound target RNAs by microarray or NGS profiling[35]. Crosslinking of IP before cell lysis is mostly preferred since artifactual RNA hybridization have been seen at the time of cell lysis [36]. An elective approach is extraction of specific biotinylated microRNAs tand than target identification[37]. This has the upside of catching focuses of a solitary known microRNA, however requires ectopic introduction of the biotinylated microRNA. While these physical methodologies have been effectively used to define target mRNA of microRNA complexes, not all bounded mRNA targets might be repressed. Studies have shown that some target binding sites are in coding regions of mRNAs, and are bound to Argonaute however are not degrading targets [35]. Like all methodologies, approval of individual targets is fundamental.

2.1.3.3 omics-based strategies for target identification

The third broad way to deal with target identification is a proteomic or transcriptomic investigation of cells/tissues in the nearness and nonappearance of a microRNA. Quantitative proteomic investigation straightforwardly measures the impact of a microRNA on protein creation, and is perhaps more reflective of the genuine target set, but at the same time is in fact testing. Since most microRNA targets have lessened mRNA consistent state levels, the more straightforward transcriptome research can be performed [38]. This should be possible by microarray profiling and a few investigation devices are accessible[39]. A case of this approach is the identification of focuses of the neutrophil-specific microRNA miR-223[40]. Neutrophils were disengaged from wild sort and miR-223 knockout mice. Microarray and quantitative mass spectrometry were performed, and contrasts in mRNA and protein content was utilized to define focuses of miR-223. It ought to be noticed that the competitor target sets will likewise contain downstream auxiliary targets, requiring approval of individual targets.microRNA detection methods

microRNA recognition strategies can be partitioned into two general categorie [41] . Disclosure strategies are intended to profile the outflow of numerous microRNAs without a moment's delay, frequently with high throughput. These methodologies are to a great extent focused on microarray hybridization and NGS profiling. The last is particularly great since it is conceivable to portray novel microRNAs, while most different techniques are restricted to location of known microRNA successions. There are a few NGS stages accessible however all start with the development of format libraries. RNA connectors are ligated to the two finishes of little RNA portions and the objectives are RT-PCR amplified with preliminaries coordinated at the connectors. This procedure amplifies all RNAs in the coveted size range. The libraries would then be able to be sequenced on a few instruments, with the Illumina stages presumably the most well-known. on the other hand, single particle instruments are accessible that keep away from the PCR amplification step inside and out[42]. Since current instruments are fit for 200 million or more peruses per library run, it is conceivable to multiplex 48 libraries (or more) in a solitary run and still accomplish sufficient grouping read profundity[42]. Sequence peruses are checked and quantitative articulation profiles are got. As stated above, novel microRNAs and other little RNA species can be identified. While NGS based profiling has clear points of interest and is turning into the default innovation, nucleotide predispositions brought about amid ligation steps has been watched[43]. So, profiling method a validation step is necessary .

While NGS stages are prepared to do high throughput profiling of the whole microRNA populace, most clinical demonstrative methodologies depend on fast investigation of a little quality mark set. In this way, RT-PCR and Nanostring are the focal point of current analytic stages[44]. Nanostring is a solitary particle hybridization technique that permits quantitation of ~ 500 targets, either mRNA or microRNA, in a quick exploratory run. A case of a LDT microRNA symptomatic is the pancreatic growth test from Asuragen. This RT-PCR based test utilizes a 7 microRNA mark to separate between pancreatic ductal adenocarcinoma and amiable tissue.

2.2 MircoRna Target Prediction

2.2.1 Common features of microRNA target prediction tools

There are mainly four frequently used features for microRNA target prediction algorithms: seed match, site accessibility, free energy and conservation. These will be described in the following sections.

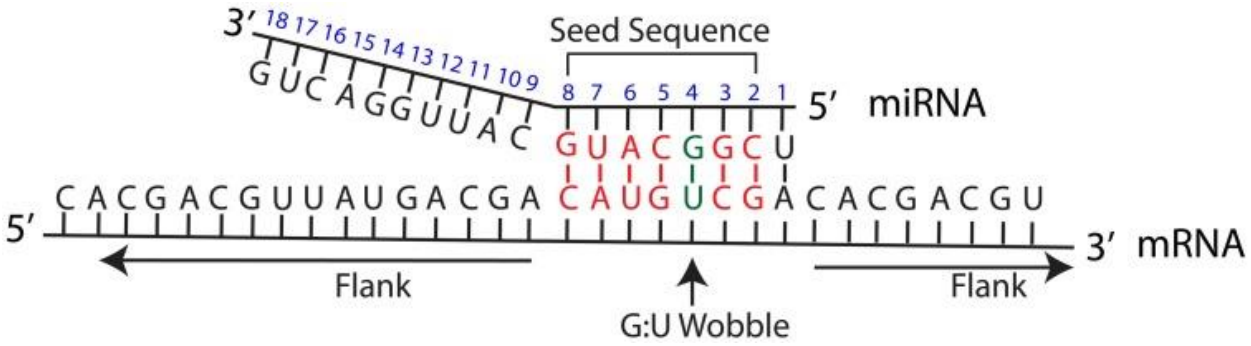


Figure 2: microRNA:mRNA target interaction

2.2.1.1 Seed match

The seed sequence arrangement of a microRNA is characterized as the initial 2– 8 nucleotides beginning at the 5' end and checking toward the 3' end[45] (Figure 1). The seed sequence arrangement of a microRNA is characterized as the initial 2– 8 nucleotides beginning at the 5' end and checking toward the 3' end [45] (Figure 1). For most algorithms, a seed complimentary in accordance to Watson-Crick (WC) pairing rules between a microRNA and its target site is require. A Watson-Crick match between a microRNA and mRNA nucleotide happens when adenosine guanine (G) sets with cytosine (C) and adenosine (A) sets with uracil (U).

There are a few kinds of seed coordinates that can be thought about relying upon the calculation. The accompanying sorts are the fundamental kinds of seed matches are [45]–[48]:

1. 6mer: A perfect WC match between the microRNA seed and mRNA for six nucleotides.

2. 7mer-m8: A perfect WC match from nucleotides 2–8 of the microRNA seed.

3. 7mer-A1: A perfect WC match from nucleotides 2–7 of the microRNA seed in addition to an A across from the microRNA nucleotide 1.

4. 8mer: A perfect WC match from nucleotides 2–8 of the microRNA seed in addition to an A across from the microRNA nucleotide 1.

2.2.1.2 Conservation

Conservation is to maintain same pattern of sequence across species. Analysis of conservation may concentrated on regions of the 3' UTR, the 5' UTR, the microRNAs, or any amalgamation of the three. Generally it is seen that the microRNA seed region has the higher conservation than in the non-seed region[45]. In a small proportion of microRNA-mRNA target interactions , there is conserved pairing at the 3' end of the microRNA which can reimburse for seed mismatches, and these sites are called 3' compensatory sites [49]. In the context of predicting microRNA targets in 3 UTRs, conservation analysis may provide evidence that a predicted microRNA target is functional because it is being selected for. Also, there is a increasing interest in conservation analysis of the genomic regions flanking the microRNA gene and microRNA target genes. As examples, conservation analysis has been applied to the promoter regions of microRNAs and their target genes [50], and to the colocalization of independently transcribed microRNAs and flanking protein coing genes [51]. Thus, the role of conservation in microRNA target prediction is broad and analysis may focus on regions in the 3 UTR, the 5 UTR, the microRNA, or any combination of the three. In general, there is higher conservation in the microRNA seed region than in the non-seed region[45]. In a small proportion of microRNA: mRNA target interactions, there is conserved pairing at the 3' end of the microRNA which can compensate for seed mismatches, and these sites are called 3' compensatory sites [49]. With regards to anticipating microRNA focuses in 3 UTRs, preservation examination may give confirm that an anticipated microRNA target is utilitarian since it is being chosen for. Also, there is expanding enthusiasm for conserva-tion examination of the genomic districts flanking the microRNA quality and microRNA target qualities. As illustrations, preservation investigation has

been connected to the promoter areas of microRNAs and their objective qualities [50], and to the co-confinement of independently interpreted microRNAs and flanking protein coding genes [51]. In this way, the part of conservation in microRNA target prediction is widely used

2.2.1.3 Free energy

Free vitality (or Gibbs free vitality) can be utilized as a measure of the steadiness of an biological framework. In the event that the binding of a microRNA to a candidate target mRNA is anticipated to be steady, it is viewed as more prone to be a genuine target of the microRNA. Given the trouble in estimating free vitality specifically, typically the change in free energy during a reaction is considered (ΔG). Since responses with a negative ΔG have less energy to respond later on, they result in frameworks with expanded solidness. By anticipating how the microRNA and its competitor target hybridize, areas of high and low free energy can be deduced and the general ΔG can be utilized as a pointer of how firmly bound they are [52].

2.2.1.4 Site accessibility

Site accessibility is a measure of the ease with which a microRNA can locate and hybridize with an mRNA target. Following transcription , mRNA assumes a secondary structure [53] which can interfere with a microRNA's ability to bind to a target site. MicroRNA mRNA hybridization involves a two-step process in which a microRNA binds first to a short accessible region of the mRNA. The mRNA secondary structure then unfolds as the microRNA completes binding to a target[54]. Therefore, to assess the likelihood that an mRNA is the target of a microRNA, the predicted amount of energy required to make a site accessible to a microRNA should be evaluated .

2.2.2 Less common features of microRNA target prediction tools

The highlights talked about above are those most ordinarily joined into microRNA target forecast apparatuses. As new advances are made in the portrayal of microRNA mRNA target associations, extra highlights are consolidated. These may be utilized to foresee the viability of

the objective or specifically fused into the objective forecast itself. Target-site wealth is a measure of what number of target locales happen in a 3 UTR[55]. Local AU content alludes to the grouping of An and U nucleotides flanking the relating seed district of the microRNA[56]. GU wobble in the seed coordinate alludes to the stipend of a G blending with a U rather than a C. 3 compensatory blending alludes to base combine coordinating with microRNA nucleotides 12–17. Seed matching solidness is the ascertained free vitality of the anticipated duplex. Position commitment dissects the situation of the objective site inside the mRNA. Machine-learning approaches utilize preparing information to build up a model of microRNA targets, and afterward utilize the model as a major aspect of the microRNA-expectation process. Machine-learning procedures are probably going to utilize more highlights in their expectations since they can be prepared to decide the prescient intensity of each component on positive and negative datasets. A machine-learning approach utilized by a few of these instruments is support vector machines (SVM). Apparatuses that utilization SVM are noted.

2.2.3 Review of commonly used microRNA target prediction tools

In this section, we outline 10 popular microRNA target prediction tools, using the characteristics previously described. A summary table comparing these tools is provided in the Comparison of microRNA Target Prediction Tools section.

Table 1: Summary table of microRNA target prediction tools

FEATURES USED IN microRNA TARGET PREDICTION							
Tool name	**Seed match**	**Conservation**	**Free energy**	**Site accessibility**	**Target-site abundance**	**Machine learning**	**References**
miRanda	X	X	X				[57]
miRanda-mirSVR	X	X	X	X		X	[56]
TargetScan	X	X					[55]
DIANA-	X	X	X	X	X	X	[58]

microT-CDS							
MirTarget2	X	X	X	X		X	[59]
RNA22-GUI	X		X				[60]
TargetMiner	X	X	X	X	X	X	[61]
SVMicrO	X	X	X	X	X	X	[62]
PITA	X	X	X	X	X		[63]
RNAhybrid	X		X		X		[64]

2.3 Neural Networks

2.3.1 Machine Learning General

Machine learning methods are general-purpose approaches to learn functional relationships from data without the need to define them a priori [65]. In computational biology, their allure is the capacity to infer prescient models without a requirement for solid presumptions about basic instruments, which are much of the time obscure or inadequately characterized. As a for example, the most exact forecast of quality articulation levels is presently produced using a wide arrangement of epigenetic highlights utilizing scanty straight models[66] or random forests ; how the selected features determine the transcript levels remains an active research topic. Predictions in genomics [67], proteomics [68], metabolomics [69] or sensitivity to compounds [70] all rely on machine learning approaches as a key ingredient.

The greater part of these applications can be depicted inside the standard machine learning work process, which includes four stages: information cleaning and pre-processing, highlight extraction, demonstrate fitting and assessment (Figure 3). It is standard to signify one information test, including all covariates and highlights as info x (more often than not a vector of numbers), and mark it with its reaction variable or yield esteem y (as a rule single number) when accessible..

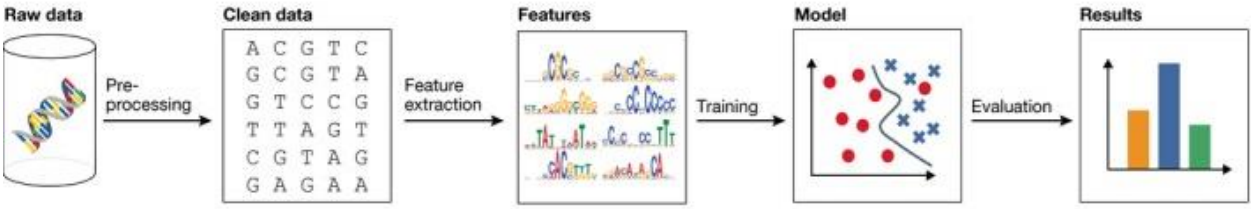


Figure 3The classical machine learning workflow can be broken down into four steps: data pre-processing, feature extraction, model learning and model evaluation.

A supervised machine learning model aims to learn a function f(x) = y from a list of training pairs (x 1,y 1), (x 2,y 2), … for which data are recorded. one typical application in biology is to predict the viability of a cancer cell line when exposed to a chosen drug [70]. The input features (x) would capture somatic sequence variants of the cell line, chemical make-up of the drug and its concentration, which together with the measured viability (output label y) can be used to train a support vector machine, a random forest classifier or a related method (functional relationship f). Given a new cell line (unlabelled data sample x*) in the future, the learnt function predicts its survival (output label y*) by calculating f(x*), even if f resembles more of a black box, and its inner workings of why particular mutation combinations influence cell growth are not easily interpreted. Both regression (where y is a real number) and classification (where y is a categorical class label) can be viewed in this way. As a counterpart, unsupervised machine learning approaches aim to discover patterns from the data samples x themselves, without the need for output labels y. Methods such as clustering, principal component analysis and outlier detection are typical examples of unsupervised models applied to biological data.

The inputs x, calculated from the raw data, represent what the model "sees about the world", and their choice is highly problem-specific. Deriving most informative features is essential for performance, but the process can be labour-intensive and requires domain knowledge. This bottleneck is especially limiting for high-dimensional data; even computational feature selection methods do not scale to assess the utility of the vast number of possible input combinations. A major recent advance in machine learning is automating this critical step by learning a suitable

representation of the data with deep artificial neural networks [71] . Briefly, a deep neural network takes the raw data at the lowest (input) layer and transforms them into increasingly abstract feature representations by successively combining outputs from the preceding layer in a data-driven manner, encapsulating highly complicated functions in the process. Deep learning is now one of the most active fields in machine learning and has been shown to improve performance in image and speech recognition [72], natural language understanding [73], and most recently, in computational biology [74].

2.3.2 Artificial Neural Networks Details

An artificial neural network, initially inspired by neural networks in the brain, consists of layers of interconnected compute units (neurons). The depth of a neural network corresponds to the number of hidden layers, and the width to the maximum number of neurons in one of its layers. As it became possible to train networks with larger numbers of hidden layers, artificial neural networks were rebranded to “deep networks”.

In the canonical configuration, the network receives data in an input layer, which are then transformed in a nonlinear way through multiple hidden layers, before final outputs are computed in the output layer (panel A). Neurons in a hidden or output layer are connected to all neurons of the previous layer. Each neuron computes a weighted sum of its inputs and applies a nonlinear activation function to calculate its output $f(x)$ (panel B). The most popular activation function is the rectified linear unit (ReLU; panel B) that thresholds negative signals to 0 and passes through positive signal. This type of activation function allows faster learning compared to alternatives (e.g. sigmoid or tanh unit).

The weights $w^{(i)}$ between neurons are free parameters that capture the model's representation of the data and are learned from input/output samples. Learning minimizes a loss function $L(w)$ that measures the fit of the model output to the true label of a sample (panel A, bottom). This minimization is challenging, since the loss function is high-dimensional and non-convex, similar to a landscape with many hills and valleys (panel C). It took several decades before the *backward propagation algorithm*was first applied to compute a loss function gradient via chain rule for derivatives, ultimately enabling efficient training of neural networks using stochastic gradient

descent. During learning, the predicted label is compared with the true label to compute a loss for the current set of model weights. The loss is then backward propagated through the network to compute the gradients of the loss function and update (panel A). The loss function *L(w)* is typically optimized using gradient-based descent. In each step, the current weight vector (red dot) is moved along the direction of steepest descent *dw* (direction arrow) by learning rate η (length of vector). Decaying the learning rate over time allows to explore different domains of the loss function by jumping over valleys at the beginning of the training (left side) and fine-tune parameters with smaller learning rates in later stages of the model training. While learning in deep neural networks remains an active area of research, existing software packages can already be applied without knowledge of the mathematical details involved.

Alternative architectures to such fully connected feedforward networks have been developed for specific applications, which differ in the way neurons are arranged. These include convolutional neural networks, which are widely used for modelling images, recurrent neural networks for sequential data [75], or restricted Boltzmann machines [76] and autoencoders [77] for unsupervised learning. The choice of network architecture and other parameters can be made in a data-driven and objective way by assessing the model performance on a validation data set.

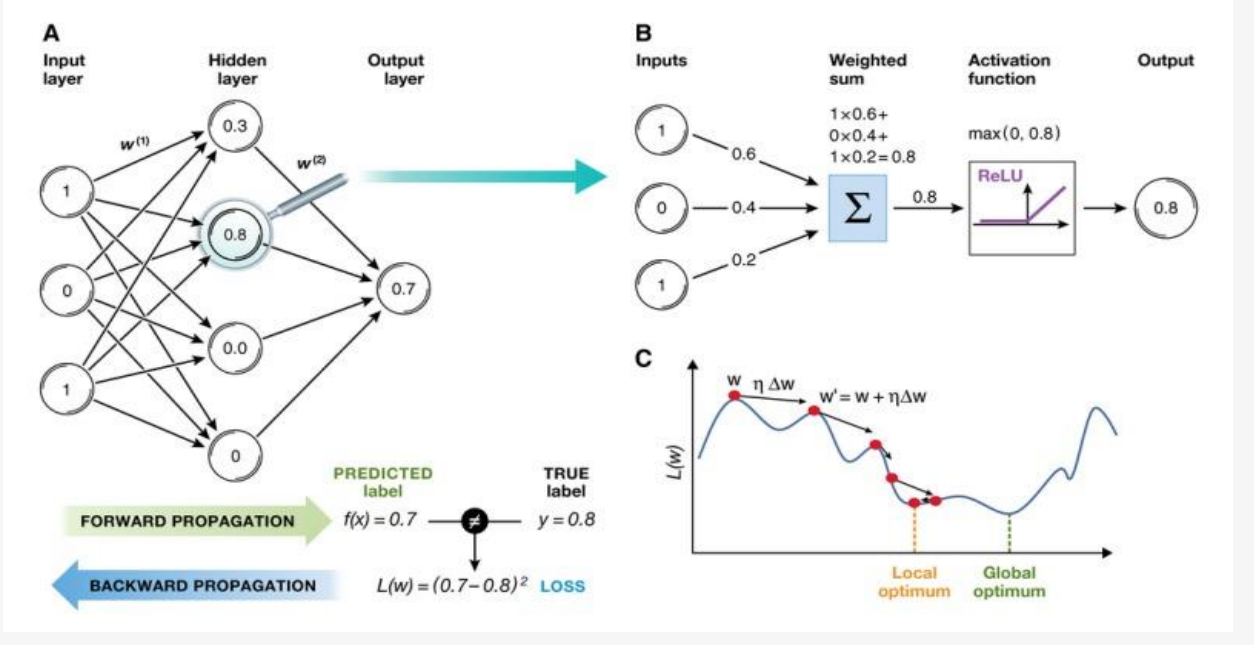


Figure 4: Working of Artificial Neural Network

2.3.3 Convolutional Neural Network

Convolutional neural systems (CNNs) were initially enlivened by intellectual neuroscience and Hubel and Wiesel's original work on the feline's visual cortex, which was found to have basic neurons that react to little themes in the visual field, and complex neurons that react to bigger one[78].

CNNs are intended to display include information as multidimensional exhibits, for example, two-dimensional pictures with three shading channels or one-dimensional genomic arrangements with one channel for each nucleotide. The high dimensionality of these information (up to a large number of pixels for high-resolution pictures) renders preparing a completely associated neural system testing, as the quantity of parameters of such a model would normally surpass the quantity of preparing information to fit them. To evade this, CNNs make extra presumptions on the structure of the system, in this manner decreasing the powerful number of parameters to learn.

A convolutional layer consists of multiple maps of neurons, so-called feature maps or filters, with their size being equal to the dimension of the input image (Figure 5 panel A). Two concepts allow reducing the number of model parameters: local connectivity and parameter sharing. First, unlike in a fully connected network, each neuron within a feature map is only connected to a local patch of neurons in the previous layer, the so-called receptive field. Second, all neurons within a given feature map share the same parameters. Hence, all neurons within a feature map scan for the same feature in the previous layer, however at different locations. Different feature maps might, for example, detect edges of different orientation in an image, or sequence motifs in a genomic sequence. The activity of a neuron is obtained by computing a discrete convolution of its receptive field, that is computing the weighted sum of input neurons, and applying an activation function (Figure 5 panel B).

In most applications, the exact position and frequency of features is irrelevant for the final prediction, such as recognizing objects in an image. Using this assumption, the pooling layer summarizes adjacent neurons by computing, for example, the maximum or average over their activity, resulting in a smoother representation of feature activities (Figure 5 panel C). By applying the same pooling operation to small image patches that are shifted by more than one pixel, the

input image is effectively down-sampled, thereby further reducing the number of model parameters.

A CNN typically consists of multiple convolutional and pooling layers, which allows learning more and more abstract features at increasing scales from small edges, to object parts, and finally entire objects. one or more fully connected layers can follow the last pooling layer (Figure 5 panel A). Model hyper-parameters such as the number of convolutional layers, number of feature maps or the size of receptive fields are application-dependent and should be strictly selected on a validation data set.

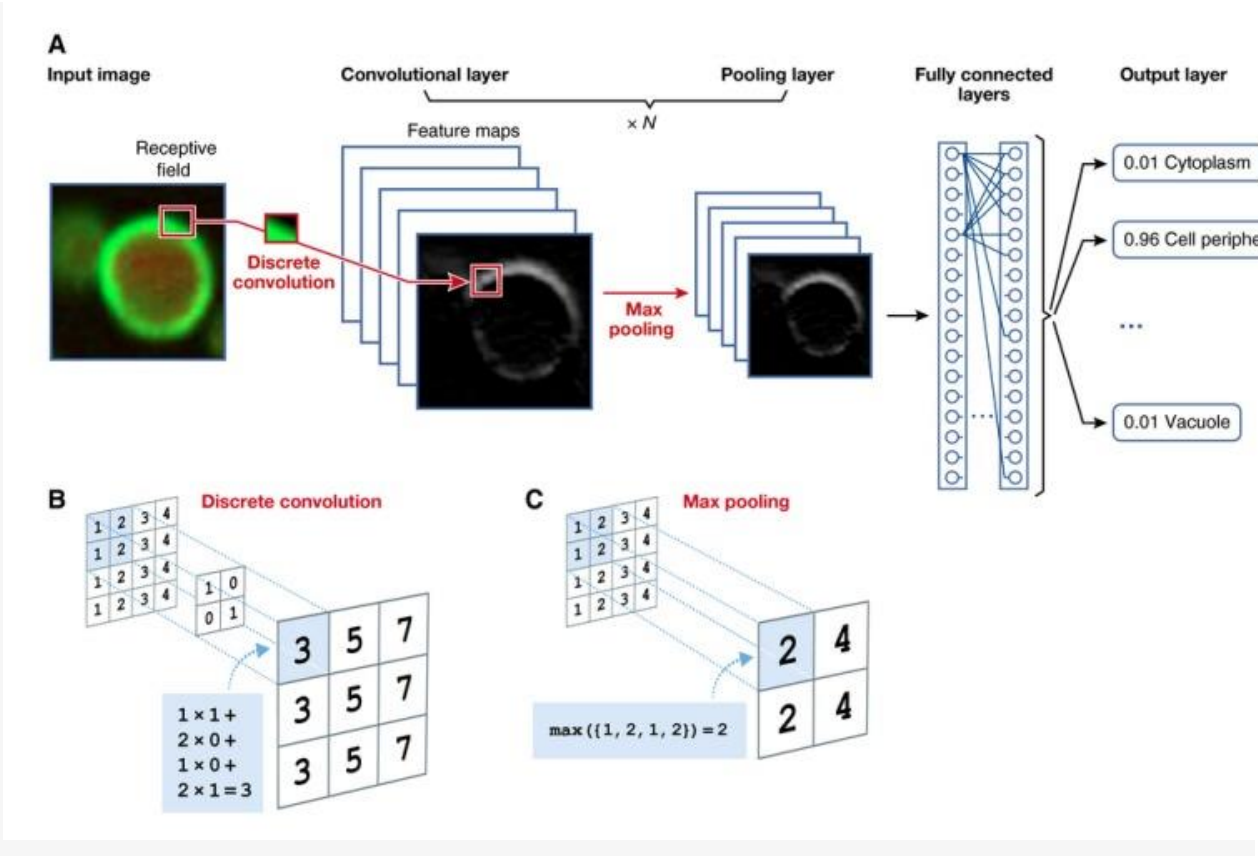


Figure 5: Working CNNs

2.3.4 Recurrent Neural network

Humans don't start their thinking from scratch every second. Traditional neural networks can't do this, and it seems like a major shortcoming. For example, imagine if one want to classify what kind of event is happening at every point in a movie. It's unclear how a traditional neural network could use its reasoning about previous events in the film to inform later ones.

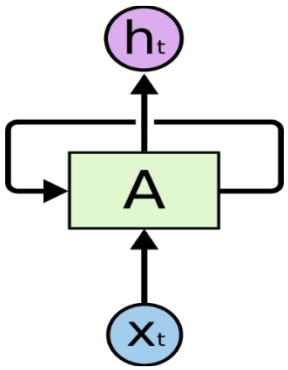


Figure 6: Recurrent Neural Networks have loops

Recurrent neural networks address this issue. They are networks with loops in them, allowing information to persist.

In the above diagram, a chunk of neural network, AA, looks at some input xt and outputs a value ht. A loop allows information to be passed from one step of the network to the next. These loops make recurrent neural networks seem kind of mysterious. It turns out that they aren't all that different than a normal neural network. A recurrent neural network can be thought of as multiple copies of the same network, each passing a message to a successor. Consider what happens if we unroll the loop:

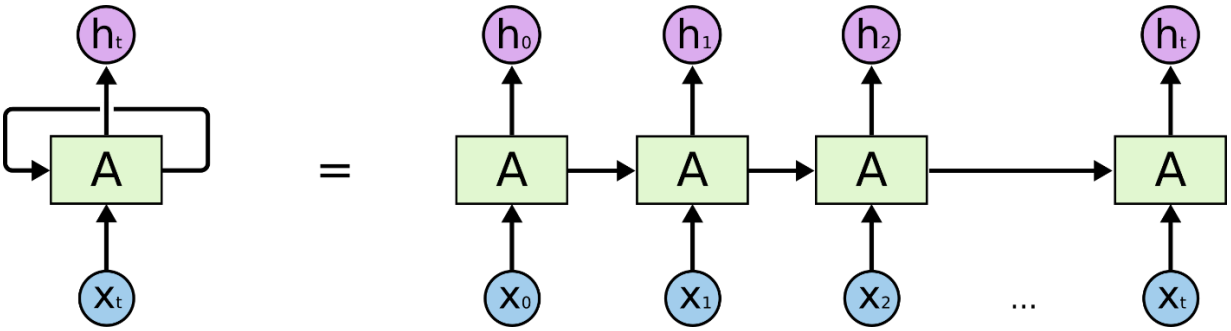


Figure 7: An unrolled recurrent neural network

.

This chain-like nature reveals that recurrent neural networks are intimately related to sequences and lists. They're the natural architecture of neural network to use for such data.

And they certainly are used! In the last few years, there have been incredible success applying RNNs to a variety of problems: speech recognition, language modeling, translation, image captioning… The list goes on.

Essential to these successes is the use of "LSTMs," a very special kind of recurrent neural network which works, for many tasks, much much better than the standard version. Almost all exciting results based on recurrent neural networks are achieved with them.

The Problem of Long-Term Dependencies

one of the appeals of RNNs is the idea that they might be able to connect previous information to the present task, such as using previous video frames might inform the understanding of the present frame.Sometimes, we only need to look at recent information to perform the present task. For example, consider a language model trying to predict the next word based on the previous ones. If we are trying to predict the last word in "the clouds are in the *sky*," we don't need any further context – it's pretty obvious the next word is going to be sky. In such cases, where the gap between the relevant information and the place that it's needed is small, RNNs can learn to use the past information.

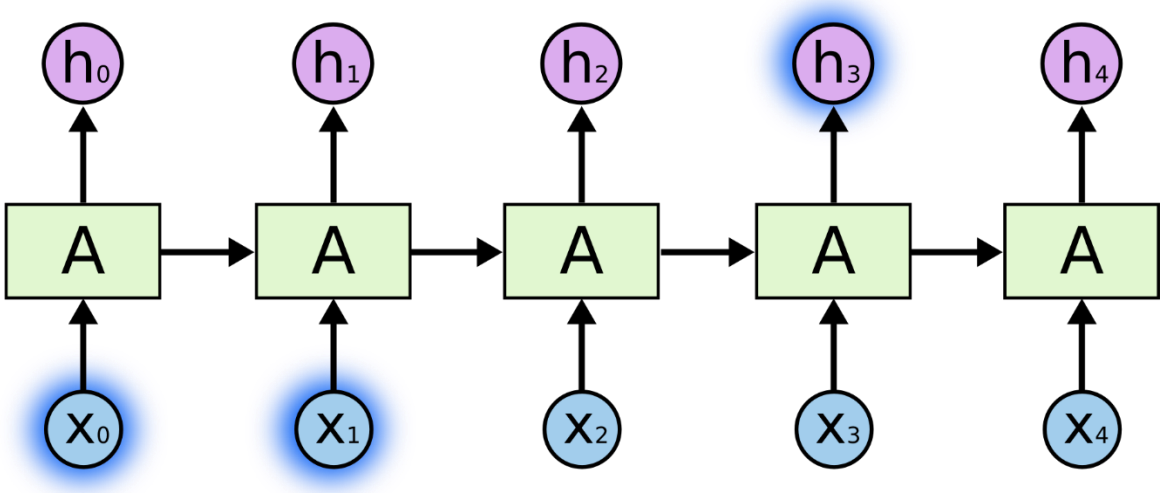


Figure 8: Working of RNNs

But there are also cases where we need more context. Consider trying to predict the last word in the text "I grew up in France… I speak fluent *French*." Recent information suggests that the next word

is probably the name of a language, but if we want to narrow down which language, we need the context of France, from further back. It's entirely possible for the gap between the relevant information and the point where it is needed to become very large.

Unfortunately, as that gap grows, RNNs become unable to learn to connect the information.

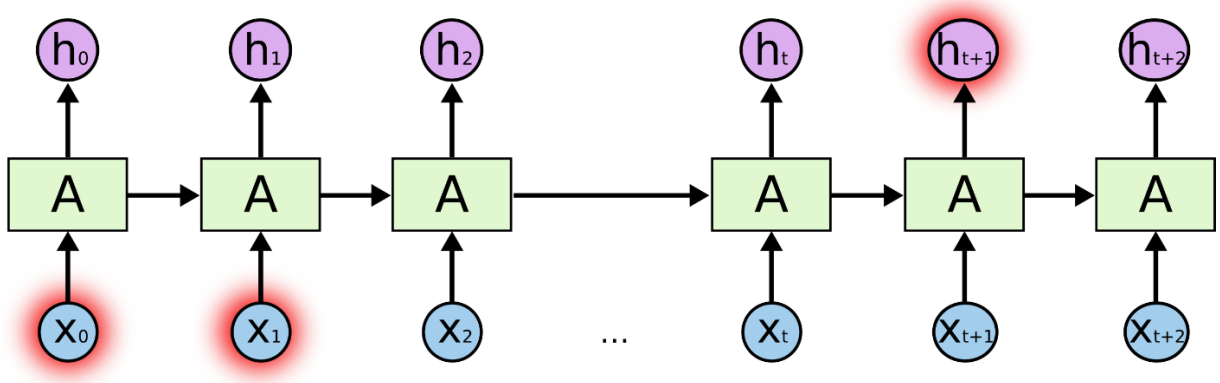


Figure 9: RNNs become unable to learn to connect the information

In theory, RNNs are absolutely capable of handling such "long-term dependencies." A human could carefully pick parameters for them to solve toy problems of this form. In practice, RNNs don't seem to be able to learn them. LSTMs don't have this problem!

LSTM Networks

Long Short Term Memory networks – usually just called "LSTMs" – are a special kind of RNN, capable of learning long-term dependencies. They were introduced by Hochreiter & Schmidhuber [79], and were refined and popularized by many people in following work. They work tremendously well on a large variety of problems, and are now widely used.

LSTMs are explicitly designed to avoid the long-term dependency problem. Remembering information for long periods of time is practically their default behavior, not something they struggle to learn!

All recurrent neural networks have the form of a chain of repeating modules of neural network. In standard RNNs, this repeating module will have a very simple structure, such as a single tanh layer.

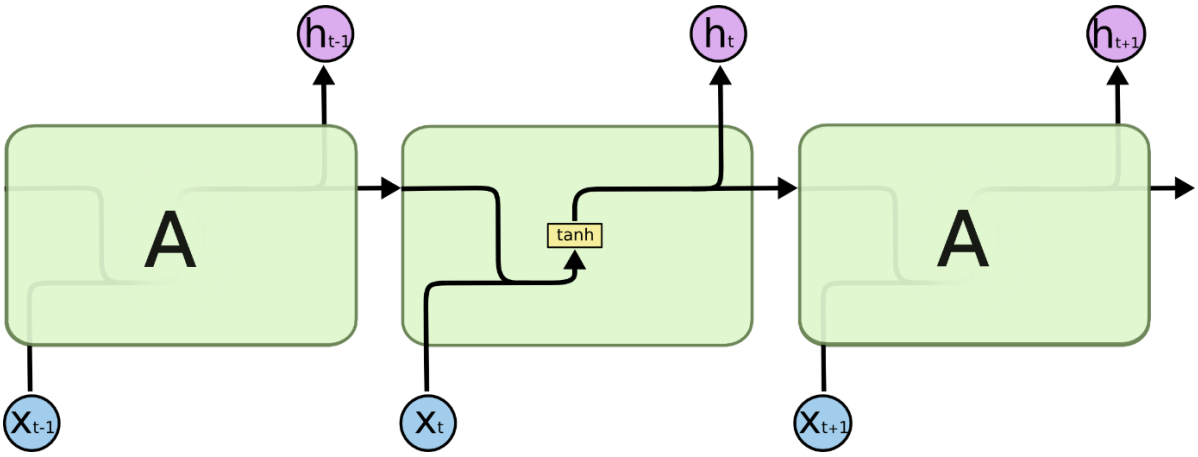


Figure 10: The repeating module in a standard RNN contains a single layer

LSTMs also have this chain like structure, but the repeating module has a different structure. Instead of having a single neural network layer, there are four, interacting in a very special way.

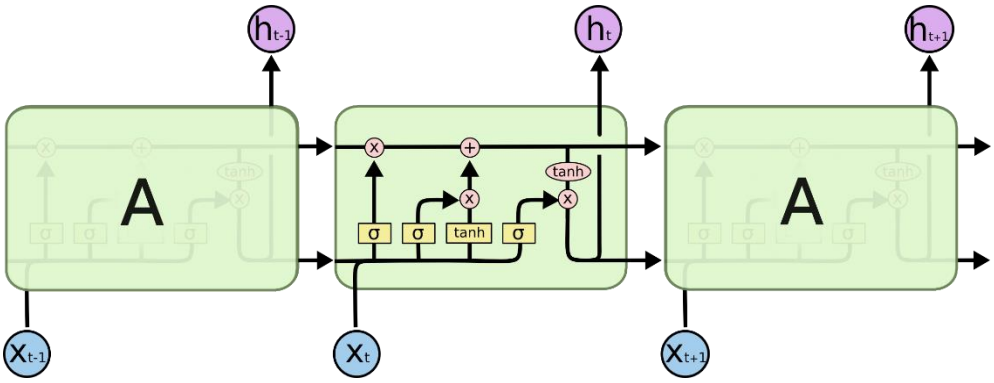
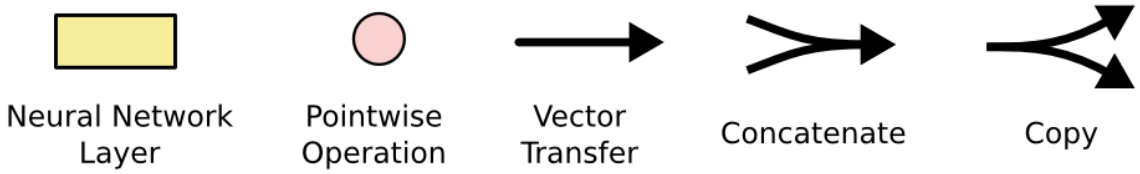


Figure 11: The repeating module in an LSTM contains four interacting layers



In the above diagram, each line carries an entire vector, from the output of one node to the inputs of others. The pink circles represent pointwise operations, like vector addition, while the yellow boxes

are learned neural network layers. Lines merging denote concatenation, while a line forking denote its content being copied and the copies going to different locations.

A basic sequence-to-sequence model consists of two recurrent neural networks (RNNs): an *encoder* that processes the input and a *decoder* that generates the output. This basic architecture is depicted below.

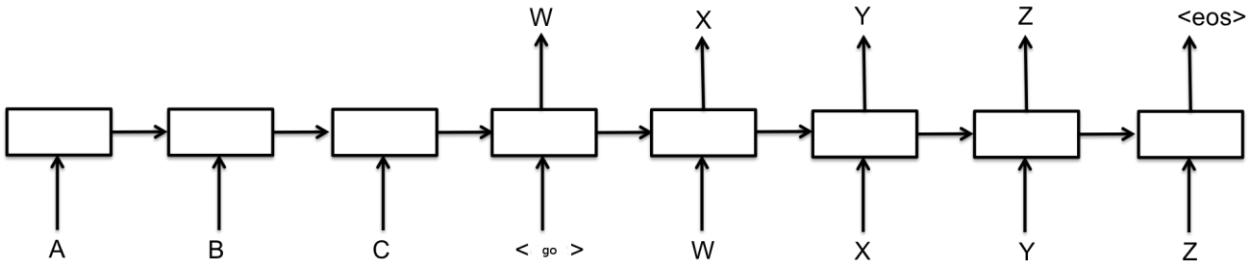


Figure 12: Typical Seq2Seq architecture

Each box in the picture above represents a cell of the RNN, most commonly a GRU cell or an LSTM cell. Encoder and decoder can share weights or, as is more common, use a different set of parameters. Multi-layer cells have been successfully used in sequence-to-sequence models too, e.g. for translation.

In the basic model depicted above, every input has to be encoded into a fixed-size state vector, as that is the only thing passed to the decoder. To allow the decoder more direct access to the input, an *attention* mechanism was introduced in [80]. It allows the decoder to peek into the input at every decoding step. A multi-layer sequence-to-sequence network with LSTM cells and attention mechanism in the decoder looks like this.

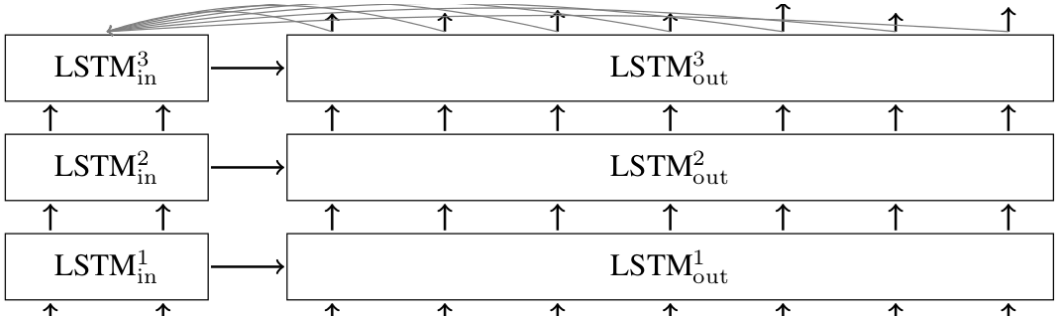


Figure 13: Multi-layer seq2seq network with LSTM cells and attention mechanism

2.3.5 Principles of using neural networks for predicting molecular traits from DNA sequence

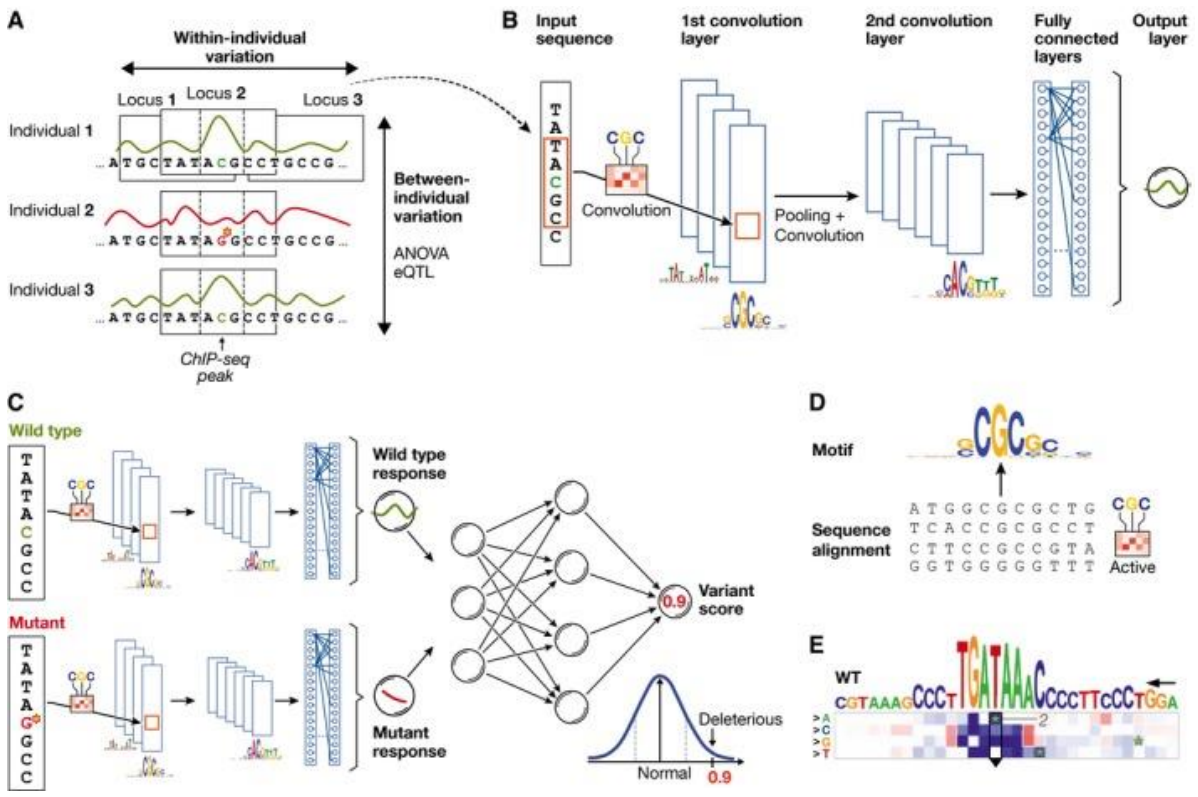


Figure 14: Principles of using neural networks for predicting molecular traits from DNA sequence

(A) DNA sequence and the molecular response variable along the genome for three individuals. Conventional approaches in regulatory genomics consider variations between individuals, whereas deep learning allows exploiting intra-individual variations by tiling the genome into sequence DNA windows centred on individual traits, resulting in large training data sets from a single sample.

(B) one-dimensional convolutional neural network for predicting a molecular trait from the raw DNA sequence in a window. Filters of the first convolutional layer (example shown on the edge) scan for motifs in the input sequence. Subsequent pooling reduces the input dimension, and additional convolutional layers can model interactions between motifs in the previous layer.

(C) Response variable predicted by the neural network shown in (B) for a wild-type and mutant sequence is used as input to an additional neural network that predicts a variant score and allows to discriminate normal from deleterious variants.

(D) Visualization of a convolutional filter by aligning genetic sequences that maximally activate the filter and creating a sequence motif.

(E) Mutation map of a sequence window. Rows correspond to the four possible base pair substitutions, columns to sequence positions. The predicted impact of any sequence change is colour-coded. Letters on top denote the wild-type sequence with the height of each nucleotide denoting the maximum effect across mutations (figure panel adapted from [81]).

CHAPTER 3 METHODOLOGY

3.1 Data Curation

. Here we intended to use CNN and LSTM network in the form of seq2seq architecture to predict the matching sequence of microRNA from mRNA. For seq2seq architecture we require sequence of microRNAs and their corresponding binding target sites in a mRNA.

microRNA sequence were retrieve from miRBase release 22, march 2018[82]. miRBase provides the following services:

- The miRBase database is a searchable database of published microRNA sequences and annotation. Each entry in the miRBase Sequence database represents a predicted hairpin portion of a microRNA transcript (termed mir in the database), with information on the location and sequence of the mature microRNA sequence (termed miR). Both hairpin and mature sequences are available for searching and browsing, and entries can also be retrieved by name, keyword, references and annotation. All sequence and annotation data are also available for download.
- The miRBase Registry provides microRNA gene hunters with unique names for novel microRNA genes prior to publication of results. Visit the help pages for more information about the naming service.

The corresponding microRNA target site sequence were curated from DIANA-TarBase v8[83].

DIANA-TarBase v8 (http://www.microrna.gr/tarbase) is a reference database devoted to the indexing of experimentally supported microRNA (microRNA) targets. Its eighth version is the first database indexing >1 million entries, corresponding to ~670 000 unique microRNA-target pairs. The interactions are supported by >33 experimental methodologies, applied to ~600 cell types/tissues under ~451 experimental conditions. It integrates information on cell-type specific microRNA-gene regulation, while hundreds of thousands of microRNA-binding locations are reported. TarBase is coming of age, with more than a decade of continuous

support in the non-coding RNA field. A new module has been implemented that enables the browsing of interactions through different filtering combinations. It permits easy retrieval of positive and negative microRNA targets per species, methodology, cell type and tissue. An incorporated ranking system is utilized for the display of interactions based on the robustness of their supporting methodologies. Statistics, pie-charts and interactive bar-plots depicting the database content are available through a dedicated result page. An intuitive interface is introduced, providing a user-friendly application with flexible options to different queries.

For retrieving data we created a web crawler for TarBase v8 in python using package Beautifulshoup. our crawler takes microRNA name as input and finds its target sites form every entry in TarBase v8. It gives output CSV file containing microRNA Name, Gene Symbol, Chromosome location of the target binning site according to Ensemble Human (GRCh38.p12) annotation. The code for this web crawler can be found in Appendix I.

High-throughput sequencing of RNA isolated by crosslinking immunoprecipitation (HITS-CLIP, also known as CLIP-Seq) is a genome-wide means of mapping protein–RNA binding sites or RNA modification sites in vivo[84][85]. HITS-CLIP was originally used to generate genome-wide protein-RNA interaction maps for the neuron-specific RNA-binding protein and splicing factor NoVA1 and NoVA2; since then a number of other splicing factor maps have been generated, including those for PTB, RbFox2, SFRS1, hnRNP C, and even N6-Methyladenosine (m6A) mRNA modifications.

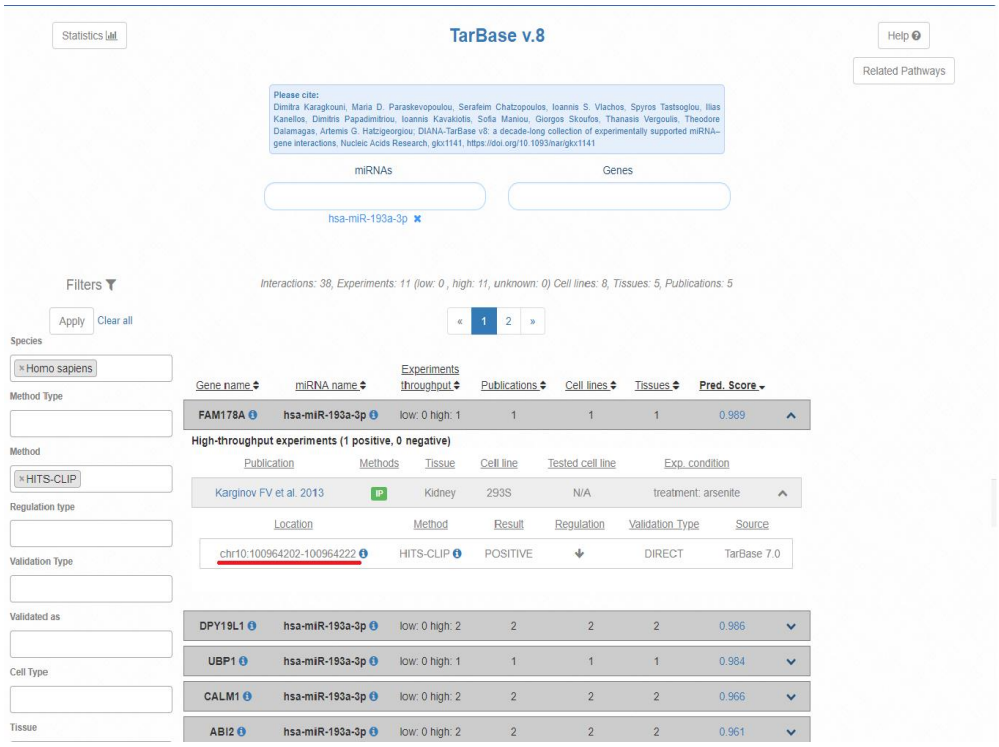


Figure 15: Screenshot of result page of TarBase v.8

HITS-CLIP of the RNA-binding protein Argonaute has been performed for the identification of microRNA targets[86] by decoding microRNA-mRNA and protein-RNA interaction maps in mouse brain[87], and subsequently in Caenorhabditis elegans, embryonic stem cells and tissue culture cells.

A sample of retrieve data can be found below.

Table 2: Sample of Retrieve data for training

microRNA_name	microRNA_sequence	Chromosome Location	mRNA_Sequence	Gene Symbol
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	15:98960051-98960068	ACUCCAUCUAUUUACAAA	IGF1R

hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	13:48480030-48480053	ACUCCAUAGGUACGAUAGUAAGUA	RB1
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	1:205602438-205602456	ACUCCAUCCCAUCCAUGAA	MFSD4
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	1:38863569-38863589	ACUCCAUCCGUAGUGCCUGUA	MYCBP
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	1:35853829-35853851	ACUCCAUUUUUAAGUCAGGUCAC	AGO4
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	17:49051062-49051083	ACUCCAUCAAAUGAAGCGUGUG	IGF2BP1
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	1:207048984-207049003	ACUCCAUCAUCCGAAGUUGG	YOD1
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	3:47736093-47736115	ACUCCAUCUACAACCCAGACCAG	SMARCC1
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	15:52065475-52065494	ACUCCAUCUAUUGUGUACAC	MAPK6
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	14:52642082-52642099	ACUCCAUAUAUCGAAGAA	ERO1L
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	9:124522310-124522334	ACUCCAUCAGUAGAAACAGUUGUAA	NR6A1
hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	5:41921316-41921344	AUUCCAUCUUUCUCUUUUCUCCGUUCGUU	C5orf51

hsa-let-7a-5p	UGAGGUAGUAGGUUGUAUAGUU	5:154475055-154475073	ACUCCAUCUUUUCCCAACC	HAND1

3.2 Data Preparation

3.2.1 Data Cleaning

As data were scraped with crawler all, all sequences are manually checked to see the polarity of the strands. All microRNA sequence strands were in polarity 3' to 5' and all mRNA strands are in polarity of 5' to 3' here by conserving the seed pair features of 3' end and 5' end of microRNA and mRNA respectively.

microRNA – 3' UGAGGUAGUAGGUUGUAUAGUU 5'
| | | | | | | | | | :
mRNA- 5' ACUCCAUCAGUAGAAACAGUUGUAA 3'

To bring a uniformity in length of sequences and difference in length of sequence so that our model could able to extract features based on patterns in the sequence not the length of sequences, we decided to set some threshold values. The threshold values are as follows:

- The difference between the length of pairs of microRNA and mRNA were varying from 0 to 18, which was very divorced, so we looked in to the distribution of the pair sequence length difference. We saw that the distribution was left side skewed with having around 90% of the difference in length pair of sequences between 0 to 6. So we took threshold as 6. After doing that we are left with 19300 data sets.

3.2.2 DNA sequence one-hot encoded as binary vectors using codes

Categorical features such as mRNA and microRNA nucleotides first need to be encoded numerically. They are typically represented as binary vectors with all but one entry set to zero, which indicates the category (*one-hot coding*). For example, DNA nucleotides (categories) are commonly encoded as A = (1 0 0 0), G = (0 1 0 0), C = (0 0 1 0) and U = (0 0 0 1). A DNA

sequence can then be represented as a binary string by concatenating the encoding nucleotides, and treating each nucleotide as an independent input feature of a feedforward neural network.

Binary coded strings as input of a neural network should be of same length, i.e. all microRNAs will be of same length and all mRNA target site sequence should be of same length which is not the scenario of real world. So. We took the lengthiest sequence among the microRNA sequence as a default length of all microRNAs by filling any blank places with zeros in any microRNA sequence whose length is less than the default length which turns to be 28. In the same way we treat the mRNAs sequences and their default length turns to be 29.

For mRNA we encoded the sequences in binary strings by making matrix of 29 X 4. Where we put A as [1,0,0,0] , C as [0,1,0,0], G as [0,0,1,0] and U as [0,0,0,1].

Table 3 Binary matrix representing mRNA binding site sequence 'UUGUGUAGUAACGUGUAAUGUCG'

U	0	0	0	1
U	0	0	0	1
G	0	0	1	0
U	0	0	0	1
G	0	0	1	0
U	0	0	0	1
A	1	0	0	0
G	0	0	1	0
U	0	0	0	1
A	1	0	0	0
A	1	0	0	0
C	0	1	0	0
G	0	0	1	0
U	0	0	0	1
G	0	0	1	0
U	0	0	0	1
A	1	0	0	0
A	1	0	0	0
U	0	0	0	1
G	0	0	1	0
U	0	0	0	1
C	0	1	0	0
G	0	0	1	0
0	0	0	0	0
0	0	0	0	0

0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0

on the other hand as we have to predict microRNAs from mRNA sequence using Seq2Seq architecture, so we require to provide a starting and ending token. So we used '\t' as starling token and '\n' and ending token. So resultant microRNA sequences are in the from of target_microRNA = '\t' + target_text + '\n'. For microRNA we encoded the sequences in binary strings by making matrix of 28 X 6. Where we put A as [1,0,0,0,0,0] , C as [0,1,0,0,0,0], G as [0,0,1,0,0,0], U as [0,0,0,1,0,0], '\t' as [0,0,0,0,1,0] and lastly '\n' as [0,0,0,0,0,1]

Than we shuffled the data for introducing randomness in every batch during training. Python code for reading the data from file and hot-encoded the input mRNA sequences and microRNA sequences are given in APPENDIX II.

3.3 Building mirBoT model

3.3.1 Proposed model

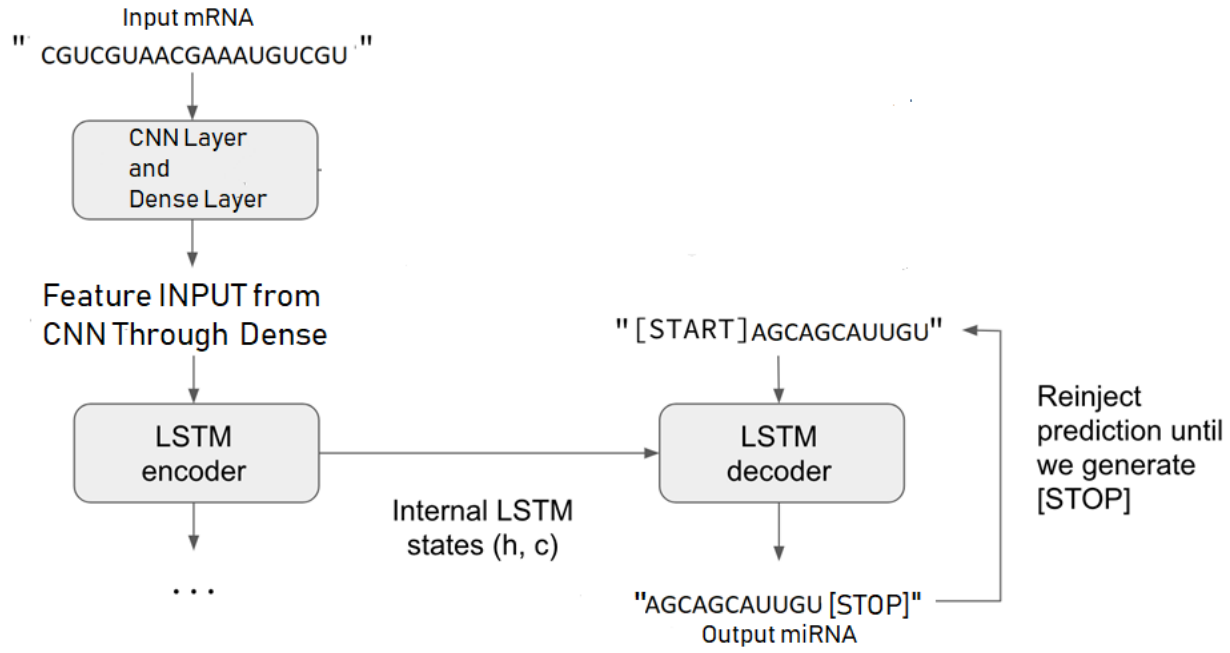


Figure 16: Proposed Seq2Seq model using CNNs and LSTMs for microRNA sequence prediction

Above given figure is our proposed model for prediction of microRNA sequences from mRNA sequences. In python we used Keras library of deep learning using Tensorflow in background for building and training our model.

TensorFlow is an open-source programming library for dataflow programming over a scope of undertakings. It is a representative math library, and is likewise utilized for machine learning applications, for example, neural systems. It is utilized for both research and generation at Google.

Keras is a high-level neural networks API, written in Python and capable of running on top of TensorFlow, CNTK, or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research. Keras offers easy and fast prototyping (through user friendliness, modularity, and extensibility). Supports both convolutional networks and recurrent networks, as well as combinations of the two. Runs seamlessly on CPU and GPU.

3.3.2 Developing the model

First step was to extract feature using CNN from mRNA sequence. We had use a window size of 8 for extracting 128 features. The method of feature extraction by CNNs is explained in point 2.3.3 of this report.

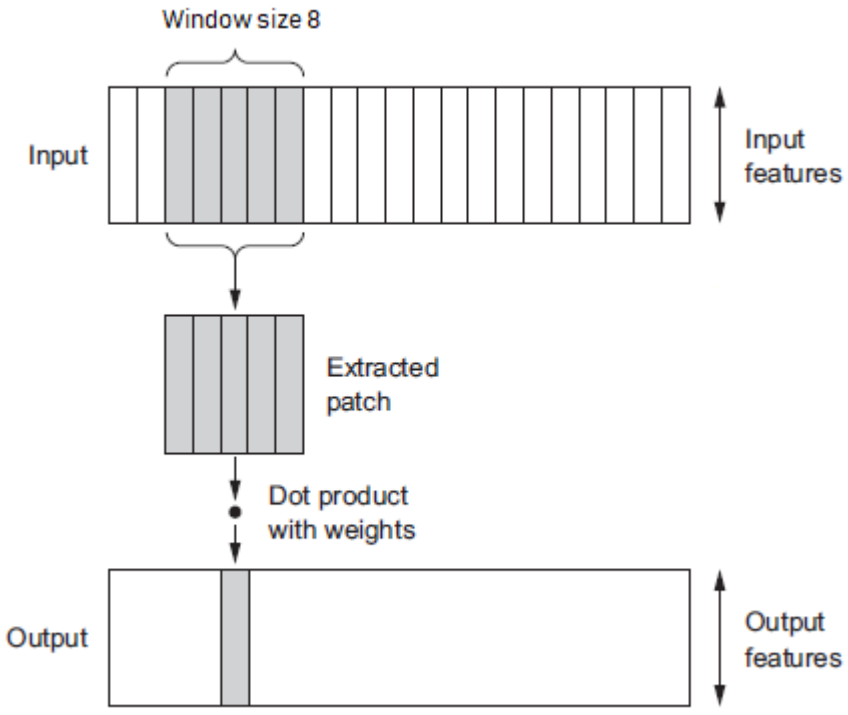


Figure 17: Feature extraction using CNN on mRNA sequence.

Then we used a dense layer of 128 neurons for adjusting weight of 128 features which were extracted from mRNA sequence which were than feed into LSTM networks.

Building Seq2Seq LSTM Network:

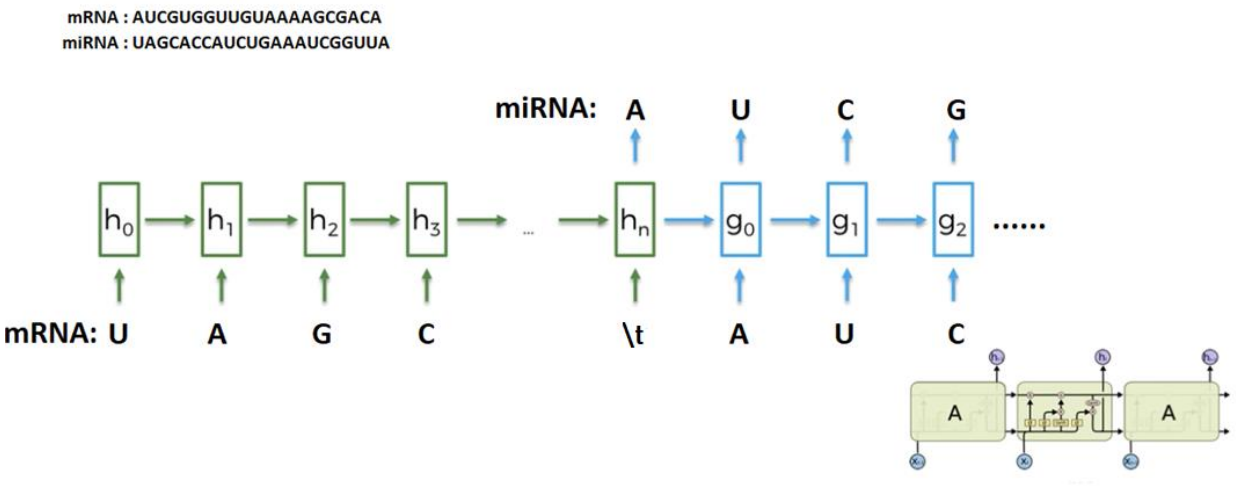


Figure 18: Seq2Seq LSTM model

A RNN layer (or stack thereof) goes about as "encoder": it forms the information succession and returns its own particular interior state. Note that we dispose of the yields of the encoder RNN, just recouping the state. This state will fill in as the "specific circumstance", or "molding", of the decoder in the subsequent stage (Figure 15).

Another RNN layer (or stack thereof) goes about as "decoder": it is prepared to foresee the following characters of the objective grouping, given past characters of the objective succession. In particular, it is prepared to transform the objective successions into similar arrangements yet counterbalance by one timestep later on, a preparation procedure called "educator constraining" in this specific situation. Essentially, the encoder utilizes as starting state the state vectors from the encoder, which is the means by which the decoder gets data about what it should create. Adequately, the decoder figures out how to create targets[t+1...] given targets[...t], molded on the

information grouping (Figure 15).In prediction mode mode, i.e. when we want to decode unknown input sequences, we go through a slightly different process:

1) Encode the input sequence into state vectors.

2) Start with a target sequence of size 1 (just the start-of-sequence character).

3) Feed the state vectors and 1-char target sequence to the decoder to produce predictions for the next character.

4) Sample the next character using these predictions (we simply use argmax).

5) Append the sampled character to the target sequence

6) Repeat until we generate the end-of-sequence character or we hit the character limit.

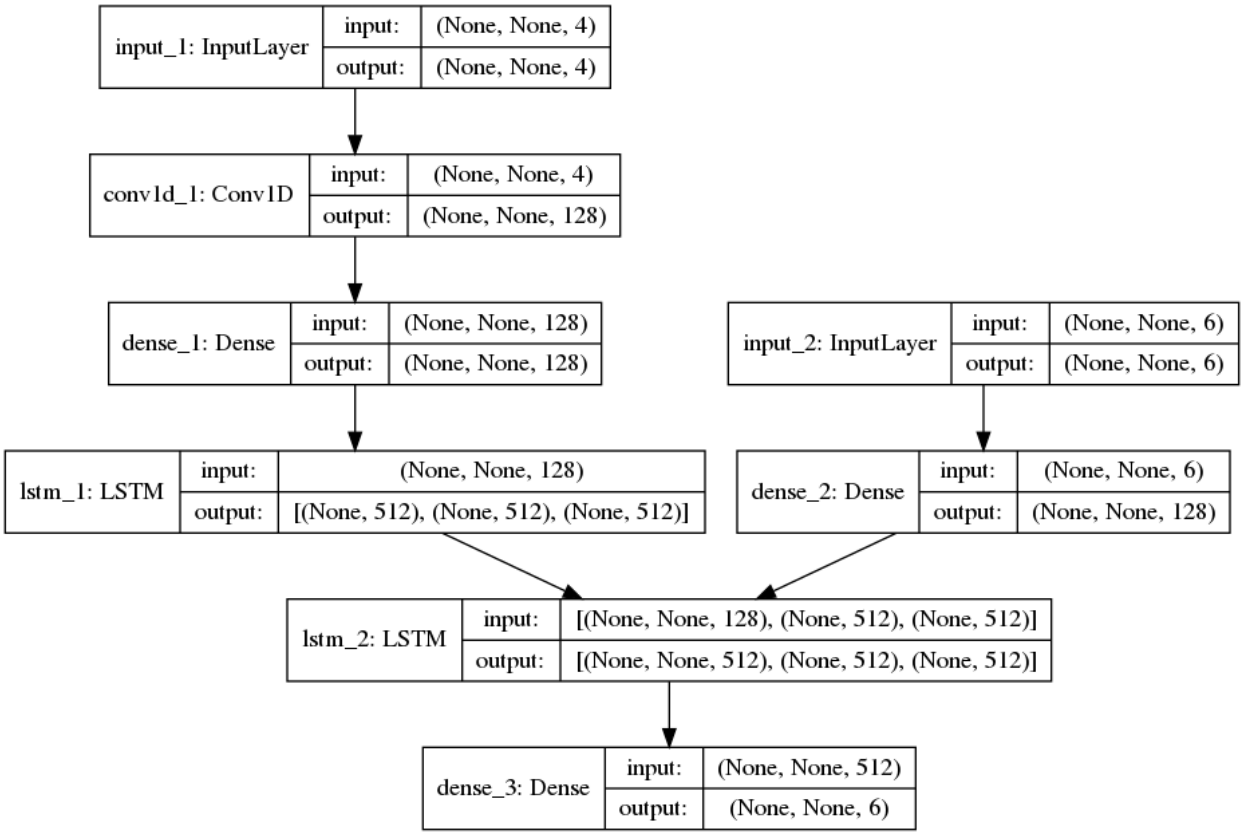


Figure 19: Final Model of mirBoT constitute of Conv1D, Dense and LSTMs layers.

The python code for the model development training and prediction is shown in APPENDIX III

3.4 Training of mirBoT

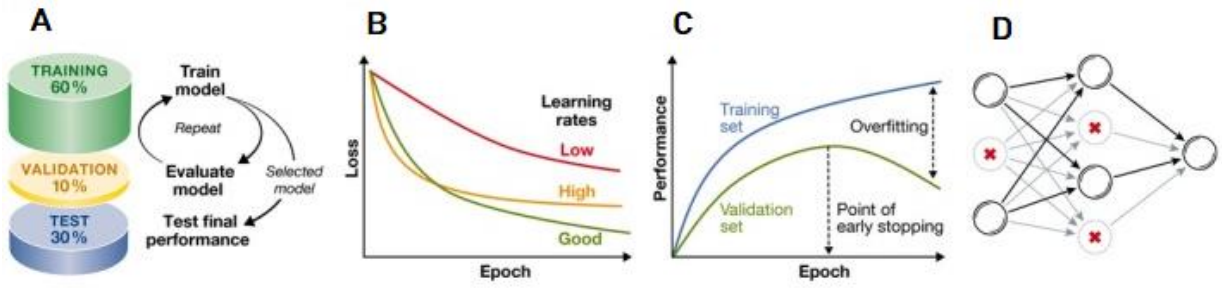


Figure 20: Typical process for training a neutral network model

The goal of model training is to find parameters *w* that minimize an objective function *L*(*w*), which measures the fit between the predictions the model parameterized by *w* and the actual observations. The most common objective functions are the cross-entropy for classification and mean-squared error for regression. Minimizing *L*(*w*) is challenging since it is high-dimensional and non-convex (Figure 19 A); see also2.3.2 section.

3.4.1 Determining the number of neurons in a network

The optimal number of hidden layers and hidden units is problem-dependent and should be optimized on a validation set. one common heuristic is to maximize the number of layers and units without overfitting the data. More layers and units increase the number of representable functions and local optima, and empirical evidence shows that it makes finding a good local optimum less sensitive to weight initialization. Here we used various numbers of neurons combinations in all layers, and found 128 neurons for Dense layers and 512 neurons for LSTMs as optimum.

3.4.2 Partitioning data into Training and Validation sets

Machine learning models need to be trained, validation and tested on independent data sets to avoid overfitting and assure that the model will generalize to unseen data. For proper training partitioning the data into a training, validation and test sets, is the standard for deep neural networks. The training set is used by the models to learn with different hyper-parameters, which

are later evaluated on the validation set. The model with best performance, for example prediction accuracy or mean-squared error, is selected and further evaluated on the test set to quantify the performance on unseen data and for comparison to other methods. out of 19300 data sets we used 80% of the data for training and 20% for validation. And for further testing we used data of microRNA associated with skin diseases from for in house developed database miDerma. It consists of microRNA and mRNA pairs which as associated with dermatological disorders.

3.4.3 Learning Rate and Batch size

The learning rate and clump size of stochastic inclination plunge should be picked with mind, since they can emphatically affect preparing rate and model execution. Diverse learning rates are typically investigated on a logarithmic scale, for example, 0.1, 0.01 or 0.001, with 0.01 as the prescribed default esteem. A clump size of 128 preparing tests is reasonable for generally applications. The group size can be expanded to accelerate preparing or diminished to lessen memory utilization, which can be vital for preparing complex models on memory-limited GPUs. The ideal learning rate and bunch measure are associated, with bigger group sizes regularly requiring littler learning rates.In our work we used a default learning rate of 0.01 and batch size of 50.

3.4.4 Avoiding overfitting

Profound neural systems are famously hard to prepare, and overfitting to information is a noteworthy test, since they are nonlinear and have numerous parameters. overfitting comes about because of an excessively complex model relative, making it impossible to the span of the preparation set, and would thus be able to be diminished by diminishing the model many-sided quality, for instance the quantity of hidden layers and units, or by expanding the measure of the preparation set, for instance by means of information expansion. We have taken the following precautions for avoiding overfitting:

- We had used a dropout rate of 0.5 in LSTM layers.
- We had put L2 regularization penalty of 0.001 in every Dense layers.

Finally we trained our model in floydhub cloud computing instance having 32GB RAM, 11GB NVIDIA Tesla K80 GPU and Intel Xeon 8 Cores CPU for 100 epochs.

Figure 21: Training setup for mirBot

3.5 Obtaining Surface Area accessibility Regions as target binding site of microRNA in 3'UTR of mRNAs

our mirBoT requires a gene symbol for predicting respective microRNAs. Sequences of all protein coding transcripts of a query gene and their 3'UTRs are retrieve using ensemble REST API. As ensemble REST API accept transcript ID so a local SQLite database was developed for mapping Gene Symbols with all respective protein coding ensemble transcript IDs.

Site accessibility is a measure of the ease with which a microRNA can locate and hybridize with an mRNA target. Following transcription, mRNA assumes a secondary structure [53] which can interfere with a microRNA's ability to bind to a target site. MicroRNA:mRNA hybridization involves a two-step process in which a microRNA binds first to a short accessible region of the mRNA. The mRNA secondary structure then unfolds as the microRNA completes binding to a target[54]. Therefore, to assess the likelihood that an mRNA is the target of a microRNA, the predicted amount of energy required to make a site accessible to a microRNA should be evaluated.

So we used RNAplfold program from ViennaRNA Package 2.0 [88] to compute calculate locally stable secondary structure − pair probabilities. This package has python wapper and computes local pair probabilities for base pairs with a maximal span of L. The probabilities are averaged over all windows of size L that contain the base pair.

The output is a plain tuple of matrix containing on each line a position x followed by the probability that x is unpaired, [x-1..x] is unpaired [x-2..x] is unpaired and so on to the probability that [x-i+1..x] is unpaired.

We had set the total accessibility (with RNAplfold). Total accessibility means the sum of P_{free} 's (Probapility of 4-mers being unpaired) over all accessible 4-mers contained in all complementary sites. If P_{free} >=0.2 than those 4-mers are said to be accessible. We used local folding: $W = 80, L = 40$ as referred in [89] and feed sequences of all protein coding transcripts of a query gene to RNAplfold for finding accessibility region or accessible 4-mers. After that we use accessible 4-mers in 3'UTR region for finding the microRNA binding site in respective mRNA i.e. these 4-mers and next 22-mers total 26-mers.

Full python code for finding accessibility region in 3'UTR of a gene's transcript can be found in APPENDIX IV

3.6 Developing Final package

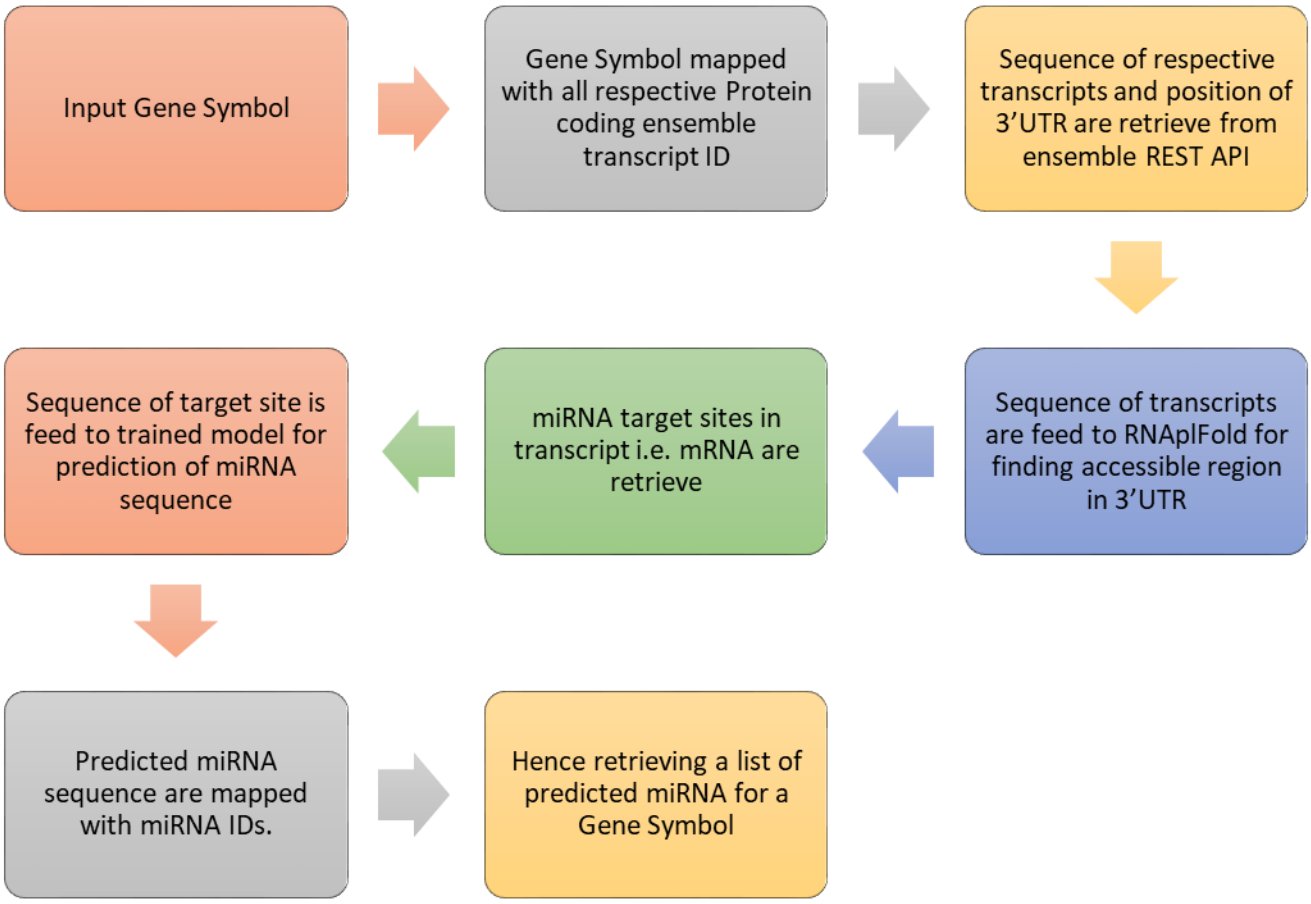


Figure 22: Workflow of mirBoT

When running the program the user will prompt to enter a Gene symbol. Then the Gene symbol will be used to retrieve all associated protein coding ensemble transcript IDs. RNA sequence of these transcript IDs and location of 3'UTR will be retrieve using ensemble REST API. These sequence will be feed to RNAplfold package of ViennaRNA package for finding accessibility region or accessible 4-mers. After that we select accessible 4-mers in 3'UTR region using location of 3'UTRs for finding the microRNA binding site in respective mRNA i.e. these 4-mers and previous 22-mers total 26-mers which are in polarity 3' to 5'. Then these mRNA segments are

feed to our trained model for predicting respective microRNA sequences. Then these predicted microRNA sequences will be mapped to their microRNA IDs using a local SQLite database containing microRNA ID and respective sequence retrieve from mirBase Release 22, March 2018. Hence giving output a list of predicted microRNA IDs. Whole flowchart is shown in figure 22.

CHAPTER 4 RESULTS

mirBoT, a package for predicting microRNA associated with a gene has been developed using neural networks particularly CNN and LSTMs which are according to a well-known seq2seq architecture which are used for prediction of sequence based on sequence. our model are trained on data set containing sequences of microRNA and their respective target binding sites in mRNA which are retrieve from TarBase v8. The model was trained for 100 epochs.

4.1 Accuracy

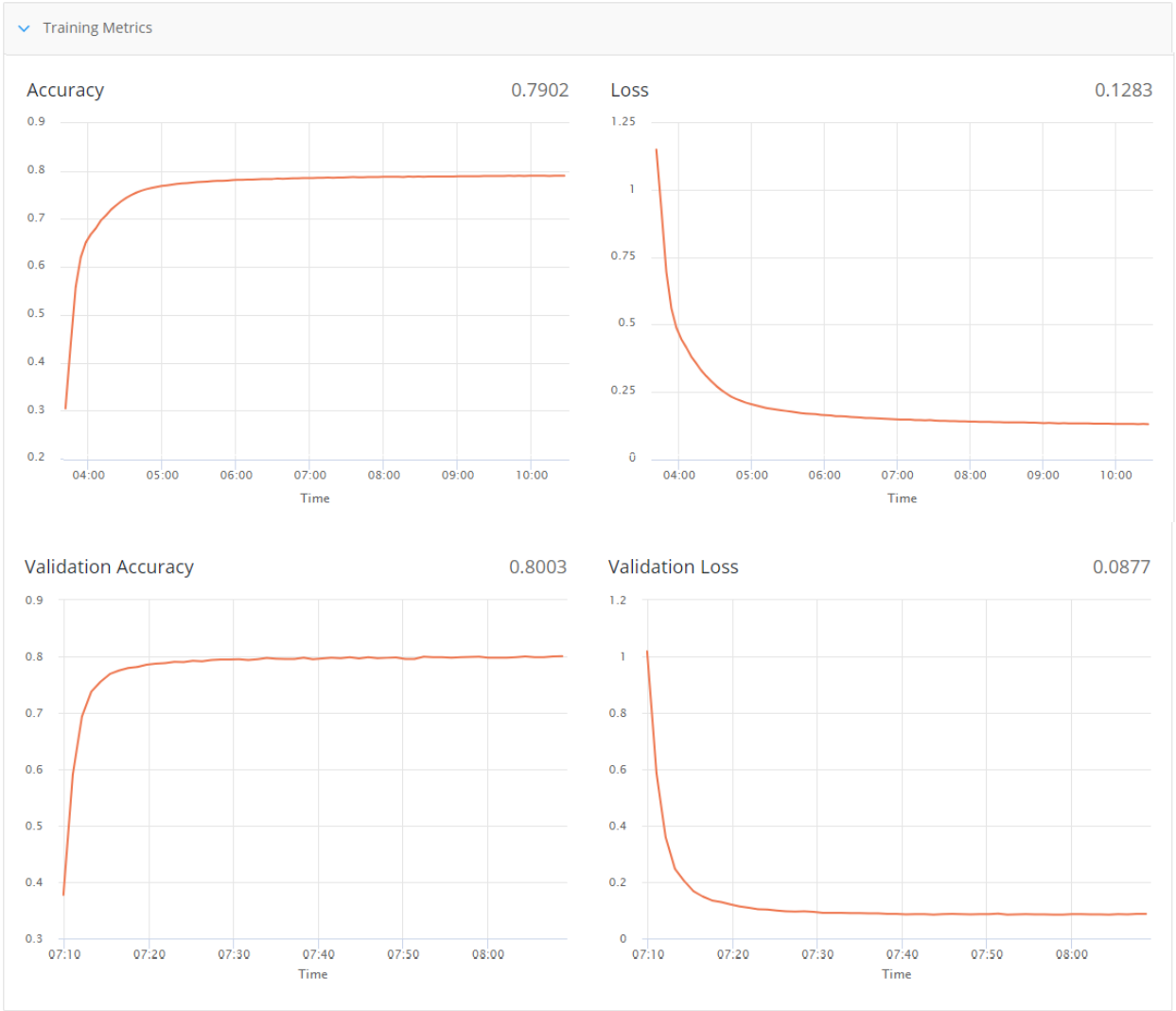


Figure 23: Training Matrices of Model

After training for 100 epochs we found that the training accuracy of our model for predicting microRNA sequence base on its binding site in mRNA is around 79% i.e. accuracy in training set, while in validation set we got an 1% increase in accuracy i.e. around 80%. As training accuracy is less than validation accuracy we can state that out model is not overfitted. Also there is a decrease in validation loss than training loss i.e. training loss is 0.128 where validation loss in 0.087 which also implies the same. From training matrices it can be seen that microRNA sequence can be predicted with upto 80% similarity using its target binding segment in mRNA (figure 23).

4.2 Validation

To test our package with experimentally validated list of microRNAs associated with Gene symbol. 200 Genes we randomly selected from our in house developed database miDerma which contains microRNA and Gene pair associated with dermatological disorders. microRNAs associated with individual genes were retrieved. Also those genes were feed to our package and microRNAs were predicted.

Here our model was able to predict on average 72% of microRNA for each Genes from the list of 200 Genes correctly and also predicted some noble microRNA sequences.

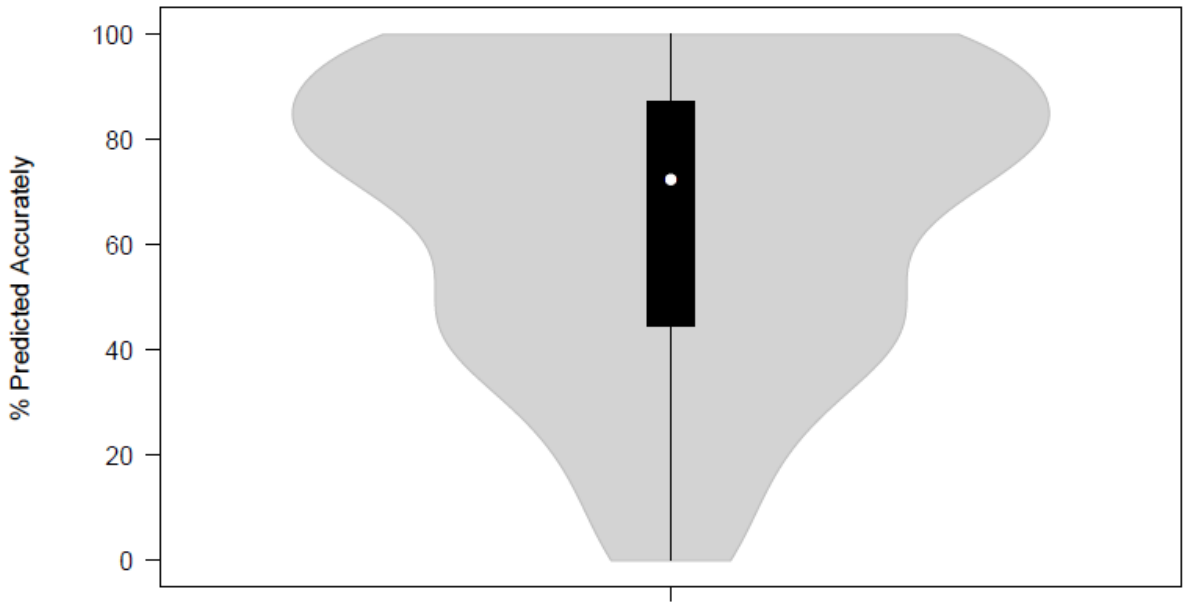


Figure 24: Violin plot showing distribution showing % of accurately predicted microRNAs from 200 Gene Symbols associated with Dermatological disorders among known microRNAs.

Figure above is a violin distribution for percentage of microRNAs predicted accurately among known experimentally validated microRNAs for individual genes in test set of 200 genes. It can be noted that width of the plot is more in the range of 95% to 75%. Also our model was able to predict some noble microRNA sequence for targeted genes.

Table 4: Some of the well predictions done through our mirBoT.

Gene Symbol	No. of Experimentally validated microRNAs associated with Gene	No. of. microRNAs accurately predicted among the validated microRNAs	Percentage of accurately predicted microRNAs among the validated microRNAs (%)
ABCC1	26	26	100
ADAMTS1	13	13	100
ELK3	14	14	100
EPB41L3	20	19	95
BMPR2	51	48	94.11764706
ITSN2	17	16	94.11764706
FGF10	26	24	92.30769231
NT5C3A	12	11	91.66666667
EIF2S2	24	22	91.66666667
CCNE1	34	31	91.17647059
ABCG2	30	27	90
DNMT1	39	35	89.74358974
HOXD11	37	33	89.18918919
DSC3	18	16	88.88888889
GPI	27	24	88.88888889
JAG1	27	24	88.88888889
CUL5	17	15	88.23529412
ESRRA	8	7	87.5
FN1	8	7	87.5
CDK6	188	164	87.23404255
CD28	46	40	86.95652174
ARRDC3	23	20	86.95652174
CADM1	46	40	86.95652174
BAX	15	13	86.66666667
CYP24A1	15	13	86.66666667
FHL2	87	75	86.20689655
IGFBP5	144	124	86.11111111
AKAP12	21	18	85.71428571

DNMT3A	34	29	85.29411765
HOXB13	20	17	85
E2F1	59	50	84.74576271
KMT2D	168	141	83.92857143
ABCB1	12	10	83.33333333
CENPF	12	10	83.33333333
ENO1	12	10	83.33333333
FGFR1	60	50	83.33333333
FOXQ1	42	35	83.33333333
BRIP1	29	24	82.75862069
ESR1	72	59	81.94444444
EZH2	44	36	81.81818182
GUCY1A2	11	9	81.81818182
JAG2	22	18	81.81818182
DKK1	16	13	81.25
FBXW7	58	47	81.03448276
FTO	31	25	80.64516129
HADHB	36	29	80.55555556
ALDOA	190	152	80
APRT	5	4	80
BARD1	10	8	80
CD109	10	8	80
CXCL12	20	16	80
CXCL3	5	4	80
DAP3	5	4	80
CDH5	28	22	78.57142857

CHAPTER 5 DISCUSSION AND CONCLUSION

The miRNA are small generally 28 bp long non-coding RNAs that are comprehensively involved in various physiological and disease processes. one of the major challenge in microRNA studies is the identification of mRNA targeted by miRNAs. Most researchers depends on computational programs to initially finding the target candidates for subsequent validation. Although many advancement has been made in recent years for prediction of targets computationally, but there is still a significant scope for algorithmic improvement.

We have seen that neural networks are categories as a very efficient class of machine learning and are applicable in solving almost every kind of problem starting from classification, clustering, regression, Natural language processing, sequence prediction etc. The fundamental way of learning of a neural network is by adjusting input weights of every neuron. CNNs are a class of ANNs which are used for feature extraction or selection. They are widely used in image recognition for finding unique features from images. They can be also utilizing in extraction of features or recognize specific patterns from sequences which very difficult to de consider by humans. 1D ConvNet can be using for selecting features from a 1D data i.e. a text sequence. RNNs are another class of ANNs which are efficient in learning from a sequence data. They basically use their internal state (memory) to process sequences of inputs which enable them to remember some instant of previous input data which is very helpful in dealing with sequence data. RNNs are used sequence classification and sequence prediction. But RNNs are tend to have a problem of vanishing gradient where it tends to forget instance from very initial states. For overcoming this problem researcher have come up with a up gradation in RNNs i.e. LSTMs. LSTMs tackle the vanishing gradient by adding another memory unit which takes accounts of all necessary states and stores them.

Looking for an improved algorithm, in this work we used sequence pair data of miRNAs and corresponding bound target mRNA from TarBase v8 to trained a ANN network for prediction of miRNA from their bounded target segment in mRNA. We particularly used CNNs for recognizing patterns in mRNA segments and extraction of features. We further used Two LSTMs in seq2seq

architecture for predicting sequences of miRNA. Also two layers of dense network were stacked between CNN and LSTM1, another between input_2 and LSTM1 (ref Figure 19). We trained this model on 19000 experimentally validated and cleaned pair of mRNA and miRNA sequences archiving accuracy of 80%.

As we know that surface area accessibility is a crucial feature for binding of a miRNA with targeted mRNA segment. Atleast 4-mers should be exposed and unbounded in a 3D structure of mRNA for bounded by a miRNA. So we used RNAplfold from RNA vienna package for finding the probability of 4-mers unpaired in mRNA 3D structure. From the regions where those unpaired are located we select segment which can be targeted by miRNAs. Then those segments are feed into our trained model of prediction of possible miRNA which can bind with that target segment.

While running this package the used will prompt to enter a gene symbol, using the gene symbol all the protein coding transcript's sequence will be retrieve from the ensemble rest API. Then these mRNA are processed for predicting a list of miRNAs. Finally we validated our model using experimentally verified microRNA and RNA pairs involved in skin diseases we were retrieved from our in house developed database miDerma. Here our model was able to predict on average 72% of microRNA from mRNA in each cases correctly. We named our package "mirBoT: A MircoRNA sequence prediction tool from RNA sequence base on CNNs, LSTMs and seq2seq architecture, as the neural network model is in the form of seq2seq architecture which are often used in developing chatbots. It this way we particularly tried to bring the natural language processing models to process core language of nature i.e. A, T, G, C. This study, we mainly focus on identifying miRNA target sites which can be bounded by the miRNA. However, subsequent improvement can be achieved by including more features which are stated in section 2.2. Knowing the target of a miRNA is one approach for discovering the role of the miRNA in normal or aberrant biological processes. Perhaps a huge number of targets exist, for any single miRNA. In the course of the most recent 17 years, several tools have been developed to solve this complex issue. Each of these projects has contributed to our understanding of the relationship between miRNA and mRNA targets and how that relationship can be used to make accurate predictions. Prediction miRNA may help the scientific community in the field of therapeutics, biomarker selection etc.

CHAPTER 6 REFERENCES

[1] H. R. Horvitz and J. E. Sulston, "Isolation and genetic characterization of cell-lineage mutants of the nematode Caenorhabditis elegans.," *Genetics*, 1980.

[2] R. C. Lee, R. L. Feinbaum, and V. Ambros, "The C. elegans heterochronic gene lin-4 encodes small RNAs with antisense complementarity to lin-14," *Cell*, 1993.

[3] B. Wightman, I. Ha, and G. Ruvkun, "Posttranscriptional regulation of the heterochronic gene lin-14 by lin-4 mediates temporal pattern formation in C. elegans," *Cell*, 1993.

[4] B. J. Reinhart *et al.*, "The 21-nucleotide let-7 RNA regulates developmental timing in Caenorhabditis elegans," *Nature*, 2000.

[5] A. E. Pasquinelli *et al.*, "Conservation of the sequence and temporal expression of let-7 heterochronic regulatory RNA," *Nature*, 2000.

[6] S. M. Hammond, E. Bernstein, D. Beach, and G. J. Hannon, "An RNA-directed nuclease mediates post-transcriptional gene silencing in Drosophila cells," *Nature*, 2000.

[7] G. Hutvágner, J. McLachlan, A. E. Pasquinelli, É. Bálint, T. Tuschl, and P. D. Zamore, "A cellular function for the RNA-interference enzyme dicer in the maturation of the let-7 small temporal RNA," *Science (80-.).*, 2001.

[8] S. Griffiths-Jones, R. J. Grocock, S. van Dongen, A. Bateman, and A. J. Enright, "miRBase: microRNA sequences, targets and gene nomenclature.," *Nucleic Acids Res.*, 2006.

[9] M. Lagos-Quintana, R. Rauhut, W. Lendeckel, and T. Tuschl, "Identification of novel genes coding for small expressed RNAs," *Science (80-.).*, 2001.

[10] A. Rodriguez, S. Griffiths-Jones, J. L. Ashurst, and A. Bradley, "Identification of mammalian microRNA host genes and transcription units," *Genome Res.*, 2004.

[11] G. L. Papadopoulos, M. Reczko, V. A. Simossis, P. Sethupathy, and A. G. Hatzigeorgiou, "The database of experimentally supported targets: A functional update of TarBase," *Nucleic Acids Res.*, 2009.

[12] M. L. Bortolin-Cavaille, M. Dance, M. Weber, and J. Cavaille, "C19MC microRNAs are processed from introns of large Pol-II, non-protein-coding transcripts," *Nucleic Acids Res.*, 2009.

[13] X. Cai, C. H. Hagedorn, and B. R. Cullen, "Human microRNAs are processed from capped, polyadenylated transcripts that can also function as mRNAs," *RNA*, 2004.

[14] M. Bhattacharyya, L. Feuerbach, T. Bhadra, T. Lengauer, and S. Bandyopadhyay, "MicroRNA transcription start site prediction with multi-objective feature selection.," *Stat. Appl. Genet. Mol. Biol.*, 2012.

[15] V. N. Kim, J. Han, and M. C. Siomi, "Biogenesis of small RNAs in animals," *Nature Reviews Molecular Cell Biology*. 2009.

[16] M. Morlando, M. Ballarino, N. Gromak, F. Pagano, I. Bozzoni, and N. J. Proudfoot, "Primary microRNA transcripts are processed co-transcriptionally," *Nat. Struct. Mol. Biol.*, 2008.

[17] A. M. Denli, B. B. J. Tops, R. H. A. Plasterk, R. F. Ketting, and G. J. Hannon, "Processing of primary microRNAs by the Microprocessor complex," *Nature*, 2004.

[18] M. T. Bohnsack, K. Czaplinski, and D. Görlich, "Exportin 5 is a RanGTP-dependent dsRNA-binding protein that mediates nuclear export of pre-miRNAs," *RNA*, 2004.

[19] E. Bernstein, A. A. Caudy, S. M. Hammond, and G. J. Hannon, "Role for a bidentate ribonuclease in the initiation step of RNA interference," *Nature*, 2001.

[20] T. P. Chendrimada *et al.*, "TRBP recruits the Dicer complex to Ago2 for microRNA processing and gene silencing," *Nature*, 2005.

[21] S. Yang *et al.*, "Widespread regulatory activity of vertebrate microRNA* species," *RNA*, 2011.

[22] M. ohanian, D. T. Humphreys, E. Anderson, T. Preiss, and D. Fatkin, "A heterozygous variant in the human cardiac miR-133 gene, MIR133A2, alters miRNA duplex processing and strand abundance.," *BMC Genet.*, 2013.

[23] L. W. Lee *et al.*, "Complexity of the microRNA repertoire revealed by next-generation sequencing," *RNA*, 2010.

[24] K. C. Vickers, P. Sethupathy, J. Baran-Gale, and A. T. Remaley, "Complexity of microRNA function and the role of isomiRs in lipid homeostasis," *J. Lipid Res.*, 2013.

[25] S. M. Hammond, S. Boettcher, A. A. Caudy, R. Kobayashi, and G. J. Hannon, "Argonaute2, a link between genetic and biochemical analyses of RNAi," *Science (80-.).*, 2001.

[26] X. Ye *et al.*, "Structure of C3Po and mechanism of human RISC activation," *Nat. Struct. Mol. Biol.*, 2011.

[27] X. Liu, D. Y. Jin, M. T. McManus, and Z. Mourelatos, "Precursor MicroRNA-Programmed Silencing Complex Assembly Pathways in Mammals," *Mol. Cell*, 2012.

[28] E. C. Lai, B. Tam, and G. M. Rubin, "Pervasive regulation of Drosophila Notch target genes by GY-box-, Brd-box-, and K-box-class microRNAs," *Genes Dev.*, 2005.

[29] K. C. Miranda *et al.*, "A Pattern-Based Method for the Identification of MicroRNA Binding Sites and Their Corresponding Heteroduplexes," *Cell*, 2006.

[30] J. Liu *et al.*, "Argonaute2 is the catalytic engine of mammalian RNAi," *Science (80-.).*, 2004.

[31] M. R. Fabian and N. Sonenberg, "The mechanics of miRNA-mediated gene silencing: A look under the hood of miRISC," *Nature Structural and Molecular Biology*. 2012.

[32] S. Ekimler and K. Sahin, "Computational Methods for MicroRNA Target Prediction," *Genes (Basel)*, 2014.

[33] P. SETHUPATHY, "TarBase: A comprehensive database of experimentally supported animal microRNA targets," *RNA*, vol. 12, no. 2, pp. 192–197, 2005.

[34] Y. Huang *et al.*, "A study of miRNAs targets prediction and experimental validation," *Protein and Cell*. 2010.

[35] M. Hafner *et al.*, "Transcriptome-wide Identification of RNA-Binding Protein and MicroRNA Target Sites by PAR-CLIP," *Cell*, 2010.

[36] K. J. Riley, T. A. Yario, and J. A. Steitz, "Association of argonaute proteins and microRNAs can occur after cell lysis," *RNA*, 2012.

[37] U. A. Ørom, F. C. Nielsen, and A. H. Lund, "MicroRNA-10a Binds the 5′UTR of Ribosomal Protein mRNAs and Enhances Their Translation," *Mol. Cell*, 2008.

[38] S. W. Eichhorn *et al.*, "MRNA Destabilization Is the dominant effect of mammalian microRNAs by the time substantial repression ensues," *Mol. Cell*, 2014.

[39] P. Sood, A. Krek, M. Zavolan, G. Macino, and N. Rajewsky, "Cell-type-specific signatures of microRNAs on target mRNA expression," *Proc. Natl. Acad. Sci.*, 2006.

[40] D. Baek, J. Villén, C. Shin, F. D. Camargo, S. P. Gygi, and D. P. Bartel, "The impact of microRNAs on protein output.," *Nature*, vol. 455, no. 7209, pp. 64–71, 2008.

[41] C. C. Pritchard, H. H. Cheng, and M. Tewari, "MicroRNA profiling: Approaches and considerations," *Nature Reviews Genetics*. 2012.

[42] L. Q. Gu, M. Wanunu, M. X. Wang, L. McReynolds, and Y. Wang, "Detection of miRNAs with a nanopore single-molecule counter," *Expert Review of Molecular Diagnostics*. 2012.

[43] M. Hafner *et al.*, "RNA-ligase-dependent biases in miRNA representation in deep-sequenced small RNA cDNA libraries," *RNA*, 2011.

[44] E. Knutsen *et al.*, "Performance comparison of digital microRNA profiling technologies applied on human breast cancer cell lines," *PLoS one*, 2013.

[45] B. P. Lewis, I. H. Shih, M. W. Jones-Rhoades, D. P. Bartel, and C. B. Burge, "Prediction of Mammalian MicroRNA Targets," *Cell*, vol. 115, no. 7, pp. 787–798, 2003.

[46] B. P. Lewis, C. B. Burge, and D. P. Bartel, "Conserved seed pairing, often flanked by adenosines, indicates that thousands of human genes are microRNA targets," *Cell*, vol. 120, no. 1, pp. 15–20,

2005.

[47] J. Brennecke, A. Stark, R. B. Russell, and S. M. Cohen, “Principles of microRNA-target recognition,” in *PLoS Biology*, 2005, vol. 3, no. 3, pp. 0404–0418.

[48] A. Krek *et al.*, “Combinatorial microRNA target predictions,” *Nat. Genet.*, 2005.

[49] R. C. Friedman, K. K. H. Farh, C. B. Burge, and D. P. Bartel, “Most mammalian mRNAs are conserved targets of microRNAs,” *Genome Res.*, vol. 19, no. 1, pp. 92–105, 2009.

[50] T. Fujiwara and T. Yada, “miRNA-target prediction based on transcriptional regulation,” *BMC Genomics*, 2013.

[51] U. ohler, S. Yekta, L. P. Lim, D. P. Bartel, and C. B. Burge, “Patterns of flanking sequence conservation and a characteristic upstream motif for microRNA gene identification,” *RNA*, 2004.

[52] D. Yue, H. Liu, and Y. Huang, “Survey of Computational Algorithms for MicroRNA Target Prediction,” *Curr. Genomics*, 2009.

[53] E. M. Mahen, P. Y. Watson, J. W. Cottrell, and M. J. Fedor, “mRNA secondary structures fold sequentially but exchange rapidly in vivo,” *PLoS Biol.*, 2010.

[54] D. Long, R. Lee, P. Williams, C. Y. Chan, V. Ambros, and Y. Ding, “Potent effect of target structure on microRNA function,” *Nat. Struct. Mol. Biol.*, 2007.

[55] D. M. Garcia, D. Baek, C. Shin, G. W. Bell, A. Grimson, and D. P. Bartel, “Weak seed-pairing stability and high target-site abundance decrease the proficiency of lsy-6 and other microRNAs,” *Nat. Struct. Mol. Biol.*, 2010.

[56] D. Betel, A. Koppal, P. Agius, C. Sander, and C. Leslie, “Comprehensive modeling of microRNA targets predicts functional non-conserved and non-canonical sites,” *Genome Biol.*, 2010.

[57] B. John, A. J. Enright, A. Aravin, T. Tuschl, C. Sander, and D. S. Marks, “Human microRNA targets,” *PLoS Biol.*, 2004.

[58] M. D. Paraskevopoulou *et al.*, “DIANA-microT web server v5.0: service integration into miRNA functional analysis workflows.,” *Nucleic Acids Res.*, 2013.

[59] X. Wang and I. M. El Naqa, “Prediction of both conserved and nonconserved microRNA targets in animals,” *Bioinformatics*, 2008.

[60] P. Loher and I. Rigoutsos, “Interactive exploration of RNA22 microRNA target predictions,” *Bioinformatics*, 2012.

[61] S. Bandyopadhyay and R. Mitra, “TargetMiner: MicroRNA target prediction with systematic identification of tissue-specific negative examples,” *Bioinformatics*, 2009.

[62] H. Liu, D. Yue, Y. Chen, S.-J. Gao, and Y. Huang, “Improving performance of mammalian microRNA target prediction.,” *BMC Bioinformatics*, 2010.

[63] M. Kertesz, N. Iovino, U. Unnerstall, U. Gaul, and E. Segal, “The role of site accessibility in microRNA target recognition,” *Nat. Genet.*, 2007.

[64] J. Krüger and M. Rehmsmeier, “RNAhybrid: MicroRNA target prediction easy, fast and flexible,” *Nucleic Acids Res.*, 2006.

[65] R. S. Michalski, J. G. Carbonell, and T. M. Mitchell, *Machine learning: An artificial intelligence approach*. Springer Science & Business Media, 2013.

[66] C. Cheng *et al.*, “A statistical framework for modeling gene expression using chromatin features and application to modENCoDE datasets,” *Genome Biol.*, 2011.

[67] K. Märtens, J. Hallin, J. Warringer, G. Liti, and L. Parts, “Predicting quantitative traits from genome and phenome with near perfect accuracy,” *Nat. Commun.*, 2016.

[68] A. L. Swan, A. Mobasheri, D. Allaway, S. Liddell, and J. Bacardit, “Application of Machine Learning to Proteomics Data: Classification and Biomarker Identification in Postgenomics Biology,” *omi. A J. Integr. Biol.*, 2013.

[69] D. B. Kell, “Metabolomics, machine learning and modelling: towards an understanding of the language of cells,” *Biochem. Soc. Trans.*, 2005.

[70] F. Eduati *et al.*, “Prediction of human population responses to toxic compounds by a collaborative competition,” *Nat. Biotechnol.*, 2015.

[71] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015.

[72] G. Hinton *et al.*, “Deep Neural Networks for Acoustic Modeling in Speech Recognition,” *IEEE Signal Process. Mag.*, 2012.

[73] C. Xiong, S. Merity, and R. Socher, “Dynamic Memory Networks for Visual and Textual Question Answering,” in *ICML 2016*, 2016.

[74] D. R. Kelley, J. Snoek, and J. L. Rinn, “Basset: Learning the regulatory code of the accessible genome with deep convolutional neural networks,” *Genome Res.*, 2016.

[75] Z. C. Lipton, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” *CoRR*, 2015.

[76] A. Fischer and C. Igel, “Training restricted Boltzmann machines: An introduction,” *Pattern Recognit.*, 2014.

[77] G. E. Hinton and R. R. Salakhutdinov, “Reducing the dimensionality of data with neural networks,”

Science (80-.)., 2006.

[78] D. H. Hubel and T. N. Wiesel, "The period of susceptibility to the physiological effects of unilateral eye closure in kittens," *J. Physiol.*, 1970.

[79] S. Hochreiter and J. J. Urgen Schmidhuber, "LoNG SHoRT-TERM MEMoRY," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[80] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," Sep. 2014.

[81] B. Alipanahi, A. Delong, M. T. Weirauch, and B. J. Frey, "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning," *Nat. Biotechnol.*, 2015.

[82] A. Kozomara and S. Griffiths-Jones, "MiRBase: Annotating high confidence microRNAs using deep sequencing data," *Nucleic Acids Res.*, vol. 42, no. D1, 2014.

[83] D. Karagkouni *et al.*, "DIANA-TarBase v8: A decade-long collection of experimentally supported miRNA-gene interactions," *Nucleic Acids Res.*, 2018.

[84] R. B. Darnell, "HITS-CLIP: Panoramic views of protein-RNA regulation in living cells," *Wiley Interdiscip. Rev. RNA*, 2010.

[85] D. D. Licatalosi *et al.*, "HITS-CLIP yields genome-wide insights into brain alternative RNA processing," *Nature*, 2008.

[86] D. W. Thomson, C. P. Bracken, and G. J. Goodall, "Experimental strategies for microRNA target identification," *Nucleic Acids Research*. 2011.

[87] S. W. Chi, J. B. Zang, A. Mele, and R. B. Darnell, "Argonaute HITS-CLIP decodes microRNA-mRNA interaction maps," *Nature*, 2009.

[88] R. Lorenz *et al.*, "ViennaRNA Package 2.0," *Algorithms Mol. Biol.*, 2011.

[89] H. Tafer *et al.*, "The impact of target site accessibility on the design of effective siRNAs," *Nat. Biotechnol.*, 2008.

CHAPTER 7 APPENDIX

7.1 APPENDIX I: Code for fetching miRNA and mRNA sequences from TarBase V8.

```
from bs4 import BeautifulSoup
import requests
import re

def deleteContent(pfile):
    pfile.seek(0)
    pfile.truncate()


mirna_file=open('mirna_list.txt')
mirna_list = mirna_file.read().split()
result = open("Mirna_batch_last_remain.csv","w+")
deleteContent(result)
result.write("Gene Symbol,miRNA_name,Sequence,Location"+'\n')
count = 0
list_not = []
for mirna in mirna_list:
    print(mirna)
    page=0
    #pagination loop
    try:
        for page_no in range(151):
            page = page+1
            if page>151:
                break
            source = requests.get("http://carolina.imis.athena-
innovation.gr/diana_tools/web/index.php?r=tarbasev8%2Findex&miRNAs%5B%5D=&miR
NAs%5B%5D={}&genes%5B%5D=&species%5B%5D=1&methods%5B%5D=8&sources%5B
%5D=1&sources%5B%5D=7&sources%5B%5D=9&publication_year=&prediction_score=0.7
5&sort_field=&sort_type=&query=1&page={}".format(mirna,page))
            soup = BeautifulSoup(source.text,'lxml')

            try:
                if int(soup.find('li',class_='active').text) < page: #last page check
```

```
            break
        except:
           page = 160


        main = soup.find('tbody')



        for tds in main.find_all('div'):
            try:
               td = tds.find_all('td',class_='first-level-block-bold')
               gene = td[0].text.strip()
               mirna = td[1].text.strip()

            except:
               continue
            tr = tds.find('a', attrs={'href':
re.compile("^http://www.ensembl.org/Homo_sapiens/Location")})

            if tr:
               link_temp = tr.get('href')
               link_contain=link_temp.split(';')

               g = link_contain[1].split('=')[1]

               r = link_contain[2].split('=')[1]
   #              print (r)
               #print(page)

               seq_fasta =
requests.get('http://asia.ensembl.org/Homo_sapiens/Export/Output/Location?db=core;flank
3_display=0;flank5_display=0;g={};output=fasta;r={};strand=feature;genomic=unmasked;p
eptide=yes;intron=yes;exon=yes;cdna=yes;coding=yes;utr5=yes;utr3=yes;_format=Text'.for
mat(g,r))
               seq = seq_fasta.text.split('\n')[1].replace('T','U')[::-1].replace('\n','')
               result.write("{},{},{},{}".format(seq,gene,mirna,r))
               #print("wrote")
               count=count+1
```

```
                print(count)



            else:
                continue
    except:
        list_not.append(mirna)
        continue
mirna_file.close()
result.close()
        # -*- coding: utf-8 -*-
```

7.2 APPENDIX II: Code for one-hotencoding of mRNA sequences

```
input_texts = []
target_texts = []
input_characters = set()
target_characters = set()
with open(data_path, 'r', encoding='utf-8') as f:
    lines = f.read().split('\n')
for line in lines[: min(num_samples, len(lines) - 1)]:
    target_text, input_text = line.split(',')
    # We use "tab" as the "start sequence" character
    # for the targets, and "\n" as "end sequence" character.
    target_text = '\t' + target_text + '\n'
    input_texts.append(input_text)
    target_texts.append(target_text)
    for char in input_text:
        if char not in input_characters:
            input_characters.add(char)
    for char in target_text:
        if char not in target_characters:
            target_characters.add(char)

input_characters = sorted(list(input_characters))
target_characters = sorted(list(target_characters))
num_encoder_tokens = len(input_characters)
num_decoder_tokens = len(target_characters)
max_encoder_seq_length = max([len(txt) for txt in input_texts])
max_decoder_seq_length = max([len(txt) for txt in target_texts])
print('Number of samples:', len(input_texts))
print('Number of unique input tokens:', num_encoder_tokens)
print('Number of unique output tokens:', num_decoder_tokens)
print('Max sequence length for inputs:', max_encoder_seq_length)
print('Max sequence length for outputs:', max_decoder_seq_length)
input_token_index = {'A': 0, 'C': 1, 'G': 2, 'U': 3}
target_token_index = {'\t': 4, '\n': 5, 'A': 0, 'C': 1, 'G': 2, 'U': 3}
encoder_input_data = np.zeros(
    (len(input_texts), max_encoder_seq_length, num_encoder_tokens),
    dtype='float32')
decoder_input_data = np.zeros(
    (len(input_texts), max_decoder_seq_length, num_decoder_tokens),
```

```
    dtype='float32')
decoder_target_data = np.zeros(
    (len(input_texts), max_decoder_seq_length, num_decoder_tokens),
    dtype='float32')
for i, (input_text, target_text) in enumerate(zip(input_texts, target_texts)):
    for t, char in enumerate(input_text):
        encoder_input_data[i, t, input_token_index[char]] = 1.
    for t, char in enumerate(target_text):
        # decoder_target_data is ahead of decoder_input_data by one timestep
        decoder_input_data[i, t, target_token_index[char]] = 1.
        if t > 0:
            # decoder_target_data will be ahead by one timestep
            # and will not include the start character.
            decoder_target_data[i, t - 1, target_token_index[char]] = 1.
```

7.3 APPENDIX III: Code for traing ANN Model.

```
from __future__ import print_function
import keras
from keras.models import Model
from keras.layers import Input, LSTM, Dense, Conv1D
from keras import regularizers
from keras.utils import plot_model
import numpy as np
batch_size = 50  # Batch size for training.
epochs = 100  # Number of epochs to train for.
latent_dim = 512  # Latent dimensionality of the encoding space.
num_samples = 19000
encoder_inputs = Input(shape=(None, num_encoder_tokens))
cnn = Conv1D(128,8, activation='relu')
cnn_output =cnn(encoder_inputs)
#dropout_layer = Dropout(0.5)
#decoder_outputs = dropout_layer(decoder_outputs)
encoder_dense_1 = Dense(128, activation='relu',kernel_regularizer=regularizers.l2(0.001))
encoder_dense_output = encoder_dense_1(cnn_output)
encoder = LSTM(latent_dim, return_state=True,recurrent_dropout=0.4, dropout = 0.1)
encoder_outputs, state_h, state_c = encoder(encoder_dense_output)
# We discard `encoder_outputs` and only keep the states.
encoder_states = [state_h, state_c]
# Set up the decoder, using `encoder_states` as initial state.
decoder_inputs = Input(shape=(None, num_decoder_tokens))
#cnn_decoder = Conv1D(128,8, activation='relu')
#cnn_decode_output =cnn_decoder(decoder_inputs)
#dropout_layer = Dropout(0.5)
#decoder_outputs = dropout_layer(decoder_outputs)
decoder_dense_1 = Dense(128, activation='relu',kernel_regularizer=regularizers.l2(0.001))
decoder_dense_output = decoder_dense_1(decoder_inputs)
# We set up our decoder to return full output sequences,
# and to return internal states as well. We don't use the
# return states in the training model, but we will use them in inference.
```

```
decoder_lstm = LSTM(latent_dim, return_sequences=True, return_state=True,recurrent_dropout=0.4, dropout = 0.1)
decoder_outputs, _, _ = decoder_lstm(decoder_dense_output,
                                     initial_state=encoder_states)
#dropout_layer = Dropout(0.5)
#decoder_outputs = dropout_layer(decoder_outputs)
#decoder_dense_1 = Dense(256, activation='relu',kernel_regularizer=regularizers.l2(0.001))
#decoder_outputs = decoder_dense_1(decoder_outputs)
decoder_dense = Dense(num_decoder_tokens, activation='softmax',kernel_regularizer=regularizers.l2(0.001))
decoder_outputs = decoder_dense(decoder_outputs)
# Define the model that will turn
# `encoder_input_data` & `decoder_input_data` into `decoder_target_data`
model = Model([encoder_inputs, decoder_inputs], decoder_outputs)
plot_model(model,show_shapes=True, to_file = '/output/model.png')
print(model.summary())
# Run training
model.compile(optimizer='rmsprop', loss='categorical_crossentropy',metrics=['mae', 'acc'])
callbacks = [keras.callbacks.TensorBoard(log_dir='/output/my_log_dir')]
model.fit([encoder_input_data, decoder_input_data], decoder_target_data,
        batch_size=batch_size,
        epochs=epochs,
        validation_split=0.2, callbacks=callbacks)
model.load_weights('s2s_batch_50.h5')
# Save model
model.save('/output/s2s_batch_50_cnn_encoder_dense_dim.h5')
# Next: inference mode (sampling).
# Here's the drill:
# 1) encode input and retrieve initial decoder state
# 2) run one step of decoder with this initial state
# and a "start of sequence" token as target.
# Output will be the next target token
# 3) Repeat with the current target token and current states
# Define sampling models
cnn_output = cnn(encoder_inputs)
encoder_dense_output = encoder_dense_1(cnn_output)
encoder_outputs, state_h, state_c = encoder(encoder_dense_output)
encoder_states = [state_h, state_c]
encoder_model = Model(encoder_inputs, encoder_states)
```

```
#cnn_decode_output =cnn_decoder(decoder_inputs)
#decoder_dense_output = decoder_dense_1(cnn_decode_output)
decoder_dense_output = decoder_dense_1(decoder_inputs)
decoder_state_input_h = Input(shape=(latent_dim,))
decoder_state_input_c = Input(shape=(latent_dim,))
decoder_states_inputs = [decoder_state_input_h, decoder_state_input_c]
decoder_outputs, state_h, state_c = decoder_lstm(
    decoder_dense_output, initial_state=decoder_states_inputs)
decoder_states = [state_h, state_c]
#decoder_outputs = dropout_layer(decoder_outputs)
#decoder_outputs = decoder_dense_1(decoder_outputs)
decoder_outputs = decoder_dense(decoder_outputs)
decoder_model = Model(
    [decoder_inputs] + decoder_states_inputs,
    [decoder_outputs] + decoder_states)
# Reverse-lookup token index to decode sequences back to
# something readable.
reverse_input_char_index = dict(
    (i, char) for char, i in input_token_index.items())
reverse_target_char_index = dict(
    (i, char) for char, i in target_token_index.items())
def decode_sequence(input_seq):
    # Encode the input as state vectors.
    states_value = encoder_model.predict(input_seq)
    # Generate empty target sequence of length 1.
    target_seq = np.zeros((1, 1, num_decoder_tokens))
    # Populate the first character of target sequence with the start character.
    target_seq[0, 0, target_token_index['\t']] = 1.
    # Sampling loop for a batch of sequences
    # (to simplify, here we assume a batch of size 1).
    stop_condition = False
    decoded_sentence = ''
    while not stop_condition:
        output_tokens, h, c = decoder_model.predict(
            [target_seq] + states_value)
        # Sample a token
        sampled_token_index = np.argmax(output_tokens[0, -1, :])
        sampled_char = reverse_target_char_index[sampled_token_index]
        decoded_sentence += sampled_char
```

```
        # Exit condition: either hit max length
        # or find stop character.
        if (sampled_char == '\n' or
           len(decoded_sentence) > max_decoder_seq_length):
            stop_condition = True
        # Update the target sequence (of length 1).
        target_seq = np.zeros((1, 1, num_decoder_tokens))
        target_seq[0, 0, sampled_token_index] = 1.
        # Update states
        states_value = [h, c]
    return decoded_sentence
for seq_index in range(100):
    # Take one sequence (part of the training set)
    # for trying out decoding.
    input_seq = encoder_input_data[seq_index: seq_index + 1]
    decoded_sentence = decode_sequence(input_seq)
    print('-')
    print('Input sentence:', input_texts[seq_index])
    print('Decoded sentence:', decoded_sentence)
```

7.4 APPENDIX IV: Code for finding surface area accessibility

```
# -*- coding: utf-8 -*-
import RNA
import requests, sys
from bs4 import BeautifulSoup
import sqlite3
conn = sqlite3.connect('linker.db')
c = conn.cursor()

def get_seq_by_GeneSymbol(GeneSymbol):
   c.execute("SELECT * FROM link WHERE
Gene_symbol=:Gene_symbol",{'Gene_symbol':GeneSymbol})
   return c.fetchall()

def get_seq_to_predict(GeneSymbol):
   GeneSymbol = GeneSymbol.upper()
   seq_to_predict=[]

   linker = get_seq_by_GeneSymbol(GeneSymbol)


   if linker is None:
       print("no match gene found")

   else:
      for line in linker:

         id_e = line[2]

         server = "https://rest.ensembl.org"
         ext = "/sequence/id/{}?".format(id_e)

         r = requests.get(server+ext, headers={ "Content-Type" : "text/plain"})

         if not r.ok:
           r.raise_for_status()
           sys.exit()

         link = ("https://asia.ensembl.org/Homo_sapiens/Export/Output/Gene?db=core;"
              +"flank3_display=0;flank5_display=10;t={};output=fasta;".format(id_e)
```

```
        +"strand=feature;param=utr3;genomic=unmasked;_format=HTML")

    utr = requests.get(link)

    soup = BeautifulSoup(utr.text, "html5lib")
    utr_split= soup.find('pre').text
    utr_split=utr_split.split('>')[1]


    utr_split = utr_split.split('\n')
    utr_seq=''
    for fasta in utr_split[1:]:
       utr_seq=utr_seq+fasta


    seq=r.text.replace('T','U')# l in negative(-l)


    if len(seq) > len(utr_seq)*1.5:
       l=len(seq)-int(len(utr_seq)*1.5)
    else:
       l= len(seq)-len(utr_seq)


    # compute minimum free energy (MFE) and corresponding structure

    n= RNA.pfl_fold_up(seq,16,40,80)
    for i in range(l,len(seq)):
       if n[i][4]>0.2:

          s = seq[i-24:i]
          seq_to_predict.append(s[::-1])
return list(set(seq_to_predict))
```