Genomic Language Processing Using Machine Learning

THESIS

Submitted to the Delhi Technological University for the award of the degree of

DOCTOR OF PHILOSOPHY

Submitted by

Rajkumar Chakraborty

Guide

Prof. Yasha Hasija, PhD
Department of Biotechnology
Delhi Technological University, DELHI

Department of Biotechnology
Delhi Technological University
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-110042, INDIA

January 2023

Genomic Language Processing Using Machine Learning

THESIS

Submitted to the Delhi Technological University for the award of the degree of

DOCTOR OF PHILOSOPHY

Submitted by

Rajkumar Chakraborty

Guide

Prof. Yasha Hasija, PhD
Department of Biotechnology
Delhi Technological University, DELHI

Department of Biotechnology
Delhi Technological University
(Formerly Delhi College of Engineering)
Shahbad Daulatpur, Main Bawana Road, Delhi-110042, INDIA

January 2023

Dedicated

to

My

Parents

DECLARATION

I hereby declare that the thesis entitled **"Genomic Language Processing Using Machine Learning"** submitted by me, for the award of the degree of ***Doctor of Philosophy*** to **Delhi Technological University (Formerly DCE)** is a record of bona fide work carried out by me under the guidance of Prof. Yasha Hasija.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this Institute or any other Institute or University.

Name: Rajkumar Chakraborty
Reg No: 2K18/PHDBT/01
Department of Biotechnology
Delhi Technological University (DTU)
Shahbad Daulatpur, Bawana Road, Delhi-110042
Place: Delhi
Date: 15.06.2023

CERTIFICATE

This is to certify that the thesis entitled **"Genomic Language Processing Using Machine Learning"** submitted by **Mr. Rajkumar Chakraborty** to **Delhi Technological University (Formerly DCE)**, for the award of the degree of "Doctor of Philosophy" in Biotechnology is a record of *bona fide* work carried out by him. Rajkumar Chakraborty has worked under my guidance and supervision and has fulfilled the requirements for the submission of this thesis, which to our knowledge has reached requisite standards.

The results contained in this thesis are original and have not been submitted to any other university or institute for the award of any degree or diploma.

Prof. Yasha Hasija, PhD
Department of Biotechnology
Delhi Technological University
Place: Delhi
Date: 15.06.2023

Prof. Pravir Kumar
Head of the Department/DRC Chairman
Department of Biotechnology
Delhi Technological University
Place: Delhi
Date: 15.06.2023

ABSTRACT

The purpose of developing biological language models (BLMs) is to enhance our capacity to comprehend and analyse biological sequences, such as DNA, RNA, and protein sequences. These sequences contain crucial information about the structure and function of living organisms and are involved in virtually every biological process. Nonetheless, analysing biological sequences can be difficult due to their complexity and enormous potential. Specifically, the functions and properties of a large number of coding and non-coding DNA and RNA sequences remain poorly understood. This thesis presents three objectives related to the application of natural language processing techniques in the field of bio-molecule sciences. The first objective involves using a combination of a Convolutional Neural Network (CNN) and a Long Short-Term Memory (LSTM) network, stacked in a sequence-to-sequence (Seq2Seq) architecture, to predict microRNA sequences from mRNA sequences. The microRNA are small, generally 28 bp long, non-coding RNAs that play a role in various physiological and disease processes. Identifying mRNA targeted by microRNAs is a challenge, and researchers often rely on computational programs to initially identify target candidates for subsequent validation. In this work, a neural network was trained to predict microRNA from the bound target segment in mRNA using a dataset of experimentally validated and cleaned microRNA-mRNA sequence pairs from TarBase v8. Convolutional neural networks (CNNs) were used to recognize patterns in mRNA segments and extract features, while long short-term memory (LSTM) networks in a seq2seq architecture were used to predict microRNA sequences. The model achieved an accuracy of 80% and was validated using experimentally verified microRNA-RNA pairs involved in skin diseases from an in-house database called miDerma, correctly predicting an average of 72% of the microRNAs from mRNA in each case. The package, called "model: A MicroRNA sequence prediction tool from RNA sequence based on CNNs, LSTMs, and seq2seq architecture," allows users to input a gene symbol and retrieves

the protein coding transcript's sequence from the Ensemble REST API to predict a list of microRNAs that may bind to potential target segments in the mRNA.

The second objective involves using natural language processing techniques, including an embedding layer, a CNN layer, and a bidirectional LSTM layer, to predict disordered regions in proteins. Intrinsically disordered regions (IDRs) are important for various physiological processes and diseases and play a complementary role to the functions of structured proteins. They can be identified through multiple experimental techniques, but these methods can be costly and time-consuming. As a result, researchers rely on computational strategies to predict probable IDRs/IDPs before conducting further validation through experimental studies. While there have been significant advancements in predicting long and short IDRs in recent years, there is still scope for algorithmic improvement. This study aims to improve the prediction of IDRs by using neural networks, specifically convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory (LSTM) networks, as well as natural language processing (NLP) techniques. The study also explores the use of different input sequence lengths and various embedding sizes for the CNN and LSTM models. The results show that the CNN and LSTM models outperform state-of-the-art techniques for predicting IDRs, with the LSTM model achieving the highest accuracy of 85.7%. The study also demonstrates the effectiveness of using NLP techniques for analyzing protein sequences and the importance of carefully selecting model architectures and hyperparameters to achieve good performance.

The third objective involves using an autoencoder, a type of deep learning architecture, to generate drug analogues by reconstructing chemical SMILES (Simplified Molecular-Input Line-Entry System) representations of molecules and varying the batch size and latent space dimensionality of the autoencoder. The design of drug analogues involves the creation of modified versions of existing drugs to improve their efficacy, stability, and safety. Deep learning techniques, such as autoencoders, can be used to generate new drug analogues through a process of chemical structure reproduction. In this study, an autoencoder was trained on chemical SMILES data from the ChEMBL database and used to generate 157 variants of the drug Vandetanib by adding noise to its latent representation and reconstructing the resulting compounds using a decoder. Molecular docking and dynamics simulations were then performed to determine which of these analogues had a higher binding affinity than Vandetanib. At least two of the analogues had a higher binding affinity than the control compound. While this model has the potential to generate a wide range of molecules, it may have difficulty generating molecules with SMILES strings longer than 80 characters due to a lack of training data of SMILES string length above 80 characters. The synthesis and laboratory testing of the generated molecules to determine their potential as drugs also presents a challenge. However, this study has the potential to make significant contributions to the field of automatic drug analogue prediction and could be a valuable addition to the current scientific literature.

The study presents several potential applications for its microRNA, protein disorder region finding, and drug analogue generation models. The microRNA prediction model could aid in the development of therapies for diseases by identifying microRNA sequences that regulate gene expression. The protein disorder prediction model could be used in drug design and protein engineering by identifying disordered regions in proteins that play a role in various protein functions. The drug analogue generation model has the potential to generate new drug analogues with desired properties and could be used in drug discovery and the optimization of existing drugs. Overall, this research has the potential to make significant contributions to biomedical research and could lead to the development of new therapies and drugs for diseases, as well as new bio-molecular language models for other tasks.

ACKNOWLEDGEMENT

I acknowledge the important role that cosmic events have played in bringing us to where we are today. From the big bang, which marked the beginning of the universe, to the formation of the first galaxies and the solar system, and eventually to the emergence of life on Earth, these events have shaped the history of the cosmos and continue to influence our world in countless ways. I am grateful for the endless mysteries that these events continue to reveal to us, and I stand in admiration of the majesty and grandeur of the universe. "Your presence serves as a reminder of the incredible journey we have undertaken since big bang, and the endless possibilities that lay ahead."

I would like to express my gratitude to ***Prof. Jai Prakash Saini****, Vice Chancellor, Delhi Technological University, Delhi for providing this opportunity to carry out this work in this prestigious institute. Further, I express my gratitude to* ***Prof. Yogesh Singh****, former Vice Chancellor, Delhi Technological University, Delhi for providing me opportunity and infrastructure to carry out this work.*

With pleasure, I acknowledge my deep sense of gratitude and indebtedness to ***Prof. Pravir Kumar****, Professor, DRC Chairman and Head, Department of Biotechnology, and* ***Prof. Jai Gopal Sharma****, Former Head of the Department, Department of Biotechnology, for providing me the infrastructure and smooth functioning of official work.*

I would like to express my deepest gratitude to my supervisor, ***Professor Yasha Hasija,*** *for her invaluable guidance, support and encouragement throughout my PhD journey. Her wisdom, expertise and patience have been an invaluable resource, and I am deeply grateful for the opportunity to work with her. Professor Hasija's mentorship has been instrumental in shaping my research, and her insights and guidance have helped me to grow as a researcher and a scholar. I am deeply thankful for her support and for the time and effort she has dedicated to my work.*

I wish to record my thanks and gratitude to my External DRC and SRC experts, ***Prof. Sonika Bhatnagar*** *(Department of Biological Sciences and Engineering, NSUT, Delhi),* ***Prof. Kamal K Aggarwal*** *(University School of Biotechnology, Guru Gobind Singh Indraprastha University, Delhi) and* ***Prof. A.K. Dubey*** *(Department of Biological Sciences and Engineering, NSUT, Delhi), for their valuable guidance, critical and constructive discussion during this work.*

I would like to thank my fellow lab mates especially ***Jaishree Meena****,* ***Bidhisa Bhowal****,* ***Neha Kumari*** *and* ***Priya Rai*** *for helping and encouraging me throughout my research. Thanks are due to the wonderful friends in my life who were always on the standby to bring me to positivity, hope and smiles when things didn't seem favoring, and it seemed a far-fetched journey. Special mention is deserved for all my friends, especially* ***Sunil Kumar****,* ***Bharat Manna****,* ***Richa Nayak****,* ***Sidharth Sharma****, and* ***Rohan Gupta*** *who have always been my guiding light when I didn't believe in myself. Moreover, I wish to thank my juniors, namely* ***Swati Sharma*** *and* ***Shashank Kumar Singh*** *for their support during my research work.*

I would also like to thank the Senior Management and technical staff of DTU, who help me to carry out my research work.

Lastly, I want to express my gratitude to my brother ***Rakesh Kumar Chakraborty****, who always stood beside me in my ups and downs.*

Finally, I want to dedicate this thesis to my parents, ***Mr. Debasis Chakraborty,*** *and* ***Mrs. Madhu Chhanda Chakraborty****. Their love and support have been a constant source of strength, and I am deeply thankful for their constant presence in my life. Their sacrifices and dedication to my education have not gone unnoticed, and I am forever grateful for all that they have done to help me achieve my dreams.*

List of Figures

List of Tables

TABLE OF CONTENTS

CHAPTER I

OVERVIEW OF THE THESIS

Chapter 1. Overview of the thesis

i. Rationale of the study

A computer is a machine that follows a set of mathematical rules and can perform complex calculations quickly, but it lacks the ability to comprehend and interpret data in the same manner as humans. A concept must be able to be expressed mathematically for it to be usable. This limitation restricts the capacity of computers to comprehend and work with natural languages. The term "natural language processing" refers to the application of computational linguistics to problems in the real world involving multiple languages. This includes efforts to teach computers to understand languages and interpret them using algorithms. There are language advancement tools that use machine learning algorithms, such as Google's keyboard and Grammarly. Language modelling is also used by translation systems to support multiple languages. Utilizing natural language properties, computational linguistics aims to convert unstructured data into a form that computers can comprehend in response to the proliferation of text data. Without explicit programming, machine learning is the process of enabling computers to learn and adapt based on data. It has the potential to affect multiple aspects of our daily lives and is utilised in numerous fields, including biology and healthcare. Machine learning can be used to extract meaningful information from large data sets and the genetic sequences DNA and RNA contain symbols that convey meaning and are organised according to specific rules and structures. These sequences are essential to the development and function of living organisms and can evolve and adapt through mutation over time. Natural language processing (NLP) techniques can aid researchers in gaining a deeper comprehension of the functions and structures encoded in these sequences. NLP techniques, such as tokenization, part-of-speech tagging, and syntactic parsing, can be used to analyse the relationships between the tokens in a genetic sequence and their context. In addition, NLP can be used to analyse the words used to describe genetic sequences and the biological processes in which they are involved, including the extraction of key terms and concepts from scientific papers and other data sources. Protein molecules, which are essential for the structure, function, and regulation of the tissues and organs of the body, can be viewed as a type of language, with amino acids serving as "letters" and proteins as "words." Using NLP techniques such as n-grams, researchers have examined the evolution of domain architectures in proteins and attempted to predict their structures and functions based on their semantics. However, there is still much about the semantics of proteins that is unknown. The majority of research has centered on the lexical, syntactic, and semantic aspects of biological sequences, but these sequences have their own unique linguistic properties that have not yet been thoroughly investigated. To improve the use of natural language processing (NLP) in the analysis of biological sequences, it may be necessary to develop biological language models that can automatically capture linguistic features using computational biology techniques. SMILES (Simplified Molecular-Input Line-Entry System) is a language that uses symbols and characters to describe the structure of a chemical molecule. To represent atoms and bonds in a molecule, this language adheres to specific grammar rules and employs a specific vocabulary. SMILES is useful for storing, retrieving, and analysing chemical compounds and can be read and understood by both humans and computers. It has also led to the creation of numerous computational chemistry applications and tools. SMILES can be used to extract properties about chemical compounds using machine learning techniques. To handle biological and chemical sequences, various machine learning/deep learning frameworks influenced by natural language processing were developed in this study.

ii. Aim and objectives

1. Aim:

Genomic Language Processing Using Machine Learning

2. Objectives

a. Objective 1: Predicting microRNA sequence using CNN and LSTM stacked in Seq2Seq architecture.

For predicting miRNA sequences from mRNA sequence CNN was used to extract sequence features from an input mRNA segment and supplied these features to a sequence-to-sequence architecture-stacked LSTM system. Typically, this design comprises of two LSTMs, encoder and decoder. The retrieved features from CNN are sent one-by-one in time steps to the encoder LSTM to obtain a fixed-dimensional vector of internal states; these internal states are then used as initial states by the decoder LSTM to extract the output sequence based on its initial state vector(Figure 1.1). This LSTM is an RNN language model, but its training is dependent on the input sequence vector. The encoder processes the features from targeted mRNA sequences and provides its hidden internal states, which are then used by the decoder LSTM as a condition or context for miRNA sequence prediction. Here our model was able to predict on average 72% of microRNA from miRNA in each cases correctly. Here CNNs, LSTMs are used in seq2seq architecture which are often used in developing chatbots. It this way an attempt was made to bring the natural language processing models to process core language of nucleotide i.e. A, T, G, C.

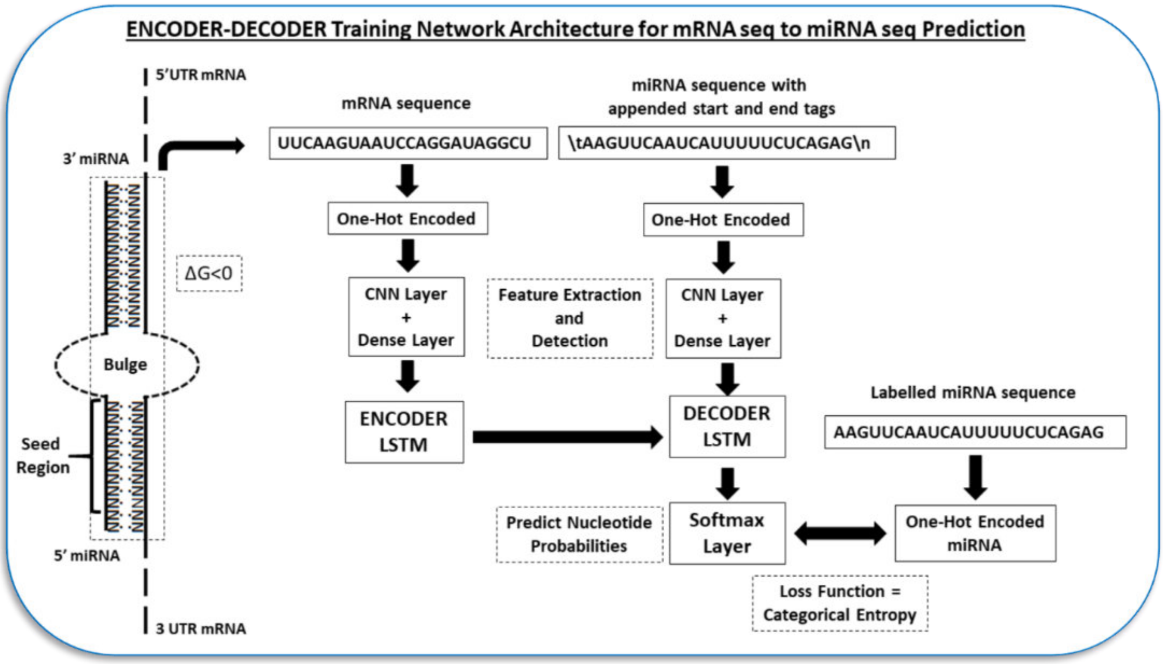
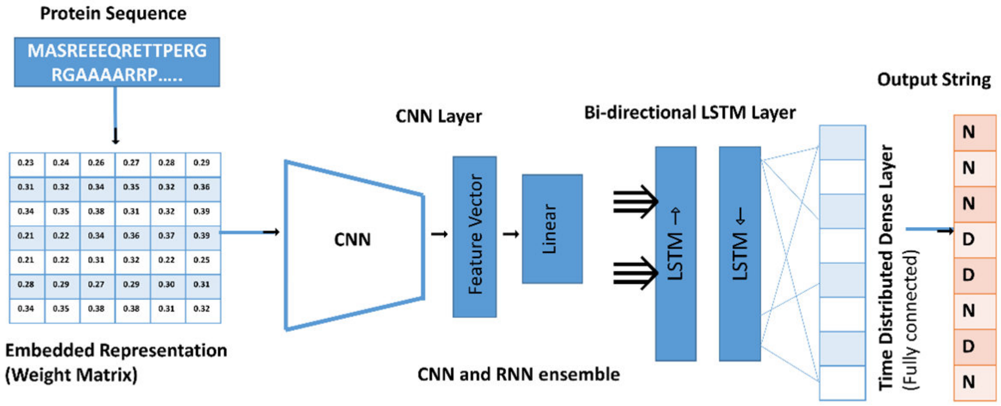


Figure 1.1:Architecture of the proposed model used for predicting miRNA from RNA sequence.

b. Objective 2: Protein disordered region prediction using natural language processing techniques.

Intrinsically Disordered Regions (IDRs) are the regions in proteins that do not possess well organized two dimensional or three-dimensional structures under physiological conditions. These regions exist extravagantly in each domain and are concerned with numerous protein functions. Our proposed model consists of four interconnected layers: an embedding layer, a CNN layer, an LSTM layer, and a basic ANN with a softmax activation function (Figure 1.2). The embedding layer is used to extract characteristics of individual amino acids. The CNN layer then extracts local features from the output of the embedding layer using multiple filters. The bidirectional LSTM layer generates global features based on the single amino acid features and local features obtained previously. A set of three neurons is applied to each amino acid location to evaluate the likelihood that the amino acid at that location is ordered or disordered based on the local and global information gathered. This model has outperformed other state-of-the-art techniques for predicting IDRs. To improve speed, depthwise separable 1D CNNs were used, which have fewer weights than vanilla 1D CNNs and can make predictions on a genomic scale more quickly. When processing 10000 random sequences from the ModiDB database on an Intel Xeon CPU W-2133, the model took 71 minutes without parallel processing. This approach suggests that amino acid sequences can be treated as letters in the language of the structural proteome and could have numerous applications in biomedical sciences.


Protein Sequence
MASREEEQRETTPERG RGAAAARRP.....
Embedded Representation (Weight Matrix)
CNN
CNN Layer
Feature Vector
Linear
Bi-directional LSTM Layer
LSTM →
LSTM ←
CNN and RNN ensemble
Time Distributed Dense Layer (Fully connected)
Output String
N
N
N
D
D
N
D
N

0.23	0.24	0.26	0.27	0.28	0.29
0.31	0.32	0.34	0.35	0.32	0.36
0.34	0.35	0.38	0.31	0.32	0.39
0.21	0.22	0.34	0.36	0.37	0.39
0.21	0.22	0.31	0.32	0.22	0.25
0.28	0.29	0.27	0.29	0.30	0.31
0.34	0.35	0.38	0.38	0.31	0.32

Figure 1.2: Architecture of the proposed model. An ensemble of Embedding, CNN and Bi-LSTM for extracting features.

c. Objective 3: Utilizing Deep Learning to Explore Chemical Space for Drug Analogues Generation

The goal of medicinal chemistry is to improve on existing drug molecules or to create new ones for use in medicine. This is frequently accomplished through the use of Analogues design, which entails creating similar but slightly modified versions of existing molecules. Generative models that use various representations of molecules, such as SMILES codes and molecular graphs, have been developed to aid in the search for hits in the unexplored chemical space. An autoencoder, a deep learning architecture, was used in this objective to create new drug Analogues (Figure 1.3). This was accomplished by reconstructing chemical SMILES and varying batch sizes to control the distribution of the autoencoder's latent space. This architecture has a small number of parameters and has the potential to generate a wide variety of molecules. the autoencoder was used to generate 157 variants/Analogues of Vandetanib by adding noise to its bell-shaped latent representation and reconstructing the resulting compounds using the decoder. Molecular docking and dynamics simulations were performed to determine which of these analogues possessed a higher binding affinity than Vandetanib. At least two of the analogues had a higher binding affinity than the control compound, according to the results.

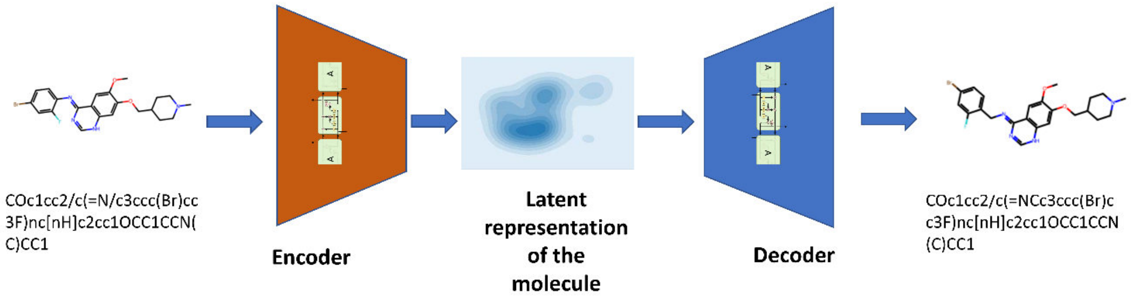


Figure 1.3: Autoencoder architecture for predicting Analogs of molecules.

CHAPTER II

BACKGROUND

Chapter 2. Background

i. Learning languages

“Learning another language is not only learning different words for the same things but learning another way to think about things.” words by well accomplished journalist Flora Lewis.

Language plays a central role in human life, culture, and cognition. It is used for communication, forming and expressing ideas, and conveying emotion through subtle nuances. Since the time of Plato, philosophers have been interested in how language performs these functions[1]. Early analytic philosophers, such as Frege and Russell, made significant contributions to the systematic study of meaning through formal semantics[2]. Other philosophers, such as Austin, Searle, and Grice, also studied the relationship between meaning and use[3], [4]. Linguists, on the other hand, have focused on analyzing the various forms of human language in terms of sound structure, morphology, and syntax. The study of linguistic meaning truly flourished when these two fields of study converged, as a thorough understanding of language form, including pronunciation and intonation, is necessary for a complete understanding of linguistic meaning[5]. Therefore, the study of linguistic meaning is interdisciplinary in nature.

Non-human creatures are capable of communication, but none have a communication system as complex as human language. Nonverbal methods of communication, such as facial expressions, vocalizations, and gestures, are common among non-human animals. Chimpanzees, gorillas, and orangutans can communicate through various facial expressions and vocalizations, while bees use specific movements to communicate about the location of food. Birds also use vocalizations, primarily for territorial purposes or to attract mates.

However, none of these forms of communication approach the complexity of human language. Language is a unique characteristic of humans, and syntax and double articulation are two key factors that distinguish linguistic from non-linguistic communication[6]. Each language consists of thousands of signs that combine form and meaning. In spoken languages, the form of a sign is a series of sounds, while in written languages it is a series of letters. Sign languages, used by deaf individuals, use specific combinations of hand and body movements as the form of a sign[7]. Double articulation refers to the fact that the forms of a language's thousands of signs are constructed from a small inventory of meaningless sounds, typically between 10 and 100[8].

English has approximately 50 sounds, approximately half of which are vowels and half of which are consonants[9]. This number may vary slightly depending on the dialect and method of phonological analysis. None of these sounds are related to the meanings of words. For example, the word "soot," pronounced /sUt/, is formed by replacing the letter "I" in the word "sit," pronounced /sIt/, with the letter "U." "Soot" is defined as "a black powdery form of carbon formed when coal, wood, or oil is burned, which rises to the surface in fine particles with the flames and smoke." The units /sIt/ and /sUt/ begin and end with the same sounds (/s/ and /t/), but the vowel in the middle is different. However, this difference in pronunciation does not affect the meaning of the word, which is unrelated to the meaning of "sit," or "to assume a position of rest supported by the buttocks." Similarly, changing the letter "t" in "sit" to the letter "k" produces the word "sick," pronounced /sIk/, which has a completely different meaning: "afflicted by an illness." This illustrates that the meanings of words in English are not directly related to their pronunciation.

In a language without double articulation, each sign would have a unique form, and the number of distinct forms would be equal to the number of signs[10]. For example, a communication system in which each sign's form is a unique cry would have a large number of distinct forms, as humans are able to distinguish hundreds of different screams. However, such a system would be impractical, as it would require a large number of distinct sounds and be sensitive to noise. It would also be inefficient, as it would require the production and recognition of a large number of distinct sounds. Double articulation, on the other hand, allows for a much more efficient and practical communication system, as it uses a small number of meaningless sounds to construct the forms of a large number of signs.

Double articulation allows humans to create languages with a large number of signs, but the inventory of signs in a language is necessarily finite[11]. This is because the number of sounds in a language is usually limited to between 10 and 100, so it would be impractical to have hundreds of thousands of distinct signs unless they were very long. Additionally, the human memory has an upper limit on the number of signs that it can remember. For example, separate signs for "man killed lion" and "lion killed man" would not be practical in a language. The total number of isolated signs in a human language is typically limited to around 10,000-20,000, and with this number of signs, it is not possible to express an infinite number of meanings unless they are combined in various ways.

Syntax is the mechanism that allows humans to communicate any and all ideas they can think of[12]. It is used to combine signs with relatively simple meanings to create sign combinations with more complex meanings. For instance, to express the meaning "man killed lion," we can combine the signs for "man," "kill," "past," and "lion." We can also use the same signs in a different order to express the meaning "lion killed man." In English, the sign sequences "man killed lion" and "lion killed man" are sentences, and a language has an infinite number of sentences. Any sentence in any language can be made longer by adding additional words and clauses, such as "the woman said that the man killed the lion," "the old woman said that the young man killed the lion," "the old woman said that the young man killed the lion, the girl believed that the old woman said that the young man killed the lion that ate the antelope," and so on. Syntax allows for an almost limitless range of expression and complexity in language. Syntax is a system that allows humans to create an infinite number of sentences from a finite set of building blocks. Without syntax, we would only be able to express the meanings associated with individual signs, and the number of possible meanings would be limited to the number of signs in the language.

To summarize, a language is a set of words used by a group of people to communicate their thoughts and ideas. Humans learn this vocabulary as part of their development and it remains relatively stable, with only a few new words added each year[13]. If a person encounters a new word, they can consult resources such as dictionaries to find out its definition. Once a person becomes familiar with a new term, it is added to their vocabulary and can be used in future conversations. Syntax allows for an almost limitless range of expression and complexity in language by enabling the creation of complex meanings from simple building blocks.

ii. Natural Language Processing

A computer is a machine that follows precise mathematical rules and is able to perform complex calculations quickly. However, it lacks the ability to interpret and understand concepts in the way that humans do. In order to work with a concept, a computer must be able to express it in the form of a mathematical model. This limitation restricts the range of natural language that a computer is able to work with.

"Natural language processing" refers to the application of computational linguistics in real-world applications that deal with languages with various structural components[14]. An attempt to train the computer to comprehend languages can be made and then we can expect it to understand them using appropriate, effective algorithms. Google's keyboard, which provides auto-correct recommendations, word predictions (upcoming terms), and other features and Grammarly which is an excellent tool for professionals and content writers who want to ensure that their postings appear professional are well known example of NLP application. They employ ML algorithms to recommend the optimal ratios of massive vocabulary, intonation, and other characteristics, guaranteeing that the written material is professional and thoroughly interests the reader[15]. Language modelling is a strategy used by translation systems to handle multiple languages successfully[16], [17]. Linguists are people who investigate linguistic patterns and linguistic characterization[18], [19]. The development of computational linguistics was spurred by the explosion of textual data[20]. Wikipedia is the best textual source accessible. The early interest in understanding data patterns, Parts-of-Speech (POS) labelling, and simplifying data processing for a variety of applications in the banking and financial industries[21], [22], educational institutions, and so on gave rise to the field of computational linguistics. NLP attempts to convert unstructured data into computer-readable language by replicating the properties of natural language[23]. Machines require complex algorithms to interpret any text content and extract relevant information from it. The obtained data is then used to teach machines more advanced logic based on natural language. Natural language processing uses syntactic and semantic analysis to instruct machines by uncovering and recognising data patterns[24]–[26]. The following are the steps involved:

- Syntax: Natural language processing applies a range of algorithms to follow grammatical rules and extract meaning from any form of text material.
- Lemmatization, morphological segmentation, word segmentation, part-of-speech tagging, parsing, sentence breaking, and stemming are common syntactic techniques. Through the immensely challenging process of semantics, machines seek to interpret the meaning of every component of any information, both individually and in context.
- Despite the fact that semantic analysis has made significant progress since its initial binary orientation, there is still much room for advancement. NER, or Named Entity Recognition, is a key process in the operation that splits text material into specific groupings.

- The method's next stage, "word sense disambiguation," deals with context-based meaning.
- The final step of the process, "natural language production," uses prior data to extract meaning and translate it into human languages.

Every day, more data are generated, which needs analysis and recording. NLP allows computers to interpret this data and translate it into human-readable languages. Many of these data sets, such as government statistics and medical information, are unstructured. NLP helps computers appropriately arrange/structure them[27], [28]. Following that, computers deduce meaning by analysing texts and sounds. Not only is the method mechanised, but it is also consistently correct. NLP is a technique for improving computers' ability to understand human language. Databases provide highly structured information. The Internet, on the other hand, has no structure and is completely unstructured. The ultimate goal of NLP is to comprehend and model human language[29], [30]. Non-linear dialogues are being perfected by Google Duplex and Alibaba's voice assistant, for example[31], [32]. Non-linear conversations are similar to how individuals communicate in real life to some extent. For example if we discuss cats in the first phrase, then abruptly flip to Tom before returning to the main subject. The listener is aware of the leap that occurs. In computer science, this expertise is currently inadequate. Because unstructured data is developing so quickly, NLP specialists are in high demand right now. Underneath this unstructured data is a wealth of knowledge that can help businesses grow and succeed. Keeping a watch on Twitter trends, for example, may help us understand the difficulties that our communities confront as well as be useful in emergency situations[33]–[36].

Text data is an unstructured data, the vocabulary of which is very large. Different words can have the same meaning, or even the same words can have different means, unlike programming languages such as python which are very well structured. As we will see in the next sections that images have to be converted into arrays based on the pixel intensities, to be used in machine learning algorithms. In the same way, to input text data to a machine learning algorithm, the text also needs to be vectorized,. i.e, it needs to be converted into numeric representations with features before applying any machine learning algorithm.

Broadly, there are three steps for any NPL task:

1) Data aggregation, where we accumulate or compile text data, i.e articles, abstracts or gene or protein sequences etc, relevant to the objective of the task.
2) Representing the text data in vector format.
3) Use a machine-learning algorithm to analyze it.

1. Text Preprocessing

Text preprocessing is necessary for NLP because it helps to clean and prepare the text data for further analysis and processing. It is important to remove any noise or irrelevant information from the text, and to standardize the text data so that it can be easily and accurately processed by NLP algorithms [37], [38]. Text preprocessing can improve the accuracy and effectiveness of NLP algorithms by removing any irrelevant or noisy data that can interfere with the analysis. It can also improve the consistency and readability of the text data, making it easier for algorithms to process and understand[39], [40]. In addition, text preprocessing can help to reduce the complexity of the text data, making it easier to work with and process. This can improve the efficiency of NLP algorithms and reduce the computational resources required for text analysis[41], [42].

a. Noise Removal

Noise removal is necessary for NLP because it helps to remove irrelevant or misleading information from the text data. This can improve the accuracy and effectiveness of NLP algorithms by reducing the amount of noise and interference in the text data. Noise can be caused by a variety of factors, such as spelling mistakes, punctuation, numbers, special characters, or stop words. These elements can interfere with the analysis of the text data and affect the accuracy of NLP algorithms. By removing noise, NLP algorithms can focus on the relevant and important information in the text data, and process it more accurately and effectively. This can improve the overall performance and reliability of NLP algorithms, and enable them to produce more accurate and useful results[43].

Overall, noise removal is an essential step in the NLP process, as it helps to improve the accuracy and effectiveness of NLP algorithms by removing irrelevant or misleading information from the text data.

There are several techniques to remove noise during text preprocessing in NLP[44], [45]:

- Stop words removal: Stop words are common words in a language that do not add much value to the meaning of a sentence. Removing these words can reduce noise and improve the effectiveness of text processing.
- Stemming and lemmatization: Stemming and lemmatization are techniques to reduce inflected words to their word stem, base or root form. This can reduce noise by reducing the number of variations of a word and improving the consistency of the text.
- Regular expression: Regular expressions are a set of rules used to match patterns in text. They can be used to identify and remove unwanted characters or words, such as punctuation, numbers, or special characters.
- Case normalization: Case normalization involves converting all words to the same case, such as lowercase or uppercase. This can improve the consistency and readability of the text, and reduce noise caused by different variations of the same word.
- Spelling correction: Spelling correction involves identifying and correcting spelling mistakes in the text. This can reduce noise and improve the accuracy of text processing.

2. **Text to Features (Feature Engineering on text data)**
3. **Bag of words**

Suppose there are two sentences in a text corpus:

1) ‘Metformin is used for treating diabetes.’

2) ‘Amlodipine is used for treating hypertension.’

Table 2.1: Vector representation of text.

‘Metformin’	**‘Amlodipine’**	**‘is’**	**‘used’**	**‘for’**	**‘treating’**	**‘diabetes’**	**‘hypertension’**
1	0	1	1	1	1	1	0
0	1	1	1	1	1	0	1

Table 2.1: Vector representation of text.

In table 2.1, the columns represent the vocabulary of the corpus, and rows represent the sentences. The sentence will be encoded based on the presence or absence of words in the vocabulary. In Table 2.1, the first row is an encoded version of the first sentence. The values in the columns for ‘Amlodipine’ and ‘hypertension’ are ‘0’, as they don't occur in the first statement, and values in rest of the columns are ‘1’, which is their counts in the respective sentence. In the same way, the vectorized representation of second sentence is made. If in case a word repeats twice in the text, then its vector representation will be ‘2’. This type of representation of text is known as ‘one-hot-encoding’ or ‘Bag’ of words. These features generally have high dimensions as vocabulary size can reach more than thousands and maximally filled with zeros, called sparse matrices. Machine learning algorithms can be applied on them, since the text is converted into features. Simple mathematical operations like finding their distance in space using Euclidean distance etc can also be performed on text[46], [47].

While ‘Bag’ of words can convert the text into vectors, it generalizes all the words as same, i.e., it does not have any weightage for the word itself, based on, for example, the importance of the word in a sentence, uniqueness of a word in the corpus etc.

4. TF-IDF

One way to assign importance to words and their uniqueness is by calculating the term frequency-inverse document frequency (TF-IDF) of the words. The term frequency (TF) measures the importance of a word within a specific text instance, such as a sentence, paragraph, or article. It is calculated by counting the number of occurrences of the word in the text instance. The inverse document frequency (IDF) measures the importance of a word in the entire corpus, or collection of texts[48]. It is calculated by taking the inverse of the frequency of documents in which the word appears. By combining the TF and IDF, the overall importance and uniqueness of a word in the corpus can be determined. IDF is calculated as:

$IDF_{(word)} = \log(D/I_{(word)})$

where $IDF_{(word)}$ is the importance of a word in the document, D is the total number of instances in the corpus, and $I_{(word)}$ is the number of text instances of the particular word.

Simply put, IDF is the inverse of the number of occurrences of a word in a corpus. The inverse document frequency (IDF) is calculated by taking the inverse of the frequency of documents in which a word appears in order to assign more importance to unique words. Common words such as auxiliary verbs and articles have a high frequency of occurrence but do not provide significant information, while rare words such as the names of drugs, proteins, or diseases have a lower frequency but can be highly informative. The value of a word in a text instance is obtained by multiplying its term frequency (TF) and its IDF, thereby preserving both the importance and uniqueness of the word[49], [50].

5. Word Embeddings

In natural language processing (NLP), word embeddings are a type of representation that is used to represent words and phrases in a way that captures the semantic and syntactic meaning of those words and phrases. Word embeddings, in contrast to more conventional methods of representing words as separate units, such as one-hot encodings, are able to take into account the relationships that exist between words as well as the context in which they are employed.

Word embeddings are typically developed by teaching a neural network to process a large amount of textual data from a corpus. The network acquires the ability to map each word to a continuous, high-dimensional vector that represents both the meaning of the word and its connection to the other words contained in the corpus. After that, the vectors that were produced can be used as input for other natural language processing activities, such as language modelling, text classification, and machine translation[51], [52].

One of the primary benefits of word embeddings is that they can capture complex relationships between words, which are difficult to represent using other methods. This is one of the most significant advantages of word embeddings. For instance, the same meaning can be conveyed by two words, but each may have a distinct connotation depending on the context in which it is used. Word embeddings can provide a representation of the words that is richer in nuance and can take into account these differences[53].

Word embeddings are not only very effective, but they are also very efficient since they can be used to represent a large number of words while utilising a relatively small number of dimensions. Because of this, they are ideally suited for use in natural language processing activities, such as language translation and sentiment analysis, that require the processing of large amounts of text data[54], [55].

Word embeddings are a useful tool for representing words and phrases in NLP tasks, and they are used in a wide variety of natural language processing applications. In general, they are a powerful tool.

iii. Important tasks of NLP

1. Text Classification

Text classification is a common task in natural language processing (NLP), where the goal is to automatically assign a piece of text to one or more pre-defined categories based on its content. For example, a text classification system might be trained to assign a movie review as positive or negative, or to classify a news article by its topic (e.g., sports, politics, technology). There are many different approaches to text classification, but most commonly, NLP techniques are used to extract features from the text data, such as the words used, the length of the text, or the presence of certain phrases or words. These features are then used as input to a machine learning model, which is trained to predict the correct category for a given piece of text. The most common machine learning algorithms used for text classification are support vector machines (SVMs) and naive Bayes classifiers. SVMs are a type of linear classifier that finds the hyperplane in the feature space that maximally separates the different classes. Naive Bayes classifiers, on the other hand, are based on the principle of maximum likelihood, and make predictions by computing the probability that a given piece of text belongs to each possible class. One of the key challenges in text classification is the large amount of data that is typically required to train a high-performing model. This can make it difficult to apply text classification to tasks with a small amount of data, or to tasks where the categories are not well-defined or highly imbalanced[56], [57].

Overall, text classification is a useful and widely used technique in NLP and has many applications in areas such as sentiment analysis, spam detection, and topic modeling. Some examples of text classification using natural language processing (NLP) include:

Sentiment analysis: This is a common application of text classification, where the goal is to automatically determine the sentiment of a piece of text, such as a movie review or customer feedback. The text is classified as positive, negative, or neutral based on the words and phrases used.

Spam detection: Text classification can be used to automatically identify spam messages in email or social media. The system is trained on a dataset of known spam and non-spam messages and uses features such as the presence of certain words or phrases to make predictions[58].

Topic modeling: Text classification can be used to automatically assign a piece of text to one or more pre-defined topics, such as sports, politics, or technology. The system is trained on a dataset of texts labeled with their corresponding topics and uses features such as the words used and the length of the text to make predictions[59].

Language detection: Text classification can be used to automatically identify the language of a piece of text. The system is trained on a dataset of texts labeled with their corresponding languages and uses features such as the words used and the character n-grams to make predictions[60].

Some of the main advantages of text classification using natural language processing (NLP) are[61], [62]:

- **Automation:** Text classification allows for the automatic assignment of text to pre-defined categories, which can save time and resources compared to manual classification.
- **Improved accuracy:** Text classification can help to improve the accuracy of predictions compared to manual classification, as the system is trained on a large amount of data and can learn complex patterns and relationships in the text.
- **Efficiency:** Text classification can process large amounts of text data quickly and efficiently, making it well-suited to tasks that require processing large amounts of data.
- **Scalability:** Text classification systems can be easily scaled up to handle larger datasets or more categories, making them well-suited to tasks that may evolve over time.

- **Versatility:** Text classification can be applied to a wide range of tasks and domains, making it a versatile and widely-used technique in natural language processing.

Overall, text classification using NLP offers many advantages compared to manual classification and is a valuable tool for many applications in natural language processing.

iv. NLP and Machine learning

The art of enabling machines to form rules and find trends from data without explicitly programming them is known as machine learning. In machine learning, computers gather intelligence from the data and adapt to their situations based on their experience. The applications of machine learning are tremendous because machine learning has the potential to affect every domain of our daily lives[63].

Throughout history, humans have utilized machines to reduce the effort required and increase efficiency in completing tasks. Machine learning is a continuation of this trend, with ongoing research being conducted globally to prepare for future challenges. Companies have begun utilizing bots and web crawlers, developing systems with the capability to learn independently. The potential applications of machine learning are vast and have the potential to significantly alter the world as we know it[64].

Machine learning is required to extract meaningful information from big data sets. It can also be used to alleviate the burden of solving many biological problems. Today we have numerous examples of applied machine learning, from protein structure prediction, image recognition, drug molecule development, drug repurposing, protein-protein interaction, finding SNPs in genomic sequences, cancer detection, solving problems in system biology, biological text mining and many more. Protein structure prediction algorithms before machine learning had an accuracy of around 70%. But machine learning has pushed the boundaries in this field as well, increasing the accuracy to a much better figure of 85%[65]. With the advancement in high throughput technologies in biology and increasing number of publications, the data is growing in size more rapidly than it ever has. Machine learning systems can use this data to perform text-mining for research purposes. These systems work just like the human brain does but have the computational power that far surpasses our capabilities. Outputs from such a system can also be used for identifying microscopic cellular images and performing various medical studies[62], [66].

1. Types Of Machine Learning Systems

There are two main types of machine learning methods: Supervised and Unsupervised[67].

2. Supervised Learning

Supervised learning is when our model gets trained on a pre-labeled dataset, and develop systems to predict outcomes for unforeseen data. The data used to train the system in supervised learning contains both input and output values.

This type of machine learning is then further classified into classification and regression. In classification, output has discrete values. Our system will have to classify these values into their respective groups correctly. For example, classifying SNPs into functional or benign class; or predicting the day to be hot or cold. In regression, output has continuous values, so the system has to predict an output value close to the actual value. For example, prediction of temperature, stock prices, etc[68].

3. Unsupervised Learning

Unsupervised learning differs from supervised learning in that it is used for clustering or inferring patterns, trends, or relationships within a dataset without any prior reference or knowledge of the labeled data. Unsupervised learning can be further divided into clustering and anomaly detection. Clustering is the most common technique in unsupervised machine learning, involving the grouping of data into clusters with similar characteristics. Anomaly detection, on the other hand, involves the identification of deviations from a general trend. For example, if a system is trained to recognize white cars on the road, the detection of a red car would be considered an anomaly. This method is useful for detecting bank frauds, human errors in data entry, and many other applications[69].

Figure 2.1and Figure 2.2 pictorially depict the differences between the supervised and unsupervised learning algorithms.

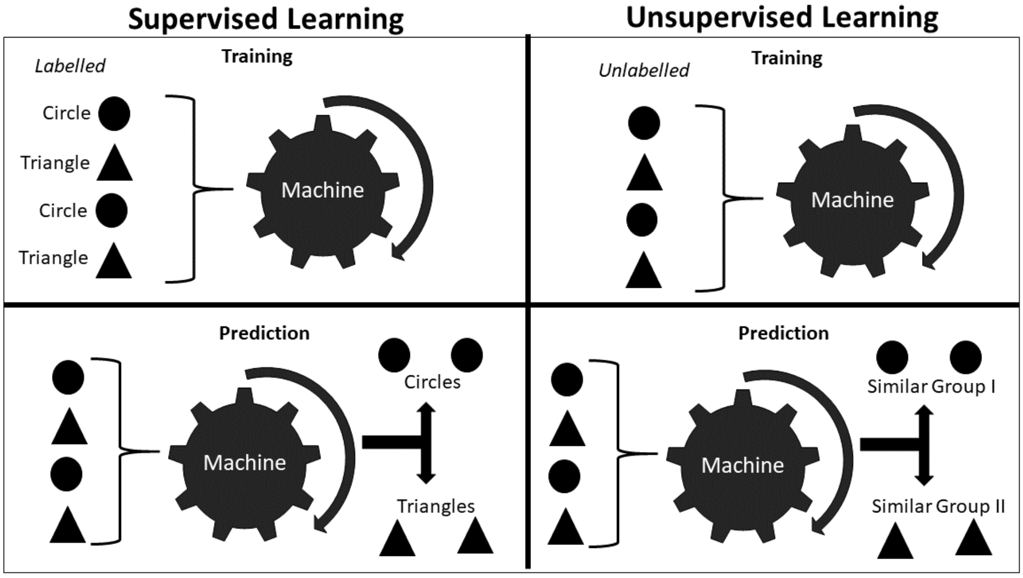


Figure 2.1: Supervised vs. unsupervised learning. During training, supervised models know the labels, while unsupervised learning doesn't have data labels.

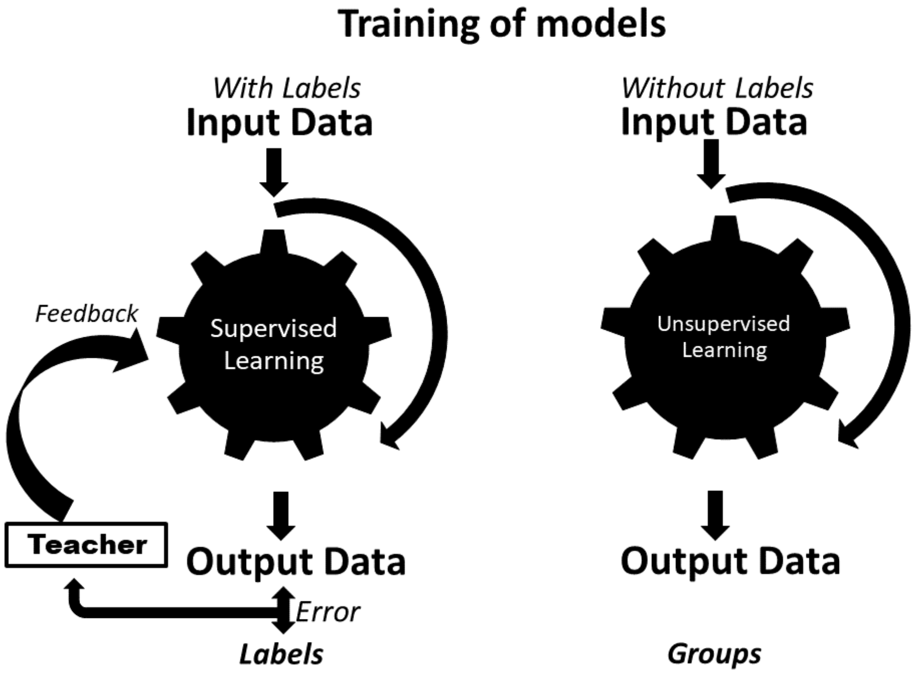


Figure 2.2: Training of supervised models required an instructor, whereas unsupervised models learn on their own.

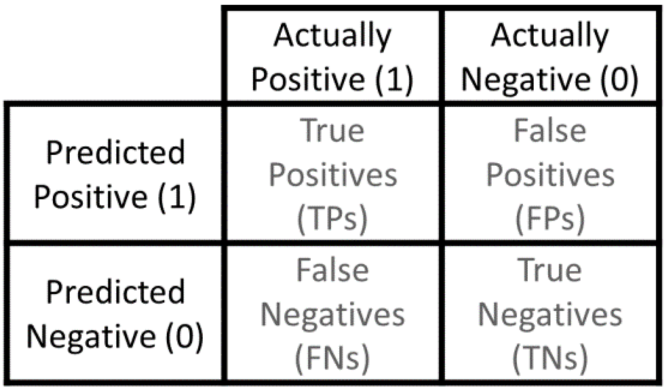
4. Other Types of machine learning

Semi-supervised Learning: Semi-supervised[70] learning is a machine learning approach that involves the use of both labeled and unlabeled data to train a model. It is typically used when there is a small amount of labeled data available, but a large amount of unlabeled data is present. In semi-supervised learning, the model is first trained on the labeled data, using this data to learn patterns and trends. The model can then be applied to the unlabeled data, using the knowledge gained from the labeled data to identify additional patterns and trends within the unlabeled data. Semi-supervised learning can be useful when it is not practical or possible to label a large amount of data, but there is still a need to use this data to train a model. It can also be used to improve the performance of a model when only a small amount of labeled data is available.

Reinforcement learning: Reinforcement learning[71] is a type of machine learning that involves training an agent to make a series of decisions in an environment in order to maximize a reward. It is often used to solve problems in which an agent must learn to interact with its environment in order to achieve a specific goal. In reinforcement learning, the agent receives feedback in the form of rewards or punishments based on its actions. The agent's goal is to learn the sequence of actions that will maximize the cumulative reward over time. This is typically achieved through trial and error, as the agent explores different actions and receives feedback on their outcomes. Reinforcement learning has been applied to a wide range of problems, including control systems, games, and natural language processing. It is particularly useful for situations in which it is difficult or impossible to specify a set of rules or a fixed decision-making process for the agent to follow.

5. Evaluation of models

Evaluating the performance of a machine learning model is crucial in order to ensure that it is operating accurately and effectively. One method of evaluating model performance is through the use of a train-test split, where the complete dataset is divided into two parts: a training set and a test set[72]. The training set, typically comprising 80% of the data, is used to train the model, while the test set, comprising the remaining 20%, is used to evaluate the model's performance. It is important that the model is not exposed to the test data during training in order to accurately assess its general performance. The efficiency of the model can be determined by comparing the predicted values with the actual values using a confusion matrix. This allows us to determine the accuracy of the model and make any necessary adjustments. To check the accuracy of the model, the predicted values are compared with actual values in a confusion matrix (Figure 2.3).



	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Figure 2.3: Confusion matrix

The values of a confusion matrix can be used to calculate various mathematical measures for evaluating the performance of a machine learning model, such as accuracy, precision, and recall. These measures provide insight into the effectiveness of the model and can be used to identify areas for improvement[73], [74].

Accuracy

Accuracy is the rate of correct prediction for a model i.e. the number of values correctly predicted divided by the total number of instances in the test set.

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

Precision

Precision is the ratio of true positives and the total number of instances predicted positive by the model.

$P = TP/\{TP+FP\}$

Recall

Recall is the true positive rate, also called the sensitivity of a model. i.e., the number of true positives divided by the total number of positive instances in the test dataset.

$R = TP/\{TP+FN\}$

F1 score

Precision and recall are always mentioned together like precision at a recall level, or measured in a single mathematical value called F1 score. It is the harmonic mean of precision and recall.

F1 Score = 2 * ((Precision * Recall)/(Precision + Recall))

Receiver Operating Characteristics (ROC) Curve

Receiver Operating Characteristics (ROC) Curve is another method for the evaluation of the classifiers[75]. It is a plot between true positive rate (TPR), or Sensitivity and False positive rate (FPR) or Specificity)

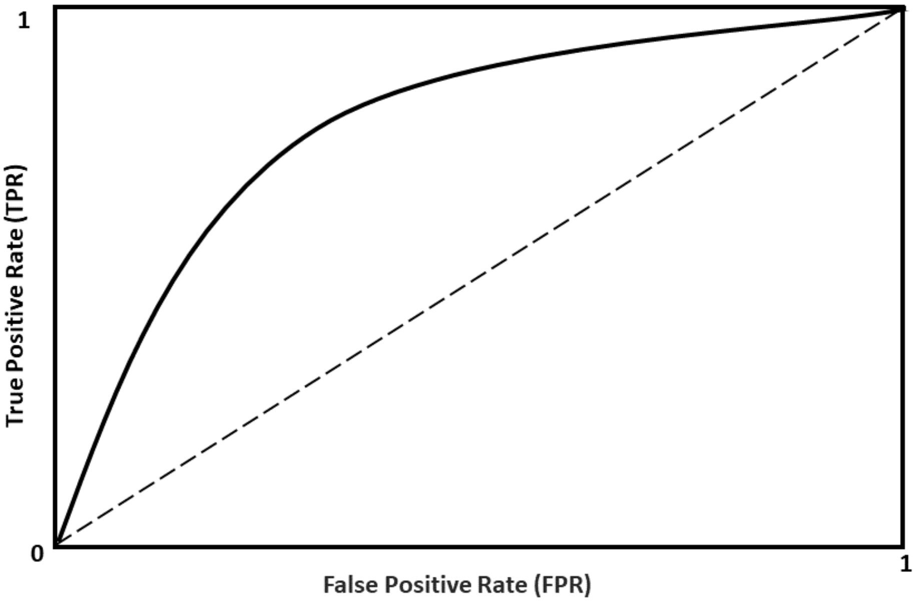


Figure 2.4: ROC curve, the curve is the change of true positive rate with false-positive rate. The more the area under the curve, the more is the accuracy of the model.

Evaluation metrics for regression models are fairly complex. Regression deals with a continuous dataset, which means advanced metrics are required for their assessment. Common metrics for regression model evaluation are variance, mean squared error and the R squared coefficient.

Matthews Correlation Coefficient (MCC)

The Matthews Correlation Coefficient is a measure used to evaluate the performance of classification models, particularly in cases where the classes are imbalanced. It takes into account true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) to provide an overall assessment of the model's performance.

The MCC is calculated using the following formula:

$MCC = (TP * TN - FP * FN) / sqrt((TP + FP) * (TP + FN) * (TN + FP) * (TN + FN))$

The MCC ranges between -1 and 1, where a value of 1 represents a perfect classifier, 0 represents a random classifier, and -1 represents a completely inverse classifier. A higher MCC indicates better performance, while a lower MCC indicates poorer performance.

The MCC is commonly used in machine learning tasks, such as binary classification problems, to evaluate the effectiveness of the model in predicting both positive and negative samples, especially when the data is imbalanced and the classes have different sizes.

6. Cross-Validation

In machine learning, it is common practice to divide a dataset into a training set and a test set in order to evaluate the performance of a model. However, this approach has the potential limitation of sacrificing a significant portion of the data for evaluation, which may result in the model not being properly trained to handle real-world challenges[76]. Additionally, this issue can be exacerbated when working with a small dataset, as valuable information may be lost during the split. To address these issues, a technique called cross-validation can be used (Figure 2.5). This involves dividing the dataset into multiple subsets and training and evaluating the model using various combinations of these subsets. Common methods of cross-validation include K-fold validation and Leave One Out Cross Validation (LOOCV)[77].

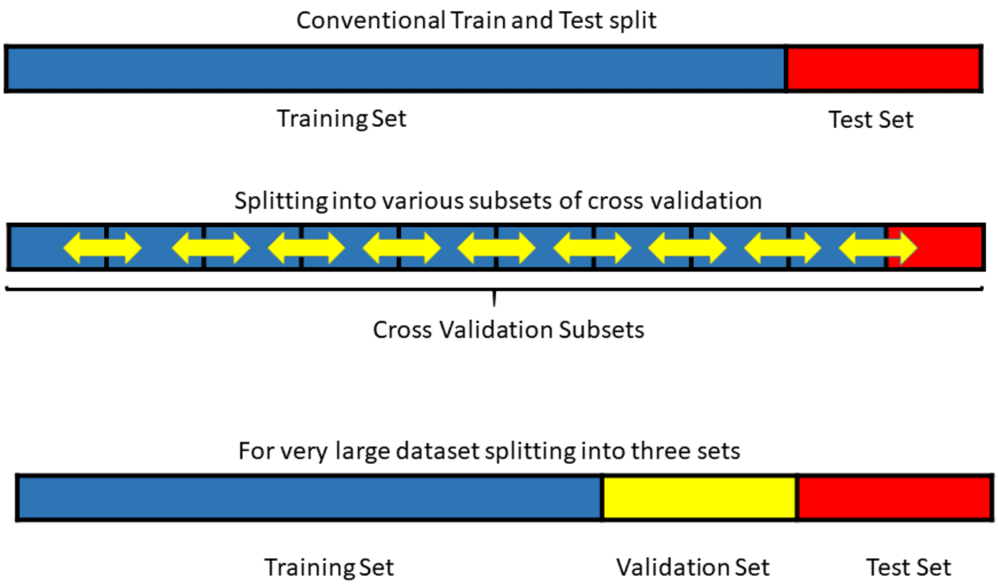

Conventional Train and Test split
Training Set
Test Set
Splitting into various subsets of cross validation
Cross Validation Subsets
For very large dataset splitting into three sets
Training Set
Validation Set
Test Set

Figure 2.5: Various arrangements for training, validating, and testing of models.

7. Optimization of models

Optimizing machine learning models is an essential step in improving their accuracy. As these models are trained on complex datasets, they often have a range of hyperparameters that can be adjusted in order to improve their performance. Finding the right combination of model and parameter values is crucial in fine-tuning a machine learning model. This process typically involves iteratively training and evaluating the model using different combinations of parameters. By continuously testing and improving the model, it is possible to optimize its performance to the greatest extent possible[78].

Grid Search

Grid searching is a method of scanning the space of all hyperparameter combinations for a model to find the best combination of hyperparameters[79]. Depending on the type of model utilized, certain hyperparameters are necessary. Grid searching can be applied on any machine learning model to calculate the best hyperparameters to use for that system. It is significant to note that grid searching is computationally intensive and may take a long time to run on your model. It iterates through every hyperparameter combination possible and stores a model for each combination.

Randomized Search

James Bergstra and Yoshua Bengio (2012)[80] proposed the idea of random searching of hyperparameters in a system. This type of search is completely different from the grid approach. Instead of sweeping through every possible combination in hyperparameter space, a randomized search only picks a few sample points from the distribution and performs the calculations on those points.

Ensemble Methods

It is a combination of procedures in which various models are used together to form a much-improved version. This allows us to have better results as compared to individual models. The two most widely used ensemble methods are averaging and voting. They are not complex and can easily be implemented to increase the accuracy of the system[81].

Averaging is performed while dealing with continuous data ranges, whereas voting is performed when with a discrete classification of data. Initially, a number of different models are trained by different subsets of training data. Then their results are combined. Therefore, instead of creating only one system and assuming it to be the most accurate, ensemble methods take several systems into account and average them to produce one final product.

8. Challenges in Machine Learning Projects

Machine learning gives us the ability to make more informed, data-driven decisions that are faster than conventional approaches. However, as with any other method, machine learning process presents its own set of challenges[82].

Challenges with data

Inadequate training data

Having sufficient data for training your system is extremely crucial for success in your machine learning project. Many of the machine learning systems fail miserably due to the lack of data. Not having sufficient data means that the system is unable to understand the trends properly and this compromises the efficiency of that system greatly. This is the reason data collection has become an integral part of the machine learning process. The amount of data needed depends both on the complexity of a problem and the algorithm selected[83].

Non-representative Training Data

For a machine learning model to be effective at generalizing to new data, it is essential that the training data be an accurate representation of the population under consideration. When sampling from the population to create a training dataset, it is important to ensure that the sample is large enough to accurately depict the complete picture of the dataset and that the sampling method is not flawed, as this can result in non-representative data known as sampling bias[84]. It is important to note that balancing the trade-off between bias and variance is key to achieving an effective model. While reducing sampling bias can increase variance, and reducing variance can increase bias, finding the right balance between the two is necessary for the model to be able to effectively generalize to new data.

Quality of Data

Poor data quality is a huge challenge in machine learning systems. These systems require high-quality data to avoid a situation in which they can fail both in the training and testing phases. Machine learning is a data-intensive technique. That is why the quality of the data used in any machine learning system has a huge effect on its development[85]. Because of this, small errors in the training data can lead to large scale errors in the system's output. Increasingly complex problems require not just massive amounts of data, but the data needs to be diverse, comprehensive as well as of good quality.

Inappropriate Features

The features that are used in the training of a machine learning system have a significant impact on its efficiency and performance. It is essential to carefully select the features to be included in the model, as the use of irrelevant or redundant features can diminish the effectiveness of the system and increase the cost of training[86]. Feature selection should be a fundamental part of the design process for any machine learning project, particularly when working with a large number of features. It is not necessary to utilize every available feature in the development of the algorithm. Instead, it is important to select only those features that are relevant and independent in order to assist the algorithm in its learning process. In some cases, reducing the dimensionality of the feature space through techniques such as principal component analysis (PCA) can improve model performance by minimizing noise. Additionally, various methodologies and techniques can be employed to select a subset of the feature space, as the inclusion of too many features can introduce additional noise and hinder the performance of the model. Thus far, we have examined the challenges and limitations of machine learning due to the lack of useful data. However, with the advances in machine learning, data has become a more valuable resource. In the next section, we will discuss the limitations of models due to hyperparameters.

Overfitting and Underfitting the Training Data

Overfitting refers to a system that models the training data too well. This happens when a model learns unnecessary details and noise in the training data. This impacts the performance of system on test data negatively[87]. Underfitting occurs when a system is not able to capture relationships between features and output variables precisely. This disturbs the ability of the algorithm to decipher the underlying trend of the data. Underfitting is a strong indicator of the algorithm not being suitable for the dataset in consideration. Underfitting is common in cases where there is limited data available for training the system. Fewer data points result in a non-accurate training of the model which then fails to perform efficiently on the test dataset. Increasing the amount of data could be one way to ensure that the algorithm has enough information through which it can detect the general trends and patterns.

9. Artificial Neural Networks

Artificial neural networks (ANNs), also known as neural nets, are a widely used type of supervised machine learning algorithm that was developed to simulate the neural networks or neurons found in the human brain[88]. They are used for statistical analysis and modeling of collected data and are often used as an alternative to traditional nonlinear regression analysis models. ANNs are particularly useful for solving problems that can be expressed in terms of prediction or classification. ANNs are made up of interconnected processing nodes, called artificial neurons, which are organized into layers. The input layer receives input data, which is then processed by the hidden layers using weights and biases. The output of the hidden layers is then passed through the output layer to produce the final prediction or classification. ANNs are able to learn and adapt based on the data they are trained on, allowing them to perform tasks such as image and speech recognition, natural language processing, and decision making. With over six decades of research, ANNs have a wide range of applications in diverse fields including speech and image recognition and classification, text recognition, medical diagnosis, and fraud detection. They are able to handle large and complex datasets and are well-suited for tasks that require the ability to learn and adapt to new data. However, they can also be more computationally intensive than some other machine learning algorithms and may require more time and resources to train and deploy.

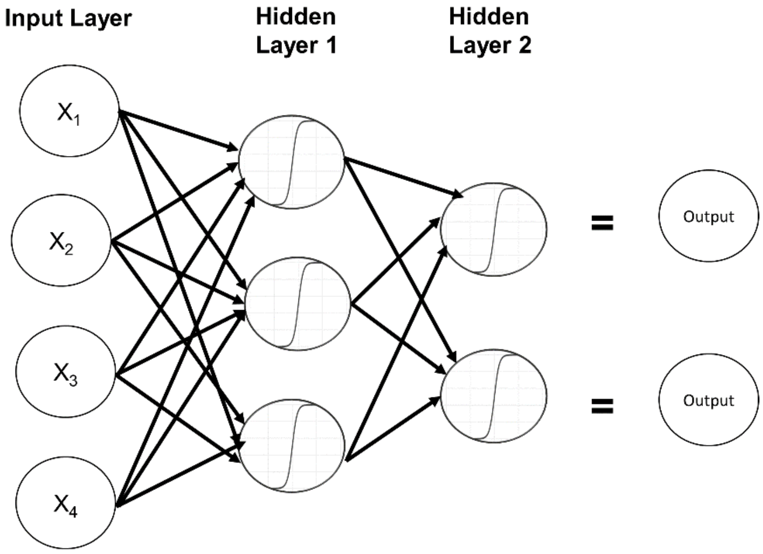
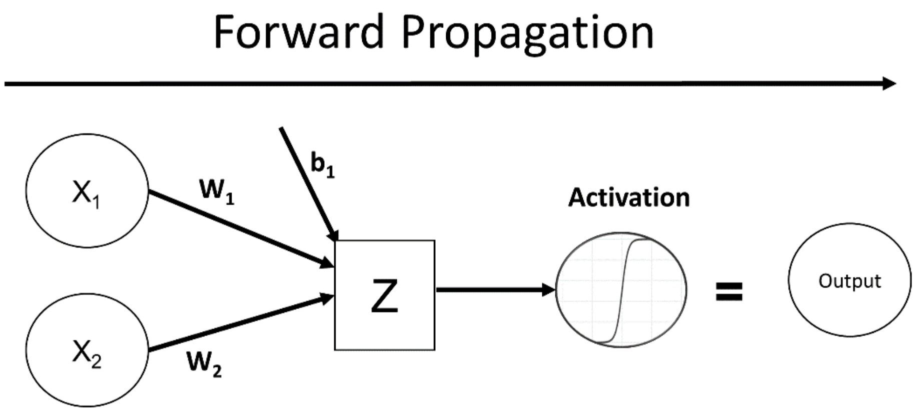


Figure 2.6: Simple architecture of artificial neural networks.

Figure 2.6 illustrates the general architecture of neural networks. It consists of three distinct layers: input layer, hidden layer and output layer. The foremost layer is the input layer that houses the input features, or input neurons. The middle layer is the hidden layer, the term 'hidden' implying the processes of mathematical computation that doesn't seem to be visible to everyone, and sometimes also termed as the black box. Diverse networks are characterized by numbers of hidden layers based on application. The ANN consisting of more than one hidden layer is called deep neural network. The last layer is the output layer, which contains the output of the network.

10. Working principle of neural networks



$$Z = (W_1X_1 + W_2X_2 + b_1)$$
$$Output = Sigmoid(Z)$$

Figure 2.7: Forward propagation, calculating the output.

In artificial neural networks, neurons are the basic processing units that are used to model the behavior of neurons in the human brain. Each neuron primarily consists of several inputs, features, weights, a bias, and an activation function[89]. The inputs to a neuron are the values that are fed into the neuron, which can be either the raw input data or the output of another neuron. The features are the characteristics or attributes of the input data that are relevant to the task being performed. The weights of a neuron define the importance or influence of each feature on the output of the neuron. Just like in a linear regression model, the weights are adjusted during training to improve the accuracy of the network's predictions. The bias of a neuron is a constant value that is added to the weighted sum of the inputs and features. It allows the neuron to shift the output of the activation function, allowing the network to model more complex patterns in the data. The activation function of a neuron is a mathematical function that takes in the weighted sum of the inputs and features, as well as the bias, and produces an output. The activation function is used to introduce nonlinearity into the network, allowing it to model more complex relationships in the data. In artificial neural networks, the weighted sum of the inputs and features for a neuron, as well as the bias, is often represented by the following equation:

$$Z = W_1X_1 + W_2X_2 + b$$

where ' W_1' and ' W_2' are the weights for the features ' X_1' and ' X_2', respectively, and 'b' is the bias. The bias is a small random number that is added to the weighted sum to ensure that 'Z' does not become zero for any values of ' W_1', ' W_2', and ' X_1', ' X_2'[90]. The bias allows the neuron to shift the output of the activation function, allowing the network to model more complex patterns in the data. (Figure 2.7).

Activation functions

The activation function in an artificial neural network determines whether a given neuron should be "activated" based on the weighted sum 'Z'. The activation function is a mathematical function that takes in the value of 'Z' and produces an output. The output of the activation function determines whether the neuron will be activated or not.

A simple activation function is the step function, which is defined as follows:

$f(z) = 1 \text{ if } z > 0$

$f(z) = 0 \text{ if } z <= 0$

This function gives an output of either 0 or 1, depending on whether the value of 'Z' is positive or negative. The step function is called a threshold function because it activates the neuron only if the value of 'Z' is above a certain threshold[91]. There are many other types of activation functions that can pass a range of values based on 'Z' to the next layer, rather than just 0 or 1. Some common activation functions include the sigmoid function, the tanh function, and the ReLU function. These activation functions introduce nonlinearity into the network, allowing it to model more complex relationships in the data[92].

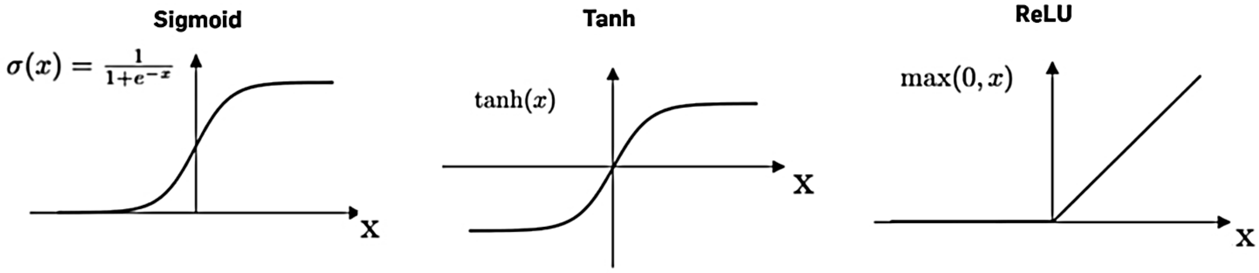


Figure 2.8: Activation functions.

Figure 2.8 shows three activation functions:

1. ‘Sigmoid’ function return a value between ‘0’ and ‘1’ for any value of ‘Z’. The sigmoid function is generally used in the output layer of a deep neural network.
2. The ‘Tanh’ function will give output ranging from ‘-1’ to ‘1’, when ‘Z’ is passed through it.
3. The ‘ReLU’ function, also known as the rectified linear unit, is the most commonly used activation function. It gives ‘0’ for negative values of ‘Z’ and will return the 1 if it’s a positive value.

Steps of forward propagation.

In an artificial neural network, the main objective is to calculate appropriate weights for the input features in order to produce an accurate output. The steps for calculating the output based on the input features are as follows:

1. Input features are assigned weights.
2. The weighted sum of all input features, along with the bias, is calculated (i.e., 'Z').
3. The value of 'Z' is passed through the activation function, which produces the output of the neuron.

After the output is calculated, it is compared with the original values, and an error or loss is calculated. This loss is then used to update the weights in order to minimize the error. This process is repeated iteratively until the loss is minimized. The process of minimizing the error is called gradient descent. In summary, the main objective of neural networks is to calculate appropriate weights for the input features in order to produce an accurate output. This is achieved through the process of gradient descent, which involves calculating the loss, comparing it to the original values, and updating the weights iteratively in order to minimize the error.

Gradient Descent

The gradient descent algorithm is a method for updating the weights in an artificial neural network in order to minimize the loss. When an output is calculated through the forward propagation method, and the actual value is known, the error can be calculated using a loss function. Mean squared error, which is the sum of all the squared errors, is a common type of loss function. For a single output value, the loss can be calculated as follows:

$$\text{Loss} = 1/2 * (y - \hat{y})^2$$

Where 'y' is the actual value and '$\hat{y}$' is the value calculated through the forward propagation method. If the output is calculated using the forward propagation method for 'n' number of times with different random weights, and the error is minimized, a plot of the loss versus the weight will be obtained, as shown in Figure 2.9. The weight for which the error is minimum can then be selected. These steps are feasible when there are a small number of features, weights, and neurons. However, with a large number of features, weights, and neurons, finding the optimal combination of weights using this method can take an indefinite amount of time.

The gradient descent algorithm was developed as an alternative method for minimizing the loss. It involves updating the weights in a way that reduces the loss with each iteration. By updating the weights using the gradient descent algorithm, we can find the optimal combination of weights that minimizes the loss in a relatively efficient manner. As a result, the model obtained will have a minimum error, making it more accurate.

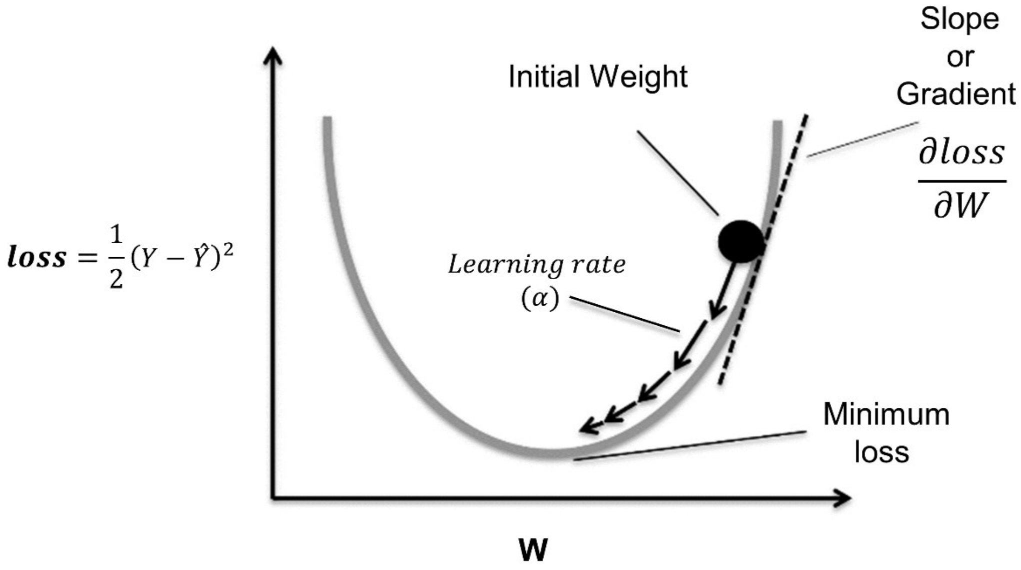

Slope
or
Gradient
$\frac{\partial loss}{\partial W}$
Initial Weight
$\boldsymbol{loss} = \frac{1}{2}(Y - \hat{Y})^2$
Learning rate
(α)
Minimum
loss
W

Figure 2.9: Gradient descent

•

The gradient descent algorithm is a method for minimizing the loss in an artificial neural network by updating the weights with each iteration[93]. The basic idea behind the gradient descent algorithm is that differentiating a function providesthe slope of the function, or gradient. The analogy of being at the top of a hill and walking downhill while blindfolded can be used to understand this concept. In this scenario, it is possible to determine the direction of the minimum error by differentiating the loss function with respect to the weights and backpropagating this information to update the weights. The amount by which the weights are updated is determined by the derivative of the loss function, or the gradient, as well as the learning rate (α).

$$W_{new} = W - \alpha * \frac{\partial loss}{\partial W}$$

Figure 2.10: Formula of updating weight.

The user defines the learning rate, and it is the length of the step the algorithm takes to reach the minimum loss value. If the step is too long then it might miss the minimum value, if the step it too short, it will take more time to reach the bottom. There are many variants of gradient descent algorithm, like RMSprop, Adam etc, they are called optimizers as they optimize the weights and hence the predictions.

Backpropagation

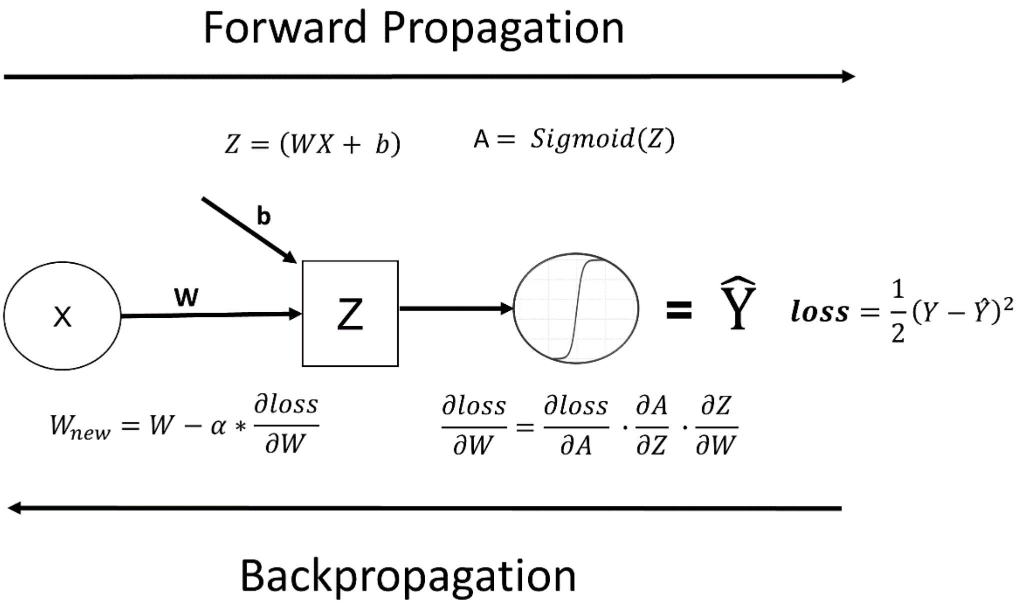

Forward Propagation
$Z = (WX + b)$
$A = Sigmoid(Z)$
b
X
W
Z
$= \hat{Y}$
$loss = \frac{1}{2}(Y - \hat{Y})^2$
$W_{new} = W - \alpha * \frac{\partial loss}{\partial W}$
$\frac{\partial loss}{\partial W} = \frac{\partial loss}{\partial A} \cdot \frac{\partial A}{\partial Z} \cdot \frac{\partial Z}{\partial W}$
Backpropagation

Figure 2.11: Forward and backward propagation for neural networks.

It is stated that the output of the activation 'A' is a function of 'Z', and 'Z' is a function of the weight 'W' and the bias 'b', which is determined through the process of forward propagation. In order to calculate the derivative of the loss with respect to the weight, partial derivatives of each function are utilized. Specifically, the derivative of the loss function with respect to the activation function is calculated first. This value is then used to find the derivative of the activation function with respect to 'Z', and finally, the derivative of 'Z' with respect to 'W' is determined. The process of propagating the loss backward in order to update the weights is referred to as backpropagation. In a deep neural network, the steps of forward propagation and backpropagation are repeated iteratively in order to update each weight. This can become complex, particularly when the number of layers is more than one, as the derivative chain becomes longer (as depicted in Figure 2.11: Forward and backward propagation for neural networks.). However, the fundamental principle behind the process remains unchanged.

Having lots of weights and activation functions provides neural networks the ability to handle linear as well as nonlinear datasets. These features of neural networks also make them susceptible to overfitting the training data. So, they require lots of parameter tuning to obtain the optimal results[94].

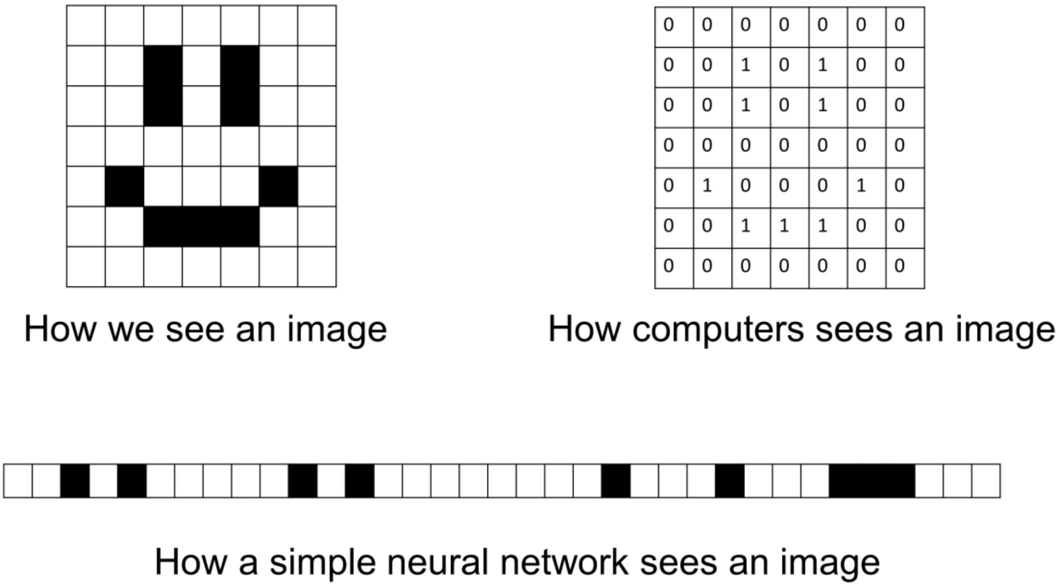
Implementing neural networks using Tensorflow

Training an artificial neural network can involve a significant amount of complex mathematical functions and processes. However, Python packages such as TensorFlow, PyTorch, and Theano make the implementation of these algorithms much simpler and more streamlined. These packages provide a range of tools and resources that allow users to easily build and train neural networks for various tasks[95].

TensorFlow is a particularly popular deep learning library that was developed by Google and is open source. It is commonly used for implementing neural networks in Python. TensorFlow has two main versions: the normal TensorFlow package, which uses a central processing unit (CPU) for calculations, and a version that utilizes a graphics processing unit (GPU) for faster processing. The GPU version of TensorFlow is often preferred for training neural networks, as it can significantly speed up calculations due to the large number of simultaneous computations required for this task. The GPU version shifts these calculations to the graphics card, allowing for faster processing and more efficient training.

11. Convolutional Neural Network

When it comes to processing images, computers typically represent them as a collection of pixel intensities arranged in a specific order. To humans, these pixel intensities correspond to various colors that make up the image. In some cases, an image may be divided into smaller parts or pieces, similar to how a puzzle is divided into individual pieces. In these instances, the individual pieces may not contain enough information on their own to fully understand the content of the image. This is similar to how simple neural networks may approach image processing. Rather than analyzing the full image at once, they may break it down into smaller parts and attempt to piece them together to understand the overall content. However, more advanced neural networks are able to analyze images as a whole and extract meaningful features and patterns from them.


How we see an image
How computers sees an image
How a simple neural network sees an image

0	0	0	0	0	0	0
0	0	1	0	1	0	0
0	0	1	0	1	0	0
0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

Figure 2.12: Image data with various prospectives.

To train an artificial neural network on image data, it is necessary to extract the individual pixels from the image and feed them to the neural network as features. While individual pixels may contain some information about the image, it is often more effective to analyze the image as a whole rather than considering each pixel in isolation (Figure 2.12). This is where convolutional neural networks (CNNs) come in[96]. CNNs are a type of deep learning model that are specifically designed to handle image or spatial data. They are able to learn local spatial features and patterns within an image, allowing them to better understand the overall content of the image. CNNs use small windows or filters to analyze images, dividing them into smaller parts and using multiple filters to learn different local features such as edges and textures. By combining these basic features, CNNs are able to create higher-order features such as eyes, hands, and other objects. This makes CNNs particularly effective at image classification tasks, as they are able to extract meaningful features from the image and use them to accurately classify the content. In contrast, traditional artificial neural networks (ANNs) are not able to learn these local features and may struggle to accurately classify images.

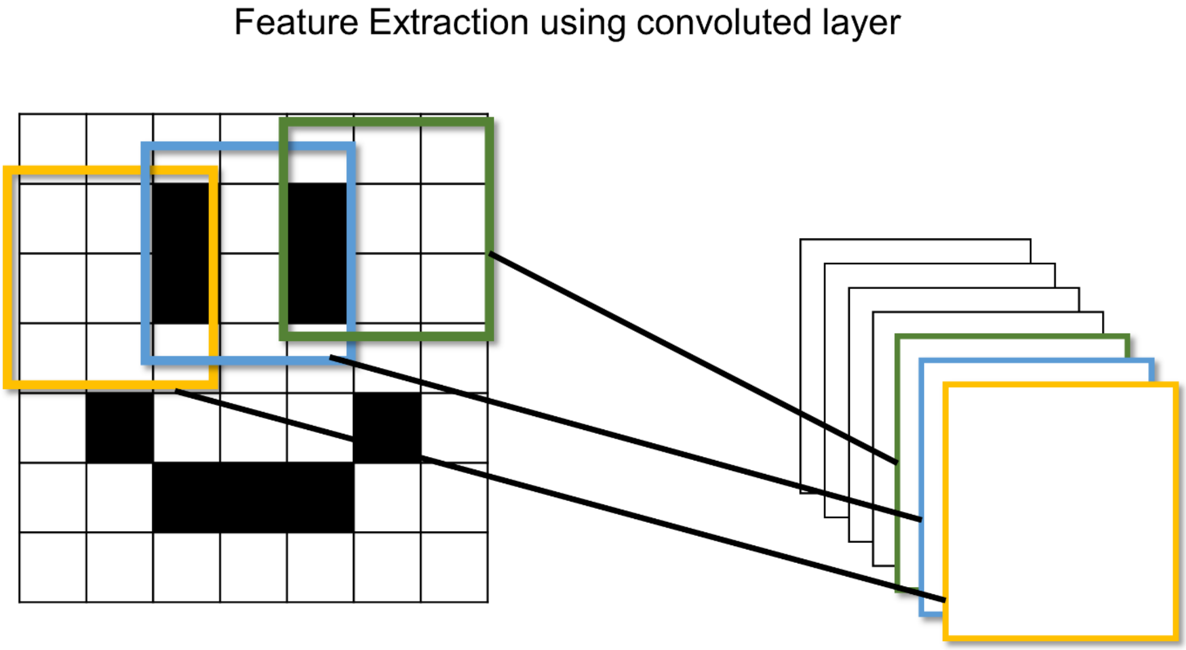


Figure 2.13: Feature extraction using CNNs.

In a convolutional neural network (CNN), filters are matrices of numbers that can be trained or updated in order to detect relevant features from the image data. These filters are used to extract predictive features from the image and are optimized through the use of the backpropagation method with each iteration. Convolution is a mathematical function that operates on two objects and results in the transformation of one object based on the other (Figure 2.13). This process, also known as feature mapping, helps the CNN to learn important features from the image data[97].

In addition to convolution layers, CNNs may also include other types of layers such as pooling and flattening layers. Pooling involves sliding a window over the pixels of an image and merging local features into a single feature. There are several types of pooling, including max pooling, which selects the maximum intensity pixel over other pixels within the window, and average pooling, which returns the average of all the pixels within the window. Pooling is primarily used to reduce the size of the data and retain important information while discarding insignificant details.

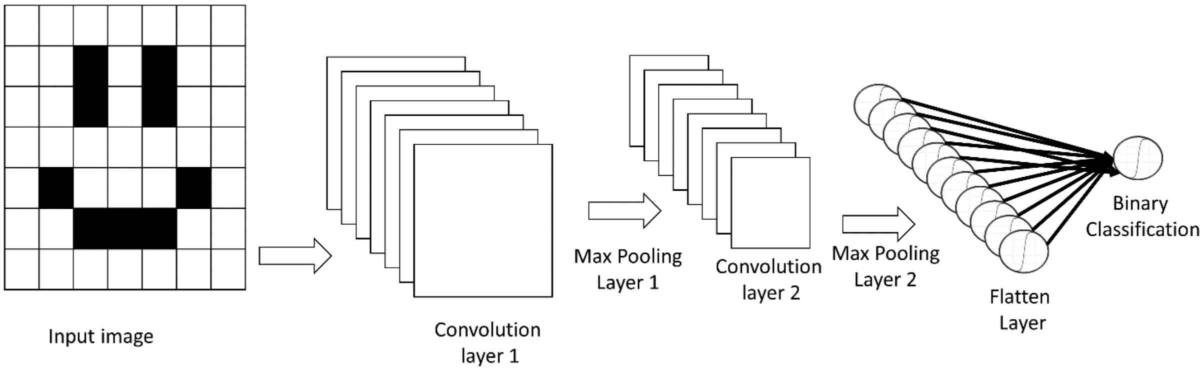


Figure 2.14: Architecture of a CNN bases neural system.

In order to use artificial neural networks for the classification of images, the local features extracted from the images using convolutional neural networks (CNNs) and the pooling operation need to be transformed into a form that can be processed by the ANNs[98]. This is done by flattening the 2D features into a 1D representation using the flattened layer. The flattened features are then connected to the ANNs, which use them to perform the desired classification. A basic architecture of a CNN based on ANNs is shown in Figure 2.14 . This architecture includes the convolutional layers, which use filters to extract local features from the images, the pooling layers, which combine local features into a single feature, and the flattened layer, which converts the 2D features into a 1D representation that can be processed by the ANNs. The combination of CNNs and ANNs in this way allows for effective classification of image data using deep learning techniques.

12. Recurrent Neural Network

Recurrent neural networks (RNNs) are a type of artificial neural network that are particularly well-suited for processing sequential data such as natural language, time series data, and audio. RNNs are designed to remember previous input by using feedback connections, which allow the network to incorporate information from the past into its current state[99], [100]. The working principle of RNNs is based on the idea of hidden states, which are intermediate representations of the input data. These hidden states are updated at each time step using the current input and the previous hidden state. The hidden state can be thought of as a memory of the network, storing information about the input it has seen so far. To understand how RNNs work, let's consider a simple example of predicting the next word in a sentence. Suppose we have a sentence "The cat sat on the". Given the input "The cat sat on", we want to predict the next word "the". To do this, we can use an RNN with a single hidden state. At each time step, the RNN takes in a single word as input and updates its hidden state using this input and the previous hidden state. The hidden state is then passed through an output layer to produce a prediction for the next word. The prediction is then compared to the actual next word and an error is calculated. This error is then used to update the weights of the network using a technique called backpropagation through time (BPTT)[101].

One of the key benefits of RNNs is their ability to process input of variable length. In our example, we could have used a different number of words as input and the RNN would still be able to make a prediction. This is because the hidden state is updated at each time step, allowing the network to incorporate information from previous time steps into its prediction. Another advantage of RNNs is their ability to handle long-term dependencies. Suppose we want to predict the next word in the sentence "The cat sat on the mat". In this case, we need to remember that "the" is an article and not the noun we are trying to predict. Without an RNN, it would be difficult to capture this long-term dependency between the first and last occurrences of "the". However, with an RNN, the hidden state can store this information and use it to make a more accurate prediction. Despite their many strengths, RNNs do have some limitations. One of the main challenges is the vanishing gradient problem, which occurs when the gradients of the weights become very small as the network is trained. This can make it difficult for the network to learn long-term dependencies and can lead to slow training times.

The Problem of Long-Term Dependencies

One of the challenges of training RNNs is the problem of long-term dependencies[99]. Long-term dependencies refer to the fact that the output of an RNN at a given time step can depend on inputs from many timesteps in the past. This can make it difficult for the RNN to effectively learn and make predictions, because the gradient signal (which is used to update the model's weights during training) can become very small or even vanish altogether as it is propagated back through many timesteps[100], [101]. This is known as the vanishing gradient problem.

There are several approaches that have been proposed to address the problem of long-term dependencies in RNNs, including using different types of RNN architectures (such as long short-term memory (LSTM) networks or gated recurrent units (GRUs)), using more advanced optimization algorithms, and using regularization techniques to prevent overfitting

Long Short-Term Memory (LSTM) Networks

Long short-term memory (LSTM)[102] is a type of recurrent neural network (RNN) that is designed to overcome the problems of long-term dependency in traditional RNNs. In traditional RNNs, the output at each time step is dependent on the input at the current time step and the hidden state at the previous time step. This means that the output at each time step is only dependent on the inputs that have been seen so far, and not on the inputs that were seen a long time ago. This can be a problem when trying to model sequences with long-term dependencies, such as language or financial data[103].

LSTMs were introduced by Hochreiter and Schmidhuber in 1997[104] as a way to address this problem. They work by explicitly remembering information for long periods of time, using three different types of gates: an input gate, an output gate, and a forget gate. The input gate controls the amount of information that is allowed to be passed through to the cell state, the output gate controls the amount of information that is allowed to be passed from the cell state to the output, and the forget gate controls the amount of information that is allowed to be forgotten from the cell state[105]. These gates are all controlled by the input and hidden state at the current time step, as well as the hidden state at the previous time step. The LSTM architecture consists of four layers of neural networks: an input layer, an output layer, and two "memory" layers. These layers interact with each other in a unique way, allowing the LSTM to remember information for long periods of time. LSTMs have been widely used and refined in the years since their introduction, and have proven to be effective in a variety of tasks, including language modeling, machine translation, and speech recognition[106].

Gated Recurrent Units (GRUs)

Gated recurrent units (GRUs) are a type of recurrent neural network (RNN) that was introduced by Cho et al. in 2014[107] as an alternative to long short-term memory (LSTM) networks. Like LSTMs, GRUs are designed to address the problem of long-term dependencies in traditional RNNs. GRUs are similar to LSTMs in that they have a gating mechanism to control the flow of information through the network. However, they differ in that they have a simpler architecture and fewer parameters[108], [109]. In particular, GRUs do not have an output gate, and they merge the hidden and cell states into a single "update" state. To update the update state in a GRU, an "update gate" is used, which is a combination of the input and forget gates in an LSTM. The update gate controls the amount of information that is allowed to be passed from the input to the update state, and the amount of information that is allowed to be forgotten from the previous update state. GRUs have gained increasing popularity in recent years due to their simplicity and good performance on various tasks, such as language modeling, machine translation, and speech recognition. They have been shown to be competitive with LSTMs on many tasks, and are often preferred due to their simplicity and faster training time[110].

v. Application of Neural Networks

Deep neural networks are a type of machine learning algorithms which are inspired by biological neural networks i.e. how our brain learns[111]. Since their development in mid of 20th century, they didn't get much application as they are computationally intensive. The inception of modern hardware systems, especially the GPUs (Graphical Processing Units) with greater computational capability, has allowed neural networks to regain popularity and applications. Additionally, special types of neural networks, such as convolutional neural networks(CNNs) and recurrent neural networks(RNNs) with long short-term memory cells(LSTM) find their applications in various fields like image processing, speech recognition and natural language processing. CNNs and LSTM have been also used in solving various biological problems including prediction of protein secondary structure, protein sub-cellular localization, peptide binding to MHC-II molecules, image recognition of skin disorders etc. [112], [113].

A typical neural network learns by adjusting its weights or priority of any given features for calculating values close to the given output, i.e. values with minimum error. The weights are calculated according to the difference between calculated output and given output using an algorithm known as Gradient descent[114]. CNNs are a type of neural networks which are used for feature extraction when exact features or patterns cannot be determined, such as in image and sequence data. CNNs slide a grid, also known as filters (a set of weights), over the input data which are fed into different neurons of further layer every time the filter is moved. These filters are not interconnected and while moving through the data, it can extract patterns or features from the input irrespective of the position where they are found[115].

RNNs are a kind of neural network architecture which can deal with sequential data such as time series data and text data. They have typical feed forward connections, but in addition, the neurons of hidden layers are connected with a time-delayed connection for retaining the weights of previous time-step[116]. In this way RNNs are capable of learning through sequence data by storing instincts form previous elements and analyzing the present element in context of the previous one. Long short-term memory (LSTM) are a special kind of RNNs where the simple matrix of hidden neuron is succeeded with the LSTM memory block which minimizes the vanishing gradient problem [117]. LSTMs have a memory block cell where context-dependent weights are saved. This block is further controlled by input, output and forget control gates so that it can read the input sequence and decide the elements it should keep in the cell for each time-step. So, it is easier for LSTMs to save a given input feature over many time frames which is its advantage over RNNs[118]. Previously, LSTMs have been used in building chatbots, language translation, image captioning etc. [119].

Deep neural networks have undergone tremendous advancement in recent years and found its application in various field one of such field in natural language processing (NLP)[120]. Natural language processing deals with a sequence of words or letters and tries to extract meaningful features form them which can be further used for various application such as spam message detection[121], text summarization[122] etc. text embedding[123] is one such tool which is used for retrieving meaningful feature for word or letter form a data set. Embedding layers works on the same principle where a user defines how many features should be retrieved for each letter, then the algorithm basically learns the feature or weights from the training data and form a numerical representation of the input text data. Upon this representation layer Depth wise separable 1d convoluted neural networks(CNNs)[124] and Long Short Term Memory networks (LSTMs)[125] can be used for deriving or learning local and global features respectively. This derived information can be used by simple neurons for taking decisions.

vi. Generative models

Generative models are a type of machine learning model that is capable of generating new data samples that are similar to the training data. These models are typically used in unsupervised learning, where the goal is to discover the underlying structure of the data, rather than to make predictions based on labeled examples[126].

There are many different types of generative models in deep learning, including:

- Generative adversarial networks (GANs)
- Variational autoencoders (VAEs)
- Autoregressive models (e.g. PixelRNN and PixelCNN)
- Flow-based models (e.g. Real NVP and Glow)
- Generative stochastic networks (GSNs)
- Generative moment matching networks (GMMNs)
- Deep Boltzmann machines (DBMs)

Each of these models has its own strengths and weaknesses, and is suitable for different types of data and applications. For example, GANs are effective for generating high-quality images, VAEs are good for producing smooth and continuous data samples, and autoregressive models are efficient for modeling structured data.

Generative adversarial networks (GANs)[127] are a type of generative model that is based on the idea of training two neural networks, a generator and a discriminator, to compete with each other. The generator network is trained to generate data samples that are similar to the training data, while the discriminator network is trained to distinguish the generated samples from real ones. During training, the generator network is provided with random noise as input and tries to produce data samples that are similar to the training data. The discriminator network then attempts to classify the generated samples as real or fake. The generator and discriminator networks are trained simultaneously, with the generator trying to fool the discriminator and the discriminator trying to accurately identify the generated samples. GANs have many potential applications, including image generation, data augmentation, and anomaly detection.

A **variational autoencoder** (VAE)[128] is a type of generative model that is used to learn a continuous and structured representation of data. A VAE consists of two parts: an encoder, which maps an input data sample to a latent representation, and a decoder, which maps the latent representation back to a reconstruction of the original data sample. The encoder and decoder are typically implemented as neural networks, and the VAE is trained by maximizing the likelihood of the data given the learned latent representation. This is done by minimizing the difference between the original data and its reconstruction, while also regularizing the latent representation to have a certain desired structure (such as a Gaussian distribution). VAEs have many potential applications, such as data generation, data interpolation, and dimensionality reduction. They are also often used as a tool for understanding and interpreting the learned features of a neural network.

Autoregressive models [129]are a type of generative model that are used to model sequential data, such as time series or natural language. These models make predictions based on previous values in the sequence, using a fixed-length context window to condition the predictions. One of the most well-known autoregressive models is the autoregressive integrated moving average (ARIMA) model, which is commonly used for time series forecasting. In the context of deep learning, autoregressive models are often implemented using neural networks, such as the PixelRNN and PixelCNN models for image generation. Autoregressive models have the advantage of being fast and efficient, as they only depend on a fixed-size context window to make predictions. However, they can struggle to capture long-term dependencies in the data, and may not be well-suited to modeling data with complex or non-linear relationships.

Flow-based models[130] are a type of generative model that uses a series of invertible transformations to map a simple prior distribution to a complex target distribution. This is done by constructing a sequence of invertible functions (called "flows") that transform the data from a simple latent distribution to the desired target distribution. One of the most popular flow-based models is the real normalizing flow (Real NVP), which uses a series of affine transformations to map a standard Gaussian distribution to the target distribution. Other examples of flow-based models include the Glow model, which uses invertible 1x1 convolutions, and the Masked Autoregressive Flow (MAF) model, which combines autoregressive models with normalizing flows. Flow-based models have many potential applications, including density estimation, generative modeling, and likelihood-based model evaluation. They are also an active area of research in deep learning, as they offer a way to model complex and high-dimensional data distributions using invertible transformations.

Generative stochastic networks (GSNs)[131] are a type of generative model that is based on the idea of decomposing a complex data distribution into a hierarchy of simpler distributions. A GSN consists of a set of stochastic layers, each of which models a different level of abstraction in the data. The GSN is trained by maximizing the likelihood of the data given the learned hierarchy of distributions. This is done by optimizing the parameters of the stochastic layers to capture the underlying structure of the data. The resulting model can be used to generate new data samples by sampling from the learned distributions at each level of the hierarchy. GSNs have many potential applications, including image generation, data compression, and representation learning. They are also an active area of research in deep learning, as they offer a way to model complex data distributions using a hierarchy of simpler distributions.

Generative moment matching networks (GMMNs)[132] are a type of generative model that is based on the idea of matching the moments of a simple distribution to those of a complex target distribution. A GMMN consists of a set of non-linear transformations that map a simple latent distribution (such as a Gaussian distribution) to the target distribution. The GMMN is trained by minimizing the difference between the moments of the target distribution and the moments of the transformed latent distribution. This is done by optimizing the parameters of the non-linear transformations to capture the underlying structure of the data. The resulting model can be used to generate new data samples by sampling from the transformed latent distribution. GMMNs have many potential applications, including density estimation, generative modeling, and representation learning. They are also an active area of research in deep learning, as they offer a way to model complex data distributions using non-linear transformations.

A **Deep Boltzmann machine (DBM)** [133]is a type of generative model that is based on the idea of representing complex data distributions using a network of stochastic units. A DBM consists of two or more layers of stochastic units, with connections between units in different layers but not within the same layer. The DBM is trained by maximizing the likelihood of the data given the learned network of stochastic units. This is done by optimizing the parameters of the connections between the units to capture the underlying structure of the data. The resulting model can be used to generate new data samples by sampling from the learned distribution using Markov Chain Monte Carlo (MCMC) methods. DBMs have many potential applications, including density estimation, generative modeling, and representation learning. They are also an active area of research in deep learning, as they offer a way to model complex data distributions using a network of stochastic units.

Generative models have many potential applications, such as image generation, data augmentation, and anomaly detection. They are also an active area of research in deep learning, as they offer a way to learn complex distributions and generate new data samples that are difficult to produce using other methods. Deep generative models are machine learning algorithms that are able to learn complex distributions of data and generate new data samples that are similar to the ones they were trained on. These models have been applied in a variety of fields, including drug design. One way in which deep generative models are being used in drug design is to generate novel chemical compounds with desired properties. For example, a generative model could be trained on a dataset of known drugs, and then be used to generate new chemical compounds that are similar to the ones in the dataset, but with improved properties such as increased potency or reduced side effects. This can help researchers identify new compounds that are worth synthesizing and testing in the laboratory. Another way in which deep generative models are being applied in drug design is to predict the properties of chemical compounds. This can help researchers identify compounds that are likely to have the desired properties before synthesizing and testing them in the laboratory, which can save time and resources. For example, a generative model could be trained on a dataset of known drugs and their properties, and then be used to predict the properties of new chemical compounds that are generated by the model. This can help researchers identify compounds that are likely to be effective drugs and prioritize them for further study.

In addition to generating and predicting the properties of chemical compounds, deep generative models are also being used to generate 3D models of molecules. This can help researchers visualize the structure of new compounds and understand how they are likely to interact with proteins or other molecules in the body. This information can be used to improve the design of drugs and make them more effective.

Overall, deep generative models are a promising tool for drug design, as they can help researchers generate novel compounds, predict their properties, and visualize their structures. This can speed up the drug discovery process and lead to the development of more effective and targeted treatments for a variety of diseases.

vii. Application of Generative models

Deep generative models are a class of machine learning algorithms that use deep neural networks to generate similar new data to a training dataset. These models have been utilised in numerous fields, including biology, to generate data with a realistic appearance, such as images or DNA or protein sequences[134], [135].

In biology, one application of deep generative models is the generation of synthetic images of cells and tissues. These images can be used to train machine learning models to identify patterns or anomalies in biological samples. A deep generative model could, for instance, be trained on a large dataset of images of healthy cells and then used to generate synthetic images of diseased cells. These synthetic images could be used to train a classifier to recognise diseased cells in real-world samples. The generation of synthetic DNA or protein sequences is yet another biological application of deep generative models. These synthetic sequences can be used to examine the effects of genetic mutations or variations. For instance, a deep generative model could be trained on a dataset of known protein sequences and then used to generate novel protein sequences with the desired properties, such as enhanced binding affinity for a specific drug target[136], [137]. Deep generative models can be used to impute missing data in biological datasets, in addition to generating synthetic data. If a dataset contains gaps or missing values, for instance, a deep generative model could be trained on the available data and used to generate realistic-looking values for the missing data. This can enhance the precision and utility of the dataset for subsequent analyses.

Deep generative models have the potential to significantly advance the field of biology by facilitating the generation of realistic synthetic data, the imputing of missing values in datasets, and the discovery of new patterns and relationships in biological systems[138].

viii. The language of LIFE (Gene, Protein and Chemicals)

Genetic sequences, such as DNA and RNA, can be considered a type of language due to the fact that they contain a specific set of symbols that convey meaning and are organised according to specific rules and structures[139]. These sequences are essential to the development and function of living organisms, and they can evolve and adapt over time through the process of mutation. The application of natural language processing (NLP) techniques to gain a comprehensive understanding of the functions and structures encoded in biological sequences has gained attention as a way to better understand and make discoveries from them. However, due to the complexity and vast potential of these sequences, the functions and properties of many coding and non-coding DNA and RNA sequences are still poorly understood. NLP can be used to analyse genetic sequences by employing tokenization techniques, which involve dividing the sequence into smaller "tokens" for analysis. A DNA sequence could be tokenized by separating it into its constituent nucleotides (A, C, G, and T) or codons (triplets of nucleotides)[140]. Then, researchers can use NLP techniques to analyse the relationships between these tokens and the context in which they appear, gaining a deeper understanding of the functions and structures encoded in the sequence. In addition to tokenization, genetic sequences can also be analysed using NLP techniques such as part-of-speech tagging and syntactic parsing. In part-of-speech tagging, tokens are labelled with their grammatical function, such as noun, verb, or adjective. Syntactic parsing is the process of analysing the syntactic structure of a sentence or sequence, including the relationships between its tokens and the rules governing their arrangement[141]. These methods can be used to identify patterns and structures within a genetic sequence that may be essential to its function or regulation. NLP can also be used to look at the words used to describe genetic sequences and the biological processes they are involved in. For instance, key terms and concepts can be extracted from scientific papers or other sources of biological data using NLP techniques, making it simpler for researchers to spot and examine trends and relationships in the data. The use of NLP in the analysis of genetic sequences has the potential to significantly enhance our understanding of the functions and structures encoded in these sequences as well as the biological processes they are involved in. It can also assist researchers in analysing and making discoveries from the vast quantities of data generated by high-throughput sequencing techniques[142], [143].

Proteins are extremely important molecules that are essential for the structure, function, and regulation of the body's tissues and organs. They are composed of chains of amino acids folded into specific three-dimensional shapes and are involved in numerous biological processes. Proteins can be viewed as a type of language, with amino acids serving as "letters" and proteins themselves as "words." In this way, proteins are similar to a language in that they are composed of specific units (amino acids) that are combined according to specific rules (the genetic code) to encode essential information for the body's proper functioning. The rules for combining protein domains constitute the "grammar" of proteins. Protein domains can be thought of as "words," and the rules for combining them constitute the "language" of proteins. Researchers have discovered a "quasi-universal grammar" underpinning the evolution of domain architectures in proteins by employing techniques such as n-grams[144], [145].

By analysing the semantics of proteins, techniques from natural language processing (NLP) have been used to attempt to predict their structures and functions. However, there is still much about the semantics of the proteome that we do not comprehend. Prior research has primarily investigated the lexical, syntactic, and semantic aspects of biological sequences, but these sequences have their own distinctive linguistic properties. For instance, there are more than 500 distinct physiochemical properties for amino acids and more than 180 for nucleotides, which is significantly more than the most complex polysemous word in a natural language. Consequently, rule-based approaches have limited performance for certain tasks, including protein disordered region prediction and enhancer identification, and they rely heavily on experience-based linguistic features. To advance the development of biological sequence analysis using NLP, it will be necessary to develop biological language models (BLMs) for DNA, RNA, and protein sequences that can automatically and systematically capture linguistic features using techniques similar to those used in bioinformatics for tasks such as protein structure prediction and function analysis[144].

Chemical SMILES is a language that allows us to describe the structure of a molecule using a set of symbols and characters. It follows certain grammar rules and uses a specific vocabulary to represent the atoms and bonds in a molecule. This language can be used to store chemical information and for chemical intelligence because the SMILES representations of structure can act as "words" in other languages[146]. SMILES ensures that there is only one representation of a molecule because the same atom and spanning tree will always produce the same SMILES string. SMILES is a symbolic representation of chemical molecules that can be read and understood by both humans and computers. It follows rules and conventions to represent the structural information of a chemical compound using short strings of characters. These strings can be interpreted and used by software to generate visual depictions of the molecules, perform simulations and calculations, and predict their properties and behavior. SMILES was developed in the 1980s as a way to represent the structure of chemical compounds in a form that can be read by computers. It is based on the concept of a linear representation, where each element of the molecule is represented by a symbol and the connectivity between them is indicated by brackets and numbers. SMILES is simple to learn but can encode a lot of structural information in a compact form, making it useful for database searches and other applications. It can also handle a wide range of chemical structures, including complex molecules with multiple rings, chiral centers, and other features. The use of SMILES has been valuable in the field of chemistry for storing, retrieving, and analyzing chemical compounds and has also led to the development of many computational chemistry tools and applications.

Natural language processing techniques can be applied to chemical SMILES strings to develop algorithms and tools for parsing and interpreting them, as well as generating SMILES strings from natural language descriptions of chemical structures[147]. These tools can be used in the development of chemical databases and the analysis of scientific literature, as well as in drug development and environmental safety by using machine learning to predict the toxicity or medicinal properties of chemical compounds based on their SMILES strings. Overall, SMILES strings are a useful tool in chemistry for efficiently representing and communicating chemical structures, and have various applications in natural language processing[148].

CHAPTER III

Predicting miRNA sequence using CNN and LSTM stacked in Seq2Seq architecture

Chapter 3. Predicting miRNA sequence using CNN and LSTM stacked in Seq2Seq architecture

i. Introduction

The human genome encodes for more than 2200 miRNAs, which are predominantly 28bp length non-coding RNA molecules that regulate post-transcriptional gene expression [149]. Since a single miRNA can target many gene transcripts, it is known that miRNAs regulate gene expression and mRNA translation [150]. Numerous recent studies have demonstrated the relevance of miRNAs in human diseases [151]. Mutations, dysregulations, or even malfunction of miRNA synthesis and their targets have been found to impede physiological and biochemical pathways, resulting in many human diseases [152]. Given the abundance of these regulatory RNAs, the diversity of their expression, and the large number of mRNA targets, it is not unexpected that miRNAs play a vital role in a wide variety of disorders, including immunological diseases[153], cancers[154], and different skin diseases [155], [156]. Although experimental approaches such as HITS-CLIP[157], PAR-CLIP[158], CLASH, etc. can be used to determine the interaction between miRNA and mRNA, this interaction has not yet been discovered. However, these procedures are quite time-consuming and require a significant number of cells for library preparation on occasion. Consequently, computational prediction of these associations is an extremely useful and currently active field of study. In-silico prediction of miRNA targets, which has been a hard problem for scientists to solve for decades, is a basic step in finding the relationship between miRNA and mRNA target. Current methods for predicting miRNA targets employ a variety of computational techniques, ranging from the demonstration of physical association algorithms to the application of machine learning algorithms [159]. Consensus features like seed site complimentary binding, homology of miRNAs, etc. have been used in the development of algorithms. As not all attributes are ubiquitous for all miRNA and RNA interactions, machine learning tools are also built with these properties as features, such as site accessibility, evolutionary conservation, and free energy. RNAhybrid[160], PicTar[161], TargetScan[162], PITA[163], Diana-microT[164] and others are In-silico tools for predicting the connection between miRNA and mRNA. As finding an experimentally validated negative set is laborious, the majority of these methods use artificial negative set data for training, resulting in limited sensitivity in real data. Furthermore, sequence level interaction is a fundamental criterion for mRNA and miRNA interaction, hence the bulk of these interactions can be found in sequence level characteristics that are difficult to produce. Therefore, in this objective the Seq2Seq architecture was proposed, which has been demonstrated to be applicable to sequence-level NLP data.

Deep neural networks are a kind of machine learning algorithms inspired by biological neural networks, i.e. how the human brain learns [165], [166]. Since their invention in the middle of the 20th century, they have not received much use since they are computationally intensive. Modern hardware systems, particularly GPUs (Graphics Processing Units) with more processing capabilities, have enabled neural networks to reclaim their popularity and uses. In addition, some types of neural networks, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs) with long short-term memory cells (LSTM), have applications in a variety of domains, including image processing, speech recognition, and natural language processing. CNNs and LSTM have also been utilised to solve a variety of biological challenges, including as the prediction of protein secondary structure, protein sub-cellular localization, peptide binding to MHC-II molecules, and picture detection of skin conditions, among others .

A typical neural network learns by altering the weights or priority of any given features to calculate output values with minimal error, i.e. values that are near to the provided output.

Using a process known as Gradient descent, the weights are computed in accordance with the difference between calculated and given output. CNNs are a sort of neural network employed for feature extraction when exact features or patterns cannot be identified, such as in image and sequence data. CNNs move a grid, also known as filters (a set of weights), across the input data, which are then supplied to the neurons of the next layer each time the filter is moved. These filters are not interconnected, and when traversing the data, they can extract patterns or characteristics from the input regardless of their position[96], [98].

RNNs are a type of neural network architecture that can process sequential data, including time series data and text data. In addition to the normal feed-forward connections, the neurons of hidden layers are connected with a time-delayed connection to maintain the weights from the previous time step.By storing instincts from past items and interpreting the current element in context of the preceding element, RNNs are able to learn from sequence data. Long short-term memory (LSTM) is a subtype of RNNs in which the simple hidden neuron matrix is replaced by an LSTM memory block that reduces the vanishing gradient problem [100], [101], [167]. The memory block cell of an LSTM is where context-dependent weights are stored. This block is additionally controlled by input, output, and forget control gates so that it may read the input sequence and determine the components to retain in the cell for each time step based on the input sequence. The advantage of LSTMs over RNNs is that it is easier for LSTMs to store an input characteristic over multiple time frames. Historically, LSTMs have been employed in the construction of chatbots, language translation, and image captioning, etc. [102], [168]

In the current study, we used CNN to extract sequence features from an input mRNA segment and then fed these features to a stacked LSTM system[143]. Typically, this design comprises of two LSTMs, encoder and decoder. The retrieved features from CNN are sent one-by-one in time steps to the encoder LSTM to obtain a fixed-dimensional vector of internal states; these internal states are then used as initial states by the decoder LSTM to extract the output sequence based on its initial state vector. This LSTM is an RNN language model[169] , but its training is dependent on the input sequence vector. The encoder processes the features from targeted mRNA sequences and outputs its hidden internal states, which are then used by the decoder LSTM as a condition or context for miRNA sequence prediction. We trained our model with mRNA and miRNA binding information obtained from TarBase version 8. (Sethupathy, 2005) [170], [171]. During training, our model's accuracy reached 80%. We discovered that the model learned to produce bases based on Watson and Crick base pairing. It was also capable of predicting the miRNA sequence based on perfect base pairing and a small number of instances of G:U wobble base pairing in the seed region. Importantly, the model generated valid miRNA sequences based on the sequence of the target mRNA.

After training the model, we utilised RNAplfold[172] from the RNA-Vienna Package[173] to locate site accessibility within mRNA where the target mRNA segment could be located. The ease with which a miRNA may identify and hybridise with an mRNA target is measured by site accessibility [174], [175]. The secondary structure of mRNA can hinder a miRNA's ability to bind to its target location. MiRNA: mRNA hybridization is a two-stage process in which a miRNA binds to a brief, accessible area of the mRNA in the first step. As soon as the miRNA completes binding to a target, the mRNA secondary structure unfolds. To determine the possibility that an mRNA segment is the target of a miRNA, it is necessary to estimate the predicted amount of energy required to make a site accessible to a miRNA. We validated our model using miRNA and RNA pairings implicated in dermatological illnesses that were retrieved from our in-house generated database miDerma. In this instance, our algorithm successfully predicted 72% of miRNAs from mRNA on average.

ii. Literature review

1. micro-RNA

a. Discovery and function of miRNAs

The discovery of miRNA, a type of short regulatory RNA, has had a significant impact on the field of genetics and gene expression. The first miRNA, lin-4, was discovered more than 30 years ago in the nematode Caenorhabditis elegans, and initially believed to be a protein-coding gene[176]. However, it was later discovered that lin-4 encodes a 22-nucleotide regulatory RNA that may bind to the mRNA of the lin-14 gene in the C. elegans developmental network, controlling protein synthesis. The discovery of lin-4 may have remained within the C. elegans scientific community if not for the discovery of a second miRNA, let-7, which is present in a wide range of species, including humans. This led to further research into the role of miRNAs in gene silencing, and the discovery of thousands of miRNAs in various organisms, including 2588 annotated miRNAs in the human genome. Each miRNA has the ability to regulate the expression of hundreds of target mRNAs, making the miRNA pathway as a whole an important mechanism for controlling gene expression[177]–[179].

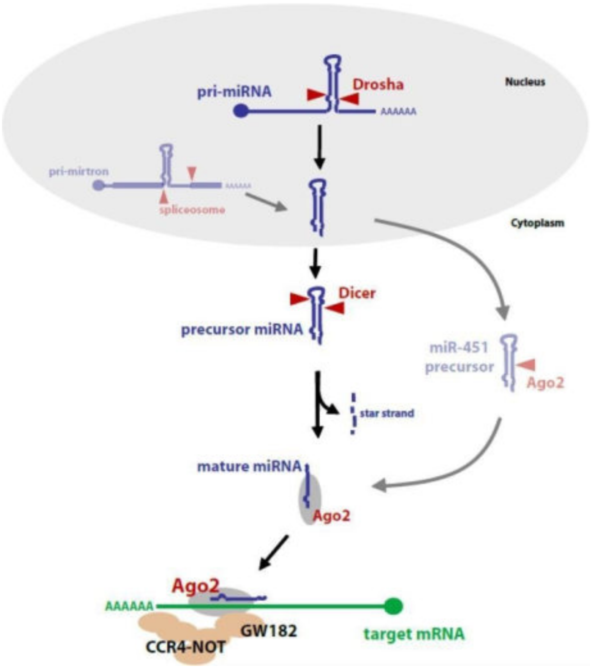


Figure 3.1: The microRNA biogenesis pathway.

b. The miRNA biogenesis pathway

The biosynthesis of miRNA involves a series of steps that convert the primary miRNA transcript into the active, 22-nucleotide mature miRNA. In mammals, the most common miRNA families follow a standard pathway for maturation. Once matured, the miRNA interacts with the RNA-induced silencing complex (RISC) to repress translation and degrade target mRNAs. There are some miRNA families that do not follow this standard pathway, but they will not be discussed in this context for the sake of simplicity[149].

c. Transcription

MiRNA genes can be located throughout the genome, and in some cases, transcription of the miRNA gene produces only the miRNA itself. In other cases, the miRNA may be located within the intron or untranslated region (UTR) of a protein-coding gene. The precursor RNA of a miRNA gene has a stem-loop structure that is shared by all miRNA genes, with either the first or second strand of the stem serving as the source of the mature miRNA. For example, the miRNA cluster MCM7 contains three stem-loops that each mature into a different miRNA with a unique set of targets[180]. If a miRNA is found in an exon that codes for a protein, removing it would delete the transcript that codes for the protein.

When the miRNA gene is transcribed by RNA polymerase II, it produces a primary miRNA transcript, known as Pri-miRNA. Like other mRNAs that code for proteins, the primary miRNA is processed through splicing, capping, and polyadenylation. While there has been limited study of miRNA promoters, those that have been analyzed show structural similarities to protein-coding gene promoters.

d. Processing by Drosha and Dicer

To become functional, the primary miRNA (pri-miRNA) must go through a two-step process of processing, which involves the action of two endonucleases. The first step occurs during pri-miRNA transcription, when the enzyme Drosha cuts the pri-miRNA to produce a stem-loop structure called the pre-miRNA[181], [182]. The RNA-binding protein DGCR8, which is associated with Drosha, is necessary for this cleavage to occur. The pre-miRNA is then transported out of the nucleus by Exportin5, a protein that acts in a Ran-GTPase-dependent manner. The second step of processing occurs in the cytoplasm, where the enzyme Dicer cuts the pre-miRNA to produce a 21-nucleotide RNA duplex. The RNA-binding protein TRBP, which is involved with both Drosha and Dicer, assists in this process. The mature miRNA is stored on one strand of the duplex in the RNA-induced silencing complex (RISC), while the other strand is usually degraded. Degraded miRNA strands are indicated by an asterisk (*) next to the miRNA name, such as miR-125*. It is possible that both strands of the miRNA duplex produced by Dicer will be incorporated into the RNA-induced silencing complex (RISC) at the same frequency. The 5p strand comes from the 5' end of the stem-loop structure, while the 3p strand comes from the 3' end. However, next-generation sequencing studies have shown that almost all miRNA families have a small fraction of their long strand loaded into RISC, suggesting that RISC may have a preference for incorporating one strand over the other. As a result, some miRNAs are expressed in a strand-specific manner depending on the cell type or biological condition. This has led to the adoption of the "5p/3p" naming scheme for miRNAs, rather than the "mature" or "star" naming scheme[183].

The cleavage sites of Drosha and Dicer are not always well-defined, leading to mature miRNAs with well-defined 3' ends[184]. However, some miRNAs have multiple cleavage sites, resulting in a variety of "isomiRs" that are variations of the parent miRNA. These isomiRs may have different target constraints and functions, and their expression can be influenced by cell-type or infection. The true mechanism by which isomiRs are regulated is not yet fully understood[185].

e. RISC loading and target repression

The main function of miRNAs is to be incorporated into the RNA-induced silencing complex (RISC), also known as miRISC[186], [187]. The core protein of RISC, argonaute, has four analogues (Ago1-Ago4), but it is not fully understood how they are organized. One proposed mechanism involves stacking the miRNA duplex into RISC and removing the passenger strand after Dicer cleavage. Ago2 can cleave and remove the passenger strand from a miRNA duplex if it has central region complementarity, after which the nuclease complex C3PO can destroy it. This is similar to the RISC stacking mechanism used in the related small interfering RNA (siRNA) pathway. However, most miRNA duplexes do not have central complementarity, which prevents them from participating in passenger strand cleavage. Some helicases have been shown to be able to unwind these miRNA duplexes[188].

During the RISC process, argonaute forms a strong bond with the miRNA and then searches for complementary target mRNAs. The "seed" region of the miRNA, which consists of nucleotides 2-7, is necessary for target binding. It has been shown that complementary target mRNAs can be identified based on their seed region, and that the 3' end of the miRNA also plays a role in target identification[189]. If a complementary seed region of the miRNA is found (nucleotides 9-11), Ago2 can degrade its mRNA target through an endonuclease mechanism. However, Ago2 is not able to effectively degrade the vast majority of human miRNA target binding sites because it lacks this property. Instead, it works together with the protein GW182 (TNRC6A/B/C) to form a complex in the cytoplasmic P bodies, where translational repression occurs. The recruitment of RISC by the CCR4-NOT deadenylase complex leads to the removal of the poly(A) tail and the subsequent degradation of the mRNA target.

2. miRNA target identification

It is important to understand which molecules miRNAs target for various reasons. To understand the biological function of a miRNA, it is necessary to identify its target list, and researchers have made significant progress in this area in recent years. Validated target binding sites are the most reliable biomarkers for determining the effectiveness of a miRNA enhancer or inhibitor, which is important information for biologists developing miRNA therapies. There are three main approaches that can be used to identify miRNA targets: bioinformatic target prediction, biochemical isolation of miRNA and mRNA complexes, and transcriptomic and proteomic analysis[190]. These approaches are briefly summarized.

a. Computational target prediction

According to the literature, bioinformatic target identification should be reliable because miRNAs bind to target mRNAs using the standard Watson-Crick base pairing rules. Although miRNAs are only 6 nucleotides long, the seed configuration is the most important predictor of target selection. As a result, there may be many false-negative competitor targets. To improve precision, bioinformatics target prediction calculations include additional components such as sequence conservation, flanking sequence determinants, flanking arrangement determinants, outside-seed compensatory matching, and target site accessibility. Some approaches have also used machine learning calculations that combine validated target sets as training sets. There are several tools available for bioinformatic target prediction, including TargetScan[191]–

[193], miRanda[194], and PicTar[195]. Bioinformatic techniques are a useful starting point for miRNA research and are widely used in research laboratories.

b. Biochemical target identification

Other approaches to identifying miRNA targets involve the physical proximity of miRNA and RISC structures to their respective target mRNAs. These approaches typically involve immunoprecipitation of the RNA-induced silencing complex (RISC) using anti-argonaute antibodies, with or without prior RNA crosslinking, followed by identification of bound target RNAs using microarray or next-generation sequencing (NGS) profiling. It is generally preferred to crosslink the immunoprecipitation (IP) before cell lysis to avoid artifactual RNA hybridization during cell lysis. Another method is to extract target-specific biotinylated miRNAs, which allows for the capture of targets of a single known miRNA, but requires the insertion of the biotinylated miRNA ectopically. These physical approaches have been useful in identifying the mRNA targets of miRNA complexes, but it is possible that not all mRNA within a specific space is repressed. Argonaute-bound targets in mRNA coding regions have been shown to be non-degrading. The success of any methodology depends on careful consideration and optimization[196]–[198].

c. Omics-based strategies for target identification

Proteomic or transcriptome analysis of cells or tissues in the presence or absence of a miRNA is the third general method for target identification. It is still being investigated if quantitative proteomic analysis, which directly evaluates a miRNA's impact on protein synthesis, is more representative of the real target set. Transcriptome analyses are made simpler by the fact that most miRNA targets have lower mRNA steady-state levels. Microarray profiling should make this practicable, and a few research tools are currently accessible. This approach led to the identification of neutrophil-specific miR-223 [199] foci. Both animals with miR-223 and those lacking it showed decreased neutrophil activation. To identify the miR-223 targets, mRNA and protein levels were compared using microarrays and quantitative mass spectrometry. It is essential to remember that the competing target sets will also contain downstream auxiliary targets that need to be individually authorised.

3. Methods for Identifying miRNA

Techniques for recognising miRNAs can be classified into two broad categories [200]. The disclosure strategies are geared toward rapid, high-throughput profiling of a large number of miRNAs. These methods excel in two areas: microarray hybridization and next-generation sequencing profiling. The latter is superior because it can depict novel miRNAs, whereas most other methods can only identify existing miRNA sequences. There are several NGS phases available, but they all begin with the creation of a library format. Priorities are synchronised at RNA connectors, which are ligated to the ends of short RNA fragments prior to RT-PCR amplification. Using this method, all RNAs within the target size range will be amplified. The libraries could then be sequenced on a variety of devices, the most common of which would most likely be Illumina stages. The PCR amplification step can, however, be entirely disregarded by single-particle instrumentation[201]. It is conceivable to multiplex 48 libraries (or more) in a single run and yet attain the proper grouping read depth because contemporary instruments are capable of 200 million or more peruses per library run[202]. Quantitative articulation profiles are derived after reading a sequence. As was already said, it is possible to identify and classify tiny RNA species, including miRNAs, that were not previously recognized. Although NGS-based profiling has clear advantages and is evolving into the industry norm, nucleotide biases brought on by ligation procedures have been noted[203]. This indicates that a validation stage is necessary for the profiling technique.

Despite the fact that NGS systems can carry out high-throughput profiling of the whole miRNA population, the majority of clinical demonstration techniques rely on quick inspection of a limited number of quality marks. As a result, RT-PCR and nano-strings are frequently utilised in recent analytical techniques[204] . In a quick exploratory run, the nanostring approach may measure up to 500 targets (mRNA or miRNA) with a single particle hybridization. The pancreatic growth test by Asuragen is an LDT miRNA symptom. In this RT-PCR-based method, a 7-miRNA marker is used to tell the difference between pancreatic ductal cancer and healthy tissue.

a. Common features of miRNA target prediction tools

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

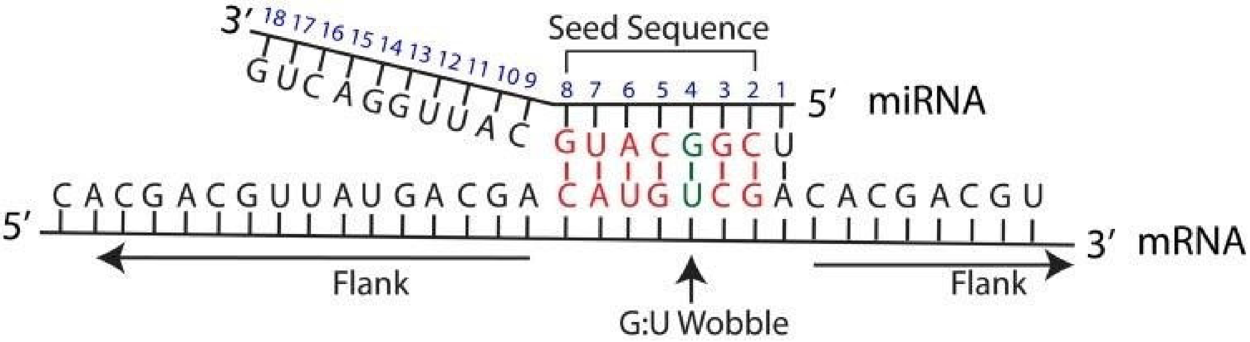


Figure 3.2: microRNA:mRNA target interaction.

Seed match

A miRNA's seed sequence is defined as the first 2–8 nucleotides, starting at the 5' end and moving toward the 3' end [45] (Figure 3.2). Most algorithms require a seed complementary to Watson-Crick (WC) pairing rules between a miRNA and its target site. When adenosine guanine (G) pairs with cytosine (C) and adenosine (A) pairs with uracil, the miRNA and mRNA nucleotide (U) form a Watson-Crick match.

There are several types of seed coordinates that can be considered based on the calculation. The following are the fundamental types of seed matches: [205]–[207]:

1. 6mer: A six-nucleotide WC match between the miRNA seed and mRNA.
2. 7mer-m8: A perfect WC match from miRNA seed nucleotides 2–8.
3. 7mer-A1: A perfect WC match from miRNA seed nucleotides 2–7, plus an A across from miRNA nucleotide 1.
4. 8mer: A perfect WC match from miRNA seed nucleotides 2–8, plus an A across from miRNA nucleotide 1.

Conservation

The preservation of the same sequence pattern across all species is the aim of conservation. Regions of the 3' UTR, the 5' UTR, miRNAs, or any combination of the three might be the subject of a conservation analysis. The conservation of the seed region of miRNAs is often greater than that of the non-seed region [208]. In a limited percentage of miRNA-mRNA target interactions, there is conserved pairing at the 3' end of the miRNA that can make up for seed mismatches; these sites are referred to as "3' compensatory sites" [209]. Because it is being selected for, conservation analysis may show that a projected miRNA target is functioning in the context of identifying miRNA targets in 3 UTRs. Studying the genomic areas around the miRNA gene and the miRNA target genes for conservation is also gaining popularity. The promoter regions of miRNAs and their target genes have been studied using conservation analysis [210], as well as the colocalization of independently transcribed miRNAs and surrounding protein-coding genes [51]. As a result, there are several ways to analyse areas in the 3 UTR, 5 UTR, miRNA, or any combination of the three to determine the impact of conservation on miRNA target prediction. The miRNA seed region often has higher conservation than the non-seed region [45]. In a tiny percentage of miRNA: mRNA target interactions, there is conserved pairing at the 3' end of the miRNA that can make up for seed mismatches; these sites are referred to as 3' compensatory sites [211]. Preservative investigation may validate that a predicted miRNA target is beneficial because it is being picked for when it comes to predicting miRNA targets in 3 UTRs. The genomic areas around miRNA quality and miRNA target characteristics are attracting increasing interest in conservation analyses. Examples include the co-convergence of independently interpreted miRNAs and surrounding protein-coding genes [51], as well as the objective characteristics of miRNA promoter areas and their relationship to preservation studies [212]. This is how conservation is frequently used for the prediction of miRNA targets.

Free energy

The stability of a biological system can be assessed using its free energy, also known as Gibbs free energy[213]. If the binding of a miRNA to a potential target mRNA is predicted to be stable, the mRNA is considered to be a real target of the miRNA. Because it can be difficult to measure free energy directly, the change in free energy that occurs during a reaction is often taken into account (ΔG). Systems are more stable when negative ΔG values are obtained because there is less energy available to respond in the future. By predicting the hybridization between the miRNA and its potential target, it is possible to identify regions of high and low free energy and use the total G as a measure of how strongly bound they are.

Site accessibility

The accessibility of a site determines the ease with which a miRNA can locate and bind to an mRNA target[174]. After transcription, mRNA adopts a secondary structure that may obstruct a miRNA from binding to a target site[214]. During the initial phase of the two-step miRNA:mRNA hybridization process, a miRNA binds to a brief, accessible region of the mRNA. Upon completion of binding to a target, the mRNA's secondary structure becomes more open. Therefore, the likelihood of an mRNA being a miRNA target can be determined by estimating the energy required to make a site accessible to a miRNA[215].

b. Less common features of miRNA target prediction tools

The majority of miRNA target prediction programmes incorporate the aforementioned features. As new developments in the characterization of miRNA:mRNA target interactions are made, additional characteristics are added. These may be integrated into the target prediction itself or utilised to forecast the target's efficacy. The quantity of target sites in a 3′ UTR is a gauge of the total number of target sites (Garcia et al., 2011). "Local AU content" refers to the quantity of A and U nucleotides surrounding the appropriate miRNA seed region. The "GU wobble" is the acceptance of a G pairing with a U rather than a C in the seed match. With miRNA nucleotides 12–17, base pair matching is known as 3′ compensatory pairing[216]. The anticipated duplex's computed free energy depends on the stability of the seed pairing. The target site's location within the mRNA is investigated via the position contribution technique. Machine-learning methods create a model of miRNA targets from training data, which is then used in the miRNA prediction process. Machine-learning algorithms are likely to include more features in their predictions since they may be trained to identify each feature's predictive potential on both positive and negative datasets. Several of these tools employ the machine-learning technique known as support vector machines (SVM)[217].

4. Review of commonly used miRNA target prediction tools

Using the previously described characteristics, ten popular miRNA target prediction tools in this section are presented. The section Comparison of MiRNA Target Prediction Tools includes a summary table comparing these tools.

Table 3.1: Summary table of miRNA target prediction tools.

	FEATURES USED IN miRNA TARGET PREDICTION						
Tool name	Seed match	Conservation	Free energy	Site accessibility	Target-site abundance	Machine learning	**References**
miRanda	X	X	X				[218]
miRanda- mirSVR	X	X	X	X		X	[219]
TargetScan	X	X					[207]
DIANA-microT-CDS	X	X	X	X	X	X	[220]
MirTarget2	X	X	X	X		X	[221]
RNA22-GUI	X		X				[222]
TargetMiner	X	X	X	X	X	X	[223]
SVMicrO	X	X	X	X	X	X	[224]
PITA	X	X	X	X	X		[225]
RNAhybrid	**X**		**X**		**X**		[226]

iii. Methodology

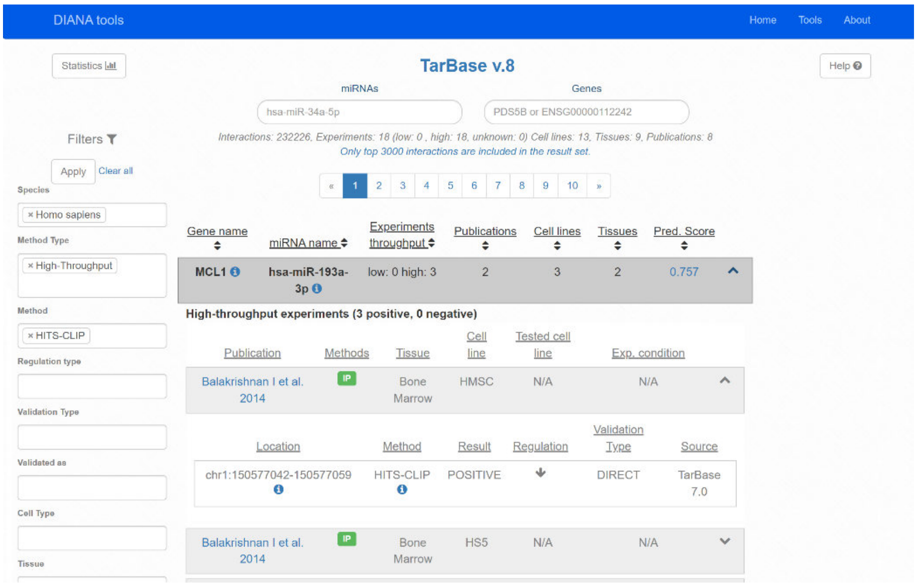
1. Data Curation

A web crawler was implemented in Python using the Beautiful Soup package for TarBase v8 in order to retrieve data. The crawler, upon receiving a miRNA name as input, searches for its target sites in every entry in TarBase v8 and provides an output in the form of a CSV file containing the miRNA name, gene symbol, and chromosome location of the target binding site according to Ensemble Human (GRCh38.p12) annotation. The code for this web crawler can be found in GitHub repository (https://github.com/rajkumar1501/sequence-prediction-using-

CNN-and-LSTMs).

A genome-wide method for identifying protein-RNA binding sites or RNA modification sites in vivo is high-throughput sequencing of RNA obtained by crosslinking immunoprecipitation (HITS-CLIP, also known as CLIP-Seq)[157], [227]. The neuron-specific RNA-binding proteins and splicing factors NOVA1 and NOVA2 were first mapped using HITS-CLIP; since then, a number of other splicing factor maps have been produced, including those for PTB, RbFox2, SFRS1, hnRNP C, and even N6-Methyladenosine (m6A) mRNA modifications.

By decoding miRNA-mRNA and protein-RNA interaction maps in mouse brain[228], [229] and subsequently in Caenorhabditis worms, embryonic stem cells, and tissue culture cells, HITS-CLIP of the RNA-binding protein Argonaute has been used to identify miRNA targets[157].



miRNAs: hsa-miR-34a-5p

Genes: PDS5B or ENSG00000112242

Interactions: 232226, Experiments: 18 (low: 0 , high: 18, unknown: 0) Cell lines: 13, Tissues: 9, Publications: 8

Only top 3000 interactions are included in the result set.

Species: × Homo sapiens

Method Type: × High-Throughput

Method: × HITS-CLIP

Regulation type

Validation Type

Validated as

Cell Type

Tissue

« 1 2 3 4 5 6 7 8 9 10 »

Gene name	miRNA name	Experiments throughput	Publications	Cell lines	Tissues	Pred. Score
MCL1	hsa-miR-193a-3p	low: 0 high: 3	2	3	2	0.757

High-throughput experiments (3 positive, 0 negative)

Publication	Methods	Tissue	Cell line	Tested cell line	Exp. condition
Balakrishnan I et al. 2014	IP	Bone Marrow	HMSC	N/A	N/A

Location	Method	Result	Regulation	Validation Type	Source
chr1:150577042-150577059	HITS-CLIP	POSITIVE	↓	DIRECT	TarBase 7.0

Publication	Methods	Tissue	Cell line	Tested cell line	Exp. condition
Balakrishnan I et al. 2014	IP	Bone Marrow	HS5	N/A	N/A

Figure 3.3: Screenshot of result page of TarBase v.8.

A sample of retrieve data can be found below.

Table 3.2: Sample of Retrieve data for training.

miRNA _name	miRNA_sequence	Chromosome Location	mRNA_Sequence	Gene Symbol
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	15:98960051-98960068	ACUCCAUCUAUUUACAAA	IGF1R
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	13:48480030-48480053	ACUCCAUAGGUACGAUAGUAAGUA	RB1
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	1:205602438-205602456	ACUCCAUCCCAUCCAUGAA	MFSD4
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	1:38863569-38863589	ACUCCAUCCGUAGUGCCUG UA	MYCBP
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	1:35853829-35853851	ACUCCAUUUUUAAGUCAGGUCAC	AGO4
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	17:49051062-49051083	ACUCCAUCAAAUGAAGCGUGUG	IGF2BP 1
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	1:207048984-207049003	ACUCCAUCAUCCGAAGUUG G	YOD1
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	3:47736093-47736115	ACUCCAUCUACAACCCAGA CCAG	SMARC C 1
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	15:52065475-52065494	ACUCCAUCUAUUGUGUACAC	MAPK6
hsa-let-7a- 5p	UGAGGUAGUAGGUUG UAUAGUU	14:52642082-52642099	ACUCCAUAUAUCGAAGAA	ERO1L

2. Data Preparation

a. Data Cleaning

Since the data was fetched with crawler, all sequences were manually curated to see the polarity of the strands. All miRNA sequence strands were in polarity 3’ to 5’ and all mRNA strands were in polarity of 5’ to 3’, thereby conserving the seed pair features of 3’ and 5’ end of miRNA and mRNA.

The lengths difference between the pairs of miRNA and mRNA varied from 0 to 18. In order to ensure uniformity in the difference in length of the sequences, so that the model may extract features based on patterns in the sequence rather than the length of the sequences, the distribution of the pair sequence length difference was analyzed. It was found that the distribution was skewed, with approximately 90% of the difference in length pair of sequences falling between 0 and 6 (Figure 3.4). As a result, a threshold of 6 was established, and miRNA and mRNA sequences with a sequence length difference of 6 or less were used for further analysis.

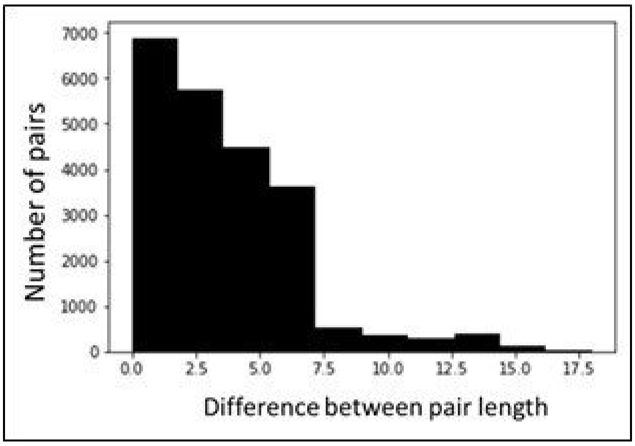


Figure 3.4: Distribution difference in length pair of sequences.

Subsequently, the binding energies for sequence pairs were analyzed, as negative ΔG values are necessary for the binding of a sequence. The RNAFold package [173], [230] from the ViennaRNA Package 2.0 was utilized for this purpose. The two sequences were combined by adding a spacer, such as 8 'L' nucleotide bases, which are not considered meaningful according to the RNAfold package [231]. The RNAfold tool was then run using the sequence in order to determine the binding energy (Figure 3.5). After filtering out sequences with positive ΔG values, a set of 19300 sequence pairs was obtained and used for training.

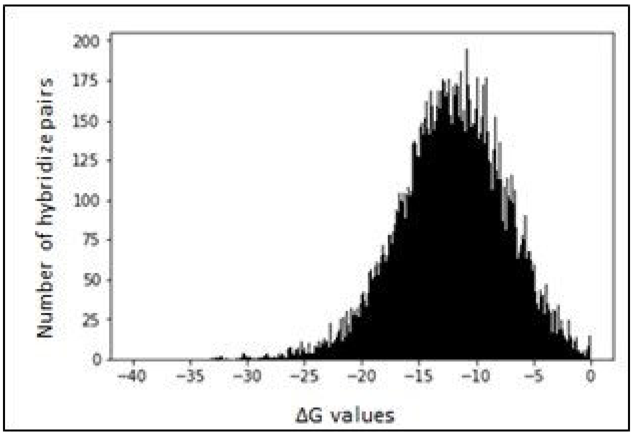


Figure 3.5: Distribution ΔG values of hybridized pairs

In order to proceed, it is necessary to encode the categorical information of mRNA and miRNA nucleotides numerically. This is typically done using binary vectors with one hot encoded embeddings, in which all entries are set to zero except for one, indicating the category. For example, the typical encoding for RNA nucleotides (categories) is A = (1 0 0 0), G = (0 1 0 0), C = (0 0 1 0), and U = (0 0 0 1). By concatenating the encoding nucleotides together and using each nucleotide as a separate input feature in a feed-forward neural network, a DNA sequence can be represented as a binary string. In our case, we have added sequence start and end tags, denoted by the letters "\t" and "\n," to the miRNA sequences. These modified sequences will be passed to the decoder, as explained in the section on constructing the model.

Table 3.3: 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

Binary coded strings as input of a deep neural network are made be of same length, i.e. all miRNAs should be of same length and all mRNA-target site sequence should be of same length, which is not the scenario of real world. To address this issue, we took the longest miRNA sequence as the default length for all miRNA sequences, and padded any miRNA sequence whose length was less than the default length with zeros. The default length for miRNA sequences was determined to be 28. The same approach was applied to the mRNA sequences, and the default length for these sequences was determined to be 29.

3. Model Building

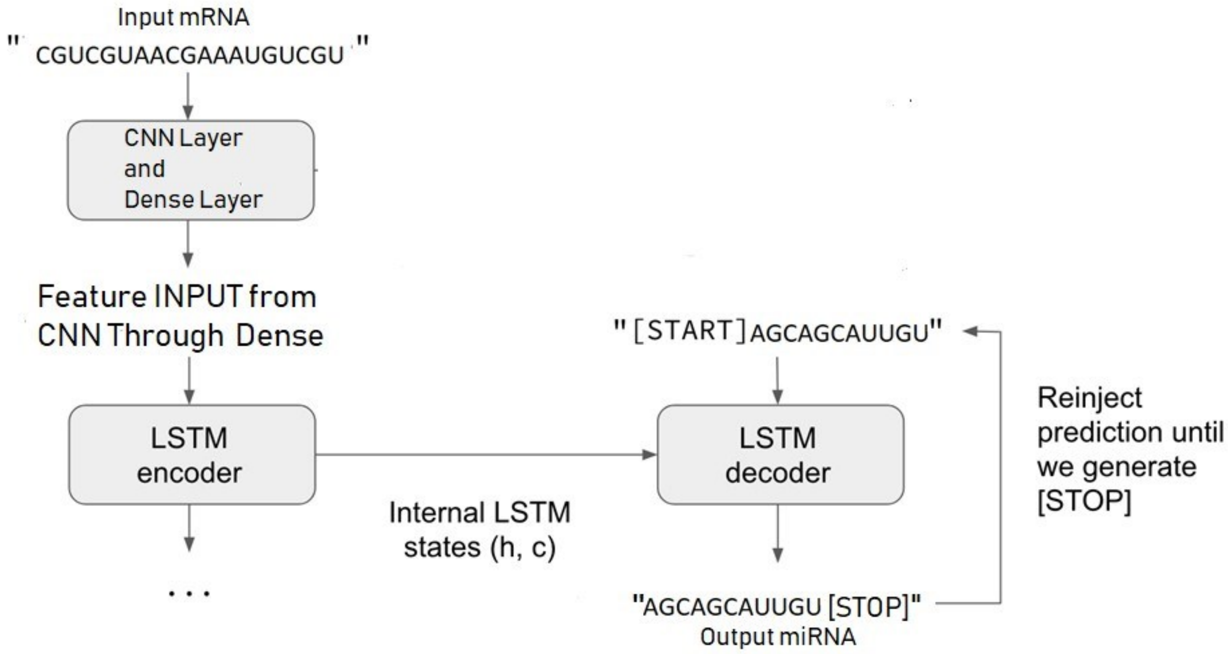

Input mRNA
" CGUCGUAACGAAAUGUCGU "
CNN Layer and Dense Layer
Feature INPUT from CNN Through Dense
LSTM encoder
...
Internal LSTM states (h, c)
"[START] AGCAGCAUUGU"
LSTM decoder
"AGCAGCAUUGU [STOP]"
Output miRNA
Reinject prediction until we generate [STOP]

Figure 3.6: Proposed Seq2Seq model using CNNs and LSTMs for microRNA sequence prediction.

The Figure 3.6 provided represents our proposed model for predicting miRNA sequences from mRNA sequences. The Keras library of deep learning was utilized in Python, with Tensorflow as the backend, to construct and train the model.

An open-source programming framework called TensorFlow is used to programme dataflow across a variety of tasks. It is a representational math library that is also used in machine learning systems like neural networks. At Google, it is used for both research and devlopmnet. A high-level neural network API called Keras was created in Python and may be used with TensorFlow, CNTK, or Theano. It was created with the goal of facilitating quick experimentation. Good research relies on being able to move quickly from concept to conclusion. Keras provides quick and simple prototyping (through user friendliness, modularity, and extensibility). supports both recurrent and convolutional networks, as well as hybrids of the two. runs well on both the CPU and GPU.

The first step in the process was to extract features from the mRNA sequence using a convolutional neural network (CNN). A window size of 8 was used to extract 128 features.

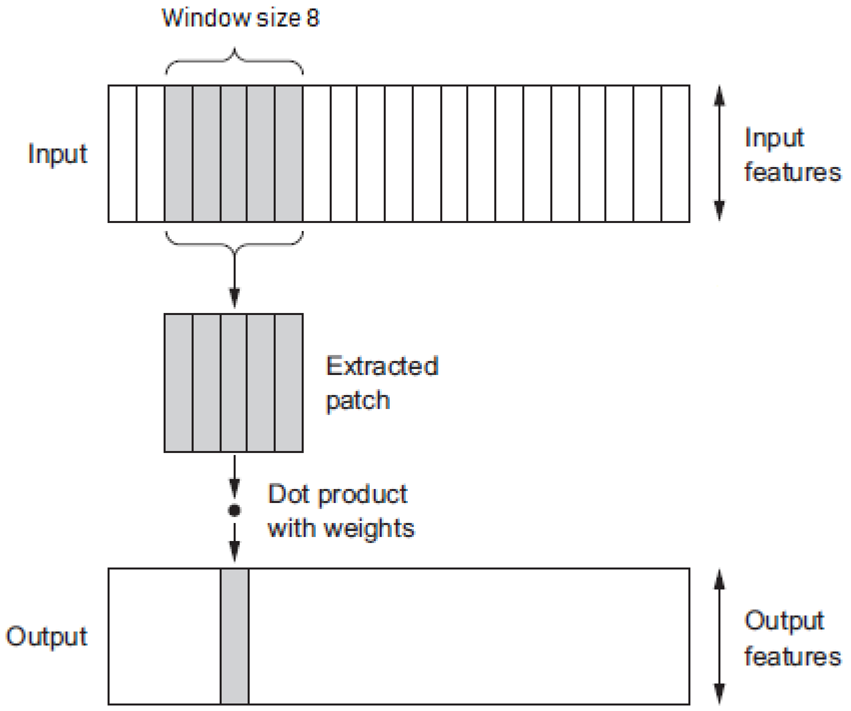


Figure 3.7: Feature extraction using CNN on mRNA sequence.

A dense layer consisting of 128 neurons was then utilized to adjust the weights of the 128 features extracted from the mRNA sequence. These adjusted features were then fed into an LSTM network.

Building Seq2Seq LSTM Network:

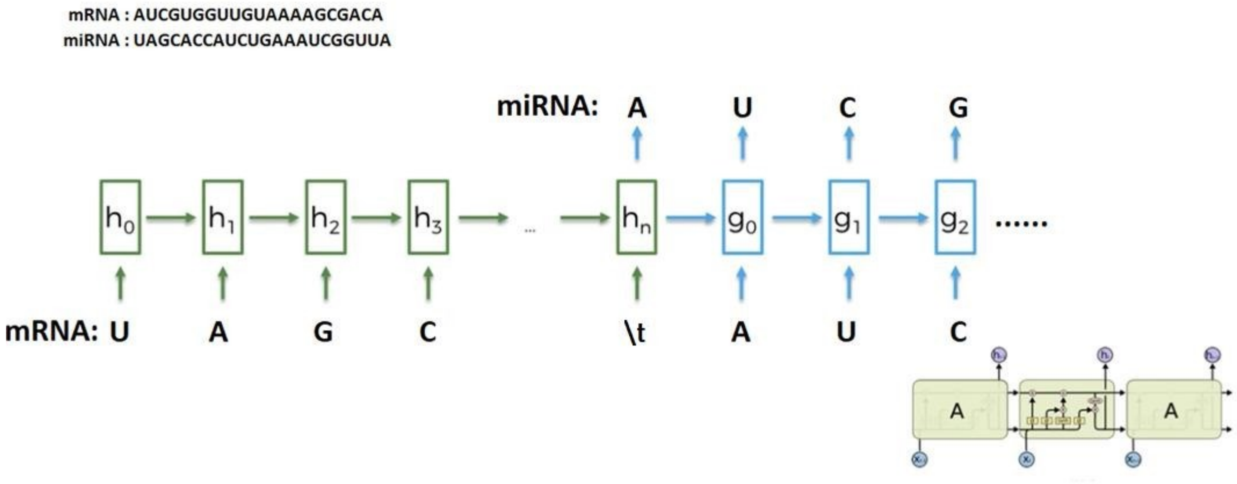

mRNA : AUCGUGGUUGUAAAAGCGACA
miRNA : UAGCACCAUCUGAAAUCGGUUA
miRNA: A U C G
h_0 h_1 h_2 h_3 ... h_n g_0 g_1 g_2 ......
mRNA: U A G C \t A U C
A
A

Figure 3.8: Seq2Seq LSTM model.

Given the input sequences from one domain (such as mRNA sequences) and corresponding target sequences from another domain (such as miRNA). An LSTM encoder converts the input sequences into two-state vectors, keeping the final state of the LSTM and discarding the outputs. A decoder LSTM is trained to transform the target sequences into an identical sequence, but offset by one timestep in the future, using a training process known as "teacher forcing." The decoder uses the encoder's state vectors as the initial state. In practice, the decoder learns to generate targets[t+1...] given targets[...t] based on the input sequence.

To decode unknown input sequences in inference mode, the following steps are taken:

1. Encode the input sequence into state vectors, starting with a target sequence of size 1 (just the start-of-sequence character).
2. Feed the state vectors and 1-char target sequence to the decoder to generate predictions for the next character.
3. Sample the next character using these predictions (argmax was used for this purpose).
4. Append the sampled character to the target sequence.

5. Repeat until the end-of-sequence character is reached or the character limit is reached.

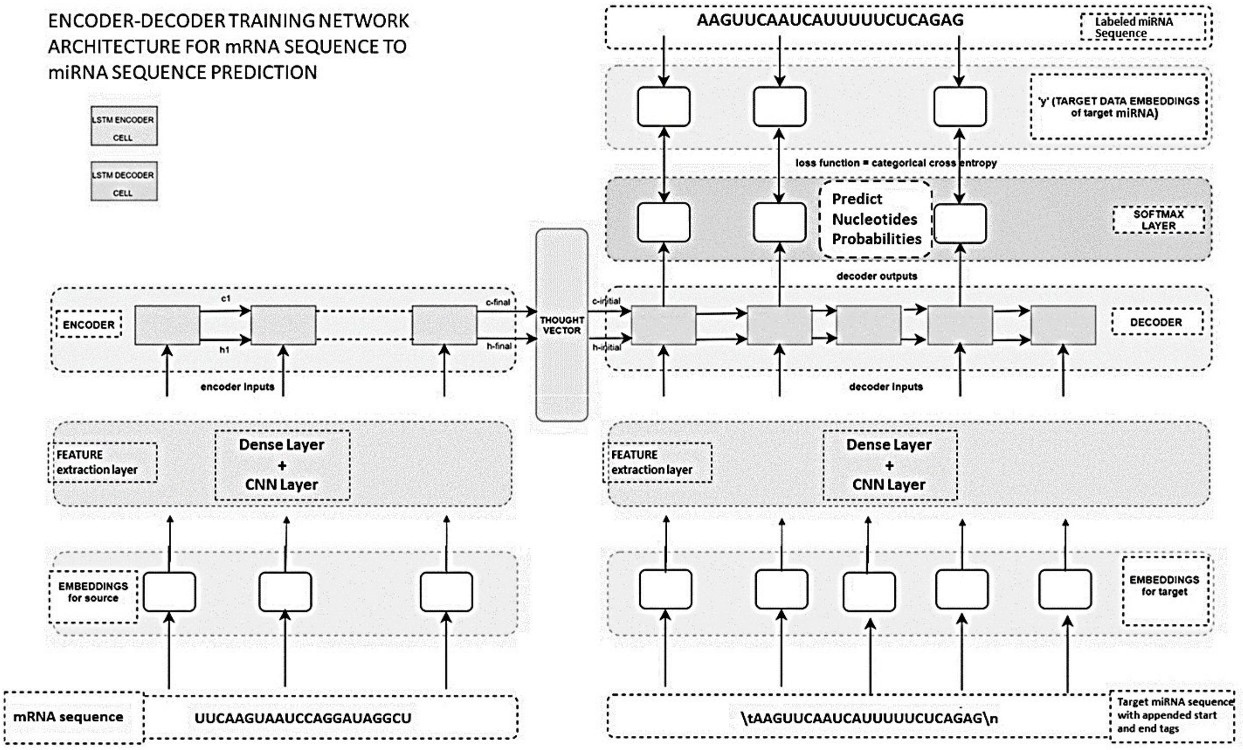


Figure 3.9: Final Model of model constitute of Conv1D, Dense and LSTMs layers.

The python code for the model development training and prediction can be found in github repository (https://github.com/rajkumar1501/sequence-prediction-using-CNN-and-LSTMs)

4. Training of model

The goal of model training is to find parameters w that minimise an objective function L(w), which measures the fit of the model's predictions to the actual observations. The crossentropy for classification and the mean squared error for regression are the two most commonly used objective functions. L(w) is difficult to minimise because it is high-dimensional and nonconvex (Figure 3.9)

5. Determining the number of neurons in a network

It is desirable to utilize a validation set in order to determine the optimal number of hidden layers and hidden units for a given task. A common strategy is to increase the number of layers and units while avoiding overfitting the data. Empirical research has shown that adding layers and units can increase the number of local optima and representable functions, while decreasing the sensitivity of finding a suitable local optimum to weight initialization. After testing various combinations of neurons across all layers, it was determined that 128 neurons for dense layers and 512 neurons for LSTMs produced the best results.

6. Partitioning data into Training and Validation sets

To avoid overfitting and ensure that the model will generalise to previously unseen data, machine learning models must be trained, validated, and tested on independent data sets. Deep neural networks require data to be partitioned into training, validation, and test sets for proper training. The models use the training set to learn different hyperparameters, which are then tested on the validation set. The model with the best performance, such as prediction accuracy or mean squared error, is chosen and evaluated further on the test set to quantify performance on unseen data and to compare to other methods. For training, 80% of the data from the 19300 data sets was used, with the remaining 20% reserved for validation. Additionally, data on miRNAs associated with skin diseases from the in-house developed database, miDerma, was utilized for further testing. This database consists of miRNA and mRNA pairs that have been linked to dermatological disorders.

7. Learning Rate and Batch size

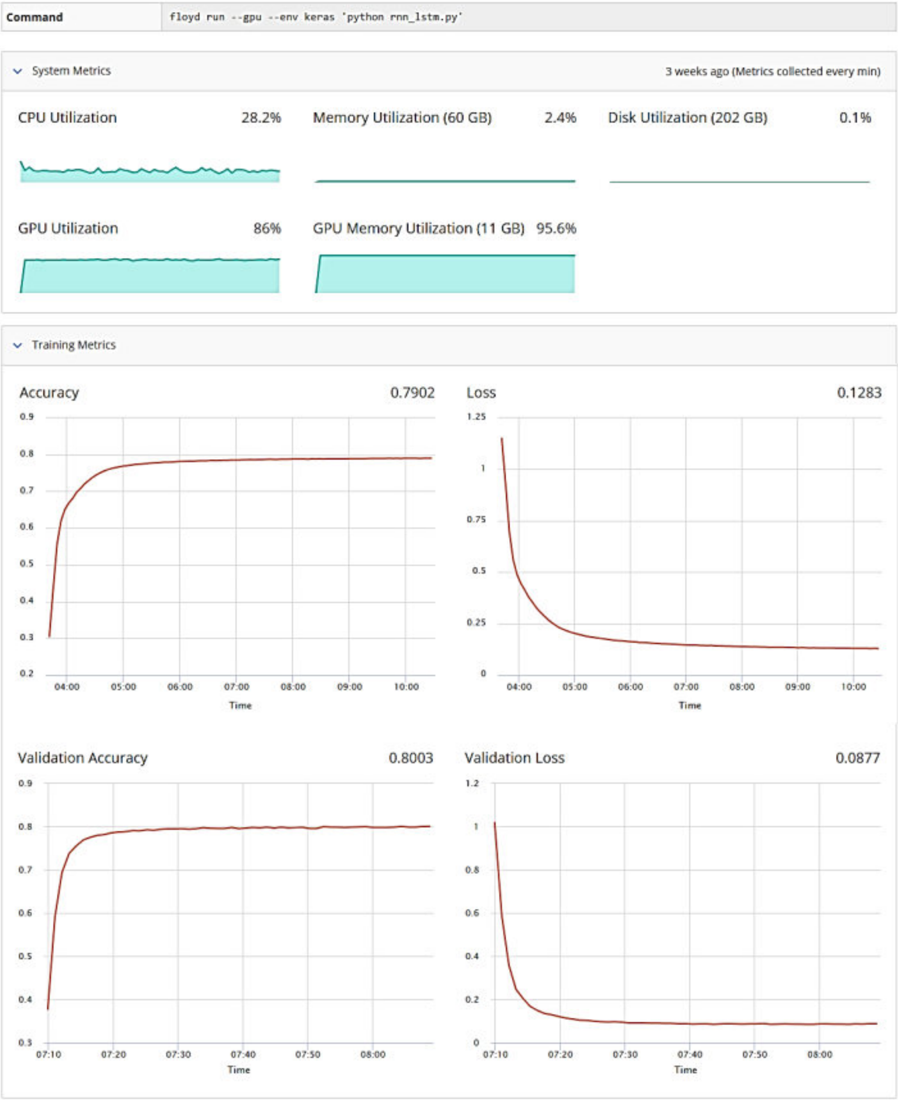
The stochastic gradeint decents learning rate and batch size should be chosen with care, as they have a significant impact on the preparing rate and model execution. Various learning rates are commonly investigated on a logarithmic scale, such as 0.1, 0.01 or 0.001, with 0.01 serving as the prescribed default orders. For most applications, a batch size of 128 is appropriate for test preparation. The batch size can be increased to speed up training or decreased to reduce memory usage, which is important when working with complex models on GPUs with limited memory. Larger batch sizes frequently necessarily require lower learning rates. In this study, a default learning rate of 0.01 and a batch size of 50 was used.

8. Avoiding overfitting

Because they are nonlinear and have numerous parameters, complex neural systems are notoriously difficult to prepare. Overfitting to information is a noteworthy test. Overfitting occurs as a result of an excessively complex model relative, making it impossible to span the preparation set, and can thus be reduced by reducing the model's many-sided quality, such as the number of hidden layers and units, or by expanding the size of the preparation set, such as through information expansion. To prevent overfitting, the following precautions were taken:

- A dropout rate of 0.5 was used in the LSTM layers.
- A 0.001 L2 regularization penalty was applied to each Dense layer.

The model was trained for 100 epochs on a cloud computing instance with 32GB RAM, an 11GB NVIDIA Tesla K80 GPU, and an Intel Xeon 8-core CPU.



Command: `floyd run --gpu --env keras 'python rnn_lstm.py'`

System Metrics — 3 weeks ago (Metrics collected every min)

CPU Utilization 28.2%

Memory Utilization (60 GB) 2.4%

Disk Utilization (202 GB) 0.1%

GPU Utilization 86%

GPU Memory Utilization (11 GB) 95.6%

Training Metrics

Accuracy 0.7902

Loss 0.1283

Validation Accuracy 0.8003

Validation Loss 0.0877

Figure 3.10: Training setup for model .

9. Obtaining Surface Area accessibility Regions

For predicting miRNAs, our model requires a gene symbol. The ensemble REST API is used to retrieve the sequences of all protein coding transcripts and their 3'UTRs from a query gene. Because the ensemble REST API accepts transcript IDs, a local SQLite database was created to map gene symbols to all protein coding ensemble transcript IDs.

The ease with which a miRNA can locate and hybridise with an mRNA target is measured by the site's accessibility. Following transcription, mRNA takes on a secondary structure [53], which can interfere with the ability of a miRNA to bind to a target site. MiRNA:mRNA hybridization is a two-step process that begins with a miRNA binding to a short, accessible region of the mRNA. As the miRNA binds to a target, the secondary structure of the mRNA unfolds. So, to figure out if an mRNA is the target of a miRNA, you should look at how much energy you think it will take to make a site accessible to a miRNA.

To compute locally stable secondary structure pair probabilities, the RNAplfold programme from ViennaRNA Package 2.0 [173] was utilised. This Python wapper package computes local pair probabilities for base pairs with a maximum span of L. The probabilities are averaged over all L-dimensional windows containing the base pair.

The result is a simple tuple of matrices, with each line containing a position x followed by the probability that x is unpaired, [x-1..x] is unpaired, [x-2..x] is unpaired, and so on until the probability that [x- i+1..x] is unpaired is reached.

Total accessibility was calculated with RNAPlfold. Total accessibility is the sum of Pfree's (probability of unpaired 4-mers) over all accessible 4-mers in all complementary sites. If Pfree is greater than 0.2, those 4-mers are said to be accessible. Following the approach described in [89], we used W = 80 and L = 40 and fed sequences of all protein coding transcripts of a query gene to RNAplfold in order to identify accessibility regions or accessible 4-mers. These accessible 4-mers were then used to locate miRNA binding sites in the 3'UTR region of the respective mRNA, resulting in 26-mers consisting of the 4-mers plus the following 22-mers. The complete Python code for locating the accessibility region in the 3' UTR of a gene's transcript can be found in GitHub repository (https://github.com/rajkumar1501/sequence-prediction-using-CNN-and-LSTMs).

10. Developing Final package

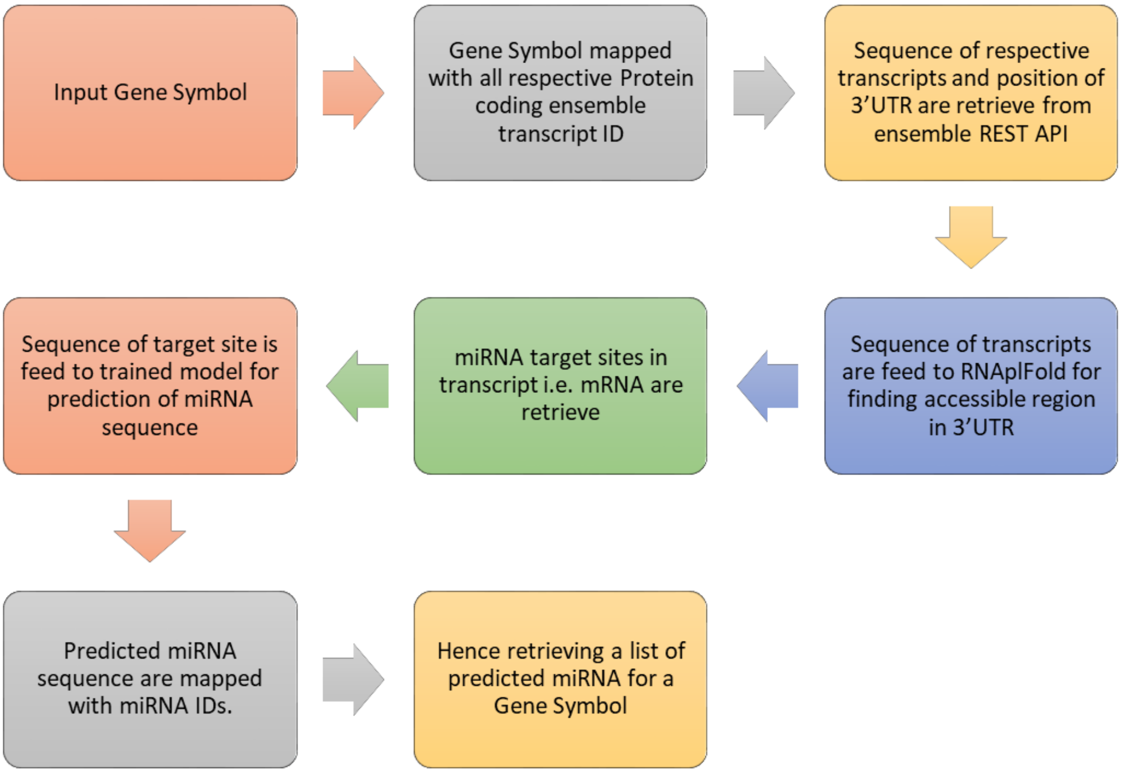

Input Gene Symbol
Gene Symbol mapped with all respective Protein coding ensemble transcript ID
Sequence of respective transcripts and position of 3'UTR are retrieve from ensemble REST API
Sequence of transcripts are feed to RNAplFold for finding accessible region in 3'UTR
miRNA target sites in transcript i.e. mRNA are retrieve
Sequence of target site is feed to trained model for prediction of miRNA sequence
Predicted miRNA sequence are mapped with miRNA IDs.
Hence retrieving a list of predicted miRNA for a Gene Symbol

Figure 3.11: Workflow of model .

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 miRNA 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 miRNA sequences. Then these predicted miRNA sequences will be mapped to their miRNA IDs using a local SQLite database containing miRNA ID and respective sequence retrieve from mirBase Release 22, March 2018. Hence giving output a list of predicted miRNA IDs. Whole flowchart is shown in Figure 3.11.

iv. Results

A package for predicting miRNA 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 was trained on data set containing sequences of miRNA and their respective target binding sites in mRNA which are retrieve from TarBase v8. The model was trained for 100 epochs.

1. Accuracy

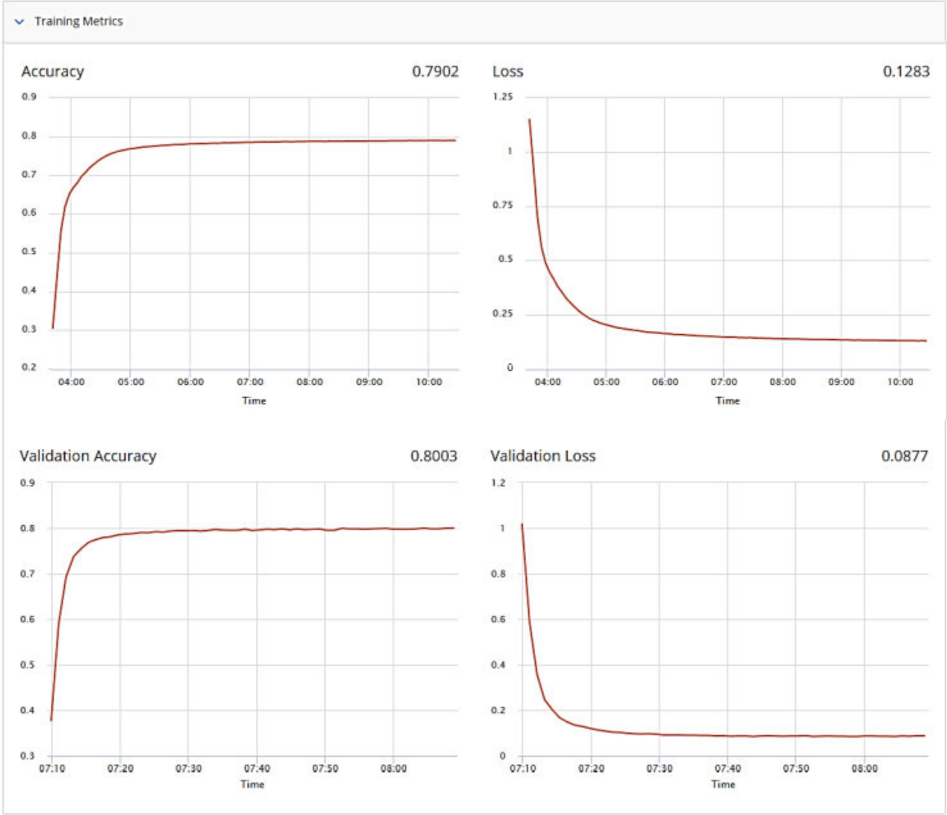


Figure 3.12: Training Matrices of Model.

After training the model for 100 epochs, the training accuracy for predicting the miRNA sequence based on its binding site in the mRNA was approximately 79%, as measured on the training set. In the validation set, an increase in accuracy of approximately 1% was observed, resulting in an overall accuracy of around 80%. Since the training accuracy is lower than the validation accuracy, therefore it can be concluded that the model is not overfitted. Additionally, the validation loss is lower than the training loss, with a training loss of 0.128 and a validation loss of 0.087, further indicating that the model is not overfitted. The training matrices show that the miRNA sequence can be predicted with up to 80% similarity using its target binding segment in the mRNA (Figure 3.12).

2. Validation

To test our package with experimentally validated list of miRNAs associated with Gene symbol. 200 Genes were randomly selected from our in house developed database miDerma which contains miRNA and Gene pair associated with dermatological disorders. miRNAs associated with individual genes were retrieved[232]. Also those genes were feed to our package and miRNAs were predicted.

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

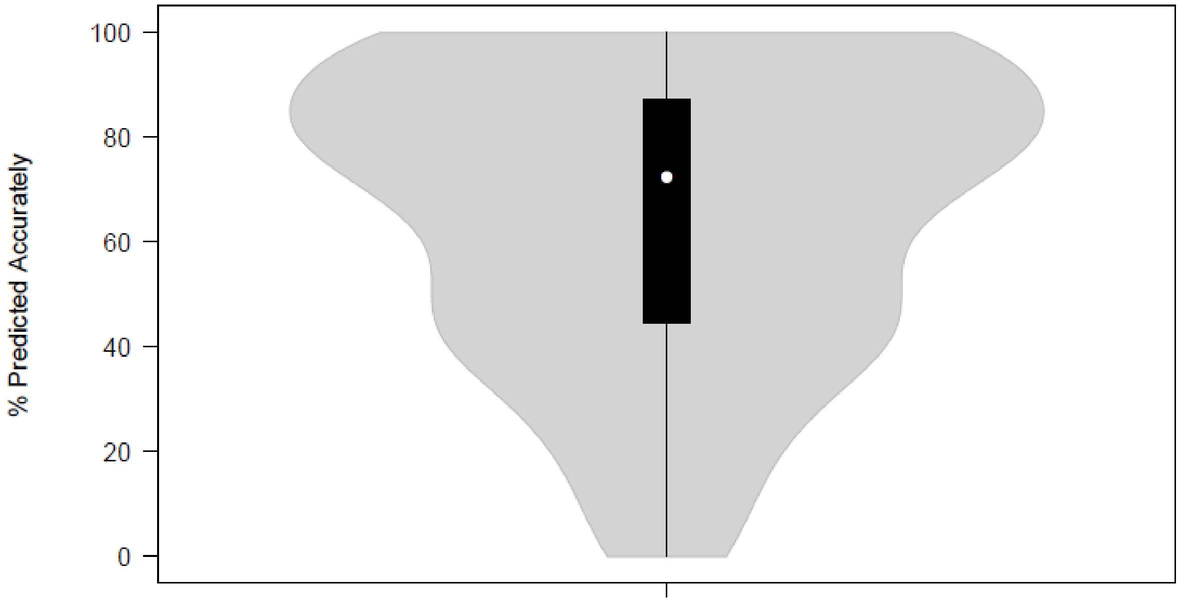


Figure 3.13: Violin plot showing distribution of accurately predicted microRNAs from 200 Gene Symbols associated with Dermatological disorders among known microRNAs.

Figure 3.13 above is a violin distribution for percentage of miRNAs predicted accurately among known experimentally validated miRNAs 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 miRNA sequence for targeted genes.

Table 3.4: Some of the well predictions done through our model .

Gene symbol	No. of Experimentally validated miRNAs associated with gene	No. of. miRNAs accurately predicted among the validated miRNAs	Percentage of accurately predicted miRNAs among the validated miRNAs (%)
ABCC1	26	26	**100**
ADAMTS1	13	13	**100**
ELK3	14	14	**100**
EPB41L3	20	19	**95**
BMPR2	51	48	**94.12**
ITSN2	17	16	**94.12**
FGF10	26	24	**92.31**
NT5C3A	12	11	**91.67**
EIF2S2	24	22	**91.67**
CCNE1	34	31	**91.18**
ABCG2	30	27	**90**
DNMT1	39	35	**89.74**
HOXD11	37	33	**89.19**
DSC3	18	16	**88.89**
GPI	27	24	**88.89**
JAG1	27	24	**88.89**
CUL5	17	15	**88.23**
ESRRA	8	7	**87.5**
FN1	8	7	**87.5**
CDK6	188	164	**87.23**
CD28	46	40	**86.96**
ARRDC3	23	20	**86.96**
CADM1	46	40	**86.96**
BAX	15	13	**86.67**
CYP24A1	15	13	**86.67**
FHL2	87	75	**86.21**
IGFBP5	144	124	**86.11**
AKAP12	21	18	**85.71**
DNMT3A	34	29	**85.29**
HOXB13	20	17	**85**

For comparing the prediction accuracy of our model with other well-known miRNA target predictive tools such as TargetScan, DINA-MicroT, miRanda and RNA22, all the sated models and our proposed model were evaluated using a independent microarray study. The publicly available data is result of a study where 25 miRNAs were inhibited concurrently using antisense miRNAs and the change of RNA expression are recoded [158]. Here predicted targets by various tools were studied with respect to the gene expression changes. As shown in Figure 3.14 the mRNAs predicted by our proposed model were most upregulated as compared with the mRNA predicted by other algorithms.

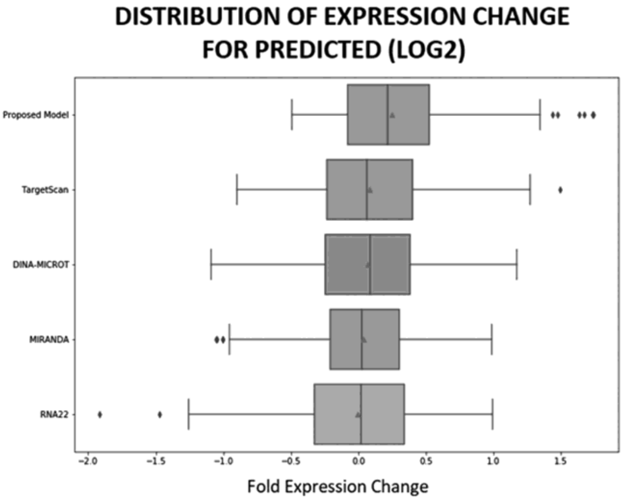


Figure 3.14: Distribution of the expression changes for top-ranking targets predicted by individual algorithms

v. 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 miRNA 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.

It has been observed that neural networks, which are a highly effective category of machine learning, can be applied to a wide range of problems including classification, clustering, regression, natural language processing, and sequence prediction. The way neural networks learn is by adjusting the input weights of each neuron. Convolutional neural networks (CNNs), a type of artificial neural network (ANN), are used for feature extraction or selection and are frequently utilized in image recognition for identifying distinctive features in images. They can also be employed in extracting features or recognizing specific patterns from sequences that are difficult for humans to discern. A 1D ConvNet can be used for selecting features from a 1D data, such as a text sequence. Recurrent neural networks (RNNs), another type of ANN, are proficient at learning from sequence data and utilize their internal state (memory) to process sequences of inputs, allowing them to retain information from previous input data, which is useful when working with sequence data. RNNs are used in sequence classification and sequence prediction, but they may suffer from a problem known as the vanishing gradient, where they tend to forget earlier instances. To address this issue, researchers have developed an enhanced version of RNNs called long short-term memory (LSTMs), which addresses the vanishing gradient issue by adding an additional memory unit that keeps track of all relevant states and stores them.

Looking for an improved algorithm, in this work sequence pair data of miRNAs and corresponding bound target mRNA from TarBase v8 was obtained to trained a ANN network for prediction of miRNA from their bounded target segment in mRNA. Particularly CNNs were used for recognizing patterns in mRNA segments and extraction of features. Two LSTMs in seq2seq architecture were placed for predicting sequences of miRNA. Also two layers of dense network were stacked between CNN and LSTM1, another between input_2 and LSTM1 (Figure 3.6). This model trained on 19000 experimentally validated and cleaned pair of mRNA and miRNA sequences archiving accuracy of 80%.

It is important for a miRNA to have access to a sufficient surface area in order to bind to a targeted mRNA segment. In particular, at least four nucleotides should be exposed and unbounded in the 3D structure of the mRNA in order for the miRNA to bind. To identify regions in the mRNA that are suitable for binding, we used RNAplfold from the RNA Vienna package to determine the probability of four nucleotide stretches being unpaired in the mRNA's 3D structure. We then selected segments from these regions as potential targets for miRNA binding and fed them into our trained model for predicting possible miRNAs that could bind to these target segments.

The user will be prompted to enter a gene symbol when running this package. Using this gene symbol, the package will retrieve the protein coding transcript's sequence from the Ensemble REST API. These mRNA sequences are then processed to predict a list of miRNAs. The model was validated using experimentally verified miRNA and RNA pairs involved in skin diseases that were retrieved from an in-house database called miDerma. On average, the model was able to correctly predict 72% of the miRNAs from mRNA in each case. The package, named "model: A MircoRNA sequence prediction tool from RNA sequence based on CNNs, LSTMs, and seq2seq architecture," utilizes a seq2seq architecture neural network model similar to those used in chatbot development. In this way, the package aims to apply natural language processing models to the fundamental language of nature, i.e. A, T, G, C.

The primary goal of this work is to find miRNA target locations that can be restricted by the miRNA. However, further enhancements can be made by introducing new features. Knowing a miRNA's target is one method for determining the miRNA's participation in normal or abnormal biological processes. For each one miRNA, there might be a plethora of targets. Several tools have been created in the last 17 years to address this difficult topic. Each of these experiments has helped us understand the link between miRNA and mRNA targets, as well as how that relationship can be utilised to create accurate predictions. MiRNA prediction can aid the scientific community in the fields of medicines, biomarker identification, and so on.

CHAPTER IV

Predicting protein intrinsically disordered regions by applying natural language processing practices

Chapter 4. Predicting protein intrinsically disordered regions by applying natural language processing practices

i. Introduction

Intrinsically Disordered Regions (IDRs) are the regions in proteins that do not posses well organized two dimensional or three dimensional structures under physiological conditions. These regions exit extravagantly in each domain [233] - [234] and concerned with numerous protein functions [235] - [236]. They are involved in chemical reactions, they recognize nucleic acids, proteins, influence molecular interactions between bound partners. These properties of disordered regions have been well explored by the researchers to delineate the potential of disordered regions in molecular interactions [237]. It has been observed that these disordered regions are accessory for biological activities carried out by most of the structured proteins [238]. Studies have been conducted revealing the importance of mobile flexibility and structural instability in natural proteins, that they are more intrinsically disordered than the protein with the random sequence [239].

Multiple reports exist pointing towards the implications of disordered regions to various diseases including neurodegenerative diseases and cancer [238]. Recent studies demonstrate the significant role of disorder prediction in identification of disease as well as in epidemiological examinations due to strong connection between disorder regions and various human disease [240]. These disorder regions serve as potential targets and undergo disordered-to-order transitions in the binding regions and ultimately prompt considerable research in drug discovery process [239]. Moreover, health care has likewise been connected to disorder prediction in identifying risk and studying the progression of diseases in patients [241]. Recognizing their widespread presence in proteins prompt the development of quick and accurate computational approaches for their prediction.

Number of experimental techniques are available to determine the Intrinsically Disordered Proteins (IDPs) or Intrinsically Disordered Regions such as missing regions in X-ray crystallography or dynamics in Nuclear Magnetic Resonance experiments. Because of high cost of identifying disordered regions experimentally, it is essential to compute probable regions/proteins before conducting experimental studies [242].

It has been estimated that around 60 or more computational techniques have been developed so far [243] - [244] , many of these techniques utilize protein sequences [245] and information derived from them such as statistical potentials (FoldIndex) [246], physio-chemical properties (IUpred) [247], propensities (Globplot) [248] for analysis on protein. Based on the studies conducted it has been shown that these methods outperformed by sequence machine learning approaches [249] - [250] (CSpritz [251], DisEMBL [252], PONDR series [253]). However, these single-sequence methods are considered to be less accurate than sequence profile based machine-learning techniques obtained from multiple sequence alignment [254]. This is on the grounds that sequence profiles, for the most part made by programs, for example, PSI-Blast [254] and HHBlits [255], contain significant data relating to the absence or presence of preserved residues due to their functional and structural roles. Instances of ongoing techniques dependent on profiles are SPINE- D [256], SPOT-Disorder [257] and AUCpred [258].

However, due to cheaper sequencing techniques, protein sequences in libraries have been increased exponentially and obtaining evolutionary profiles for these sequences are computationally intensive. As a result, genome wide scale analysis using profile-based techniques are difficult and time consuming. Furthermore, in real-world applications, the large number of amino acid chains, greater than 90 percent, do not correspond to a large sequence cluster [259]. In other words, due to lack of evolutionary information the quality of sequence profiles for large number of proteins is poor. As this is the case, sequence dependent methods may be more reliable and accurate than profile dependent methods as revealed from the determination of secondary structure and solvent accessible surface area [260] computationally using single sequence based method. Thus, it is highly advisable to have a highly precise sequence based method as the intrinsic disorder regions can also be displayed by protein sequences alone [261]. Hence, improving already existing sequence dependent techniques also addresses the fundamental question of how far we can push the accuracy limit considering only sequence information irrespective of evolutionary profiles.

Improvements in already existing single sequence based techniques are possible as most of these rely on algorithms such as [262] simple neural network, support vector machine, recurrent Neural Network. On the other hand, advanced learning algorithms have been utilized in profile based predictors in order to improve disorder prediction. Such as deep long short-term memory (LSTM) bidirectional RNN, deep convolutional neural fields and combined LSTM and convolutional networks [263] - [264].

Present study was encouraged by recent progress in employing ensemble of Long Short Term Memory (LSTM) and Convolutional Neural Networks (CNN) [262]. Such an ensemble not only enhance robustness of performance but also removes noise that prediction more reliable. As LSTM Networks have already been known to provide high accuracies in disorder prediction, their amalgamation with Convolutional Neural Network can increase the effectiveness of disorder prediction. Hence we showed a model with enhanced accuracy utilizing an ensemble of embedding and convolutional and bi-directional LSTM layer for making predictions on disordered regions in proteins in contrast to already existing state of art methods.

ii. Review of literature

1. Overview of disordered proteins

Disordered regions of proteins, which do not have a well-organized three-dimensional or two-dimensional structure under physiological conditions, are common in eukaryotic genomes and are highly conserved in terms of their sequence and composition between species. These disordered regions, which can be larger than 50 amino acids, play important roles in a wide range of protein functions, such as cell signal transduction, transcriptional regulation, and translation[265]–[268]. Disordered regions can be detected using X-ray crystallography experiments or spectroscopic techniques such as NMR, which can provide more detailed information about the dynamics and structural tendencies of these regions in solution[269].

In addition to their role in various protein functions, disordered regions have also been shown to play a role in protein-protein interactions and in the regulation of protein activity. These regions can act as "adaptors" that help bring proteins together to form complexes, and they can also modulate the activity of other proteins by blocking or activating enzymatic activity[270]. Disordered regions can also act as "molecular glue," helping to stabilize protein-protein interactions and maintaining the integrity of protein complexes[271]. Despite the important roles that disordered regions play in protein function, they are often overlooked in traditional studies of protein structure and function because they do not have a well-defined three-dimensional structure[265], [272]. However, recent advances in techniques for studying disordered proteins have helped to shed light on their functional importance.

2. Are the domains unfolded or folded?

It is challenging to predict the three-dimensional structure of globular proteins based solely on their amino acid sequence, with the exception of cases where the structures of highly homologous sequences are already known. However, it is relatively straightforward to identify sequences that are likely to be intrinsically disordered, meaning that they are unable to spontaneously fold into well-defined structures[271], [273]. These disordered proteins are characterized by the absence of a stable three-dimensional structure under physiological conditions and can play important roles in various protein functions[274]. Despite the difficulties in predicting their structure, the functional importance of disordered proteins has led to increased efforts to understand their behavior and to develop methods for studying and manipulating them[275].

a. Sequence Characteristics of disordered regions

A probable disordered region is characterized by an amino acid compositional bias, low complexity in the sequence, low content of bulky hydrophobic residues such as Phe, Trp, Ile, Val, and Tyr, and high content of charged and polar residues such as Ser, Glu, Lys, Pro, Gln, Ser, and occasionally Ala and Gly[275]–[277]. There are several computer programs available that can be used to identify disordered proteins or regions, including FoldIndex[278], DisEMBL[279], PONDR[280], and GLOBPLOT[281]. Genome-wide studies have shown that disordered regions are very common, and the extent of disordered regions in proteins tends to increase with the complexity of an organism. Disordered proteins are often associated with diseases and are involved in eukaryotic cell signaling processes. There are many proteins that have been shown to be either fully or partially disordered, as documented in the DisProt database of experimentally annotated protein disorder.

3. Experimental techniques for characterization

The most commonly used experimental technique for identifying disordered regions of proteins is Nuclear Magnetic Resonance (NMR) spectroscopy. However, other techniques such as Fluorescence Spectroscopy, Hydrodynamic Measurements, Raman Spectroscopy, Vibrational

CD Spectroscopy, and Circular Dichorism can also provide useful information about disordered regions[282], [283]. These techniques can be used to complement information obtained through NMR or to provide additional insights into the properties and behavior of disordered proteins[284].

4. General attributes of disordered regions

Proteins can be classified along a structural continuum based on the degree of structure they possess. At one end of the continuum are proteins with a firmly collapsed single domain, while at the other end are proteins with highly extended, heterogeneous disordered regions. In between these extremes, there are proteins with multi-domain structures that may contain flexible or unstructured regions, as well as compact but unstructured molten globules[283], [285]. This structural continuum has been divided into three or four categories, although there is a wide range of different structure types within each category. Generally, proteins that are characterized as intrinsically disordered lack a sufficient hydrophobic core to spontaneously fold into a well-organized three-dimensional structure. As a result, these proteins often do not have a stable three-dimensional structure under physiological conditions and may exist as disordered or partially ordered states[282], [283].

5. Functions of disordered regions

Intrinsically disordered proteins (IDPs) are a class of proteins that do not have a well-defined, stable three-dimensional structure under physiological conditions. IDPs are characterized by a high degree of conformational flexibility and disorder, and they often contain regions that are rich in flexible, unstructured amino acid residues such as proline, glycine, and glutamine. Despite their lack of a stable structure, IDPs play important roles in many cellular processes, including signal transduction, transcription regulation, protein phosphorylation, and the assembly of multi-protein complexes[286]–[289].

One of the key features of IDPs is their ability to undergo a "disordered-to-ordered" transition upon binding to a target molecule[290]. This process involves the IDP folding into a more stable, ordered structure upon recognition and interaction with its target. An example of this process is the activation domain in the protein cAMP response element-binding protein (CREB), which contains an intrinsically disordered Kinase-Inducible transcriptional activation Domain (KID)[291]. The KID is disordered as a detached peptide as well as in the full-length protein, but upon binding to its target molecule, it folds into orthogonal helices and becomes more stable. The intrinsic disordered nature of the KID can be predicted from its sequence, and this disordered-to-ordered transition is thought to play a crucial role in the protein's function[292].

IDPs are increasingly being recognized as important players in many cellular processes, and their involvement in various functions is continuously emerging. In addition to the examples mentioned earlier (e.g., signal transduction, transcription regulation, protein phosphorylation, and the assembly of multi-protein complexes)[286], [287], [293], IDPs have also been implicated in other functions such as the storage of small molecules and the regulation of the translational process. IDPs can also behave like chaperones, binding to misfolded RNA and protein molecules and helping to unfold and relax kinetically unfavorable intermediates.

One important aspect of IDPs is their ability to interact with and bind to other proteins and molecules. These interactions often involve the intrinsically disordered regions of the IDP, which can bind to and stabilize specific conformations of the target molecule. This ability to recognize and bind to specific target molecules is thought to be critical for the function of IDPs in many cellular processes[294].

In summary, IDPs are a class of proteins that are characterized by their intrinsic disorder and conformational flexibility. Despite their lack of a stable structure, IDPs play important roles in many cellular processes, and their involvement in various functions is continuously emerging. IDPs have the ability to undergo a disordered-to-ordered transition upon binding to a target molecule, and their ability to interact with and bind to other proteins and molecules is thought to be crucial for their function.

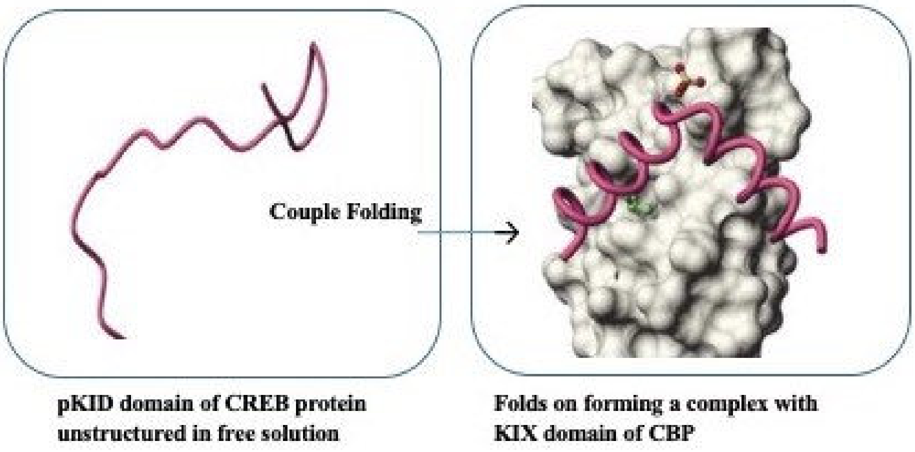

Couple Folding
pKID domain of CREB protein unstructured in free solution
Folds on forming a complex with KIX domain of CBP

Figure 4.1: Couple folding and binding.

6. Roles of disordered regions

a. Recognition Elements- Nucleic acid and protein recognition

DNA-binding proteins have developed specialized methods for identifying and interacting with specific DNA sequences, many of which involve partial folding or unfolding[294], [295]. Induced fit folding has been suggested as a key mechanism in the sequence-specific binding of proteins to DNA, based on the large heat changes that occur when these complexes are formed. RNA-binding proteins also have disordered regions, and they tend to remain disorganized when they form complexes, similar to their structure in the free state[296]. The formation of a complex between 5S ribosomal RNA and L5 ribosomal protein is thought to involve an induced fit mechanism[297].

b. Regulation through degradation

The instability of IDPs/IDRs involved in translation, transcription, and cell signaling may contribute to the regulation of cell behavior through proteolytic cleavage and degradation. For example, the ubiquitin-proteasome complex system may play a key role in activating transcription by targeting the degradation of transcriptional activation domains. The stability of β-catenin and the regulation of cadherin are also controlled through targeted degradation. The cadherin cytoplasmic terminal domain, which is disordered and contains exposed Pest-Sequence motifs, is targeted for degradation by the ubiquitin-proteasome system, but this process is prevented when cadherin binds to β-catenin[298]. Linker sequences, which should be flexible and have a moderate level of stability, may be important for proper function. These sequences may need to be resistant to proteolysis, but also be unfolded, and a low content of hydrophobic residues may be critical for this function. Misfolded proteins are targeted for degradation, and it has been suggested that this process involves the recognition of hydrophobic solvated residues by the ubiquitin-proteasome complex system[299]. Poly-glutamine residue repeats may be resistant to degradation by the eukaryotic ubiquitin-proteasome complex system. The specific amino acid residues in linker sequences may also allow them to fold and form structures that can interact non-specifically with other proteins.

These proteins and sequences are often unstable and prone to degradation, which can be regulated through the ubiquitin-proteasome complex system and other mechanisms[300]. However, the specific characteristics of these proteins and sequences, such as their flexibility, stability, and amino acid composition, can affect their function and how they interact with other proteins. Understanding these features and how they are regulated may provide insights into how cells control various processes and respond to different signals[301].

c. The natural ‘cost’ of disordered regions

Disordered regions of proteins are important for certain functions, such as in the case of transcriptional activators, cell signaling molecules, and regulatory proteins. However, this can also come with negative consequences, as disordered regions are often involved in chromosomal translocations that can lead to diseases, such as leukemia. For example, translocations in the N-terminal unstructured regions of CBP/p300 or in the linker between the KIX and bromo-domains have been linked to leukemia[302], [303]. These translocations cause the segments of CBP/p300 to become attached to the MLL or MOZ regions, resulting in the proteins acquiring abnormal functions[304]. In contrast, translocations or truncations in genes that encode structured domains are more likely to produce misfolded proteins that are quickly destroyed by cellular machinery and do not cause a diseased phenotype.

There is still much work to be done in understanding and characterizing functional disordered proteins. Computational techniques that can analyze protein chains and even entire genomes for IDPs (intrinsically disordered proteins) will likely reveal more proteins that belong to this class. Advances in functional genomics will also help us understand the functional properties of IDRs (intrinsically disordered regions). The role of lower complexity in protein sequences is just starting to be explored, and our understanding of the function of proteins is limited to a static view rather than a dynamic one, where different conformations can correspond to different functions.

7. Computational methods for IDRs prediction

Over the past two decades, numerous methods for identifying IDPs (intrinsically disordered proteins) and IDRs (intrinsically disordered regions) have been developed. These methods can be broadly classified into four categories: physicochemical techniques, machine learning techniques, template or homology techniques, and meta techniques. Physicochemical techniques rely on the chemical and physical properties of protein sequences, while machine learning techniques use classification algorithms based on learning algorithms. Template or homology techniques rely on examining known structures of proteins, and meta techniques incorporate the results from a variety of predictors. It's worth noting that these categories are not mutually exclusive, and predictors in one category may also use techniques from other categories. For example, some physicochemical techniques may also use features from machine learning predictors or meta predictors.

a. Physicochemical-based techniques

These techniques for predicting IDPs (intrinsically disordered proteins) and IDRs (intrinsically disordered regions) are based on chemical and physical properties that can directly affect the binding and folding of proteins[305], [306]. These properties include the affinity of specific residues, overall charge, hydrophobicity, and contact angle, among others. For example, Uversky used a combination of low hydrophobicity and high overall charge to identify IDPs. Based on this principle, the FoldIndex was developed to identify IDRs using a pre-defined sliding window. Another method, GlobPlot, uses a parameter called P to predict IDRs. The basic theory behind GlobPlot is that the probability of an amino acid being disordered can be described as $P = RC - SS$, where RC represents the propensity of a specific residue to exist in a "random coil" and SS represents the propensity to exist in "secondary structure". The basic calculation behind GlobPlot is a summation function of P, which is simple and fast. Physicochemical-based strategies are efficient and have low computational cost, and their predicted results are easy to interpret. These physical and chemical properties are also used as features in machine learning techniques and in meta techniques[307], [308].

b. Template/Homology based techniques

These techniques for predicting IDRs (intrinsically disordered regions) rely on proteins with known structures. These techniques first try to find known structures of homologous protein sequences (templates), and then analyze the query protein to predict its IDRs. For example, GSmetaDisorder3D[309] and PrDOS[310] use different structural principles as features rather than independent predictors. The advantage of these techniques is that the predicted results are easy to interpret. However, even when similar homologous sequences to the target protein are available, they may not be reliable, and in some cases, homologous proteins may not be able to be identified at all. It is worth noting that template-based techniques are generally more accurate than other methods, but they are limited in their ability to predict the behavior of proteins that do not have a known structure. In these cases, other methods, such as physicochemical or machine learning techniques, may be more effective.

c. Meta techniques

Meta-predictors are methods that combine the predictions of multiple computational predictors into a single model in order to enhance prediction accuracy. There are two main categories of meta-predictors: direct combination and machine learning combination. Direct combination methods, such as PrDOS[310], CSpritz[311], and MobiDB-light[312], combine the results of different predictors using a weighted voting technique. Machine learning combination methods, on the other hand, use the prediction results of different predictors as features to train a final model using a machine learning algorithm. Examples of machine learning combination methods include meta-PrDOS[310], DISOPRED3 [313], and MD. MetaPrDOS is a two-stage meta-predictor that uses the results of seven predictors to determine the probability of each residue being disordered using a support vector machine model. MD is a neural network-based method that uses the results of four predictors and protein sequence properties as input. DISOPRED3 is based on DISOPRED2[313], a predictor of long disordered regions and closest neighbor, and consists of a main layer with three predictors and a neural network layer that combines the results of the main layer. Meta-predictors are known to achieve state-of-the-art performance, but their high computational cost has limited their application to a limited number of proteins.

There are several benefits to using meta-predictors for protein disorder prediction. One benefit is that they can take advantage of the strengths of multiple predictors, potentially leading to improved prediction accuracy. Another benefit is that meta-predictors can be more robust to changes in the input data, as they can use information from multiple sources to make predictions. However, there are also some limitations to the use of meta-predictors. One limitation is that they can be computationally expensive, which may limit their use to smaller proteins or proteins with specific characteristics. In addition, meta-predictors may be more difficult to interpret and understand compared to single predictor models, as they involve the combination of multiple prediction approaches. Overall, meta-predictors can be a useful tool for protein disorder prediction, but their use should be carefully considered based on the specific needs and constraints of the application[314].

d. Machine-learning techniques

Machine learning algorithms have been developed to address the limitations of other techniques, such as physicochemical and template methods, for protein disorder prediction. These algorithms use negative and positive sets to identify disorder regions and incorporate various features. In the process of feature extraction, which is a crucial component of any machine learning algorithm, the features used to predict protein disorder can be divided into three categories: sequence properties, evolutionary relationships, and structural data. Sequence properties include amino acid residue propensity, composition, flexibility, hydrophobicity, and low complexity. Evolutionary relationships refer to profiles obtained from multiple sequence alignment. Structural data includes secondary structure, solvent accessibility, and torsion angles. Machine learning techniques for protein disorder prediction can be further divided into classification models and sequence labeling models.

Classification models

Traditional models were designed to handle feature vectors of fixed length. Classification models are trained in a supervised manner using both negative and positive datasets and then predict the label of unseen data samples based on the trained model. Predicting whether an amino acid is disordered or ordered is a binary classification problem. One challenge in using these techniques for protein disorder prediction is converting proteins, which can vary in length, into fixed-length vectors. The sliding window method is a common approach for addressing this challenge, as it incorporates information about an amino acid residue and its adjacent residues into a fixed-length vector. There are several algorithms that can be used to build classification models for protein disorder prediction, including Random Forest, Neural Network (NN), and Support Vector Machine (SVM). PONDR[315] is an example of a primary predictor based on Neural Networks for different types of intrinsic disorder regions, including short disordered regions (SDRs), medium disordered regions (MDRs), long disordered regions (LDRs), and intrinsic disorder regions of all lengths.

Sequence labeling models

Sequence labeling models for protein disorder prediction are based on supervised learning using both negative and positive datasets. In these models, the input is a protein sequence and the output is a labeled sequence indicating the presence or absence of disordered residues. Several algorithms have been developed for sequence labeling models, including Conditional Random Field (CRF), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM). One example of a sequence labeling model is SPOT-disorder, which is a deep-bidirectional model comprising a feed-forward RNN layer followed by two LSTM layers. This model has been shown to perform well for predicting both long disordered regions (LDRs) and short disordered regions (SDRs).[311] The performance of sequence labeling methods can be improved by combining them with deep learning methods, as LSTM and RNN can automatically identify relevant information about residues and capture both local and global context in proteins[316].

Table 4.1: Summary of some of IDPs/IDRs predictors .

PREDICTOR	CATEGORY	CLASSIFIER	FEATURES	YEAR	AUC
GlobPlot[281]	P	_	Amino acid propensity difference	2003	NA
IUPred[279], [317], [318]	P	_	Amino acid composition	2005	0.66
FoldIn[319], [320]	P	_	Net charge and hydrophobicity	2005	NA
DisEMBL [279]	C	NN	Protein Sequence	2003	NA
PONDR VL3[315]	C	NN	Residue frequency, flexibility and sequence composition	2001	0.69
Spritz [321]	C	SVM	PSSM and secondary structure predictions	2006	NA
SPINE-D [322]	C	NN	Residue and window level information from different parameters	2012	0.82
SLIDER[323]	C	LR	Physicochemical properties, complexity of sequence and amino acid composition	2014	NA
DisPredict [314]	C	SVM	Amino acids, physical-chemical properties, ASS	2015	NA
DISpro [324]	L	RNN	PSSM, solvent accessibility, secondary structure	2005	NA
Espritz[325]	L	BRNN	Sequence or add PSSM	2012	0.855
AUCpreD [316]	L	CNN CRF	Residue-related features include identity, physical-chemical propensities	2016	0.88
SPOT-Disorder[326]	L	LSTM RNN	SSM, Shannon entropy, physical-chemical properties, structural properties	2016	0.903
PrDOS[327]	M	_	Combination of a SVM predictor and a homolgy predictor	2007	0.907
MD[327]	M	_	Combination of DISOPRED2, PROFbval, IUPred and a few sequence properties	2009	0.849

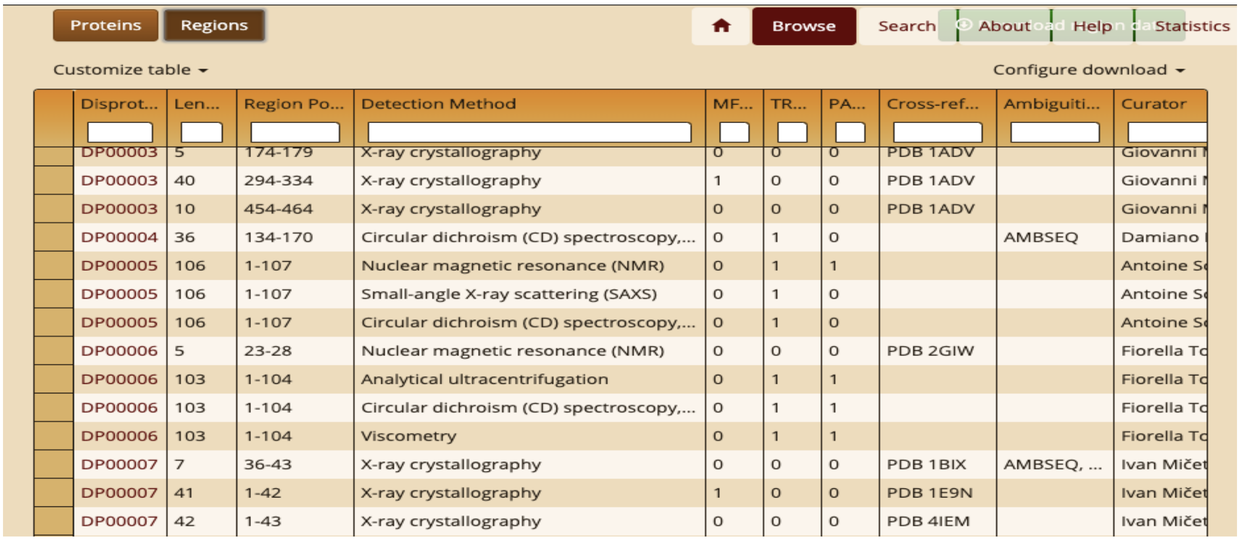
PONDR-FIT [315]	M	_	Combination of PONDR, PONDR VL3, IUPred, TopIDP and FoldIndex	2010	0.818
MetaDisorder[309], [328]	M	_	FloatCons: combination of 13 predictors	2012	0.753
DISOPRED3[3 28]	M	_	Combination of DISOPRED2, LDRs and a nearest neighbor predictor	2015	NA
MobiDB-lite[329]	M	_	Combination of two variants of IUpred, and three variants of ESpritz, DisEMBL and GlobPlot	2017	NA

iii. Methodology

1. Data curation

In this study we intended to use an ensemble of long short term memory and convolutional networks to predict DRs. For this experimentally validated disordered protein sequences were required. These sequences were collected and from manually curated and annotated databases such as DisProt 7 [330] (v0.5 release 11-05-2017) and MobiDB [331] (release 24 October 2017).

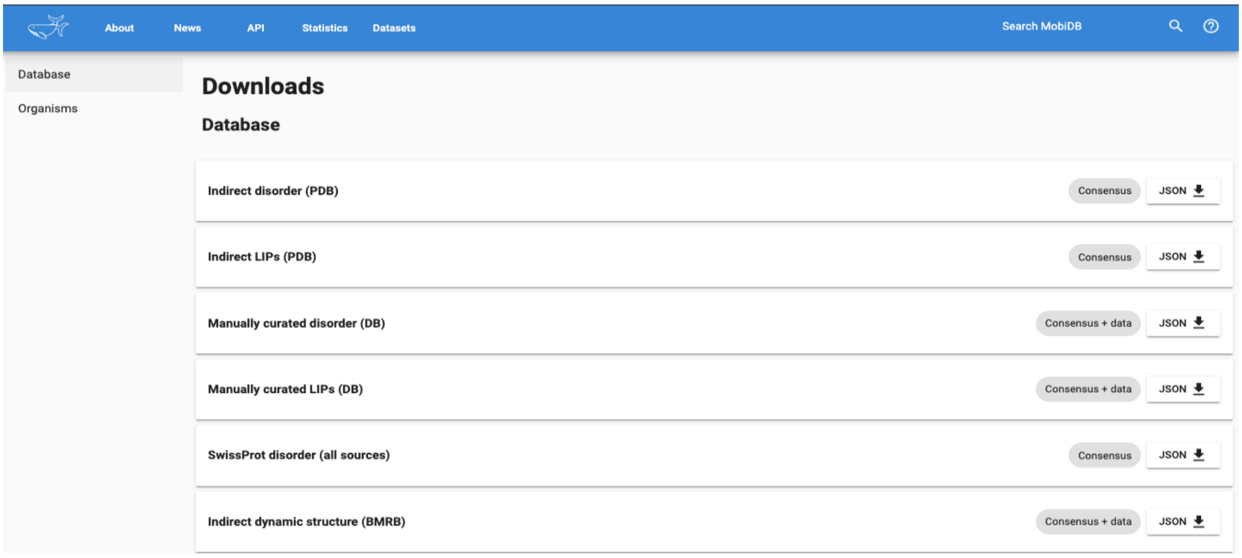
DisProt is a database comprising of experimentally validated and annotated information of disordered regions manually gathered from literature. Statistics includes 803 proteins and 2167 regions. Each evidence is recognized by at least one experiment. DisProt disorder region (DR) is unambiguously distinguished by literature, the first and the last residue of the DR, and the experimental method utilized in the paper. DisProt can be annotated with functions and another ontology has been made to portray disorder-specific viewpoints. This has following structure, Molecular function of disorder region, the kind of basic structural transition of disorder region, the type of associating or interacting partners.



Disprot...	Len...	Region Po...	Detection Method	MF...	TR...	PA...	Cross-ref...	Ambiguiti...	Curator
DP00003	5	174-179	X-ray crystallography	0	0	0	PDB 1ADV		Giovanni
DP00003	40	294-334	X-ray crystallography	1	0	0	PDB 1ADV		Giovanni
DP00003	10	454-464	X-ray crystallography	0	0	0	PDB 1ADV		Giovanni
DP00004	36	134-170	Circular dichroism (CD) spectroscopy,...	0	1	0		AMBSEQ	Damiano
DP00005	106	1-107	Nuclear magnetic resonance (NMR)	0	1	1			Antoine S
DP00005	106	1-107	Small-angle X-ray scattering (SAXS)	0	1	0			Antoine S
DP00005	106	1-107	Circular dichroism (CD) spectroscopy,...	0	1	0			Antoine S
DP00006	5	23-28	Nuclear magnetic resonance (NMR)	0	0	0	PDB 2GIW		Fiorella T
DP00006	103	1-104	Analytical ultracentrifugation	0	1	1			Fiorella T
DP00006	103	1-104	Circular dichroism (CD) spectroscopy,...	0	1	1			Fiorella T
DP00006	103	1-104	Viscometry	0	1	1			Fiorella T
DP00007	7	36-43	X-ray crystallography	0	0	0	PDB 1BIX	AMBSEQ, ...	Ivan Mičet
DP00007	41	1-42	X-ray crystallography	1	0	0	PDB 1E9N		Ivan Mičet
DP00007	42	1-43	X-ray crystallography	0	0	0	PDB 4IEM		Ivan Mičet

Figure 4.2: DisProt- Database of Protein Disorder.

MobiDB[312] was intended to brought together asset for annotations of protein disorders and its functions. The database covers diverse issue perspectives. MobiDB highlights three quality levels of annotation from high to low quality (pyramid). Various sources present an unmistakable tradeoff among quality and coverage, for example, manually curated (annotations from external databases), indirect (Derived/determined data from experimental information, for example PDB structures and additionally chemical shifts), predicted (Predicted annotations).



Downloads

Database

Indirect disorder (PDB)	Consensus JSON
Indirect LIPs (PDB)	Consensus JSON
Manually curated disorder (DB)	Consensus + data JSON
Manually curated LIPs (DB)	Consensus + data JSON
SwissProt disorder (all sources)	Consensus JSON
Indirect dynamic structure (BMRB)	Consensus + data JSON

Figure 4.3: Screenshot of MobiDB.

All sequences and annotation data are also available for download. Manually curated consensus sequences were retrieved from the database. It gives output in CSV/JSON file containing protein name, protein sequence, start and end region, disordered region sequence. So, the dataset of 9604 unique disordered region sequence were extracted from the databases.

2. Data analysis

12375 protein sequences with annotated IDRs and non-IDRs were extracted and used for further analysis. Here since only amino acid sequences were being used for prediction of IDRs, hence it was essential to check if sequences of amino acids in IDRs and Non-IDRs are contrasting to each other. For the same reason, all protein sequences were fragmented according to ordered and disordered regions, then the average and $\log_2$ fold change for occurrence of various k-mers of amino acid sequences, i.e. 1-mer to 6-mers, among IDRs and Non-IDRs were calculated. Also, Structural and Physico-chemical property enrichment in IDRs with respect to the non-IDRs was calculated using Composition Profiler [332]. Composition profiler compares the probability of frequent amino acids between a test dataset and background dataset for the particular structural and physicochemical property, here test dataset was fragments of IDRs, and background dataset constitute fragments of IDRs and ordered regions.

All 12375 sequences were clustered according to sequence identity with the help of CD-HIT server [333] with sequence identity cut-off value of 0.75, and the representative sequence of each cluster was selected. Further, the sequences have more than 25% identity with SL329 [334], i.e. benchmark dataset, which was used for testing the generalization of the trained model, were being removed. After this step 7065 sequences were selected for future analysis.

3. Data preparation

a. Pre-processing

Length of protein chains ranges widely. Figure 4.4 shows the distribution of protein's chains according to their length plotted from the data acquired from PDB. For training a machine learning algorithm, the consistency in data is the primary requirement. This vast inconsistency in length may make it hard for deep learning models to extract features as extreme ends of the graph plotted has very few representations of data or data points. Therefore, we took the range 60 to 600, which contained around 95% of the total proteins present in PDB, which was the length limit for our model for prediction.

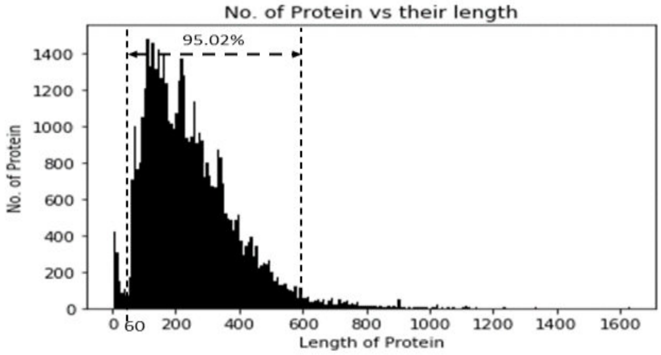


Figure 4.4: Distribution of proteins present in PDB according to their length.

After applying a length limit filter on the dataset, a total of 7008 sequences remained. These sequences were all appended with a '0' at the beginning as a starting tag. To be compatible with a deep learning model, the sequences needed to be of fixed length, meaning that every sequence must be the same length. Therefore, sequences that were less than 600 were padded with '1's up to a length of 600, and the remaining amino acids were numerically encoded from 2 to 23. The data set had a padding to amino acid ratio of 0.66. Because the dataset had a higher proportion of padding to amino acids, this may have hindered the learning of the LSTM network in the model. To address this issue, the dataset was divided into two subsets: one with protein sequences of a maximum length of 300 amino acids and the other with protein sequences of a maximum length of 600 amino acids. The first set contained 4274 protein sequences and the second set contained 2734 sequences. Both sets had a padding to amino acid ratio of 0.33. These two datasets were used to train two models with the same architecture but different input and output lengths. The model using the 300 amino acid long length sequences was named "model_300," and the other was named "model_600".

4. Building a model

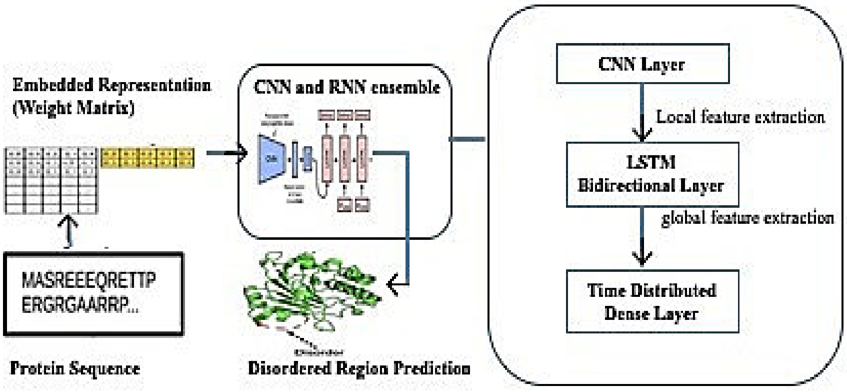

Embedded Representation (Weight Matrix)
CNN and RNN ensemble
CNN Layer
Local feature extraction
LSTM Bidirectional Layer
global feature extraction
Time Distributed Dense Layer
MASREEEQRETTP ERGRGAARRP...
Protein Sequence
Disordered Region Prediction

Figure 4.5: Proposed model for prediction of disordered regions.

Above given Figure 4.5 delineates the proposed model for prediction of disorder regions in protein sequences. Keras Library was utilized for structure and preparing our model. This library is a high-level NNs API, scripted in python equipped for running over Theano, TensorFlow or CNTK. TensorFlow is open source math programming library used in learning applications. It was developed with an attention on empowering quick experimentation. It is vital for doing great research because it provides a platform for going directly from thought a to desired outcome with the least conceivable delays. Keras offers simple and quick prototyping (through ease of use, extensibility and modularity). Supports both CNNs and RNNs, just as blends of the two. Runs consistently on CPU and GPU.

5. Developing a model

a. Weight vectors using embedding layer

As mentioned by Heffernan, utilizing an alternate matrix representation, for example, the BLOSUM62 or physical-chemical properties of every residue do not give considerable variations in performance as they can be effectively represented on one- hot vector therefore, can be learnt as linear transformations by the network. In embedding, each residue is encoded as unique integer and defined as a vector in a continuous vector space. This layer necessitates that each residue is initialized to arbitrary weights and get familiar with an embedding for each residue. The weight initialization step usually carried out by Tokenizer API available in keras library. Finally, layer has optimized weights that are learned so the final output is a two dimensional vector with an embedding for every residue of input sequences in training set. Therefore, categorical features such as amino acid sequences are encoded numerically in a matrix of 23 X 23 using embedding layer. The weight matrix representation is given in code ocean repository (https://codeocean.com/capsule/3457808/tree).

b. Feature extraction using layers

The next step was to extract features from the sequences using a CNN layer. A window size of 40 amino acids was used to extract 64 features. A bidirectional LSTM layer with 512 neurons was then used for global feature extraction along the sequence. Finally, a time distributed dense layer of 552 neurons was used to concatenate the outputs from the LSTM layer using the softmax function to produce the final output.

c. Training the model

The objective of training a model is to discover parameters, that is weights in a network which limits the error function. This error function estimates the fit between sample's true label (the real observation) and the model prediction output. The widely recognized error function in case of classification problems is categorical cross-entropy and in case of regression, it is mean squared function. It is difficult to minimize this function L(w) because of high-dimensional and non-convex nature.

d. Deciding the quantity of neurons in a network

The ideal quantities of neurons and hidden layers in the network are problem-specific and ought to be optimized. A regular approach is to enlarge the quantity of neurons and layers without over-fitting the information. Larger number of neurons and layers increase the quantity of representable functions, and experimental evidences demonstrates that it makes initialization of weights less sensitive for finding a local optimum. Here we utilized different quantities of neurons blends in all layers, and found 552 neurons for Dense layer and 512 neurons for LSTM as ideal. The weights in the model were optimized utilizing Adam optimizer, in order to minimize categorical cross-entropy function, using default parameters.

e. Partitioning dataset into Training and Validation datasets

Learning models should be build, learn and validate on autonomous data sets to abstain from over-fitting. This makes sure that the model will generalize to new data. For appropriate training apportioning data into training, testing and validation datasets, and is the typical step for any machine learning system. The training set is utilized by the models to learn various parameters, that are later assessed on validation set. The model with minimized mean-squared error function or highest prediction accuracy, is chosen and further assessed the performance of model on the test set to evaluate the correlation with different techniques. We used 80% of the data for training our model and 20% for its validation.

f. Batch Size and Learning Rate Estimate

The batch size and training rate of stochastic tendency ought to be picked up correctly, since they directly influence validation accuracy and rate of training of any model. Various learning rates have been generally explored, for instance, 0.001, 0.01 or 0.1, where 0.01 (on a logarithmic scale) is the suggested model training rate. For most applications, batch size 128 and learning rate 0.01 are the most sensible and generally used as default. However, accelerate the training process by increasing the size of a batch or it can essentially be reduced to diminish memory use in cases where the learning of complex models is carried out on GPUs with limited memory. The perfect batch size and training rate are related; smaller learning rates require greater batch sizes consistently. In our work we have utilized a default learning rate 0.01 and batch size 50.

g. Avoid Over-Fitting

Training neural network-based models can be challenging due to the risk of data overfitting. Overfitting occurs when the model is too complex for the training set and is therefore unable to generalize to new data. It can be prevented by reducing the complexity of the model, such as decreasing the number of hidden layers and neurons, or by increasing the size of the training set through data augmentation. To avoid overfitting in our model, we applied a dropout rate of 0.2 in the LSTM layer and used softmax activation and L2 regularization with a penalty of 0.001 in the dense layer.

The model was trained on Floyd Hub cloud computing using 32GB of RAM, an Intel Xeon 8-

core CPU, and an NVIDIA Tesla K80 GPU for 50 epochs. The Python code for training the model is provided in code ocean repository (https://codeocean.com/capsule/3457808/tree).

iv. Results

1. Data preparation

These sequences data were retrieved from manually curated and annotated databases such as DisProt7 and MobiDB, release October 2017. The sample of the retrieved data can be found in the table below.

To summarize, a total of 9604 protein sequences were obtained and pre-processed to ensure consistency in the length of the sequences. This allowed the model to focus on patterns in the sequences rather than the differences in length. A threshold value was set for the range of protein sequence lengths at 150-550 amino acid residues. The final dataset consisted of 7008 protein chains, which were divided into a training set of 5606 chains (80%) and a validation set of 1401 chains (20%). These sequences had a similarity of less than 25% as determined by BlastClust.

Table 4.2: Sample of data retrieved from databases.

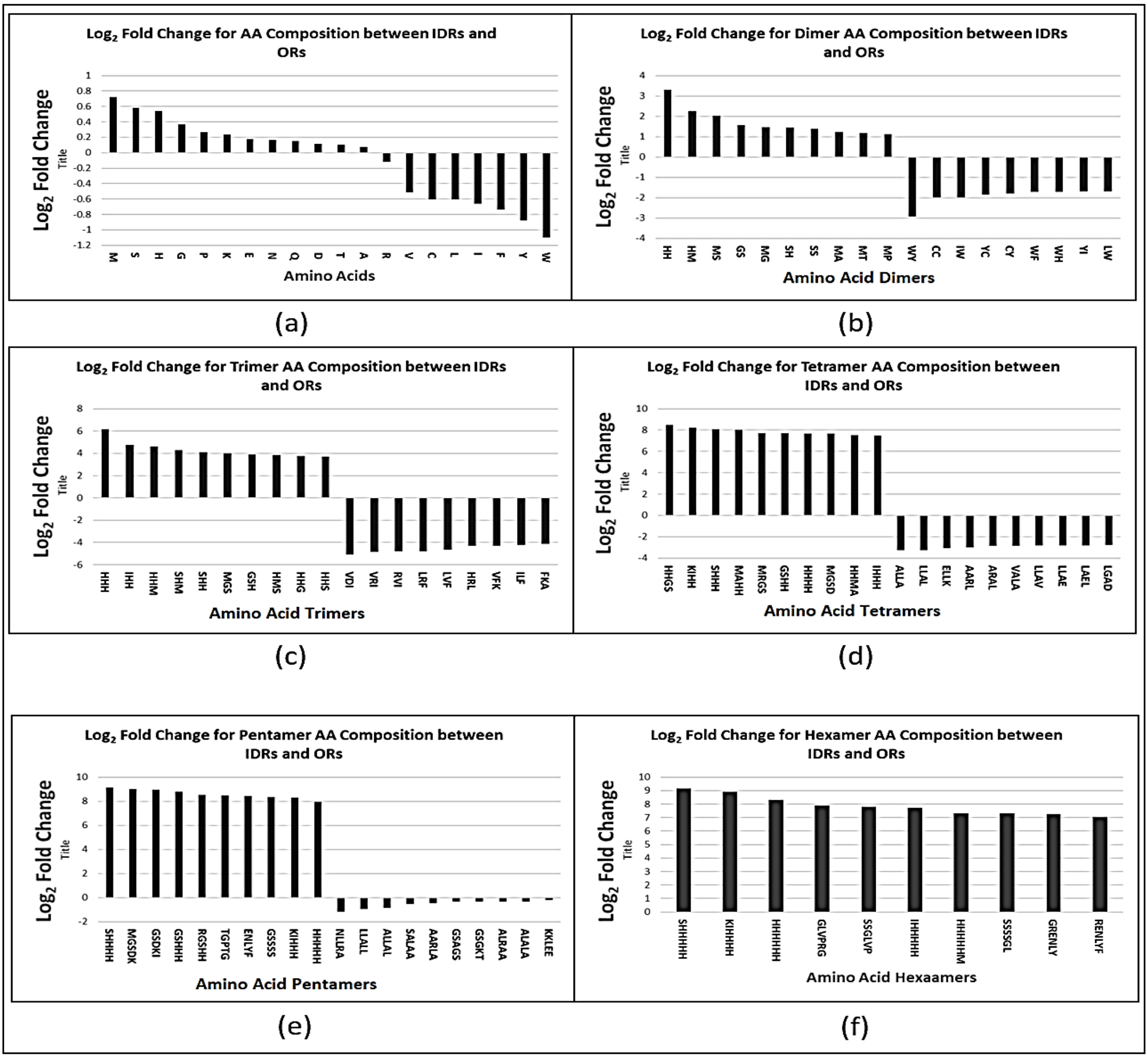
Disprot_id	End	Name	Method	Start	Sequence	Protein _type	pmid
DP00733	391		XRAY	385	LHLCSGT	Native	15713488
DP00450	536		XRAY	532	KDKCG	Native	11005854
DP00962	719	AF1 domain, Amino Terminal Domain (NTD)	NMR	710	SEVHPSRLQT	Native	19214187
DP00962	750	AF1 domain, Amino Terminal Domain (NTD)	NMR	720	TDNLLPMSPEEFDEVSRIVGSVEFDSMMNTV	Native	19214187
DP01091	81		XRAY	60	AEHQTAGRGRHGRGWAATARAQ	Native	20169168

DP01091	172		XRAY	159	VTQAPEEVDPDATS	Native	20169168
DP01099	10		XRAY	1	MASPPPFHSQ	Native	12923182
DP01099	350		XRAY	339	GQASETPHPRPS	Native	12923182
DP00981	173	C-terminal extension	PNMR	165	EKPSSAPSS	Native	1397302
DP00324	102		XRAY	92	REDSQRPGAHL	Native	11917013
DP00142	209		NMR	192	RAQIGGPEAGKSEQSGAK	Native	7649277
DP00142	209		FCD	192	RAQIGGPEAGKSEQSGAK	Native	10727931
DP00142	209		NCD	192	RAQIGGPEAGKSEQSGAK	Native	10727931
DP00023	199		XRAY	191	TAFMEKVLG	Native	9525918
DP00023	328		XRAY	289	PAKAEAGAEAGGGAGPGAEDEAGRGAVGDPELGDPPAAPQ	Native	9525918
DP00324	7		XRAY	1	MSKSESP	Native	11917013
DP00733	326	VPg	XRAY	321	LVKEVT	Native	15713488
DP00733	353	VPg	XRAY	346	CSKLPKSL	Native	15713488
DP00324	196		XRAY	182	SKQEMASASSSQRGR	Native	11917013
DP00322	31		XRAY	14	SALPDPAGAPSRRQSRQR	Native	15525646
DP00733	378		XRAY	372	LLEEVSP	Native	15713488
DP00324	102		XRAY	92	REDSQRPGAHL	Native	11917013

The amino acids in proteins chains are labeled as ‘N’ or ‘D’ considering the structured or disordered regions respectively. The python code for reading and preparing data can be found in code ocean repository (https://codeocean.com/capsule/3457808/tree).

2. Exploratory analysis

To identify the IDRs and structured regions of proteins based on their sequences, it is necessary for the sequences to be distinct from one another. The amino acid composition and occurrence averages for k-mers (i.e. dimers to hexamers) were calculated and compared based on the log2 fold change of amino acid k-mers present in IDRs and ordered regions. The composition of all amino acids and the top ten positive and negative log2 fold change for other k-mers were plotted.



(a)

(b)

(c)

(d)

(e)

(f)

Figure 4.6: The composition of all amino acid and top ten positive-negative log2 fold change for amino acids k-mers present in IDRs and ordered regions.

The plots are shown in Figure 4.6, where a contrasting difference and biasness of specific amino acids and their repeaters for IDRs such as histidine and serine were observed. In Figure 4.6(f), no common hexamers were found between IDRs and ordered regions which have a higher presence in the ordered region than IDRs. It was also observed that the sequence of amino acids like S-S-G-L-V-P-R-G, G-R-E-N-L-Y-F etc. were rich in disordered regions. All other values of k-mers can be found in the supplementary files. From these data, it was concluded that IDRs and ordered regions have amino acids and k-mers biasness and that our sequence-based model can successfully be able to learn from these contrasting features for the prediction of IDRs. The results of structural and physicochemical property enrichment in IDRs with respect to the non-IDRs from Composition Profiler[332] is shown in Table 4.3: Structural and physicochemical property enrichment in IDRs.

Table 4.3: Structural and physicochemical property enrichment in IDRs

Cumulative Amino acid Properties	**Enriched/Depleted in IDRs**	**P-value**
Aromatic content	Depleted	P-value < 0.05
Charged residues	Enriched	P-value < 0.05
Positively charged	Enriched	P-value < 0.05
Negatively charged	Enriched	P-value < 0.05
Polar (Zimmerman)	Enriched	P-value < 0.05
Hydrophobic (Eisenberg)	Depleted	P-value < 0.05
Hydrophobic (K-D)	Depleted	P-value < 0.05
Hydrophobic (F-P)	Depleted	P-value < 0.05
Exposed (Janin)	Enriched	P-value < 0.05
Flexible (Vihinen)	Enriched	P-value < 0.05
High interface propensity (J-T)	Depleted	P-value < 0.05
High solvation potency (J-T)	Enriched	P-value < 0.05
Frequent in alpha helix. (N)	Not significant	P-value > 0.05
Frequent in beta structure (N)	Depleted	P-value < 0.05
Frequent in coils (N)	Enriched	P-value < 0.05
High linker propensity (G-H)	Depleted	P-value < 0.05
Disorder promoting (Dunker)	Enriched	P-value < 0.05
Order promoting (Dunker)	Depleted	P-value < 0.05

Bulky (Zimmerman)	Depleted	P-value < 0.05
Large (Dawson)	Depleted	P-value < 0.05

3. Building a model

Our model utilizes an ensemble of embedding layer, convolutional layer and bidirectional LSTM layer. The description of each of these architectures embedding, convolutional, bidirectional LSTM and time distributed dense layer are represented in a Figure 4.7.

Where embedding layer has optimized weights for every amino acid residue that are learned and the final output is represented in a two dimensional weight vector matrix of 23 X 23 vector. The one dimensional convolution layer connected in our model uses a window size of 40 amino acid residues for extracting 64 features along the length of sequences. Bidirectional LSTM layer consists of a one cell memory state in each direction concatenating together to give an output of 2X NLSTM size. Different combinations of neuron quantities were tested in all layers, and it was determined that 552 neurons in the dense layer and 512 neurons in the LSTM layer were optimal. To prevent overfitting, a dropout rate of 0.2 was applied in the LSTM layer and an L2 regularization penalty of 0.001 was applied in the dense layer. The ReLU activation function was used for the Conv1D layer and the softmax function was used for the time distributed dense layer. The weights in the model were optimized using the Adam optimizer and the categorical cross-entropy function was minimized using default parameters.

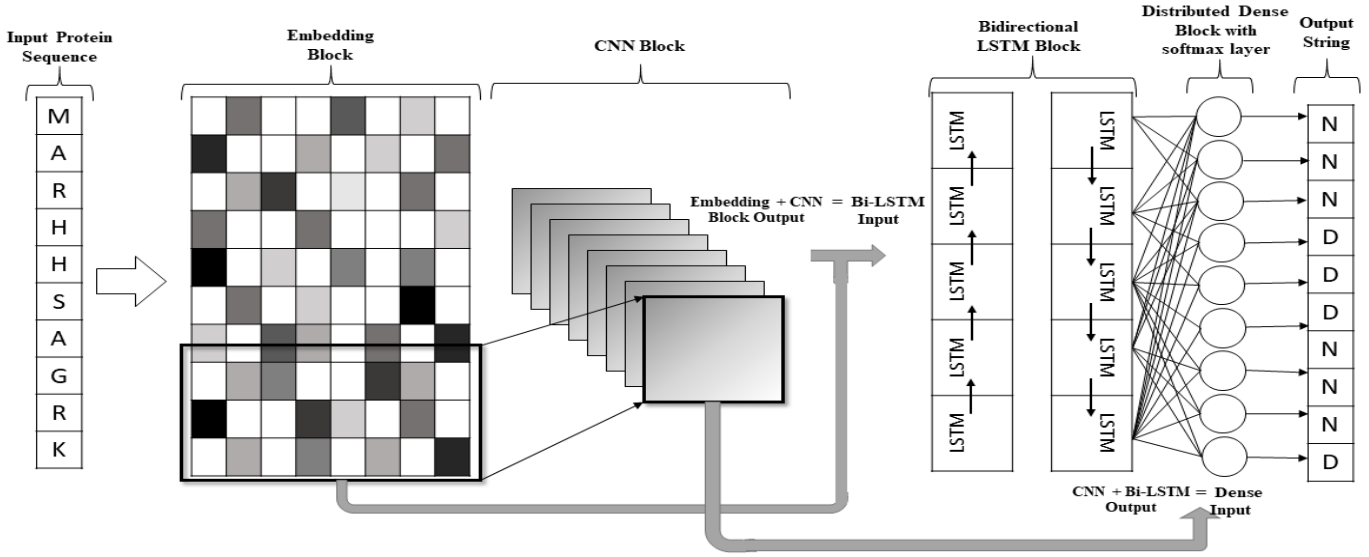

Input Protein Sequence
Embedding Block
CNN Block
Bidirectional LSTM Block
Distributed Dense Block with softmax layer
Output String
M
A
R
H
H
S
A
G
R
K
Embedding + CNN Block Output = Bi-LSTM Input
LSTM
CNN + Bi-LSTM Output = Dense Input
N
N
N
D
D
D
N
N
N
D

Figure 4.7: Architecture of the proposed model. An ensemble of Embedding, CNN and Bi-LSTM for extracting features. Extracted features were fed to step distributed dense layer with softmax function for determining the probability of being ordered or disordered.

4. Performance evalution

After training the model on 80% of the dataset, it was evaluated or validated on the remaining 20% data-set. Both models were concatenated in a package where the protein sequences of lengths between 60 to 300 were processed by model_300, and the protein of length between 301 and 600 will be processed by model_600. The model evaluation parameters on validation-set are given in Table 4.4: Evaluation of the model on the validation set.. ROC curve on the validation set is shown in Figure 4.8.

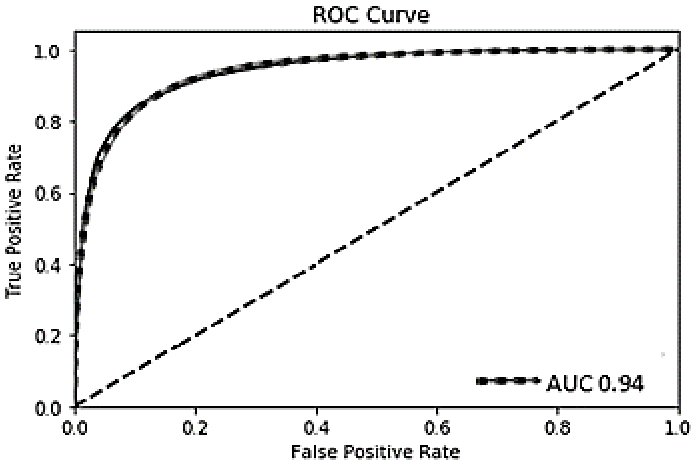


Figure 4.8: ROC curve of proposed model on validation set.

Table 4.4: Evaluation of the model on the validation set.

AUC of ROC	Sensitivity	Specificity	Precision	MCC
0.94	0.859	0.877	0.878	0.736

To compare our proposed model with other state-of-the-art models, we evaluated our trained model on benchmarked data set SL329, which was filtered according to our model's length limit SL293. ROC curve of the predicted results by our model on the mentioned test set is shown in Figure 4.9

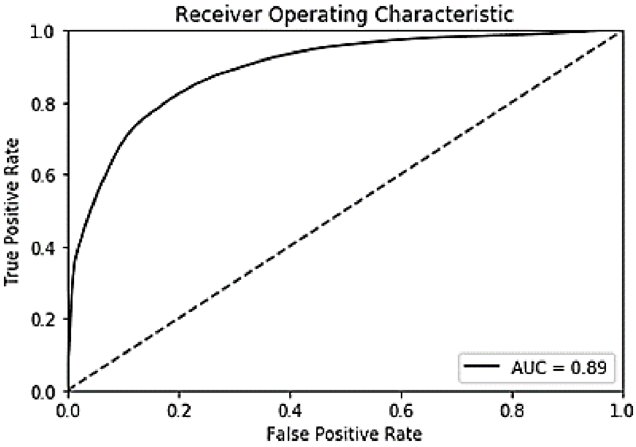


Figure 4.9: ROC curve of proposed model on test set.

Table 4.5: Comparison of the proposed model with various state of the art models on test set SL293.

Method	**AUC of ROC**	**Sensitivity**	**Specificity**	**MCC**
SPINE-D[335]	0.886	0.78	0.85	0.63
MFDp[336]	0.873	0.88	0.62	0.51
Disopred2[328]	0.858	0.69	0.90	0.59
MD[337]	0.864	0.66	0.89	0.58
DISOClust[338]	0.846	0.81	0.70	0.51
PONDR-FIT[339]	0.843	0.61	0.91	0.55
Dispro[340]	0.837	0.28	0.99	0.40
Spot-Disorder-Single[341]	0.887	0.67	0.97	0.604
NORSnet[342]	0.815	0.54	0.92	0.51
PONDR-VLXT[343]	0.755	0.59	0.78	0.38
Proposed model	**0.89**	0.87	0.8	**0.677**

Table 4.5: Comparison of the proposed model with various state of the art models on test set SL293. shows the comparison among some published models and our proposed model in terms AUC of ROC, specificity, sensitivity and MCC (Matthews Correlation Coefficient) . Our model outperformed other well-known servers and programs in terms of AUC and MCC. In terms of sensitivity, our, model scored 0.87. The specificity of our model on test data was found to be 0.8; Specificity here can be stated as a tendency of a model for prediction of a residue in the ordered region when it is genuinely in the ordered region. Higher specificity and lower sensitivity often indicate the biasness of a model for predicting more negative value; in this case, amino acids in the ordered region. Our model seems to have a balanced ratio which can be seen in high MCC value.

v. Discussion and Conclusion

The intrinsically disordered regions are comprehensively being implied to various physiological processes and disease, and also complement the functions of structured proteins. These regions can be determined by multiple experimental techniques. Because of high cost and time for identifying disordered regions experimentally, researchers depend on computational strategies in order to predict probable IDRs/IDPs before conducting subsequent validation through experimental studies. Although many advancements have been made in recent years for prediction of long and short intrinsically disordered regions, but there is still a significant scope for algorithmic improvement.

It has been observed that the neural networks are exceptionally the most effective class of machine and pertinent in taking care of pretty much every sort of issue beginning from classification, clustering, regression, natural language processing, sequence prediction, structure prediction and so forth. The fundamental way of learning a neural network is by altering input loads of each neuron. CNNs are a class of ANNs which are utilized for feature extraction or selection. They are generally utilized in picture image recognition for discovering extraordinary features from pictures. They can additionally be used in extraction of features or perceive specific patterns from sequences which are exceptionally hard to be considered manually. 1D Convolutional Network can be utilizing for choosing features from a 1D information for example a text sequence. RNNs are another class of ANNs which are effective in gaining from a sequence information. They fundamentally utilize their interior state (memory) to process input sequence which enable them to recollect some past information which is useful in managing 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 an 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.

Recent advancements in natural language processing techniques can be seen from more accurate language translations, text summarization, sentiment analysis, text generation etc. [344]. The most difficult part in the NLP pipeline is feature extraction and to find out the relation between words or letters through their numerical representations[345]. Previously various machine learning techniques have been used for solving the problems of identifying IDRs form protein sequences, but their accuracies are limited due to lack of proper feature selection and small dataset[346]. In the present work, amino acids were treated as letters of proteome language. An exploratory data analysis of amino acid sequences of IDRs and ordered regions was carried out to check if they have contrasting differences in the arrangement of sequences, structural and physicochemical properties. Analysis of various kmers in (Figure 4.6) showed us that IDRs are biased towards some amino acid arrangements. Further structural and physicochemical properties enrichment in IDRs support our observation of amino acid biasness. In Table 1 and Figure 4.6, the aromatic amino acid content in IDRs are less as these amino acids tend to stabilize structures due to their aromatic ring[347]. Charged residues are present in more number in IDRs; IDRs have less hydrophobic residues and more hydrophilic residues which shows IDRs presence on the protein surface. IDR enrichment analysis of 12375 protein sequences displays an increase in flexibility, solvation potency and the frequency for forming coils. Through the analysis, it appears that IDRs can be characterized by a decrease in interface propensity, formation beta structures, linker propensity and bulky amino acid significantly (Table 1). These observations can be generalized, and several have been proved previously[348]. After observing the differences in Amino acid arrangements in IDRs and ordered regions, NLP pipeline was used for extracting features via 1D convoluted layers on the top of the embedding layer. Embeddings are much successful in NLP projects for feature assigning than, one-hot-encoding, n-grams, bag of words etc. [349]. Embedding layer learns or updates and assigns the required number of features during training, to each amino acid by adjusting its weights, where the user has to define the number of features per letter or amino acids. On top of the embedding layer, a 1D CNN was placed for retrieving local sequence features. CNN takes a user define window size to apply filters, i.e. an abstract representation of that window size, and the user also defines the number of filters or features. On top of embedding layer and CNN layer a Bidirectional LSTM layer was placed which take account of the output of embedding (single amino acid features) and CNN (derived local features) layers to derive global features. Lastly, these local and global derive features are feed to three neurons for each position of residues to learn and decide if the residue is at IDRs or ordered region. After training the proposed NLP based model achieve AUC of 0.94 with sensitivity, specificity and precision, of 0.859, 0.877 and 0.878, respectively (Table-2). When testing the proposed model on SL293 dataset and comparing with other states of art IDRs prediction models, it was seen that the proposed model have a slight increase in AUC with 0.89, however it was able to balance the Sensitivity, Specificity with an MCC of 0.67 (Table 3). This significant increase in MCC may be due to firstly the use of larger datasets which were retrieved from most updated databases, large dataset means the model will train better. Secondly, fragmentation of the datasets in two length limits i.e. 300 and 600 respectively have help the models to focus on learning the features of amino acids than pads as padding to amino acid ratio was 0.33 in both the data set which was 0.66 in accumulated data which would force the model to learn more features of pads[350]. One of the limitations for the proposed model is the length range i.e. maximum length in can take as input is 600 aa residues, as there were less data available for proteins with more than 600 aa residues and which can be addressed once ample amount of data is available for training. The dataset and ready to execute package is available at https://doi.org/10.24433/CO.3457808.v1, where users can run the application with custom inputs. The proposed model has outperformed the other state-of-the-art techniques for the prediction of IDRs. In terms of speed, the depthwise separable 1D CNNs which have fewer weights than vanilla 1D CNNs, making it faster for making predictions on a genomic scale. The model took 71 mins for processing random 10000 sequences from ModiDB database on Intel Xeon CPU W-2133 @ 3.60GHz without any parallelize processing. NLP techniques are used for processing of languages, this approach showed that amino acid sequences can be treat as the letter for deciphering language of the structural proteome and may have enormous applications in the field of biomedical sciences.

CHAPTER V

Utilizing Deep Learning to Explore Chemical Space for Drug Analogues Generation

Chapter 5. Utilizing Deep Learning to Explore Chemical Space for Drug Analogues Generation

i. Introduction

Medicinal chemists plays an important role to enhance an existing active molecule, whether it be natural or synthetic (i.e., through the practise of Analogues design) [351]. The tweaking of a drug molecule or any other bioactive product in order to create a new molecule that shares similar properties with the original model compound on both a chemical and biological level is known as Analogues design. The Analogues is typically considered to have certain advantages over the original pharmaceutical molecule. The use of Analogues design has been successful, simple to manufacture, and widely accepted in pharmaceutical research since its inception. According to a review, 10% of pharmaceuticals on the market are unaltered compound which exists in nature, 29% are their derivatives (semi-synthetics), and the rest 61% are synthetic[352]. Using commercially accessible drug structures as a starting point for research (i.e., Analogues-based drug design), results in iterative adjustments that improve therapeutic agent efficacy and safety[353], [354]. For example, the 55 years of Analogues design that followed the creation of the historical antibiotic penicillin G allowed for the development of broad-spectrum, modern, orally active-lactam antibiotics such as ampicillin and amoxicillin[355]. Neuroleptics, antidepressants, and antihypertensive medications all underwent similar changes[356]. Analogues design accounts for two-thirds of all small molecule sales.

Analogues, which are similar to a known therapeutic molecule, are usually created through small molecular changes such as the formation of homologues, vinylogues, isosteres, positional isomers, optical isomers, and altered ring systems[357]. The basic structure of the molecule is usually preserved or only slightly modified. Substituent effects, which can be used to fine-tune the molecule, can be analyzed using techniques like QSAR [358] . Given that the Analogues are derived from a known therapeutic molecule as a starting point, drug-like properties are not a major concern. When necessary, filters are used to remove reactive or toxic groups and consider qualities related to absorption, distribution, metabolism, and excretion (ADME) such as blood brain barrier permeability. In the past, discovering functional Analogues was often a result of luck, but now virtual screening and systematic screening can be used to search for these Analogues in databases like PubChem[359] , ChEMBL[360], and ZINC[361], which contain a range of 2.3 million to 750 million compounds.

As a result, only a small portion of the chemical space has been explored thus far, which includes approximately 10^{60} small, synthetically possible molecules [362]. The primary tool for exploring the molecular space is now generative models. The use of generative models to create molecules from scratch has increased over the last few years, and this area has been extensively studied. These generative models—such as recurrent neural networks (RNNs) [363], [364], variational autoencoders (VAE)[365], and generative adversarial networks (GAN) [366], [367]—come from a variety of architectural types and have been demonstrated to produce accurate, unique compounds with desirable physicochemical properties in the same chemical space as their training sets. Generative models have been developed to address two main issues: how molecules are represented and how their properties are optimized[368], [369]. Optimizing molecular properties during the generation process aims to create clusters of high-validity, novel, and synthesizable molecules in chemical space[370], [371]. These improved algorithms have been successful in goal-directed molecular design and have led to the identification of novel active molecules through organic synthesis and activity evaluation, demonstrating the potential of deep learning-based generative models in de novo drug design. Different representations, such as SMILES[372], molecular graphs[373], [374] , fingerprints [375], and 3D geometry[376], are used to help deep learning algorithms understand the features of molecules.

In this study, the SMILES representation were used as the starting point for analyzing chemical structures because it follows a set of rules and conventions for encoding the molecular structure of a compound using letters, numbers, and symbols. In this study, 2.3 million chemical structures from ChEMBL [360]were used to train an autoencoder. An autoencoder is made up of an encoder and a decoder as its basic components. The encoder is a neural network that processes the input data and maps it to a low-dimensional latent space, while the decoder is a neural network that maps the latent space back to the original data space. Together, these two networks comprise the autoencoder, which can be trained to learn a representation of the input data that captures the underlying structure of the data and can be used to generate new data that is comparable to the original dataset. Following the lowest latent layer of the autoencoder, a batch normalisation layer was added. Batch normalisation is a method for normalising the activations of a neural network layer. It often entails subtracting the batch mean and dividing by the batch standard deviation in order to normalise the activations of data. As a result, the distribution of the activations becomes more stable, which can enhance the model's efficiency and convergence. To concentrate the distribution of the latent space of the autoencoder, the model was trained using large batch sizes.

The hypothesis behind the study was that, by adding random noise to an autoencoder's latent representation, similar but slightly different SMILES could be generated because the autoencoder can "decode" the latent representation back into SMILES. By introducing noise into the latent representation, the input to the decoder can be slightly alter, resulting in the generation of new SMILES that are similar to the original but differ in some ways, i.e. generating Analogues. Further, the encoder was fed with a randomly selected FDA-approved drug, vandetanib [377], to obtain its latent representation. Random Gaussian noise was added to its latent representations and passed it into the decoder to generate various drug Analogues.

The molecules were then docked with their native receptors, and molecular-dynamic simulations of the complexes were run to determine their stability and compare them to the control. At least two generated Analogues have similar and improved affinity for its receptor, RET tyrosine kinase.

ii. Literature Review

1. Drug Analogues

Analog design is the process of creating a new chemical compound that is similar to an existing drug or bioactive compound in an effort to enhance or replace it[356]. This is commonly done for financial reasons, but it can also result in compounds that are more effective. Analog design is common in pharmaceutical research and has been used to create a wide range of complex molecules such as steroids, prostaglandins, anticancer drugs, and antibiotics. Analogue design is thought to account for roughly two-thirds of small molecule sales. This process led to the development of oral, broad-spectrum beta-lactams such as ampicillin and amoxicillin, which are safer and more effective than penicillin G[378], [379]. This method has also been used in the creation of medications for mental illness, hypertension, and other conditions. Many of the small-molecule drugs introduced or approved between 2000 and 2003 were based on existing drug structures or structures in the early stages of development[380]–[382]. However, some critics argue that this approach primarily produces "me-too" drugs as opposed to truly innovative medications. It is essential to note that there are various types of analogue drugs, and not all of them fall under this critique[383], [384].

In the field of natural science, analogy refers to structural and functional similarity between two things. An analogue of an existing drug molecule is a compound that shares chemical and pharmacological similarities with the original compound. There are three types of drug analogues: those with both chemical and pharmacological similarities; those with chemical differences but similar pharmacological properties; and those with neither chemical nor pharmacological similarities.

The first category of drug analogues consists of those with chemical and pharmacological similarities to the original compound, also known as "me-too" drugs. These analogues are pharmacologically, pharmacodynamically, or biopharmacologically superior versions of the original compound. Through simple molecular modifications, such as the synthesis of homologues, vinylogues, isosteres, positional isomers, optical isomers, modified ring systems, and twin drugs, they are produced (homodimers). These analogues are created for the same reason that other industrial products, such as laptops and automobiles, are: to provide the consumer with an advantage in the form of enhanced therapeutic benefit.

The second category of analogues, structural analogues, consists of compounds that were designed as patentable analogues of a novel lead compound but have unexpected pharmacological properties. For instance, imipramine was originally intended as an analogue of the neuroleptic drug chlorpromazine, but antidepressant properties were discovered. Sildenafil was originally developed as an antihypertensive medication, but clinical trials revealed that it is also effective for treating erectile dysfunction in men. Structural analogues may also be the result of systematic studies involving the multi-target screening of a large number of structurally similar compounds[385], [386].

Functional analogues are compounds that lack chemical similarity but share similar biological properties. Historically, these analogues were the result of chance observations; however, they can now be designed through the virtual screening of large libraries of diverse structures. Functional analogues include the neuroleptics chlorpromazine and haloperidol, which have different chemical structures but similar affinities for the dopamine receptor, and the tranquillizers diazepam and zopiclone, which have different chemical structures but similar affinities for the benzodiazepine receptor[385], [387].

The synthesis of new analogues can be accomplished through a variety of methods. Beginning with natural sources such as plants and animals can provide a wealth of chemical diversity and lead to the discovery of new medications. Natural compounds such as cocaine, morphine, and quinine have given rise to numerous analogues[388], [389]. In some instances, metabolites generated by various metabolic pathways may have superior properties to the parent molecule and can serve as lead compounds for the synthesis of additional analogues[390]. Lastly, the success of existing drugs may also inspire the creation of "fast followers" or "me-too" drugs with a comparable structure or function. The purpose of analogue synthesis is to enhance the properties of existing compounds and develop new drugs with improved efficacy, safety, and ADME (absorption, distribution, metabolism, and excretion) profiles[391]. For instance, a drug with a short half-life is rapidly metabolised and eliminated from the body, necessitating multiple doses throughout the day. By modifying the drug's chemical structure, it may be possible to increase its half-life and decrease its dosing frequency. In addition to enhancing the safety profile of a drug, drug analogue design is pursued in order to enhance the drug's safety profile. Some medications may have undesirable side effects, such as gastrointestinal discomfort or liver toxicity. By making minor modifications to the drug's chemical structure, it may be possible to reduce or eliminate these side effects[392].

2. Computational methods for Drugs analogues Designs

Designing drug analogues is the process of creating structural variations of existing drugs in an effort to enhance their potency, stability, and safety. This process typically involves making minor modifications to the original drug's chemical structure in order to enhance its therapeutic effects. For instance, a drug analogue's design algorithm may seek compounds with similar structures to an existing drug but with different functional groups or substituents that are predicted to increase its potency or decrease its toxicity.

Typically, the design of drug analogues involves multiple steps. Initially, the original molecule of the drug is identified and its structure is thoroughly analyzed. Then, various variations of the molecule are created and evaluated to determine which ones possess the desired properties. This can be a time-consuming and labor-intensive process, as it may involve synthesising and testing a large number of variants prior to determining the optimal solution. Once a promising drug analogue has been identified, it must undergo additional testing to determine its safety and efficacy in treating the condition of interest. This typically involves administering the drug to human subjects and closely monitoring the results of clinical trials. The drug can then be submitted for approval by regulatory bodies such as the Food and Drug Administration (FDA) in the United States if it is found to be safe and effective.

To create an analogue of a drug, one also has to comprehend the chemical structure and mechanism of action of the original drug. Typically, this includes studying the drug's mechanism of action, pharmacokinetics, and pharmacodynamics. Once this is understood, chemists can use a variety of techniques to modify the drug's chemical structure in order to create analogues. Altering the chemical structure of the original drug, such as by substituting one of its atoms or atom groups with a different atom or group, is a common strategy. There are a number of computational methods for designing chemical analogues[393], [394]. Utilizing computer-based modelling and simulation techniques to predict the properties and behaviour of potential analogue compounds and evaluate their potential efficacy and safety for a given application is typical of these methods.

Using structural modelling, which involves creating a computer-generated model of the chemical structure of an interesting compound, is one approach. This technique can then be used to predict the compound's behaviour in various environments, such as in the presence of other chemicals or under varying temperature and pressure conditions using molecular dynamic simulation[395]s.

3. De-novo drug generation

There are approximately 10^{23} to 10^{60} drug-like compounds in chemical space, making it difficult to investigate each one[396], [397]. To efficiently discover new lead compounds for drug discovery, high-throughput screening and virtual screening are frequently used to evaluate large chemical libraries with a number of filters. Quantitative structure-activity relationship methods based on machine learning have also been used to evaluate the physical and pharmacological properties of molecules[398]. De novo drug design entails the generation of new molecules with the desired properties from scratch, whereas traditional virtual screening methods search existing chemical libraries for molecules with the desired properties[399]. Several de novo drug design strategies employ computational growth and evolutionary algorithms to generate new molecular structures from smaller building blocks. However, these methods frequently struggle to optimise multiple objectives while simultaneously producing novel compounds. The advent of deep learning has opened up new opportunities for the design and discovery of innovative pharmaceuticals. In recent years, various deep learning-based de novo drug design algorithms have been developed, and in 2020, the Massachusetts Institute of Technology Technology Review ranked the use of deep learning in drug discovery as one of the top 10 breakthrough technologies[400]. In virtual screening, methods based on deep learning are frequently employed to predict the physical or biological properties of input molecules, which is a form of discriminative modeling. In contrast, generative models based on deep learning can be utilised to explore the vast chemical space and identify compounds with desirable properties, a process known as reverse quantitative structure-activity relationship. Generative models aim to approach desirable molecular properties through optimised strategies, similar to virtual screening techniques, which use multiple filters to narrow the chemical space until it is manageable. By exploring the continuous space of properties, generative models are believed to be able to generate molecules with novel scaffolds and desirable properties.

Recurrent neural networks, encoder-decoder models, reinforcement learning, and generative adversarial networks are the four primary classes of machine learning algorithms used to construct generative models. These models address molecular representations and optimization strategies and generative adversarial networks are the four primary classes of machine learning algorithms used to construct generative models. These models address molecular representations and optimization strategies. Different types of molecular representations, such as SMILES, molecular graphs, fingerprints, and three-dimensional geometries, are used to help deep learning algorithms comprehend the various features of a molecule. The objective of optimising molecular properties during the generation process is to create high gradients and clustered regions in chemical space for the generated molecules, thereby ensuring their validity, originality, and synthesizability. Using tensor decomposition and self-organizing map techniques, for instance, the GENTRL algorithm optimised the properties of molecules in chemical space to design novel active compounds against the Discoidin Domain Receptor Tyrosine Kinase 1.

4. SMILES chemical representations

SMILES, or Simplified Molecular Input Line Entry System, is a widely used representation of chemical structures in computational chemistry. Many generative models use SMILES as input, but a molecule can be represented by multiple different SMILES strings, which can lead to problems with atom-order invariance in the latent space[148], [401]. These issues can be addressed through the use of SMILES enumeration. Context-free language, which provides a more robust encoding of the rings and branches of molecules, can also be used to extend SMILES and has been applied successfully in generative models. A chemical smile is a string of characters that represents a chemical compound or molecule using a specific set of rules. It is also known as a SMILES string. A chemical smile uses a combination of letters, numbers, and symbols to encode the molecular structure of a compound. The letters represent the elements that make up the molecule, and the numbers and symbols indicate the connectivity between the atoms. For example, the chemical smile for water (H_2O) would be "O" because the molecule contains one oxygen atom and two hydrogen atoms.

To form a SMILES string, you need to follow a set of rules and conventions that define how the molecular structure of a compound is represented using letters, numbers, and symbols.

Here are the steps to form a SMILES string:

1. Write the element symbols for the atoms in the molecule, in the order in which they appear in the molecule. For example, the SMILES string for water (H_2O) would be "O" because the molecule contains one oxygen atom and two hydrogen atoms.

2. Use parentheses to enclose any atoms that are part of a sub-structure or branch in the molecule. Theophylline, a compound commonly found in tea leaves, can be represented by the following SMILES string:

CN1C(=O)C2=C(NC1=O)N(C)C(=O)N2C

In this SMILES string, the substructure enclosed in parentheses represents a cyclic group within the larger molecule. The substructure "(=O)" within the cyclic group represents a carbonyl group.

3. Use the symbol "=" to indicate a double bond between two atoms, and the symbol "#" to indicate a triple bond. For example, the SMILES string for ethene (C_2H_4) would be "C=C" because the molecule contains two carbon atoms that are double-bonded together.

4. Use the symbol "-" to indicate a single bond between two atoms, unless the bond is explicitly indicated using one of the other symbols. For example, the SMILES string for ethoanol (C_2H_6O) would be "CC-O" because the molecule contains one carbon atom that is

bonded to four hydrogen atoms, and the bond between the carbon and hydrogen atoms is implicit.

5. Use the symbol "@" to indicate the chirality (handedness) of an atom. For example, the SMILES string for D-glucose ($C_6H_{12}O_6$) would be "C@C(O)CC(O)C(O)C(O)O" because the molecule contains one chiral carbon atom (indicated by the "@" symbol) and five other carbon atoms that are bonded together in a ring structure.

Overall, forming a SMILES string involves following a set of rules and conventions to encode the molecular structure of a compound using letters, numbers, and symbols. By following these rules, you can create a SMILES string that accurately represents the chemical structure of a molecule. Chemical smiles are used in cheminformatics, a field that involves the use of computer technology to store, search, and analyze chemical data. They provide a compact and standardized way to represent molecules and can be used to search databases of chemical compounds, predict the properties of compounds, and perform other types of analysis.

5. Algorithms

There are two main categories of deep learning-based generative models: those that generate molecules by sampling the latent space, such as encoder-decoder models, and those that generate molecules towards predetermined molecular properties, such as recurrent neural network, reinforcement learning, and generative adversarial network models[402], [403]. To achieve goal-directed molecular generation towards a specific target, several strategies can be used, including generating a large chemical library and using virtual screening, retraining the model with active compounds for the target, introducing established quantitative structure-activity relationship models for the target into the adversarial or reinforcement architecture, and including protein-ligand interactions in the model's training process. Encoder-decoder models, including variational autoencoders and transformers, comprise an encoder that maps a molecule to a vector in the latent space and a decoder that maps the probability distribution back to the original representation[404]. Recurrent neural network models solve the gradient-vanishing problem by generating the most probable arrangement of the molecular representation using algorithms such as long short-term memory and gate recurrent unit.

Transfer learning is the process of enhancing learning in a new task by transferring knowledge acquired in a related task. Transfer learning in the context of recurrent neural network models entails using active compounds to retrain the generative model for a specific target. Several studies have successfully employed transfer learning to fine-tune recurrent neural network models in order to generate molecules with novel scaffolds aimed at particular targets. One advantage of recurrent neural networks is their capacity to generate molecules of limitless length, as well as their relatively straightforward model training[405]–[407]. A generator and a discriminator play a zero-sum game, with the discriminator attempting to distinguish generated molecules from real positive molecules and the generator attempting to generate molecules from noise and deceive the discriminator[408]–[410]. For de novo drug design tasks, generative adversarial networks are frequently combined with other deep learning frameworks. Objective-reinforced generative adversarial networks and objective-reinforced generative adversarial networks for inverse-design chemistry use a generative adversarial network discriminator and a reinforcement learning reward system to evaluate the generated molecules. Combining generative adversarial networks and variational autoencoders, adversarial autoencoders can be used to generate compounds against a specific target by operating on the latent layer of the variational autoencoder and introducing a generative adversarial network discriminator. Variational autoencoders are less suitable than adversarial autoencoders for molecular feature extraction based on the fingerprint representation of molecules.[411], [412] Deep reinforcement learning is a technique for optimising deep learning models that employs a reward function to score the actions of an agent, with the resulting scores guiding the agent's subsequent actions[369]. In the process of molecular generation, a generator (agent) generates molecules in a predetermined mode (task and action), and a built-in quantitative structure-activity relationship predictor (environment) evaluates the quality (reward or penalty) of the generated molecules. The evaluation procedure aims to enhance the desired characteristics of the generated molecules. Deep Q-Network, REINFORCE, and Monte Carlo tree search are the most common reinforcement learning-based generative model training strategies[413], [414].

In tasks such as image, language, and music generation, deep learning algorithms have exhibited significant advantages over conventional machine learning algorithms. There are at least three major differences between image/text generation and molecule generation: data representation, machine learning-based predictors, and evaluation metrics. Molecular representation is a crucial component of a generative model, and there are a variety of molecular representations to choose from, as different tasks may call for distinct molecular representations[401], [415], [416]. Moreover, the fault tolerance of the generated molecules differs from that of other representations. Machine learning-based quantitative structure-activity relationship models are frequently used as reward functions in reinforcement learning and generative adversarial network architectures to guide the generation of molecules, but the false positive rate of these models for predicting and classifying the bioactivity, physical and chemical properties, and drug-likeness of molecules should not be overlooked. Wet laboratory experiments are the best way to evaluate the quality of generated molecules, but due to the large number of generated molecules, this is typically impractical. Consequently, machine learning criteria are used to evaluate the quality of generated molecules, but there is no gold standard metric for evaluating the performance of deep generative models or the quality of generated molecules[417], [418].

Previous methodological research has primarily focused on improving the deep learning framework for generating valid molecules with some level of novelty and/or synthetic accessibility. There are, however, fewer empirical studies on the de novo design of novel molecules, followed by chemical synthesis and activity evaluation using the deep learning framework. Furthermore, the majority of open-source tools are focused on developing new algorithms rather than de novo drug design for specific targets. Therefore, novel drug design requires platforms that integrate multiple deep learning molecular generation algorithms. Traditional methods, such as the genetic algorithm, can achieve comparable results with appropriate fine-tuning for de novo drug design, despite the potential of deep learning technology[419]. Deep learning is still in its infancy in the field of drug design, and these developments in other fields may provide guidance for future drug design and discovery applications.

iii. Methodology

1. Model Building and Data curation

Autoencoders are a type of neural network used for unsupervised learning. It is frequently employed in dimensionality reduction and feature learning. It is divided into two parts: an encoder and a decoder. An autoencoder was trained to reconstruct its input data, which was chemical SMILES, by learning a compact, lower-dimensional representation of the input data known as the latent space. An autoencoder architecture was implemented that would use embeddings to represent characters in text data, an LSTM encoder to process the embedded text data and compress it into a latent space, and an LSTM decoder to reconstruct the original text data from the latent space representation. The autoencoder's layer by layer details are mentioned below and shown in Figure 5.1:

Input layer: The input to the autoencoder is a sequence of characters in the form of a list of integers, where each integer represents a character in the input chemical SMILES.

Embedding layer: The input integers are passed through an embedding layer, which maps each integer to a dense vector of floating-point values. This allows the autoencoder to represent each character in the input SMILES as a dense, continuous vector, which can be processed by the neural network. These vectors are learned during the training process and are intended to capture the relationships between characters in the SMILES vocabulary.

Encoder: The embedded input SMILES is then passed through an LSTM encoder, which processes the sequence of embedded vectors and compresses them into a fixed-length latent space representation. This is done by iteratively processing the embedded vectors one at a time and using the hidden state of the LSTM at each time step to capture the context and dependencies between the characters in the input SMILES.

Flatten layer: The output of the LSTM encoder is then passed through a flatten layer, which converts the latent space representation from a two-dimensional tensor to a one-dimensional tensor. This allows the autoencoder to learn a more compact representation of the input SMILES data.

Fully connected layers: The vectors of the Flatten layer is then passed through a fully connected (FC) layer, which further reduces the number of vectors into 100 for more compact representation of the one-dimensional latent space.

Batch normalization layer: The latent space of an autoencoder is a representation of the data that is typically not regularized and may be discrete for certain data points. To structure the latent space in a specific way or make it follow a particular distribution during training, a batch normalization layer was used. This layer normalizes the activations of the previous layer by scaling them to have a mean of zero and a variance of one. The batch size can affect the distribution of the latent space when batch normalization is used in an autoencoder in several ways[420], [421]. Larger batch sizes may provide more accurate estimates of the mean and variance of the batch normalization layer and can lead to more stable gradients during training, making it easier for the network to learn the latent space distribution[422]. This can result in a more concentrated distribution as the network more effectively learns the underlying structure of the data. Larger batch sizes can also reduce the occurrence of internal covariate shift, which can improve the stability of the network's training and prevent it from "forgetting" previously learned information, resulting in a more concentrated distribution of the latent space[423]. This can improve the training of the network and lead to more precise and stable latent representations of the input data. To determine the optimal network hyperparameters, the effect of various batch sizes on the latent space distribution was examined, including batch sizes of 32, 64, 128, 256, 512, 1024, 2048, 4096, and 8192.

Gaussian Noise layer: The gaussian noise layer adds random noise to the activations of the previous layer. This can help improve the robustness and generalization of the model, as it forces the model to learn features that are robust to noise.

Decoder: The normalized latent space representation with added gaussian noise is then passed through a time distributed layer, which reshapes the vector into a sequence of vectors that can be processed by an LSTM layer. The LSTM decoder processes the sequence and produces a reconstruction of the original input sequence. The LSTM decoder does this by iteratively processing the latent space representation, one element at a time and using the hidden state of the LSTM at each time step to generate a sequence of embedded vectors that are similar to the original input SMILES.

Output layer: The final reconstructed output sequence produced by the autoencoder is passed through a time distributed layer with a softmax activation function, which maps the output to a probability distribution over the vocabulary represented through one-hot encoding. This reconstruction is compared to the original input sequence, and the difference between the two is used to compute the loss function of the autoencoder. The model was implemented using Tensorflow 2 [95] library in python3 [424].

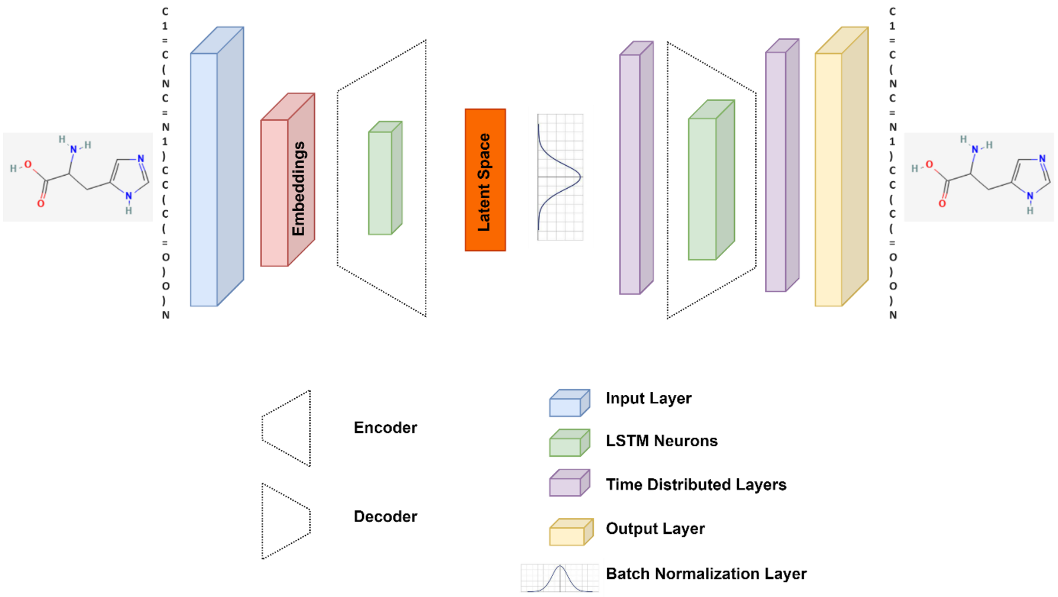


Figure 5.1: Layer by layer representation of proposed autoencoder architecture.

To facilitate easy computation, encoded strings were limited to 100 characters in length. The SMILES (simplified molecular input line-entry system) representations for 2.2 million compounds with 100-character or shorter SMILES strings were downloaded from the ChEMBL23 database. A ratio of 80:10:10 was used to divide the dataset into three sets: a training set, a testing set, and a validation set. The SMILES strings were encoded as text using a 45-character set. If a string was shorter than 100 characters, it was padded with spaces to reach that length. To avoid dealing with multiple equivalent SMILES representations for the same molecule, only "canonicalized" forms of SMILES (a standardised version of the SMILES representation of a molecule) were used for training. With the goal of minimising the difference between the input sequence and the reconstructed output sequence, the proposed deep learning model was trained with these data using the Adam optimization algorithm. Using the grid search algorithm, the optimal hyperparameters for the model were determined to be an embedding size of 32, an encoder LSTM with 64 neurons, a latent space with a 100-dimensional vector, and a decoder LSTM with 256 neurons.

The model with optimum hyperparameters was trained with different batch sizes i.e. 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192 till the reconstruction categorical accuracy reaches above 0.98 or 98%. The reconstruction categorical accuracy of a character-based autoencoder is a measure of how well the autoencoder can reconstruct the input characters after encoding and decoding them. This is typically calculated by comparing the reconstructed characters with the original input characters and computing the percentage of characters that are correctly reconstructed. Usually, the required reconstruction categorical accuracy was achieved before 128 epochs. Creating bell shaped curve representation of latent space will have a similar scale to the original data. This is important because it allows the autoencoder to capture the overall variations in the data rather than being influenced by any type of scaling. Therefore, the distribution of the latent space in all the trained models with various batch sizes were computed. The autoencoders that were trained with different batch sizes were tested by inputting 10000 randomly selected chemical SMILES and plotting the distribution of the resulting latent vectors. From these results, the model whose latent vector distribution was the most compact, or closest to a normal distribution, was selected for further investigation.

2. Generating Analogues

During training, the encoder and decoder learn to reconstruct the input chemical SMILES as accurately as possible. Once the autoencoder is trained, it can be used to generate similar SMILES i.e. Analogues by adding random noise to the latent representation of an chemical SMILES and then using the decoder to reconstruct the new SMILES. The resulting SMILES will be similar to the original input image, but with some variations or perturbations introduced by the noise in the latent representation. Simply the Gaussian noise was added to the latent representation, which will add random variations to the SMILES.

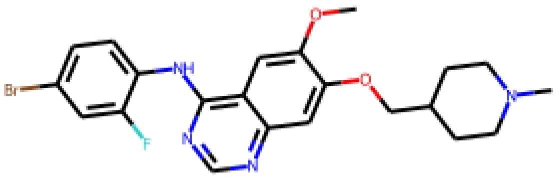


Figure 5.2: Structure of Vandetanib.

.

The FDA-approved drug Vandetanib was randomly selected, which is used to treat certain types of cancer, for the purpose of generating Analogues and conducting future competitive studies. Vandetanib belongs to a class of drugs called tyrosine kinase inhibitors, which block the activity of enzymes that contribute to the growth and spread of cancer cells. Vandetanib specifically targets RET, VEGFR, and EGFR tyrosine kinases, which become inactive and lead to tumor regression by replacing ATP in the substrate binding site of mutant proteins. An autoencoder was used to process the canonical SMILES representation of Vandetanib "COc1cc2c(Nc3ccc(Br)cc3F)ncnc2cc1OCC1CCN(C)CC1" and extract its latent space representation. Then, random noise were added to the latent space representation and passes back into the encoder to generate Vandetanib Analogue SMILES. This process was iterated with different random noise values, checked the validity of the resulting SMILES using the RDKit library, and generated 157 Analogues of Vandetanib while discarding any invalid SMILES as determined by the RDKit library [425].

3. Docking and MD simulation

Open Bebel [426] was utilized to create 3D SDF structures for all 157 Analogue compounds. The structure of the phosphorylated RET tyrosine kinase domain complexed with the inhibitor Vadetanib was obtained from PDB with the ID 2IVU[427]. The receptor was prepared using AutoDockTools [428] according to the specified procedure. EasyDock vina [429] was used to dock all the Analogues and the control compound Vadetanib with the prepared receptor, which utilizes AutoDock Vina in the background. The grid box was generated by taking the coordinates of the ligand's center from the PDB structure and extending them outward by 25 points in each axis. The top two Analogues and the Vadetanib complex were simulated using the specified protocol for 10 ns.

In order to compare the stability of two Analogue compounds to the reference compound (Vadetanib), molecular dynamics simulations were performed on all the three docked complexes. The systems were created using CHARM GUI's solution builder feature [430]. The TIP3 rectangular water box with a 10 Å gap between the protein edges and the box borders was used for the experiment. The topologies and coordinates for each system were generated using the CHARMM36 all-atomic force field and the CHARMM general protein drug complex force field[431]. Sodium and chloride ions were added to neutralize the systems. Gromacs version 20.3 was used for the simulations [432]. Each system has been subjected to a 50,000-step steepest descent energy minimization procedure in order to minimize steric repulsions. The NVT equilibration was run for 500 picoseconds (ps) to stabilize the system temperature, and a short position restraint NPT was run for 500 ps to stabilize the system pressure by relaxing the system and maintaining the protein confined. All systems were simulated for 10 nanoseconds (ns) in the absence of restrictions. GROMACS utilities were used to analyze the trajectory files. The root mean square deviation is calculated using gmx rms (RMSD).

The MM-PBSA (Molecular Mechanics Poisson-Boltzmann Surface Area) is a reliable and efficient free energy simulation tool that has been widely used to mimic molecular affinity, including protein-ligand binding interactions [433]. When used in conjunction with molecular dynamics (MD) simulations, MM-PBSA can incorporate conformational and entropic variables into the binding energy. This method has also been utilized to provide a thorough understanding of biomolecular interactions, by decomposing the total binding energy into many sections. In this study, the frames composing the MD trajectories obtained for receptor–ligand complexes were analyzed using gmx_MMPBSA tool[434]. Frames from the last 5 ns (250 frames) of the trajectory out of 10 ns were used for MMGBSA and MMPBSA analysis [435]. The complexes were refined with the oldff/leaprc.ff99SB force field. The following equation ΔGbind = ΔG complex − [ΔG receptor + ΔG ligand] was used to evaluate the systems net binding energy, where ΔG in GBSA and PBSA are the sum of the van der Waals (VDW) and electrostatic (EEL) as well as polar (EGP/EPB) and non-polar (ESURF/ENPOLAR) free energy of solvation.

iv. Results

1. Evaluation of the Laten space and Model

An autoencoder's latent space is a condensed representation of the input data that is not necessarily structured or regularized. In order to shape the latent space in a specific way or force it to conform to a certain distribution during training, a batch normalization layer can be employed. This layer normalizes the outputs of the previous layer by adjusting them so that they have a mean of zero and a variance near to one i.e. a bell shaped curve. The influence of batch normalisation on the latent space is depicted in Figure 5.3. When a batch normalisation layer is applied, the distribution of the latent space is observed to be more bell-shaped than when it is not. For adding noise with a similar distribution to make data points that are similar to the original, like similar molecular SMILEs, the latent space needs to be set up in a certain way or forced to follow a certain distribution.

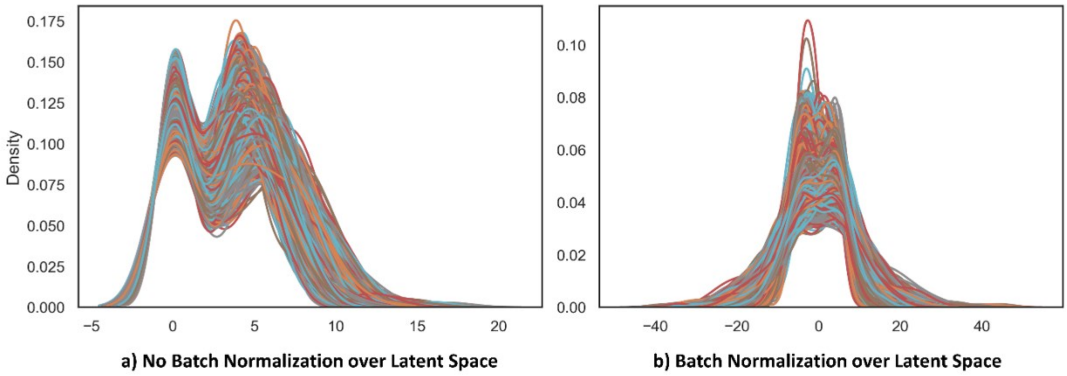


Figure 5.3: Distribution of the latent space with and without batch normalization layer.

We experimented with different batch sizes by training our optimum layered model to find the most compact latent space, which can be used to generate Analogue compounds SMILES using gaussian noises. The trained encoders were tested on 10,000 randomly selected molecules and plotted the latent representations. Our results showed that the model with a batch size of 8192 had the most compact latent space, as indicated by the smaller distance between the mean and standard deviation from normal distribution compared to other batch sizes (as seen in Table 5.1: Mean and standard deviation of the distribution of latent representations for 10000 randomly chosen compounds created using models trained with different batch sizes.and Figure 5.4). The model with larger batch sizes than 8192 were not train due to technical limitations.

Table 5.1: Mean and standard deviation of the distribution of latent representations for 10000 randomly chosen compounds created using models trained with different batch sizes.

Batch Size	Mean	Standard Deviation
32	-0.0931	8.6
64	-0.37	9.187
128	0.2801	8.3423
256	-0.2610	5.6089
512	0.4932	5.76505
1024	0.3020	5.8383
2048	0.43755	5.4940
4096	0.49061	5.53
8192	-0.1674	4.8518

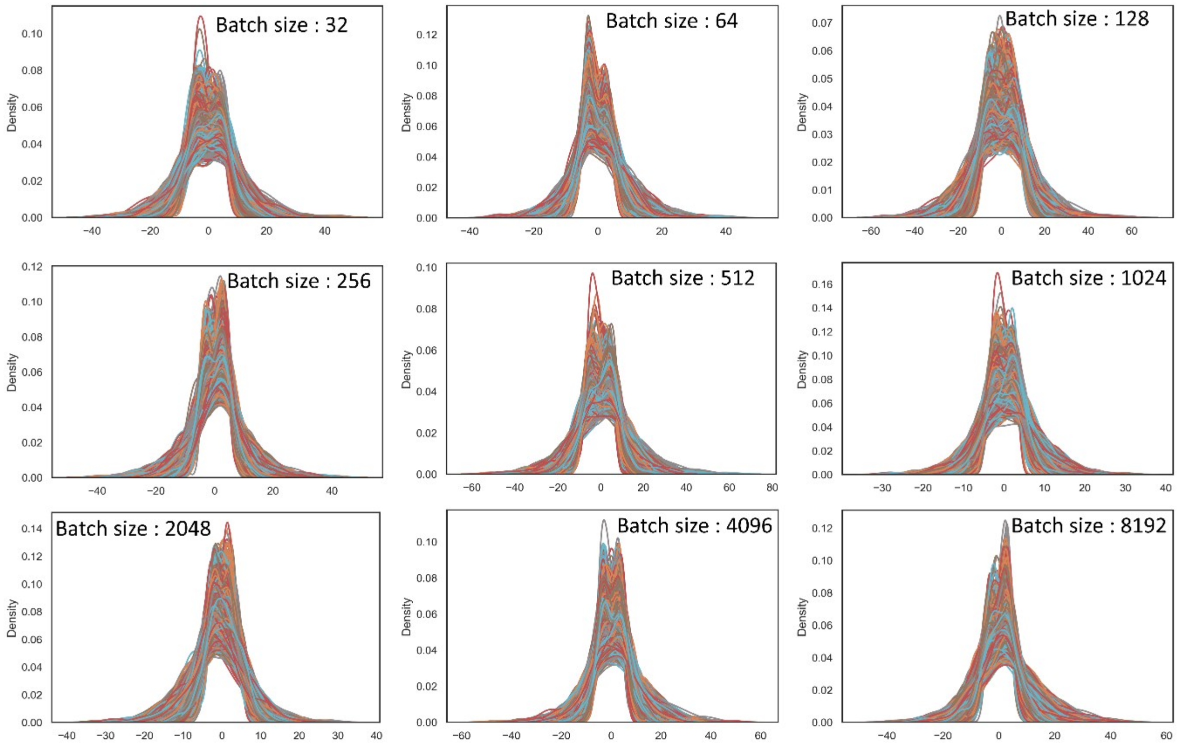


Figure 5.4: The distribution of latent representations for 10000 randomly chosen compounds created using models trained with different batch sizes.

In order to assess the effectiveness of different architectures for generating molecules, a range of metrics from the scientific literature were used, including validity (which checks whether the generated molecules are chemically feasible), uniqueness (the percentage of generated molecules that are not present in the training dataset), novelty, and the Fréchet ChemNet distance (FCD) (which measures how similar the distribution of generated data is to the molecules in the training set) [436], [437]. Given the model's objectives—in this case, the generation of Analogues—and the potential for different hyperparameter tuning to affect results, it is difficult to compare generative models. Therefore, hyperparameters based on how they were stated in the respective publications were noted. Finally, 10,000 compounds were genrated from our architecture. The proposed architectural design functioned similarly and was capable of producing compounds with a high FCD score that were valid, unique, and novel as other methods (Figure 5.5). These findings were consistent with other benchmarking studies and those described in the literature.

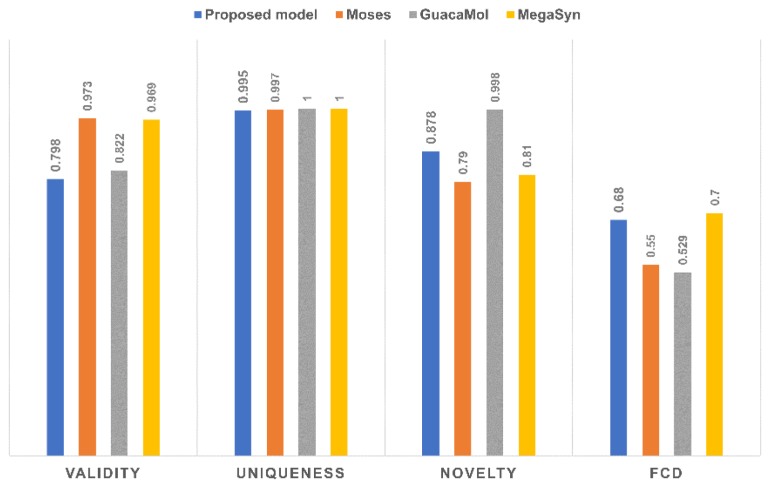


***Figure 5.5: Evaluating various model architectures for generative models using our model and comparing the results to those reported in other published benchmark resources such as MOSES*[437]*, GuacaMol*[436]*, and MegaSyn*[438]**.**

2. Generating Vadetanib Analogues

Vadetanib, an FDA-approved drug for lung cancer treatment, was selected and used the autoencoder to generate 157 similar SMILES by adding Gaussian noise to the latent representation of the drug's SMILES. Figure 5.6 illustrates a few of the Analogues produced through this process, which is described in the methodology.

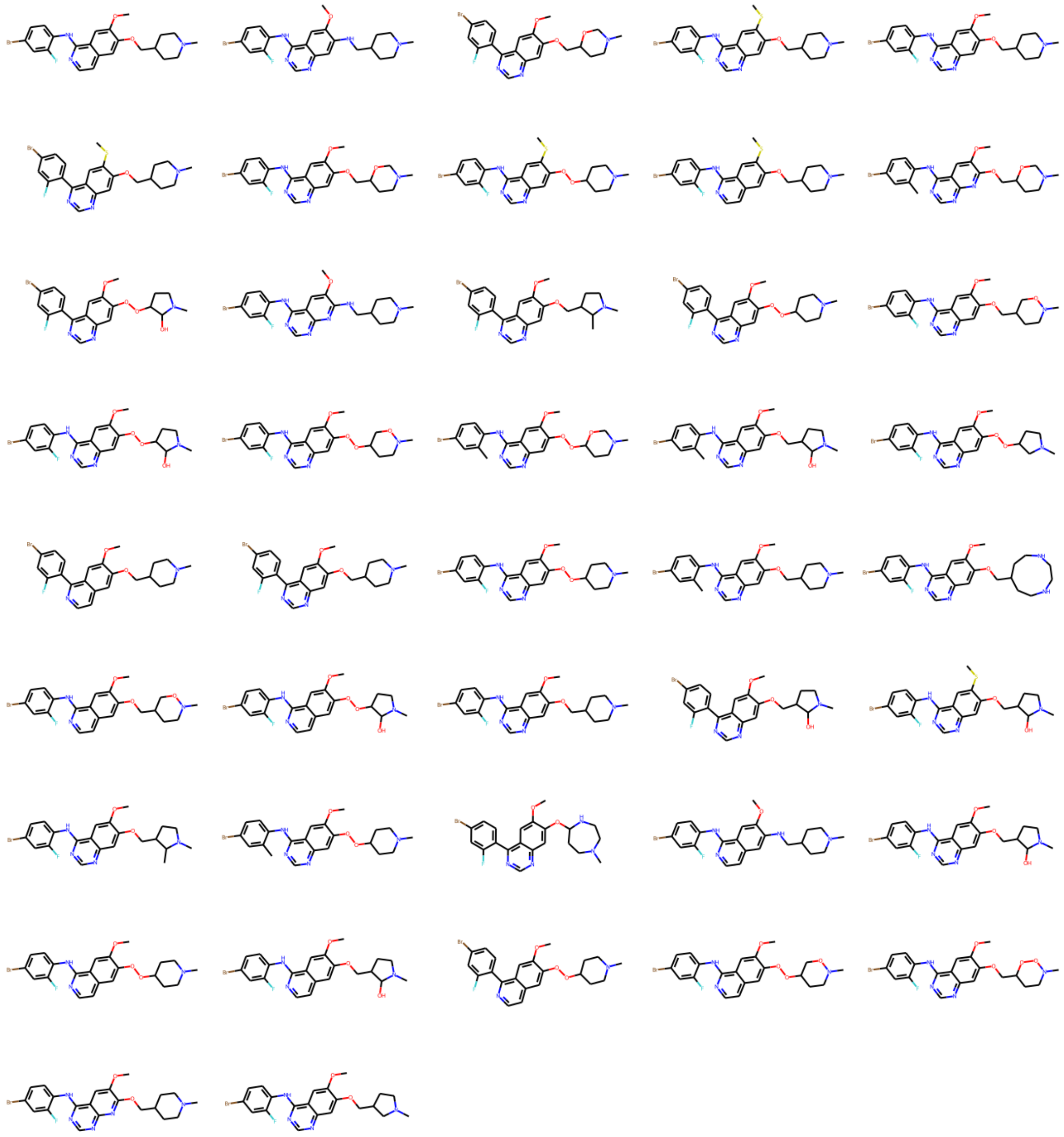
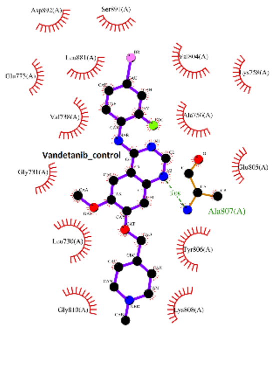
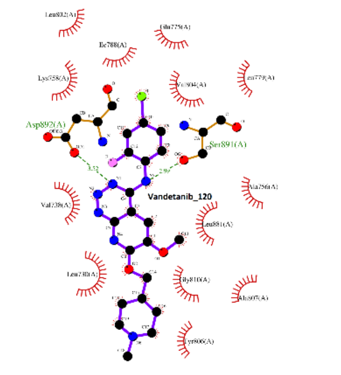
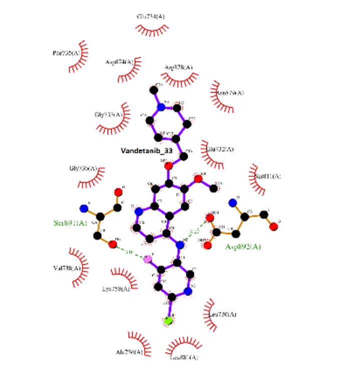


Figure 5.6: Few among the Vadetanib Analogues generated by the proposed model.

3. Virtual screening

EasyDock Vina was used to dock all 158 ligands including the control with the RET tyrosine kinase domain found in the Protein Data Bank (PDB). From these, the top two Analogues that have a stronger binding affinity than the original control were selected. Ligplot+ was employed to visually analyze the interactions between the ligands and the target protein[439].

Results	Vandetanib_Control	Vandetanib_120	Vandetanib_33
SMILES	COc1cc2c(Nc3ccc(Br)cc3F)ncnc2cc1OCC1CCN(C)CC1	COc1cc2c(Nc3ccc(Br)cc3F)nnnc2nc1OCC1CCN(C)CC1	COc1cc2c(NC3=CC=C(Br)C=CS3)ncnc2cc1OCC1CCN(C)CC1
Affinity (kcal/mol)	-8.8	-10.4	-10.2

Interactions	 Asp892(A)
	Ser891(A)
	Leu881(A)
	Val804(A)
	Lys758(A)
	Glu775(A)
	Val738(A)
	Ala756(A)
	Vandetanib_control
	Gly731(A)
	Glu805(A)
	3.08
	Ala807(A)
	Leu730(A)
	Tyr806(A)
	Gly810(A)
	Lys808(A)
		 Leu802(A)
		Glu775(A)
		Ile788(A)
		Lys758(A)
		Val804(A)
		Leu779(A)
		Asp892(A)
		Ser891(A)
		3.52
		2.99
		Ala756(A)
		Vandetanib_120
		Val738(A)
		Leu881(A)
		Leu730(A)
		Gly810(A)
		Ala807(A)
		Tyr806(A)
			 Glu734(A)
			Phe735(A)
			Asp874(A)
			Arg878(A)
			Asn879(A)
			Gly733(A)
			Vandetanib_33
			Glu732(A)
			Gly736(A)
			Ser811(A)
			Ser891(A)
			Asp892(A)
			3.27
			Val738(A)
			Lys758(A)
			Leu730(A)
			Ala756(A)
			Leu881(A)

Ligand bond
Non-ligand bond
3.0
Hydrogen bond and its length
His 53
Non-ligand residues involved in hydrophobic contact(s)
Corresponding atoms involved in hydrophobic contact(s)

Figure 5.7: Control and Analogue's interactions with receptor found trough docking study.

Vandetanib_120 and Vadetanib_30 have a stronger affinity than the original Vandetanib structure (as shown in Figure 5.7 and Table 5.2: The interaction of specific amino acid residues within the RET tyrosine kinase domain with Vandetanib and its Analogues.), possibly because they form two hydrogen bonds while the control only forms one. To confirm the reliability of these findings, the complexes were also evaluated through molecular dynamic simulations to assess them based on dynamic parameters in addition to the static parameters obtained through docking.

Table 5.2: The interaction of specific amino acid residues within the RET tyrosine kinase domain with Vandetanib and its Analogues.

Complexes	**Interactions of RET tyrosine kinase domain residues with the Vandetanib and its Analogues**		
	Hydrophobic Interactions	**Hydrogen Bonds**	**Water Bridges**
Vandetanib_Control	LEU730, VAL 738, LYS758, VAL804, TYR806, LEU881	ALA807	LYS728, SER811
Vandetanib_120	LEU730, VAL738, LYS758, ILE788,	SER891, ASP892	-

	VAL804, LEU881		
Vandetanib_33	PHE735, VAL738, ALA756, LEU 881	SER891, ASP892	-

4. Dynamic Stability

Molecular dynamics simulation is a useful method for studying the behavior of docked complexes over a set period of time by examining the interactions of the molecules [440]. The simulation includes the use of root-mean-square deviation (RMSD) to compare the simulation snapshots to the initial docked frame. The stability of the protein structure was analyzed by measuring the movement of the carbon alpha atoms in nanometers over time. The results of the simulation in Figure 5.8 showed that the RMSD values of the complexes seemed to stabilize within 10 nanoseconds, indicating stability in the complexes. The stability of the ligand atoms was also analyzed over a 10-nanosecond trajectory (Figure 5.9), and it was found that the Vandetanib_Control and Vandetanib_120 complexes had overlapping RMSD values, while the Vadetanib_30 complex had lower values, potentially indicating that the Vadetanib_30 Analogue is more stable over time.

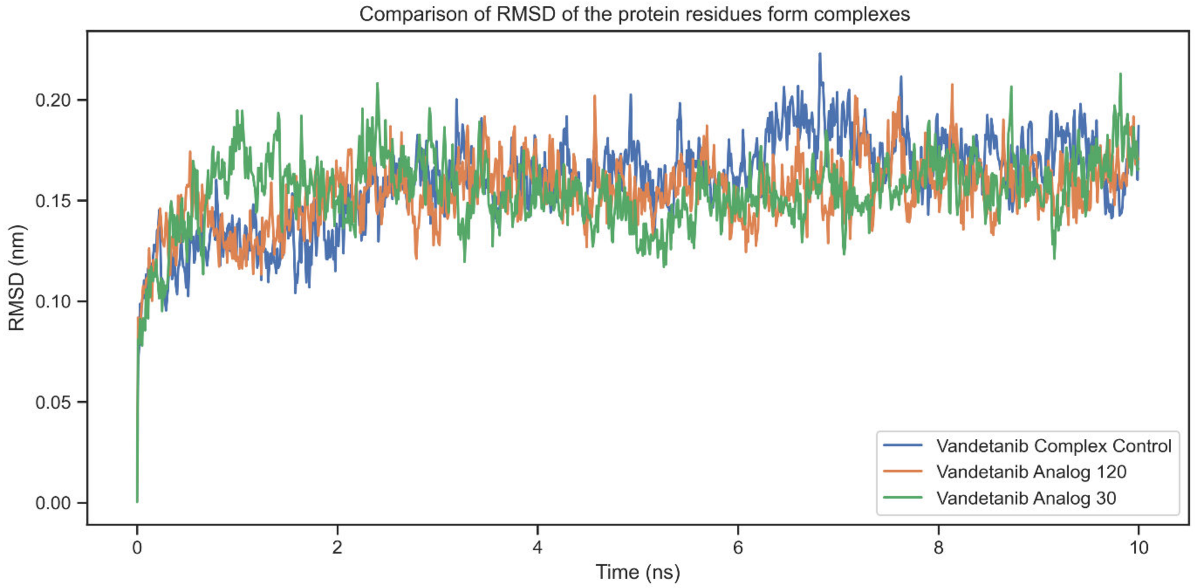


Figure 5.8: Receptor's c-alpha backbone RMSD form all the three complexs.

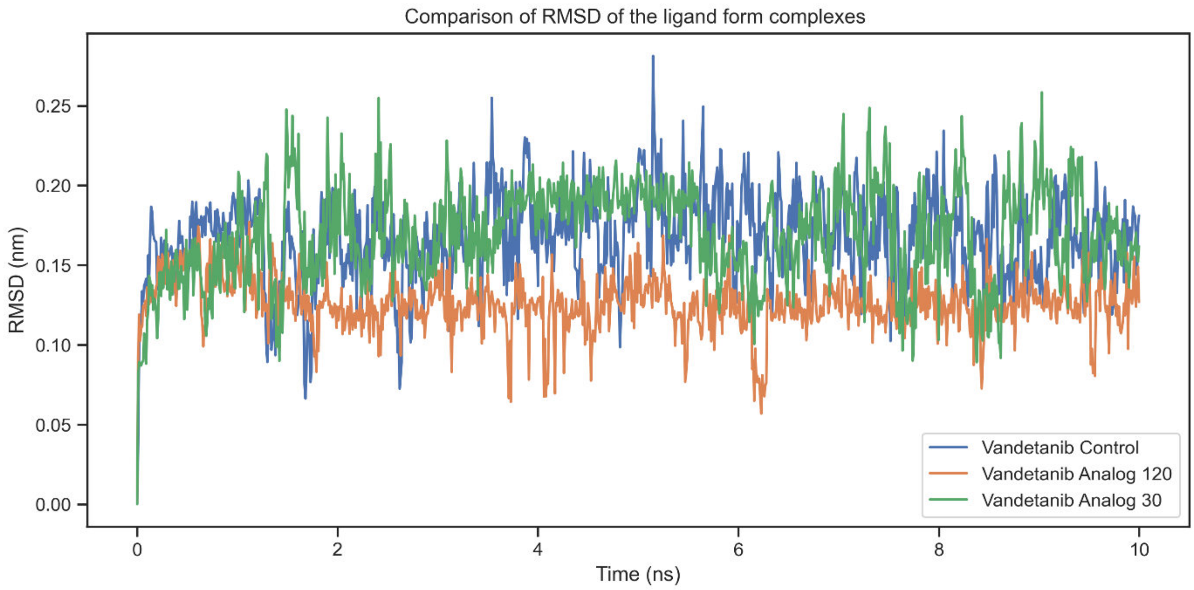


Figure 5.9: Ligand's c-alpha backbone RMSD form all the three complexs.

To emphasize atomic-level interactions, the complexes free binding energies for MM–GBSA and MM–PBSA were computed. A total of 250 frames (last 5 ns) were processed and analyzed to determine the free binding energies of MM–GBSA and MM–PBSA. Van der Waals contribution was found to be the most significant in gas phase, followed by total solvation energy and Electrostatic contribution.

The overall, net binding energies of all the systems showed favorable binding as shown in Table 5.3: MM-GBSA and MM-PBSA analysis of the complexes. The Analogues complex has a higher negative free binding energy than the control Vandetanib complex, indicating that the Analogues complex has stronger interactions.

Table 5.3: MM-GBSA and MM-PBSA analysis of the complexes

MM-GBSA Method					
Complex name	**EEL**	**VDW**	**ESURF**	**EGB**	**ΔGBSA**
Vandetanib_Control	-0.61 ± 0.04	-20.35 ± 0.14	-2.38 ± 0.02	1.84 ± 0.03	-21.50 ± 0.15
Vandetanib_120	-1.34 ± 0.03	-20.39 ± 0.14	-2.48 ± 0.02	2.43 ± 0.03	-21.78 ± 0.15
Vandetanib_33	-1.23 ± 0.03	-21.62 ± 0.25	-2.79 ± 0.03	2.63 ± 0.03	-23.01 ± 0.27
MM-PBSA Method					

Complex name	EEL	VDW	ENPOLAR	EPB	ΔPBSA
Vandetanib_Control	-6.12 ± 0.38	-20.35 ± 0.14	-2.64 ± 0.02	18.10 ± 0.31	-11.01 ± 0.18
Vandetanib_120	-13.41 ± 0.30	-20.39 ± 0.14	-2.61 ± 0.01	22.62 ± 0.32	-13.74 ± 0.17
Vandetanib_33	-12.13 ± 0.26	-21.62 ± 0.25	-2.90 ± 0.03	24.76 ± 0.37	-12.06 ± 0.21

v. Discussions and Conclusions

Designing drug analogues entails creating modified versions of existing drugs to enhance their efficacy, stability, and safety. Typically, this involves modifying the chemical structure of the original drug to improve its therapeutic properties. One reason for this is to address the limitations of current drugs, such as their short half-life, which usually requires frequent dosing. By modifying the drug's chemical structure, it may be possible to extend the drug's half-life and reduce the frequency of dosing.

De novo molecule generation involves the design and synthesis of entirely new molecules. Predicting the properties and behaviour of a molecule based on its chemical structure is a challenging task. De novo molecule generation includes computational methods that employ computer algorithms and models to design and optimise the structure of the molecule, as well as experimental methods that involve synthesising and testing the molecule in the laboratory[441]. In recent years, there has been renewed interest in using deep learning architectures to generate new compounds, particularly for high-throughput screening library generation, hit optimization, and fragment-based hit discovery. However, the number of studies that have applied these methods to drug discovery is still low compared to the number of studies that have focused on algorithm development[442]. This study developed a pipeline for the creation of new drug analogues using autoencoders, a technique of Deep Learning. A chemical structure-reproducing autoencoder was trained on various chemical SMILES from the ChEMBL database. The distribution of the autoencoder's latent space was controlled by using batch normalization layer and optimised using varying batch sizes. The latent space of an autoencoder is a compact numerical representation of the input data, and its distribution can affected by batch normalization and the batch size. Without batch normalization the latent space may have dispersed (Figure 5.3).

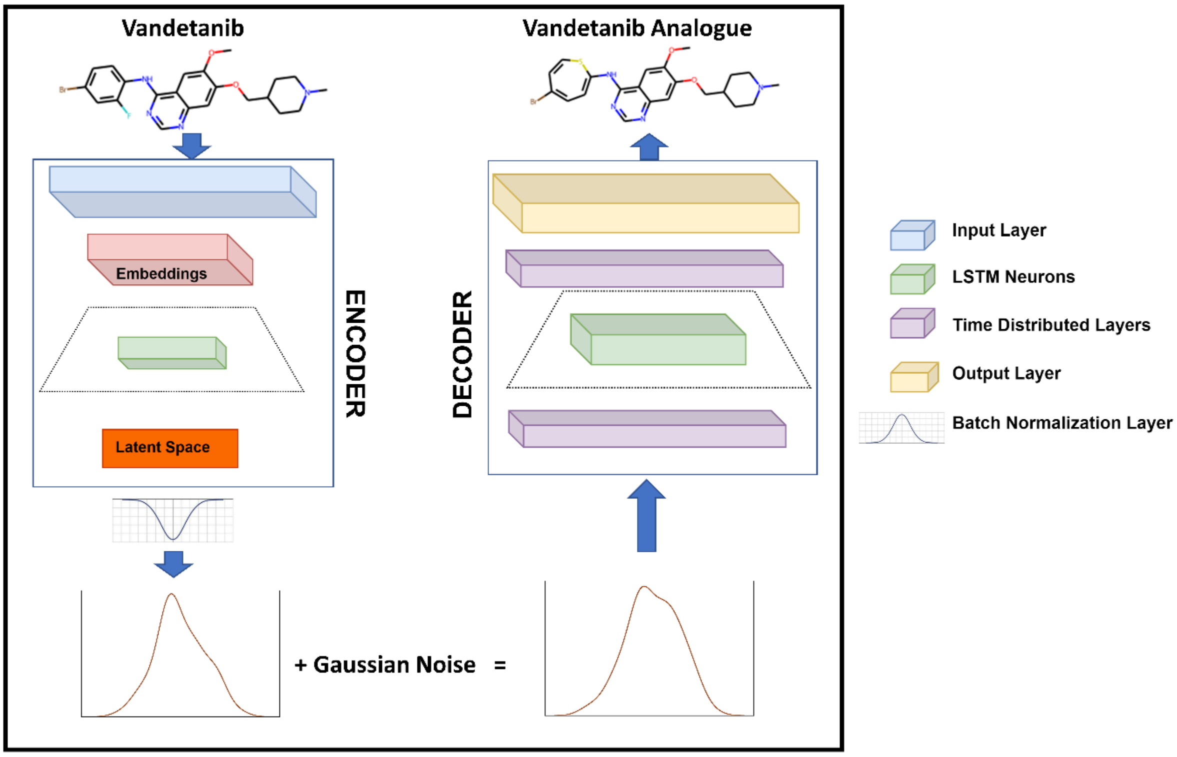


Figure 5.10: Generation of Analogues using proposed model.

As depicted in Figure 5.10, the autoencoder was used to generate 157 variants/Analogues of Vandetanib by adding noise to its bell-shaped latent representation and reconstructing the resulting compounds using the decoder. Molecular docking and dynamics simulations were performed to determine which of these analogues possessed a higher binding affinity than Vandetanib. At least two of the analogues had a higher binding affinity than the control compound, according to the results.

The current model only has about 0.42 million parameters, but it can be used to create a wide range of molecules. By adjusting the amount of noise added to the latent representation, it is possible to fine-tune the level of variation in the generated chemicals. For instance, the addition of a small amount of noise results in subtle changes, whereas the addition of a large amount of noise results in more dramatic changes. The amount of noise can be controlled by varying the mean and standard deviation of the gaussian noise. It may be possible in the future to combine this pipeline with genetic algorithms for fragment-based de novo drug generation. The model is intended to generate molecules with SMILES strings of up to 100 characters in length, but it may have difficulty generating molecules with SMILES strings longer than 80 characters due to a lack of training data. Potentially, this limitation could be overcome by increasing the availability of such data. The synthesis and laboratory testing of the generated molecules to determine their potential as drugs represents an additional potential difficulty in this procedure. This difficulty can be overcome by enhancing models for generating molecules and proposing synthesis techniques. Despite the challenges that must be addressed, this study has the potential to make significant contributions to the field of automatic drug analogue prediction and could be a significant addition to the existing scientific literature. The model was implemented and shared in a Google Colaboratory notebook for scientific community exploration and evaluation at https://colab.research.google.com/drive/1BPhw7_-_VV11dbk6s9JGE0bSX0K_-qIh?usp=sharing

CHAPTER VI

Summary and Future Prospects

Chapter 6. Summary and Future Prospects

This thesis aims to predict microRNA sequences from mRNA sequences, predict protein disorder regions, and generate new drug analogues by exploring chemical space using machine learning and natural language processing techniques. Convolutional neural networks (CNNs), long short-term memory (LSTM) networks, and autoencoders were utilised to accomplish these goals. It was observed that the models performed well on the tasks for which they were created, with the microRNA prediction model achieving an average accuracy of 72% and the protein disorder prediction model outperforming other state-of-the-art techniques. The model for generating drug analogues using an autoencoder was able to generate new molecules with the desired properties. Overall, we demonstrated the applicability of machine learning and natural language processing techniques to the field of genomic, proteomic, and chemical sequence analysis.In addition to the primary objectives stated, several additional experiments were conducted to evaluate and enhance the performance of the models. For the microRNA prediction model, the effect of utilising various CNN architectures was evaluated, and it was discovered that 1D CNN with LSTM layers enhanced performance. The use of various input sequence lengths was also investigated, and it was discovered that longer input sequences led to improved performance. For the protein disorder prediction model, we examined the effect of various embedding sizes and found that larger embedding sizes improved performance. Different CNN architectures were also evaluated, and it was discovered that using depthwise separable 1D CNNs, which have fewer weights than standard 1D CNNs, increased the model's speed without sacrificing accuracy. For the generation model of drug analogues, the effect of using different latent space sizes and distributions was evaluated, and it was determined that bell-shaped spaces produced more similar and chemically valid analogues. These additional experiments demonstrate the significance of selecting model architectures and hyperparameters with care in order to achieve good performance on these tasks.

The purpose of developing biological language models (BLMs) is to enhance our capacity to comprehend and analyse biological sequences, such as DNA, RNA, and protein sequences. These sequences contain crucial information about the structure and function of living organisms and are involved in virtually every biological process. Nonetheless, analysing biological sequences can be difficult due to their complexity and enormous potential. Specifically, the functions and properties of a large number of coding and non-coding DNA and RNA sequences remain poorly understood.

Using natural language processing (NLP) techniques, which are used to comprehend and analyse human languages, is one method for enhancing our comprehension of biological sequences. Biological sequences have structure and convey meaning, much like natural languages. For instance, the manner in which amino acids are connected by peptide bonds determines the structure and function of a protein, just as grammar and linguistic rules determine the structure and meaning of a sentence. Therefore, techniques based on linguistics have significantly contributed to the field of biological sequence analysis and have been useful for comprehending the meaning of the genome. However, natural languages and biological sequences also have significant differences. For example, amino acids have over 500 physiochemical properties, and nucleotides have over 180. This exceeds the complexity of even the most polysemous word in a natural language. Therefore, traditional rule-based approaches may not be able to fully capture the complexity of biological sequences and may have limited performance for certain tasks, such as the prediction of disordered protein regions and the identification of enhancers. To overcome these obstacles, researchers have developed BLMs, which are deep learning models designed to represent and analyse biological sequences using techniques similar to those used in natural language processing (NLP).BLMs' ability to capture complex patterns and relationships within biological sequences is one of their primary advantages. Traditional methods for analysing biological sequences frequently rely on predefined rules or heuristics that may be incapable of detecting subtle or complex patterns within the data. In contrast, BLMs can learn these patterns directly from the data, enabling them to identify relationships that may not have been detectable using other methods. This can be especially useful for tasks such as protein function prediction, in which the relationship between sequence and function is frequently unclear and complex. The annotation of biological sequences is yet another potential application of BLMs. Annotation is the process of labelling or tagging specific portions of a sequence in order to provide additional information about its function or structure. This is a crucial step in the analysis of biological sequences because it allows scientists to better comprehend the sequence and its function within the organism. Nevertheless, manual annotation is time-consuming and resource-intensive, especially for large datasets. It may be possible to automatically predict the function or structure of a new, unannotated sequence by training a BLM on a large dataset of annotated sequences. This could save time and resources by eliminating the need for manual annotation and enabling the rapid analysis of massive datasets. BLMs can also be used to identify relationships and patterns within biological sequences. For example, they can be used to identify motifs, which are repeating patterns of characters within a sequence, or to identify conserved regions, which are sections of the sequence that are highly similar across different organisms. BLMs can also be used to extract information from scientific literature or other text sources related to the biological sequences being analysed.

Biological language models (BLMs) can also be applied to the analysis of chemical compounds through the Simplified Molecular-Input Line-Entry System (SMILES) notation. SMILES is a computer-processable language used to represent the structural formula of a chemical compound. It is a series of symbols and letters representing the atoms and bonds in a molecule. BLMs can be taught to analyse SMILES-represented chemical compounds in a variety of ways.

BLMs can be used to predict the solubility, toxicity, and reactivity of chemical compounds, among other properties. The properties of chemical compounds play a significant role in their potential uses and applications.

BLMs can also be used to predict the structure of chemical compounds. This is significant because the structure of a chemical compound determines its physical and chemical properties, and understanding these properties enables researchers to predict how the compound will behave in various environments. To create BLMs for chemical compounds represented in SMILES notation, researchers will need to employ techniques comparable to those used to create BLMs for DNA, RNA, and protein sequences. This may involve utilising machine learning algorithms and bioinformatics tools to train the models, in addition to obtaining data from chemical databases and other resources. BLMs may be able to provide valuable insights and facilitate the discovery of new biological knowledge by treating biological sequences as natural language and utilising deep learning techniques. The use of BLMs in biological sequence analysis has the potential to significantly advance our knowledge of the genome and its encoded functions and structures. By treating biological sequences as natural language and applying techniques developed for understanding and analysing text, we may be able to gain a deeper understanding of the information encoded in these sequences and use it to make significant scientific discoveries.

CHAPTER VII

REFERENCES

Chapter 7. References

[1] J. R. Searle, “The future of philosophy,” *Philos Trans R Soc Lond B Biol Sci*, vol. 354, no. 1392, pp. 2069–2080, Dec. 1999, doi: 10.1098/RSTB.1999.0544.

[2] S. Soames, “The Changing Role of Language in Analytic Philosophy,” *Analytic Philosophy*, pp. 34–51, Sep. 2017, doi: 10.4324/9781315733050-3.

[3] “Speech Act Theory and Pragmatics,” *Speech Act Theory and Pragmatics*, 1980, doi: 10.1007/978-94-009-8964-1.

[4] D. C. Blair, “Information Retrieval and the Philosophy of Language,” *Comput J*, vol. 35, no. 3, pp. 200–207, Jun. 1992, doi: 10.1093/COMJNL/35.3.200.

[5] G. Finch, M. Coyle, and J. Peck, *How to study linguistics: A guide to understanding language*. Bloomsbury Publishing, 2017.

[6] J. Aitchison, *The seeds of speech: Language origin and evolution*. Cambridge University Press, 2000.

[7] U. Bellugi and S. Fischer, “A comparison of sign language and spoken language,” *Cognition*, vol. 1, no. 2–3, pp. 173–200, Jan. 1972, doi: 10.1016/0010-0277(72)90018-2.

[8] R. L. Trask, *Language: the basics*. Routledge, 2003.

[9] D. H. Klatt, “Linguistic uses of segmental duration in English: Acoustic and perceptual evidence,” *J Acoust Soc Am*, vol. 59, no. 5, p. 1208, Jun. 1998, doi: 10.1121/1.380986.

[10] J. W. F. Mulder, “ON THE ART OF DEFINITION, THE DOUBLE ARTICULATION OF LANGUAGE, AND SOME OF THE CONSEQUENCES,” *Forum for Modern*

Language Studies, vol. V, no. 2, pp. 103–117, Apr. 1969, doi: 10.1093/FMLS/V.2.103.

[11] G. Jucquois, "Language and Communication among Hominids," *http://dx.doi.org/10.1177/0392192107078774*, vol. 54, no. 2, pp. 60–80, Jul. 2016, doi: 10.1177/0392192107078774.

[12] P. Lieberman, *Uniquely human: The evolution of speech, thought, and selfless behavior*. Harvard University Press, 1991.

[13] B. Hart and T. R. Risley, "The early catastrophe: The 30 million word gap by age 3," *American educator*, vol. 27, no. 1, pp. 4–9, 2003.

[14] A. Clark, C. Fox, and S. Lappin, *The handbook of computational linguistics and natural language processing*, vol. 118. John Wiley & Sons, 2012.

[15] D. Khurana, A. Koli, K. Khatter, and S. Singh, "Natural language processing: state of the art, current trends and challenges," *Multimed Tools Appl*, pp. 1–32, Jul. 2022, doi: 10.1007/S11042-022-13428-4/FIGURES/3.

[16] M. Johnson *et al.*, "Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation," *Trans Assoc Comput Linguist*, vol. 5, pp. 339–351, Dec. 2017, doi: 10.1162/TACL_A_00065/43400.

[17] A. Conneau and G. Lample, "Cross-lingual language model pretraining," *Adv Neural Inf Process Syst*, vol. 32, 2019.

[18] J. Cowie and W. Lehnert, "Information Extraction," *Commun ACM*, vol. 39, no. 1, pp. 80–91, 1996, doi: 10.1145/234173.234209.

[19] J. Pustejovsky and A. Stubbs, *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*. 2012. Accessed: Jan. 02, 2023. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=A57TS7fs8MUC&oi=fnd&pg=PR2&dq=The+early+interest+in+understanding+data+patterns,+Parts-of-

Speech+(POS)+labelling,+and+simplifying+data+processing+for+a+variety+of+applications+in+the+banking+and+financial+industries,+educational+institutions,+and+so+on+gave+rise+to+the+field+of+computa&ots=SKhtyC-ByF&sig=902ziZdojL6k3ktDj7SSMfs_bRE

[20] J. Hirschberg and C. D. Manning, "Advances in natural language processing," *Science (1979)*, vol. 349, no. 6245, pp. 261–266, Jul. 2015, doi: 10.1126/SCIENCE.AAA8685.

[21] A. Kumar, T. S. I. S. and Applications, and undefined 2012, "Sentiment analysis: A perspective on its past, present and future," *mecs-press.com*, vol. 10, pp. 1–14, 2012, doi: 10.5815/ijisa.2012.10.01.

[22] M. El-Haj, P. Rayson, M. Walker, S. Young, and V. Simaki, "In search of meaning: Lessons, resources and next steps for computational analysis of financial discourse," *Wiley Online Library*, vol. 46, no. 3–4, pp. 265–306, Mar. 2019, doi: 10.1111/jbfa.12378.

[23] S. Weiss, N. Indurkhya, and T. Zhang, *Fundamentals of predictive text mining*. 2015. Accessed: Jan. 02, 2023. [Online]. Available: https://link.springer.com/content/pdf/10.1007/978-1-4471-6750-1.pdf

[24] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. 2009. Accessed: Jan. 02, 2023. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=KGIbfiiP1i4C&oi=fnd&pg=PR5&dq=Natural+language+processing+uses+syntactic+and+semantic+analysis+to+instruct+machines+by+uncovering+and+recognising+data+patterns.+The+following+are+the+steps+involved&ots=Y4FhC1LAJ4&sig=Kt8b9MLEXE3yF8poZxSZ_JImJMs

[25] K. R. Chowdhary, "Natural Language Processing," *Fundamentals of Artificial Intelligence*, pp. 603–649, 2020, doi: 10.1007/978-81-322-3972-7_19.

[26] N. Hardeniya, J. Perkins, D. Chopra, N. Joshi, and I. Mathur, *Natural language processing: python and NLTK*. 2016. Accessed: Jan. 02, 2023. [Online]. Available: https://books.google.com/books?hl=en&lr=&id=0J_cDgAAQBAJ&oi=fnd&pg=PP1&dq=Natural+language+processing+uses+syntactic+and+semantic+analysis+to+instruct+machines+by+uncovering+and+recognising+data+patterns.+The+following+are+the+steps+involved&ots=lfvryYotXW&sig=eDu6rIqGOUhuppjnuJGq_QWnqlY

[27] U. Kelle, "Theory building in qualitative research and computer programs for the management of textual data," *Sociol Res Online*, vol. 2, no. 2, 1997, doi: 10.5153/SRO.86.

[28] E. de Vries, K. Lund, M. B.-T. journal of the learning sciences, and undefined 2002, "Computer-mediated epistemic dialogue: Explanation and argumentation as vehicles for understanding scientific notions," *Taylor & Francis*, vol. 11, no. 1, pp. 63–103, 2002, doi: 10.1207/S15327809JLS1101_3.

[29] R. G.-J.-J. of E. T. & Society and undefined 2019, "In a world of SMART technology, why learn another language?," *JSTOR*, Accessed: Jan. 02, 2023. [Online]. Available: https://www.jstor.org/stable/26819613

[30] W. H. Hsiao and T. S. Chang, "Exploring the opportunity of digital voice assistants in the logistics and transportation industry," *Journal of Enterprise Information Management*, vol. 32, no. 6, pp. 1034–1050, Oct. 2019, doi: 10.1108/JEIM-12-2018-0271/FULL/HTML.

[31] D. Yoffie, L. Wu, J. Sweitzer, D. E.-… B. S. Case, and undefined 2018, "Voice war: Hey google vs. alexa vs. siri," *picture.iczhiku.com*, 2018, Accessed: Jan. 02, 2023. [Online]. Available: https://picture.iczhiku.com/resource/paper/sYKdjjDyORlQlCvB.pdf

[32] R. D.-N. L. Engineering and undefined 2020, "Voice assistance in 2019,"

cambridge.org, vol. 26, pp. 129–136, 2020, doi: 10.1017/S1351324919000640.

[33] R.-P. Ashir Ahmed, "Use of social media in disaster management," 2011, Accessed: Jan. 02, 2023. [Online]. Available: https://aisel.aisnet.org/icis2011/proceedings/generaltopics/16/

[34] A. Amara, M. A. Hadj Taieb, and M. ben Aouicha, "Multilingual topic modeling for tracking COVID-19 trends based on Facebook data analysis," *Applied Intelligence*, vol. 51, no. 5, pp. 3052–3073, May 2021, doi: 10.1007/S10489-020-02033-3.

[35] C. Wendling, J. Radisch, and S. Jacobzone, "The use of social media in risk and crisis communication," no. 24, 2013, doi: 10.1787/5k3v01fskp9s-en.

[36] A. Saroj, S. P.-I. J. of D. R. Reduction, and undefined 2020, "Use of social media in crisis management: A survey," *Elsevier*, Accessed: Jan. 02, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S221242091931684X

[37] B. Pahwa, N. Kasliwal, R. Scholar, B. Vidyapith, and R. S. Taruna, "Sentiment analysis-strategy for text pre-processing," *researchgate.net*, vol. 180, no. 34, pp. 975–8887, 2018, doi: 10.5120/ijca2018916865.

[38] E. Haddi, X. Liu, Y. S.-P. computer science, and undefined 2013, "The role of text pre-processing in sentiment analysis," *Elsevier*, Accessed: Jan. 02, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050913001385

[39] D. Jani, N. Patel, H. Yadav, S. Suthar, and S. Patel, "A Concise Review on Automatic Text Summarization," *Smart Innovation, Systems and Technologies*, vol. 281, pp. 523–536, 2022, doi: 10.1007/978-981-16-9447-9_40.

[40] L. Hickman, S. Thapa, L. Tay, M. Cao, and P. Srinivasan, "Text preprocessing for text mining in organizational research: Review and recommendations," *journals.sagepub.com*, vol. 25, no. 1, pp. 114–146, Jan. 2022, doi: 10.1177/1094428120971683.

[41] A. Gasparetto, M. Marcuzzo, A. Zangari, A. A.- Information, and undefined 2022, "A Survey on Text Classification Algorithms: From Text to Predictions," *mdpi.com*, 2022, doi: 10.3390/info13020083.

[42] V. Kadam, … K. K.-C. on S. C. and its, and undefined 2022, "Text Analysis and Classification for Preprocessing Phase of Automatic Text Summarization Systems," *Springer*, Accessed: Jan. 02, 2023. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-05767-0_30

[43] K. al Sharou, Z. Li, and L. Specia, "Towards a better understanding of noise in natural language processing," in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, 2021, pp. 53–62.

[44] A. S. Nayak, A. P. Kanive, N. Chandavekar, and R. Balasubramani, "Survey on pre-processing techniques for text mining," *International Journal of Engineering and Computer Science*, vol. 5, no. 6, pp. 16875–16879, 2016.

[45] S. Kannan *et al.*, "Preprocessing techniques for text mining," *International Journal of Computer Science & Communication Networks*, vol. 5, no. 1, pp. 7–16, 2014.

[46] W. A. Qader, M. M. Ameen, and B. I. Ahmed, "An Overview of Bag of Words;Importance, Implementation, Applications, and Challenges," *Proceedings of the 5th International Engineering Conference, IEC 2019*, pp. 200–204, Jun. 2019, doi: 10.1109/IEC47844.2019.8950616.

[47] Y. Zhang, R. Jin, and Z. H. Zhou, "Understanding bag-of-words model: A statistical framework," *International Journal of Machine Learning and Cybernetics*, vol. 1, no. 1–4, pp. 43–52, Dec. 2010, doi: 10.1007/S13042-010-0001-0/FIGURES/2.

[48] L. Havrlant and V. Kreinovich, "A simple probabilistic explanation of term frequency-inverse document frequency (tf-idf) heuristic (and variations motivated by this explanation)," *http://dx.doi.org/10.1080/03081079.2017.1291635*, vol. 46, no. 1, pp.

27–36, Jan. 2017, doi: 10.1080/03081079.2017.1291635.

[49] T. Peng, L. Liu, and W. Zuo, "PU text classification enhanced by term frequency–inverse document frequency-improved weighting," *Concurr Comput*, vol. 26, no. 3, pp. 728–741, Mar. 2014, doi: 10.1002/CPE.3040.

[50] N. H. Cahyana, S. Saifullah, Y. Fauziah, A. S. Aribowo, and R. Drezewski, "Semi-supervised Text Annotation for Hate Speech Detection using K-Nearest Neighbors and Term Frequency-Inverse Document Frequency," *Int. J. Adv. Comput. Sci. Appl*, vol. 13, no. 10, pp. 147–151, 2022.

[51] C. Periñán-Pascual, "Measuring associational thinking through word embeddings," *Artif Intell Rev*, vol. 55, no. 3, pp. 2065–2102, Mar. 2022, doi: 10.1007/S10462-021-10056-6/TABLES/10.

[52] N. Oubenali, S. Messaoud, A. Filiot, A. Lamer, and P. Andrey, "Visualization of medical concepts represented using word embeddings: a scoping review," *BMC Med Inform Decis Mak*, vol. 22, no. 1, pp. 1–14, Dec. 2022, doi: 10.1186/S12911-022-01822-9/TABLES/2.

[53] S. A. B. Andrabi and A. Wahid, "A Comparative Study of Word Embedding Techniques in Natural Language Processing," pp. 701–712, 2022, doi: 10.1007/978-981-16-9573-5_50.

[54] M. A. Haq, M. A. R. Khan, and M. Alshehri, "Insider Threat Detection Based on NLP Word Embedding and Machine Learning," *Intelligent Automation & Soft Computing*, vol. 33, no. 1, pp. 619–635, Jan. 2022, doi: 10.32604/IASC.2022.021430.

[55] G. Alipour, J. Bagherzadeh Mohasefi, and M. R. Feizi-Derakhshi, "Learning Bilingual Word Embedding Mappings with Similar Words in Related Languages Using GAN," *https://doi.org/10.1080/08839514.2021.2019885*, vol. 36, no. 1, 2022, doi: 10.1080/08839514.2021.2019885.

[56] A. Montejo-Ráez and S. M. Jiménez-Zafra, “Current Approaches and Applications in Natural Language Processing,” *Applied Sciences 2022, Vol. 12, Page 4859*, vol. 12, no. 10, p. 4859, May 2022, doi: 10.3390/APP12104859.

[57] S. Pudasaini, S. Shakya, S. Lamichhane, S. Adhikari, A. Tamang, and S. Adhikari, “Application of NLP for Information Extraction from Unstructured Documents,” *Lecture Notes in Networks and Systems*, vol. 209, pp. 695–704, 2022, doi: 10.1007/978-981-16-2126-0_54/COVER.

[58] J. Passmore and T. S. Rowson, “Neuro-linguistic-programming: a critical review of NLP research and the application of NLP in coaching,” 2019.

[59] S. M. Mathews, “Explainable Artificial Intelligence Applications in NLP, Biomedical, and Malware Classification: A Literature Review,” *Advances in Intelligent Systems and Computing*, vol. 998, pp. 1269–1292, 2019, doi: 10.1007/978-3-030-22868-2_90/COVER.

[60] A. F. Smeaton, “Progress in the Application of Natural Language Processing to Information Retrieval Tasks,” *Comput J*, vol. 35, no. 3, pp. 268–278, Jun. 1992, doi: 10.1093/COMJNL/35.3.268.

[61] E. Sarioglu, H.-A. Choi, and K. Yadav, “Clinical report classification using natural language processing and topic modeling,” in *2012 11th International Conference on Machine Learning and Applications*, 2012, vol. 2, pp. 204–209.

[62] P. M. Lavanya and E. Sasikala, “Deep learning techniques on text classification using Natural language processing (NLP) in social healthcare network: A comprehensive survey,” in *2021 3rd International Conference on Signal Processing and Communication (ICPSC)*, 2021, pp. 603–609.

[63] W. Khan, A. Daud, J. A. Nasir, and T. Amjad, “A survey on the state-of-the-art machine learning models in the context of NLP,” *Kuwait journal of Science*, vol. 43, no. 4, 2016.

[64] T. Kanan *et al.*, “A review of natural language processing and machine learning tools used to analyze Arabic social media,” *2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology, JEEIT 2019 - Proceedings*, pp. 622–628, May 2019, doi: 10.1109/JEEIT.2019.8717369.

[65] D. Ofer, N. Brandes, and M. Linial, “The language of proteins: NLP, machine learning & protein sequences,” *Comput Struct Biotechnol J*, vol. 19, pp. 1750–1758, Jan. 2021, doi: 10.1016/J.CSBJ.2021.03.022.

[66] E. H. Houssein, R. E. Mohamed, and A. A. Ali, “Machine Learning Techniques for Biomedical Natural Language Processing: A Comprehensive Review,” *IEEE Access*, vol. 9, pp. 140628–140653, 2021, doi: 10.1109/ACCESS.2021.3119621.

[67] Y. Hasija and R. Chakraborty, “Machine Learning and Linear Regression,” *Hands-On Data Science for Biologists Using Python*, pp. 161–181, Feb. 2021, doi: 10.1201/9781003090113-8-8/MACHINE-LEARNING-LINEAR-REGRESSION-YASHA-HASIJA-RAJKUMAR-CHAKRABORTY.

[68] B. Liu, “Supervised Learning,” *Web Data Mining*, pp. 63–132, 2011, doi: 10.1007/978-3-642-19460-3_3.

[69] Y. Hasija and R. Chakraborty, “Hands-On Data Science for Biologists Using Python,” *Hands-On Data Science for Biologists Using Python*, Apr. 2021, doi: 10.1201/9781003090113.

[70] J. E. van Engelen and H. H. Hoos, “A survey on semi-supervised learning,” *Mach Learn*, vol. 109, no. 2, pp. 373–440, Feb. 2020, doi: 10.1007/S10994-019-05855-6/FIGURES/5.

[71] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[72] P. Flach, “Performance Evaluation in Machine Learning: The Good, the Bad, the Ugly, and the Way Forward,” *Proceedings of the AAAI Conference on Artificial Intelligence*,

vol. 33, no. 01, pp. 9808–9814, Jul. 2019, doi: 10.1609/AAAI.V33I01.33019808.

[73] G. Bernieri, M. Conti, and F. Turrin, "Evaluation of Machine Learning Algorithms for Anomaly Detection in Industrial Networks," *2019 IEEE International Symposium on Measurements and Networking, M and N 2019 - Proceedings*, Jul. 2019, doi: 10.1109/IWMN.2019.8805036.

[74] B. Gaye and A. Wulamu, "Sentiment Analysis of Text Classification Algorithms Using Confusion Matrix," *Communications in Computer and Information Science*, pp. 231–241, 2019, Accessed: Jan. 03, 2023. [Online]. Available: https://www.academia.edu/91163050/Sentiment_Analysis_of_Text_Classification_Algorithms_Using_Confusion_Matrix

[75] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit*, vol. 30, no. 7, pp. 1145–1159, Jul. 1997, doi: 10.1016/S0031-3203(96)00142-2.

[76] D. Berrar, "Cross-Validation," *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, vol. 1–3, pp. 542–545, Jan. 2019, doi: 10.1016/B978-0-12-809633-8.20349-X.

[77] G. I. Webb *et al.*, "Leave-One-Out Cross-Validation," *Encyclopedia of Machine Learning*, pp. 600–601, 2011, doi: 10.1007/978-0-387-30164-8_469.

[78] "Encyclopedia of Machine Learning," *Encyclopedia of Machine Learning*, 2010, doi: 10.1007/978-0-387-30164-8.

[79] D. M. Belete and M. D. Huchaiah, "Grid search in hyperparameter optimization of machine learning models for prediction of HIV/AIDS test results," *International Journal of Computers and Applications*, vol. 44, no. 9, pp. 875–886, 2022, doi: 10.1080/1206212X.2021.1974663.

[80] J. Bergstra and Y. Bengio, "Random Search for Hyper-Parameter Optimization,"

Journal of Machine Learning Research, vol. 13, no. 10, pp. 281–305, 2012, [Online]. Available: http://jmlr.org/papers/v13/bergstra12a.html

[81] T. G. Dietterich, "Ensemble methods in machine learning," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 1857 LNCS, pp. 1–15, 2000, doi: 10.1007/3-540-45014-9_1/COVER.

[82] A. L'Heureux, K. Grolinger, H. F. Elyamany, and M. A. M. Capretz, "Machine Learning with Big Data: Challenges and Approaches," *IEEE Access*, vol. 5, pp. 7776–7797, 2017, doi: 10.1109/ACCESS.2017.2696365.

[83] Maciej Serda *et al.*, "How to avoid machine learning pitfalls: a guide for academic researchers," *Uniwersytet śląski*, vol. 7, no. 1, pp. 343–354, Aug. 2021, doi: 10.2/JQUERY.MIN.JS.

[84] L. H. Clemmensen and R. D. Kjærsgaard, "Data Representativity for Machine Learning and AI Systems," Mar. 2022, doi: 10.48550/arxiv.2203.04706.

[85] G. Fenza, M. Gallo, V. Loia, F. Orciuoli, and E. Herrera-Viedma, "Data set quality in Machine Learning: Consistency measure based on Group Decision Making," *Appl Soft Comput*, vol. 106, p. 107366, Jul. 2021, doi: 10.1016/J.ASOC.2021.107366.

[86] R. C. Chen, C. Dewi, S. W. Huang, and R. E. Caraka, "Selecting critical features for data classification based on machine learning methods," *J Big Data*, vol. 7, no. 1, pp. 1–26, Dec. 2020, doi: 10.1186/S40537-020-00327-4/FIGURES/13.

[87] D. Bashir, G. D. Montañez, S. Sehra, P. S. Segura, and J. Lauw, "An Information-Theoretic Perspective on Overfitting and Underfitting," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12576 LNAI, pp. 347–358, 2020, doi: 10.1007/978-3-030-64984-5_27/COVER.

[88] A. K. Jain, J. Mao, and K. M. Mohiuddin, “Artificial neural networks: A tutorial,” *Computer (Long Beach Calif)*, vol. 29, no. 3, pp. 31–44, Mar. 1996, doi: 10.1109/2.485891.

[89] S. Ray, “A Quick Review of Machine Learning Algorithms,” *Proceedings of the International Conference on Machine Learning, Big Data, Cloud and Parallel Computing: Trends, Prespectives and Prospects, COMITCon 2019*, pp. 35–39, Feb. 2019, doi: 10.1109/COMITCON.2019.8862451.

[90] D. M. D’Addona, “Neural Network,” *CIRP Encyclopedia of Production Engineering*, pp. 911–918, 2014, doi: 10.1007/978-3-642-20617-7_6563.

[91] G.-J. Wang, S.-Y. Li, J.-Q. Xia, J. Feng, and S. Lu, “Performance Analysis of Various Activation Functions in Artificial Neural Networks,” *J Phys Conf Ser*, vol. 1237, no. 2, p. 022030, Jun. 2019, doi: 10.1088/1742-6596/1237/2/022030.

[92] C. Bircanoglu and N. Arica, “A comparison of activation functions in artificial neural networks,” *26th IEEE Signal Processing and Communications Applications Conference, SIU 2018*, pp. 1–4, Jul. 2018, doi: 10.1109/SIU.2018.8404724.

[93] P. Baldi, “Gradient Descent Learning Algorithm Overview: A General Dynamical Systems Perspective,” *IEEE Trans Neural Netw*, vol. 6, no. 1, pp. 182–195, 1995, doi: 10.1109/72.363438.

[94] J. Schmidhuber, “Deep Learning in Neural Networks: An Overview,” *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015, doi: 10.1016/j.neunet.2014.09.003.

[95] Martín~Abadi *et al.*, “ TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.” 2015. doi: https://doi.org/10.5281/zenodo.4724125.

[96] S. Albawi, T. A. Mohammed, and S. Al-Zawi, “Understanding of a convolutional neural network,” *Proceedings of 2017 International Conference on Engineering and Technology, ICET 2017*, vol. 2018-January, pp. 1–6, Mar. 2018, doi:

10.1109/ICENGTECHNOL.2017.8308186.

[97] J. Kim, O. Sangjun, Y. Kim, and M. Lee, “Convolutional Neural Network with Biologically Inspired Retinal Structure,” *Procedia Comput Sci*, vol. 88, pp. 145–154, Jan. 2016, doi: 10.1016/J.PROCS.2016.07.418.

[98] K. O’Shea and R. Nash, “An Introduction to Convolutional Neural Networks,” Nov. 2015, doi: 10.48550/arxiv.1511.08458.

[99] X. Li and X. Wu, “Constructing Long Short-Term Memory based Deep Recurrent Neural Networks for Large Vocabulary Speech Recognition,” Oct. 2014, Accessed: Jan. 03, 2023. [Online]. Available: http://arxiv.org/abs/1410.4281

[100] S. A. Marhon, C. J. F. Cameron, and S. C. Kremer, “Recurrent Neural Networks,” *Intelligent Systems Reference Library*, vol. 49, pp. 29–65, 2013, doi: 10.1007/978-3-642-36657-4_2/COVER.

[101] E. DIao, J. DIng, and V. Tarokh, “Restricted Recurrent Neural Networks,” *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, pp. 56–63, Dec. 2019, doi: 10.1109/BIGDATA47090.2019.9006257.

[102] N. Tax, I. Verenich, M. la Rosa, and M. Dumas, *Predictive Business Process Monitoring with LSTM neural networks*, vol. 10253 LNCS. Springer Verlag, 2017. doi: 10.1007/978-3-319-59536-8_30.

[103] A. Graves, G. Wayne, and I. Danihelka, “Neural Turing Machines,” Oct. 2014, Accessed: Jan. 03, 2023. [Online]. Available: http://arxiv.org/abs/1410.5401

[104] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[105] G. Petneházi, “Recurrent Neural Networks for Time Series Forecasting,” *Cornell University*, pp. 1–22, 2019, Accessed: Jan. 03, 2023. [Online]. Available: https://arxiv.org/abs/1901.00069v1

[106] I. Sutskever, O. Vinyals, and Q. v. Le, “Sequence to sequence learning with neural networks,” *Adv Neural Inf Process Syst*, vol. 4, no. January, pp. 3104–3112, 2014.

[107] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder–decoder approaches,” *Proceedings of SSST 2014 - 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, 2014, doi: 10.3115/v1/w14-4012.

[108] R. Dey and F. M. Salemt, “Gate-variants of Gated Recurrent Unit (GRU) neural networks,” *Midwest Symposium on Circuits and Systems*, vol. 2017-August, pp. 1597–1600, Sep. 2017, doi: 10.1109/MWSCAS.2017.8053243.

[109] J. C. Heck and F. M. Salem, “Simplified minimal gated unit variations for recurrent neural networks,” *Midwest Symposium on Circuits and Systems*, vol. 2017-August, pp. 1593–1596, Sep. 2017, doi: 10.1109/MWSCAS.2017.8053242.

[110] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. E. Mohamed, and H. Arshad, “State-of-the-art in artificial neural network applications: A survey,” *Heliyon*, vol. 4, no. 11, p. e00938, Nov. 2018, doi: 10.1016/j.heliyon.2018.e00938.

[111] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, Jan. 2015, doi: 10.1016/j.neunet.2014.09.003.

[112] V. I. Jurtz *et al.*, “An introduction to deep learning on biological sequence data: Examples and solutions,” *Bioinformatics*, 2017, doi: 10.1093/bioinformatics/btx531.

[113] L. Bajaj, H. Kumar, and Y. Hasija, “Automated System for Prediction of Skin Disease using Image Processing and Machine Learning,” *Int J Comput Appl*, vol. 180, no. 19, pp. 9–12, Feb. 2018, doi: 10.5120/ijca2018916428.

[114] N. B. Karayiannis, “Reformulated radial basis neural networks trained by gradient descent,” *IEEE Trans Neural Netw*, vol. 10, no. 3, pp. 657–671, May 1999, doi: 10.1109/72.761725.

[115] Y. A. LeCun, Y. Bengio, and G. E. Hinton, "Deep learning," *Nature*, 2015, doi: 10.1038/nature14539.

[116] A. Graves, *Supervised Sequence Labeling with Recurrent Neural Networks*. 2013. doi: 10.1145/2661829.2661935.

[117] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Comput*, 2000, doi: 10.1162/089976600300015015.

[118] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[119] S. Ahn, "Deep learning architectures and applications," *Journal of Intelligence and Information Systems*, vol. 22, no. 2, pp. 127–142, 2016.

[120] J. Hirschberg and C. D. Manning, "Advances in natural language processing," *Science*. 2015. doi: 10.1126/science.aaa8685.

[121] A. Heydari, M. A. Tavakoli, N. Salim, and Z. Heydari, "Detection of review spam: A survey," *Expert Systems with Applications*. 2015. doi: 10.1016/j.eswa.2014.12.029.

[122] V. Thiagarajan, D. B. Shah, and N. Lalithamani, "Text summarization," *Journal of Advanced Research in Dynamical and Control Systems*, 2018.

[123] Y. Tsuruoka, "Deep Learning and Natural Language Processing," *Brain and nerve = Shinkei kenkyu no shinpo*. 2019. doi: 10.11477/mf.1416201215.

[124] H. Gao, Z. Wang, and S. Ji, "ChannelNets: Compact and efficient convolutional neural networks via channel-wise convolutions," in *Advances in Neural Information Processing Systems*, 2018.

[125] M. Sundermeyer, R. Schl, and H. Ney, "LSTM Neural Networks for Language Modeling," *Proc. Interspeech*, 2012.

[126] J. Xu, H. Li, and S. Zhou, "An Overview of Deep Generative Models," *http://dx.doi.org/10.1080/02564602.2014.987328*, vol. 32, no. 2, pp. 131–139, 2014, doi: 10.1080/02564602.2014.987328.

[127] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Process Mag*, vol. 35, no. 1, pp. 53–65, Jan. 2018, doi: 10.1109/MSP.2017.2765202.

[128] R. Wei and A. Mahmood, "Recent Advances in Variational Autoencoders with Representation Learning for Biomedical Informatics: A Survey," *IEEE Access*, vol. 9, pp. 4939–4956, 2021, doi: 10.1109/ACCESS.2020.3048309.

[129] M. Regis, P. Serra, and E. R. van den Heuvel, "Random autoregressive models: A structured overview," *https://doi.org/10.1080/07474938.2021.1899504*, vol. 41, no. 2, pp. 207–230, 2021, doi: 10.1080/07474938.2021.1899504.

[130] Q. Zhou, C. Du, D. Li, H. Wang, J. K. Liu, and H. He, "Neural Encoding and Decoding with a Flow-based Invertible Generative Model," *IEEE Trans Cogn Dev Syst*, 2022, doi: 10.1109/TCDS.2022.3176977.

[131] G. de Bie, G. Peyré, and M. Cuturi, "Stochastic Deep Networks," in *Proceedings of the 36th International Conference on Machine Learning*, Jan. 2019, vol. 97, pp. 1556–1565. [Online]. Available: https://proceedings.mlr.press/v97/de-bie19a.html

[132] Y. Li, K. Swersky, and R. Zemel, "Generative Moment Matching Networks," in *Proceedings of the 32nd International Conference on Machine Learning*, Jan. 2015, vol. 37, pp. 1718–1727. [Online]. Available: https://proceedings.mlr.press/v37/li15.html

[133] R. Salakhutdinov and H. Larochelle, "Efficient Learning of Deep Boltzmann Machines," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Jan. 2010, vol. 9, pp. 693–700. [Online]. Available: https://proceedings.mlr.press/v9/salakhutdinov10a.html

[134] K. Sankaran and S. P. Holmes, "Generative Models: An Interdisciplinary Perspective," *https://doi.org/10.1146/annurev-statistics-033121-110134*, vol. 10, no. 1, Nov. 2022, doi: 10.1146/ANNUREV-STATISTICS-033121-110134.

[135] Y. Miao and P. Blunsom, “Language as a Latent Variable: Discrete Generative Models for Sentence Compression,” *EMNLP 2016 - Conference on Empirical Methods in Natural Language Processing, Proceedings*, pp. 319–328, Sep. 2016, doi: 10.48550/arxiv.1609.07317.

[136] R. Lopez, A. Gayoso, and N. Yosef, “Enhancing scientific discoveries in molecular biology with deep generative models,” *Mol Syst Biol*, vol. 16, no. 9, p. e9198, Sep. 2020, doi: 10.15252/MSB.20199198.

[137] C. Bilodeau, W. Jin, T. Jaakkola, R. Barzilay, and K. F. Jensen, “Generative models for molecular discovery: Recent advances and challenges,” *Wiley Interdiscip Rev Comput Mol Sci*, vol. 12, no. 5, p. e1608, Sep. 2022, doi: 10.1002/WCMS.1608.

[138] C. G. Turhan and H. S. Bilge, “Recent Trends in Deep Generative Models: A Review,” *UBMK 2018 - 3rd International Conference on Computer Science and Engineering*, pp. 574–579, Dec. 2018, doi: 10.1109/UBMK.2018.8566353.

[139] D. B. Searls, “The language of genes,” *Nature 2002 420:6912*, vol. 420, no. 6912, pp. 211–217, Nov. 2002, doi: 10.1038/nature01255.

[140] A. U. Igamberdiev and N. E. Shklovskiy-Kordi, “Computational power and generative capacity of genetic systems,” *Biosystems*, vol. 142–143, pp. 1–8, Apr. 2016, doi: 10.1016/J.BIOSYSTEMS.2016.01.003.

[141] L. M. Rocha and W. Hordijk, “Material Representations: From the Genetic Code to the Evolution of Cellular Automata,” *Artif Life*, vol. 11, no. 1–2, pp. 189–214, Jan. 2005, doi: 10.1162/1064546053278964.

[142] R. K. Logan, “What Is Information?: Why Is It Relativistic and What Is Its Relationship to Materiality, Meaning and Organization,” *Information 2012, Vol. 3, Pages 68-91*, vol. 3, no. 1, pp. 68–91, Feb. 2012, doi: 10.3390/INFO3010068.

[143] D. B. Searls, “Linguistic approaches to biological sequences,” *Bioinformatics*, vol. 13,

no. 4, pp. 333–344, Aug. 1997, doi: 10.1093/BIOINFORMATICS/13.4.333.

[144] X. Liu *et al.*, “Protein Language Model Predicts Mutation Pathogenicity and Clinical Prognosis,” *bioRxiv*, p. 2022.09.30.510294, Nov. 2022, doi: 10.1101/2022.09.30.510294.

[145] D. Ofer, N. Brandes, and M. Linial, “The language of proteins: NLP, machine learning & protein sequences,” *Comput Struct Biotechnol J*, vol. 19, pp. 1750–1758, Jan. 2021, doi: 10.1016/J.CSBJ.2021.03.022.

[146] D. Weininger, “SMILES, a Chemical Language and Information System: 1: Introduction to Methodology and Encoding Rules,” *J Chem Inf Comput Sci*, vol. 28, no. 1, pp. 31–36, Feb. 1988, doi: 10.1021/CI00057A005/ASSET/CI00057A005.FP.PNG_V03.

[147] A. Drefahl, “CurlySMILES: A chemical language to customize and annotate encodings of molecular and nanodevice structures,” *J Cheminform*, vol. 3, no. 1, pp. 1–7, Jan. 2011, doi: 10.1186/1758-2946-3-1/FIGURES/2.

[148] I. Lee and H. Nam, “Infusing Linguistic Knowledge of SMILES into Chemical Language Models,” Apr. 2022, doi: 10.48550/arxiv.2205.00084.

[149] D. P. Bartel, “MicroRNAs: Genomics, Biogenesis, Mechanism, and Function,” *Cell*, vol. 116, no. 2. pp. 281–297, 2004. doi: 10.1016/S0092-8674(04)00045-5.

[150] D. Baek, J. Villén, C. Shin, F. D. Camargo, S. P. Gygi, and D. P. Bartel, “The impact of microRNAs on protein output,” *Nature*, 2008, doi: 10.1038/nature07242.

[151] Y. Li and K. v. Kowdley, “MicroRNAs in Common Human Diseases,” *Genomics Proteomics Bioinformatics*, vol. 10, no. 5, pp. 246–253, Oct. 2012, doi: 10.1016/j.gpb.2012.07.005.

[152] N. Meola, V. A. Gennarino, and S. Banfi, “microRNAs and genetic diseases.,” *Pathogenetics*, vol. 2, no. 1, p. 7, Nov. 2009, doi: 10.1186/1755-8417-2-7.

[153] J. Xia and W. Zhang, "MicroRNAs in normal and psoriatic skin," *Physiol Genomics*, vol. 46, no. 4, pp. 113–122, 2014, doi: 10.1152/physiolgenomics.00157.2013.

[154] Y. W. Kong, D. Ferland-McCollough, T. J. Jackson, and M. Bushell, "MicroRNAs in cancer management," *The Lancet Oncology*, vol. 13, no. 6. 2012. doi: 10.1016/S1470-2045(12)70073-6.

[155] M. Mancini *et al.*, "MicroRNAs in human skin ageing.," *Ageing Res Rev*, vol. 17, pp. 9–15, Sep. 2014, doi: 10.1016/j.arr.2014.04.003.

[156] Y. Hasija, "Role of microRNAs in Genodermatoses." Accessed: Feb. 02, 2019. [Online]. Available: http://www.krishisanskriti.org

[157] S. W. Chi, J. B. Zang, A. Mele, and R. B. Darnell, "Argonaute HITS-CLIP decodes microRNA-mRNA interaction maps," *Nature*, 2009, doi: 10.1038/nature08170.

[158] A. Helwak, G. Kudla, T. Dudnakova, and D. Tollervey, "Mapping the human miRNA interactome by CLASH reveals frequent noncanonical binding," *Cell*, 2013, doi: 10.1016/j.cell.2013.03.043.

[159] S. Ekimler and K. Sahin, "Computational Methods for MicroRNA Target Prediction," *Genes (Basel)*, 2014, doi: 10.3390/genes5030671.

[160] J. Krüger and M. Rehmsmeier, "RNAhybrid: microRNA target prediction easy, fast and flexible," *Nucleic Acids Res*, vol. 34, no. Web Server issue, Jul. 2006, doi: 10.1093/NAR/GKL243.

[161] A. Krek *et al.*, "Combinatorial microRNA target predictions," *Nat Genet*, 2005, doi: 10.1038/ng1536.

[162] S. Bandyopadhyay and R. Mitra, "TargetMiner: MicroRNA target prediction with systematic identification of tissue-specific negative examples," *Bioinformatics*, 2009, doi: 10.1093/bioinformatics/btp503.

[163] M. D. Paraskevopoulou *et al.*, "DIANA-microT web server v5.0: service integration into

miRNA functional analysis workflows.," *Nucleic Acids Res*, vol. 41, no. Web Server issue, 2013, doi: 10.1093/nar/gkt393.

[164] M. D. Paraskevopoulou *et al.*, "DIANA-microT web server v5.0: service integration into miRNA functional analysis workflows.," *Nucleic Acids Res*, 2013, doi: 10.1093/nar/gkt393.

[165] K. O. Stanley, J. Clune, J. Lehman, and R. Miikkulainen, "Designing neural networks through neuroevolution," *Nature Machine Intelligence 2019 1:1*, vol. 1, no. 1, pp. 24–35, Jan. 2019, doi: 10.1038/s42256-018-0006-z.

[166] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[167] A. L. Caterini and D. E. Chang, "Recurrent neural networks," *SpringerBriefs in Computer Science*, 2018.

[168] M. Sundermeyer, R. Schlüter, and H. Ney, "LSTM neural networks for language modeling," *13th Annual Conference of the International Speech Communication Association 2012, INTERSPEECH 2012*, vol. 1, pp. 194–197, 2012, doi: 10.21437/INTERSPEECH.2012-65.

[169] I. Sutskever, O. Vinyals, and Q. v Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[170] P. SETHUPATHY, "TarBase: A comprehensive database of experimentally supported animal microRNA targets," *RNA*, vol. 12, no. 2, pp. 192–197, 2005, doi: 10.1261/rna.2239606.

[171] 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, doi: 10.1093/nar/gkn809.

[172] S. H. Bernhart, U. Mückstein, and I. L. Hofacker, "RNA Accessibility in cubic time,"

Algorithms Mol Biol, vol. 6, no. 1, p. 3, Mar. 2011, doi: 10.1186/1748-7188-6-3.

[173] R. Lorenz *et al.*, “ViennaRNA Package 2.0,” *Algorithms for Molecular Biology*, vol. 6, no. 1, pp. 1–14, Nov. 2011, doi: 10.1186/1748-7188-6-26/TABLES/2.

[174] M. Kertesz, N. Iovino, U. Unnerstall, U. Gaul, and E. Segal, “The role of site accessibility in microRNA target recognition,” *Nat Genet*, vol. 39, no. 10, pp. 1278–1284, Oct. 2007, doi: 10.1038/NG2135.

[175] H. Tafer *et al.*, “The impact of target site accessibility on the design of effective siRNAs,” *Nat Biotechnol*, vol. 26, no. 5, pp. 578–583, May 2008, doi: 10.1038/NBT1404.

[176] 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, doi: 10.1038/nature03049.

[177] B. J. Reinhart *et al.*, “The 21-nucleotide let-7 RNA regulates developmental timing in Caenorhabditis elegans,” *Nature*, 2000, doi: 10.1038/35002607.

[178] A. E. Pasquinelli *et al.*, “Conservation of the sequence and temporal expression of let-7 heterochronic regulatory RNA,” *Nature*, 2000, doi: 10.1038/35040556.

[179] 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 (1979)*, 2001, doi: 10.1126/science.1062961.

[180] S. M. Hammond, “An overview of microRNAs,” *Adv Drug Deliv Rev*, vol. 87, pp. 3–14, Jun. 2015, doi: 10.1016/J.ADDR.2015.05.001.

[181] R. Sperling, “The nuts and bolts of the endogenous spliceosome,” *Wiley Interdiscip Rev RNA*, vol. 8, no. 1, p. e1377, Jan. 2017, doi: 10.1002/WRNA.1377.

[182] C. Blenkiron *et al.*, “MicroRNA expression profiling of human breast cancer identifies new markers of tumor subtype,” *Genome Biol*, vol. 8, no. 10, pp. 1–16, Oct. 2007, doi:

10.1186/GB-2007-8-10-R214/FIGURES/6.

[183] L. Adams, “Pri-miRNA processing: structure is key,” *Nature Reviews Genetics 2017 18:3*, vol. 18, no. 3, pp. 145–145, Jan. 2017, doi: 10.1038/nrg.2017.6.

[184] C. Roden *et al.*, “Novel determinants of mammalian primary microRNA processing revealed by systematic evaluation of hairpin-containing transcripts and human genetic variation,” *Genome Res*, vol. 27, no. 3, pp. 374–384, Mar. 2017, doi: 10.1101/GR.208900.116.

[185] S. Yin, Y. Yu, and R. Reed, “Primary microRNA processing is functionally coupled to RNAP II transcription in vitro,” *Sci Rep*, vol. 5, Jul. 2015, doi: 10.1038/SREP11992.

[186] X. A. Cambronne, R. Shen, P. L. Auer, and R. H. Goodman, “Capturing microRNA targets using an RNA-induced silencing complex (RISC)-trap approach,” *Proc Natl Acad Sci U S A*, vol. 109, no. 50, pp. 20473–20478, Dec. 2012, doi: 10.1073/PNAS.1218887109/SUPPL_FILE/SD02.XLSX.

[187] G. Tang, “siRNA and miRNA: an insight into RISCs,” *Trends Biochem Sci*, vol. 30, no. 2, pp. 106–114, Feb. 2005, doi: 10.1016/J.TIBS.2004.12.007.

[188] M. R. Fabian, N. Sonenberg, and W. Filipowicz, “Regulation of mRNA translation and stability by microRNAs,” *Annu Rev Biochem*, vol. 79, pp. 351–379, Jul. 2010, doi: 10.1146/ANNUREV-BIOCHEM-060308-103103.

[189] N. J. Lambert, S. G. Gu, and A. M. Zahler, “The conformation of microRNA seed regions in native microRNPs is prearranged for presentation to mRNA targets,” *Nucleic Acids Res*, vol. 39, no. 11, pp. 4827–4835, Jun. 2011, doi: 10.1093/NAR/GKR077.

[190] L. B. Chipman and A. E. Pasquinelli, “miRNA Targeting: Growing beyond the Seed,” *Trends in Genetics*, vol. 35, no. 3, pp. 215–222, Mar. 2019, doi: 10.1016/J.TIG.2018.12.005.

[191] A. Krek *et al.*, “Combinatorial microRNA target predictions,” *Nature Genetics 2005*

37:5, vol. 37, no. 5, pp. 495–500, Apr. 2005, doi: 10.1038/ng1536.

[192] B. R. Cullen, “Transcription and processing of human microRNA precursors,” *Mol. Cell*, vol. 16, no. 6, pp. 861–865, Dec. 2004, doi: 10.1016/j.molcel.2004.12.002.

[193] D. P. Bartel, “MicroRNAs: genomics, biogenesis, mechanism, and function,” *Cell*, vol. 116, no. 2, pp. 281–297, Jan. 2004, doi: 10.1016/s0092-8674(04)00045-5.

[194] K. C. Miranda *et al.*, “A Pattern-Based Method for the Identification of MicroRNA Binding Sites and Their Corresponding Heteroduplexes,” *Cell*, 2006, doi: 10.1016/j.cell.2006.07.031.

[195] A. Krek *et al.*, “Combinatorial microRNA target predictions,” *Nat Genet*, vol. 37, no. 5, pp. 495–500, May 2005, doi: 10.1038/NG1536.

[196] C. Shin, J. W. Nam, K. K. H. Farh, H. R. Chiang, A. Shkumatava, and D. P. Bartel, “Expanding the microRNA targeting code: functional sites with centered pairing,” *Mol Cell*, vol. 38, no. 6, pp. 789–802, Jun. 2010, doi: 10.1016/J.MOLCEL.2010.06.005.

[197] S. L. Ameres, J. Martinez, and R. Schroeder, “Molecular Basis for Target RNA Recognition and Cleavage by Human RISC,” *Cell*, vol. 130, no. 1, pp. 101–112, Jul. 2007, doi: 10.1016/J.CELL.2007.04.037.

[198] A. Arvey, E. Larsson, C. Sander, C. S. Leslie, and D. S. Marks, “Target mRNA abundance dilutes microRNA and siRNA activity,” *Mol Syst Biol*, vol. 6, 2010, doi: 10.1038/MSB.2010.24.

[199] O. Barad *et al.*, “MicroRNA expression detected by oligonucleotide microarrays: system establishment and expression profiling in human tissues,” *Genome Res.*, vol. 14, no. 12, pp. 2486–2494, Dec. 2004, doi: 10.1101/gr.2845604.

[200] V. Agarwal, G. W. Bell, J. W. Nam, and D. P. Bartel, “Predicting effective microRNA target sites in mammalian mRNAs,” *Elife*, vol. 4, no. AUGUST2015, Aug. 2015, doi: 10.7554/ELIFE.05005.

[201] S. L. Ameres, J. Martinez, and R. Schroeder, “Molecular Basis for Target RNA Recognition and Cleavage by Human RISC,” *Cell*, vol. 130, no. 1, pp. 101–112, Jul. 2007, doi: 10.1016/J.CELL.2007.04.037.

[202] 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, Jan. 2009, doi: 10.1101/GR.082701.108.

[203] E. M. Anderson *et al.*, “Experimental validation of the importance of seed complement frequency to siRNA specificity,” *RNA*, vol. 14, no. 5, pp. 853–861, May 2008, doi: 10.1261/RNA.704708.

[204] U. A. Ørom and A. H. Lund, “Experimental identification of microRNA targets,” *Gene*, vol. 451, no. 1–2, pp. 1–5, Feb. 2010, doi: 10.1016/j.gene.2009.11.008.

[205] G. Liu, R. Zhang, J. Xu, C. I. Wu, and X. Lu, “Functional Conservation of Both CDS- and 3′-UTR-Located MicroRNA Binding Sites between Species,” *Mol Biol Evol*, vol. 32, no. 3, pp. 623–628, Mar. 2015, doi: 10.1093/MOLBEV/MSU323.

[206] J. Gu, H. Fu, X. Zhang, and Y. Li, “Identifications of conserved 7-mers in 3′-UTRs and microRNAs in Drosophila,” *BMC Bioinformatics*, vol. 8, Nov. 2007, doi: 10.1186/1471-2105-8-432.

[207] 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*, vol. 18, no. 10, pp. 1139–1146, Oct. 2011, doi: 10.1038/NSMB.2115.

[208] A. Grimson, K. K. H. Farh, W. K. Johnston, P. Garrett-Engele, L. P. Lim, and D. P. Bartel, “MicroRNA Targeting Specificity in Mammals: Determinants beyond Seed Pairing,” *Mol Cell*, vol. 27, no. 1, pp. 91–105, Jul. 2007, doi: 10.1016/J.MOLCEL.2007.06.017.

[209] S. Ambs *et al.*, "Genomic profiling of microRNA and messenger RNA reveals deregulated microRNA expression in prostate cancer," *Cancer Res*, vol. 68, no. 15, pp. 6162–6170, Aug. 2008, doi: 10.1158/0008-5472.CAN-08-0144.

[210] S. Vasudevan, "Functional validation of microRNA-target RNA interactions," *Methods*, vol. 58, no. 2, pp. 126–134, Oct. 2012, doi: 10.1016/J.YMETH.2012.08.002.

[211] G. Nigita *et al.*, "microRNA editing in seed region aligns with cellular changes in hypoxic conditions," *Nucleic Acids Res*, vol. 44, no. 13, pp. 6298–6308, Jul. 2016, doi: 10.1093/NAR/GKW532.

[212] Z. Hu, "Insight into microRNA regulation by analyzing the characteristics of their targets in humans," *BMC Genomics*, vol. 10, p. 594, Dec. 2009, doi: 10.1186/1471-2164-10-594.

[213] C. Lind, M. Esguerra, W. Jespers, P. Satpati, H. Gutierrez-de-Terán, and J. Åqvist, "Free energy calculations of RNA interactions," *Methods*, vol. 162–163, pp. 85–95, Jun. 2019, doi: 10.1016/J.YMETH.2019.02.014.

[214] M. Kertesz, N. Iovino, U. Unnerstall, U. Gaul, and E. Segal, "The role of site accessibility in microRNA target recognition," *Nature Genetics 2007 39:10*, vol. 39, no. 10, pp. 1278–1284, Sep. 2007, doi: 10.1038/ng2135.

[215] H. Tafer *et al.*, "The impact of target site accessibility on the design of effective siRNAs," *Nat Biotechnol*, 2008, doi: 10.1038/nbt1404.

[216] G. Varani and W. H. McClain, "The G·U wobble base pair," *EMBO Rep*, vol. 1, no. 1, pp. 18–23, Jul. 2000, doi: 10.1093/EMBO-REPORTS/KVD001.

[217] J. G. Doench and P. A. Sharp, "Specificity of microRNA target selection in translational repression," *Genes Dev*, vol. 18, no. 5, pp. 504–511, Mar. 2004, doi: 10.1101/GAD.1184404.

[218] A. J. Enright, B. John, U. Gaul, T. Tuschl, C. Sander, and D. S. Marks, "MicroRNA

targets in Drosophila.," *Genome Biol*, vol. 5, no. 1, pp. 1–14, Dec. 2003, doi: 10.1186/GB-2003-5-1-R1/TABLES/6.

[219] 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*, vol. 11, no. 8, p. R90, Aug. 2010, doi: 10.1186/GB-2010-11-8-R90.

[220] M. Reczko, M. Maragkakis, P. Alexiou, I. Grosse, and A. G. Hatzigeorgiou, "Functional microRNA targets in protein coding sequences," *Bioinformatics*, vol. 28, no. 6, pp. 771–776, Mar. 2012, doi: 10.1093/BIOINFORMATICS/BTS043.

[221] X. Wang and I. M. el Naqa, "Prediction of both conserved and nonconserved microRNA targets in animals," *Bioinformatics*, vol. 24, no. 3, pp. 325–332, Feb. 2008, doi: 10.1093/BIOINFORMATICS/BTM595.

[222] P. Loher and I. Rigoutsos, "Interactive exploration of RNA22 microRNA target predictions," *Bioinformatics*, vol. 28, no. 24, pp. 3322–3323, Dec. 2012, doi: 10.1093/BIOINFORMATICS/BTS615.

[223] S. Bandyopadhyay and R. Mitra, "TargetMiner: microRNA target prediction with systematic identification of tissue-specific negative examples," *Bioinformatics*, vol. 25, no. 20, pp. 2625–2631, Oct. 2009, doi: 10.1093/BIOINFORMATICS/BTP503.

[224] H. Liu, D. Yue, Y. Chen, S. J. Gao, and Y. Huang, "Improving performance of mammalian microRNA target prediction," *BMC Bioinformatics*, vol. 11, no. 1, pp. 1–15, Sep. 2010, doi: 10.1186/1471-2105-11-476/FIGURES/10.

[225] M. Kertesz, N. Iovino, U. Unnerstall, U. Gaul, and E. Segal, "The role of site accessibility in microRNA target recognition," *Nature Genetics 2007 39:10*, vol. 39, no. 10, pp. 1278–1284, Sep. 2007, doi: 10.1038/ng2135.

[226] J. Krüger and M. Rehmsmeier, "RNAhybrid: microRNA target prediction easy, fast and flexible," *Nucleic Acids Res*, vol. 34, no. Web Server issue, p. W451, Jul. 2006, doi:

10.1093/NAR/GKL243.

[227] R. B. Darnell, “HITS-CLIP: Panoramic views of protein-RNA regulation in living cells,” *Wiley Interdiscip Rev RNA*, 2010, doi: 10.1002/wrna.31.

[228] J. Wu, H. Gong, Y. Bai, and W. Zhang, “Analyzing the miRNA-Gene Networks to Mine the Important miRNAs under Skin of Human and Mouse,” *Biomed Res Int*, vol. 2016, pp. 1–9, 2016, doi: 10.1155/2016/5469371.

[229] J. J. Cole *et al.*, “Diverse interventions that extend mouse lifespan suppress shared age-associated epigenetic changes at critical gene regulatory regions,” *Genome Biol*, vol. 18, no. 1, Mar. 2017, doi: 10.1186/S13059-017-1185-3.

[230] A. R. Gruber, S. H. Bernhart, and R. Lorenz, “The viennaRNA web services,” *Methods in Molecular Biology*, 2015, doi: 10.1007/978-1-4939-2291-8_19.

[231] J. S. Reuter and D. H. Mathews, “RNAstructure: Software for RNA secondary structure prediction and analysis,” *BMC Bioinformatics*, 2010.

[232] R. Chakraborty and Y. Hasija, “miDerma: An Integrated Database and Tool for Analysis of miRNAs associated with Dermatological Disorders,” in *International Conference on Bioinformatics and Systems Biology, BSB*, 2018.

[233] B. Xue, A. K. Dunker, and V. N. Uversky, “Orderly order in protein intrinsic disorder distribution: Disorder in 3500 proteomes from viruses and the three domains of life,” *J Biomol Struct Dyn*, 2012, doi: 10.1080/07391102.2012.675145.

[234] Z. Peng *et al.*, “Exceptionally abundant exceptions: Comprehensive characterization of intrinsic disorder in all domains of life,” *Cellular and Molecular Life Sciences*, 2014, doi: 10.1007/s00018-014-1661-9.

[235] H. J. Dyson and P. E. Wright, “Intrinsically unstructured proteins and their functions,” *Nature Reviews Molecular Cell Biology*. 2005. doi: 10.1038/nrm1589.

[236] A. K. Dunker, I. Silman, V. N. Uversky, and J. L. Sussman, “Function and structure of

inherently disordered proteins," *Current Opinion in Structural Biology*. 2008. doi: 10.1016/j.sbi.2008.10.002.

[237] V. N. Uversky, C. J. Oldfield, and A. K. Dunker, " Intrinsically Disordered Proteins in Human Diseases: Introducing the D 2 Concept ," *Annu Rev Biophys*, 2008, doi: 10.1146/annurev.biophys.37.032807.125924.

[238] Y. Shigemitsu and H. Hiroaki, "Common molecular pathogenesis of disease-related intrinsically disordered proteins revealed by NMR analysis," *Journal of Biochemistry*. 2018. doi: 10.1093/jb/mvx056.

[239] V. Receveur-Bréhot, J. M. Bourhis, V. N. Uversky, B. Canard, and S. Longhi, "Assessing protein disorder and induced folding," *Proteins: Structure, Function and Genetics*, 2006, doi: 10.1002/prot.20750.

[240] V. N. Uversky, "Functions of short lifetime biological structures at large: the case of intrinsically disordered proteins," *Brief Funct Genomics*, 2018, doi: 10.1093/bfgp/ely023.

[241] S. DeForte and V. N. Uversky, "Resolving the ambiguity: Making sense of intrinsic disorder when PDB structures disagree," *Protein Science*, 2016, doi: 10.1002/pro.2864.

[242] R. Konrat, "NMR contributions to structural dynamics studies of intrinsically disordered proteins," *Journal of Magnetic Resonance*, 2014, doi: 10.1016/j.jmr.2013.11.011.

[243] F. Meng, V. N. Uversky, and L. Kurgan, "Comprehensive review of methods for prediction of intrinsic disorder and its molecular functions," *Cellular and Molecular Life Sciences*. 2017. doi: 10.1007/s00018-017-2555-4.

[244] B. He, K. Wang, Y. Liu, B. Xue, V. N. Uversky, and A. K. Dunker, "Predicting intrinsic disorder in proteins: An overview," *Cell Research*. 2009. doi: 10.1038/cr.2009.87.

[245] P. Romero, Z. Obradovic, C. Kissinger, J. E. Villafranca, and A. K. Dunker, "Identifying disordered regions in proteins from amino acid sequence," in *IEEE International*

Conference on Neural Networks - Conference Proceedings, 1997. doi: 10.1109/ICNN.1997.611643.

[246] J. Prilusky *et al.*, "FoldIndex©: A simple tool to predict whether a given protein sequence is intrinsically unfolded," *Bioinformatics*, 2005, doi: 10.1093/bioinformatics/bti537.

[247] Z. Dosztányi, V. Csizmok, P. Tompa, and I. Simon, "IUPred: Web server for the prediction of intrinsically unstructured regions of proteins based on estimated energy content," *Bioinformatics*, 2005, doi: 10.1093/bioinformatics/bti541.

[248] R. Linding, R. B. Russell, V. Neduva, and T. J. Gibson, "GlobPlot: Exploring protein sequences for globularity and disorder.," *Nucleic Acids Res*, 2003.

[249] J. Hanson, Y. Yang, K. Paliwal, and Y. Zhou, "Improving protein disorder prediction by deep bidirectional long short-term memory recurrent neural networks," *Bioinformatics*, 2017, doi: 10.1093/bioinformatics/btw678.

[250] Y. Liu, X. Wang, and B. Liu, "A comprehensive review and comparison of existing computational methods for intrinsically disordered protein and region prediction," *Brief Bioinform*, 2019, doi: 10.1093/bib/bbx126.

[251] I. Walsh, A. J. M. Martin, T. Di Domenico, A. Vullo, G. Pollastri, and S. C. E. Tosatto, "CSpritz: Accurate prediction of protein disorder segments with annotation for homology, secondary structure and linear motifs," *Nucleic Acids Res*, 2011, doi: 10.1093/nar/gkr411.

[252] R. Linding, L. J. Jensen, F. Diella, P. Bork, T. J. Gibson, and R. B. Russell, "Protein disorder prediction: Implications for structural proteomics," *Structure*, 2003, doi: 10.1016/j.str.2003.10.002.

[253] P. Romero, Z. Obradovic, X. Li, E. C. Garner, C. J. Brown, and A. K. Dunker, "Sequence complexity of disordered protein," *Proteins: Structure, Function and*

Genetics, 2001, doi: 10.1002/1097-0134(20010101)42:1<38::AID-PROT50>3.0.CO;2-3.

[254] S. F. Altschul *et al.*, “Gapped BLAST and PSI-BLAST: A new generation of protein database search programs,” *Nucleic Acids Research*. 1997. doi: 10.1093/nar/25.17.3389.

[255] M. J. Mizianty, W. Stach, K. Chen, K. D. Kedarisetti, F. M. Disfani, and L. Kurgan, “Improved sequence-based prediction of disordered regions with multilayer fusion of multiple information sources,” in *Bioinformatics*, 2011. doi: 10.1093/bioinformatics/btq373.

[256] T. Zhang, E. Faraggi, B. Xue, A. K. Dunker, V. N. Uversky, and Y. Zhou, “Spine-d: Accurate prediction of short and long disordered regions by a single neural-network based method,” *J Biomol Struct Dyn*, 2012, doi: 10.1080/073911012010525022.

[257] J. Hanson, K. Paliwal, T. Litfin, Y. Yang, and Y. Zhou, “Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks,” *Bioinformatics*, 2018, doi: 10.1093/bioinformatics/bty481.

[258] S. Wang, J. Ma, and J. Xu, “AUCpreD: Proteome-level protein disorder prediction by AUC-maximized deep convolutional neural fields,” in *Bioinformatics*, 2016. doi: 10.1093/bioinformatics/btw446.

[259] C. Cheng *et al.*, “A statistical framework for modeling gene expression using chromatin features and application to modENCODE datasets,” *Genome Biol*, 2011, doi: 10.1186/gb-2011-12-2-r15.

[260] 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, doi: 10.1038/ncomms11512.

[261] 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," *OMICS*, 2013, doi: 10.1089/omi.2013.0017.

[262] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Networks*. 2015. doi: 10.1016/j.neunet.2014.09.003.

[263] V. Vapnik, *Statistical learning theory. 1998*. 1998.

[264] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," *Neural Comput*, 1997, doi: 10.1162/neco.1997.9.8.1735.

[265] B. Kumari, R. Kumar, and M. Kumar, "Low complexity and disordered regions of proteins have different structural and amino acid preferences," *Mol Biosyst*, vol. 11, no. 2, pp. 585–594, Feb. 2015, doi: 10.1039/C4MB00425F.

[266] E. Schad, P. Tompa, and H. Hegyi, "The relationship between proteome size, structural disorder and organism complexity," *Genome Biol*, vol. 12, no. 12, pp. 1–13, Dec. 2011, doi: 10.1186/GB-2011-12-12-R120/FIGURES/6.

[267] L. Piersimoni *et al.*, "Lighting up Nobel Prize-winning studies with protein intrinsic disorder," *Cellular and Molecular Life Sciences 2022 79:8*, vol. 79, no. 8, pp. 1–30, Jul. 2022, doi: 10.1007/S00018-022-04468-Y.

[268] V. N. Uversky, "Protein intrinsic disorder-based liquid–liquid phase transitions in biological systems: Complex coacervates and membrane-less organelles," *Adv Colloid Interface Sci*, vol. 239, pp. 97–114, Jan. 2017, doi: 10.1016/J.CIS.2016.05.012.

[269] M. Ota *et al.*, "An assignment of intrinsically disordered regions of proteins based on NMR structures," *J Struct Biol*, vol. 181, no. 1, pp. 29–36, Jan. 2013, doi: 10.1016/J.JSB.2012.10.017.

[270] T. A. Ngo, H. Dinh, T. M. Nguyen, F. F. Liew, E. Nakata, and T. Morii, "Protein adaptors assemble functional proteins on DNA scaffolds," *Chem Commun (Camb)*, vol. 55, no. 83, pp. 12428–12446, 2019, doi: 10.1039/C9CC04661E.

[271] E. G. Weagel, J. M. Foulks, A. Siddiqui, and S. L. Warner, "Molecular glues: enhanced protein-protein interactions and cell proteome editing," *Medicinal Chemistry Research 2022 31:7*, vol. 31, no. 7, pp. 1068–1087, May 2022, doi: 10.1007/S00044-022-02882-2.

[272] S. E. Bondos, A. K. Dunker, and V. N. Uversky, "On the roles of intrinsically disordered proteins and regions in cell communication and signaling," *Cell Commun Signal*, vol. 19, no. 1, Dec. 2021, doi: 10.1186/S12964-021-00774-3.

[273] M. M. Misiura and A. B. Kolomeisky, "Role of Intrinsically Disordered Regions in Acceleration of Protein-Protein Association," *J Phys Chem B*, vol. 124, no. 1, pp. 20–27, Jan. 2020, doi: 10.1021/ACS.JPCB.9B08793.

[274] N. E. Davey, "The functional importance of structure in unstructured protein regions," *Curr Opin Struct Biol*, vol. 56, pp. 155–163, Jun. 2019, doi: 10.1016/J.SBI.2019.03.009.

[275] Z. Dosztányi, B. Mészáros, and I. Simon, "Bioinformatical approaches to characterize intrinsically disordered/unstructured proteins," *Brief Bioinform*, vol. 11, no. 2, pp. 225–243, Dec. 2010, doi: 10.1093/BIB/BBP061.

[276] B. Kumari, R. Kumar, and M. Kumar, "Low complexity and disordered regions of proteins have different structural and amino acid preferences," *Mol Biosyst*, vol. 11, no. 2, pp. 585–594, Feb. 2015, doi: 10.1039/C4MB00425F.

[277] G. M. Ginell and A. S. Holehouse, "Analyzing the Sequences of Intrinsically Disordered Regions with CIDER and localCIDER," *Methods Mol Biol*, vol. 2141, pp. 103–126, 2020, doi: 10.1007/978-1-0716-0524-0_5.

[278] J. Prilusky *et al.*, "FoldIndex©: a simple tool to predict whether a given protein sequence is intrinsically unfolded," *Bioinformatics*, vol. 21, no. 16, pp. 3435–3438, Aug. 2005, doi: 10.1093/BIOINFORMATICS/BTI537.

[279] R. Linding, L. J. Jensen, F. Diella, P. Bork, T. J. Gibson, and R. B. Russell, "Protein

disorder prediction: Implications for structural proteomics," *Structure*, vol. 11, no. 11, pp. 1453–1459, 2003, doi: 10.1016/j.str.2003.10.002.

[280] B. Xue, R. L. Dunbrack, R. W. Williams, A. K. Dunker, and V. N. Uversky, "PONDR-FIT: A meta-predictor of intrinsically disordered amino acids," *Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics*, vol. 1804, no. 4, pp. 996–1010, Apr. 2010, doi: 10.1016/J.BBAPAP.2010.01.011.

[281] R. Linding, R. B. Russell, V. Neduva, and T. J. Gibson, "GlobPlot: exploring protein sequences for globularity and disorder," *Nucleic Acids Res*, vol. 31, no. 13, pp. 3701–3708, Jul. 2003, doi: 10.1093/NAR/GKG519.

[282] N. Kodera *et al.*, "Structural and dynamics analysis of intrinsically disordered proteins by high-speed atomic force microscopy," *Nat Nanotechnol*, vol. 16, no. 2, pp. 181–189, Feb. 2021, doi: 10.1038/S41565-020-00798-9.

[283] P. G. Righetti and B. Verzola, "Folding/unfolding/refolding of proteins: Present methodologies in comparison with capillary zone electrophoresis", doi: 10.1002/1522-2683.

[284] C. Danis *et al.*, "Nuclear magnetic resonance spectroscopy for the identification of multiple phosphorylations of intrinsically disordered proteins," *Journal of Visualized Experiments*, vol. 2016, no. 118, Dec. 2016, doi: 10.3791/55001.

[285] P. Lieutaud, F. Ferron, A. v. Uversky, L. Kurgan, V. N. Uversky, and S. Longhi, "How disordered is my protein and what is its disorder for? A guide through the 'dark side' of the protein universe," *http://dx.doi.org/10.1080/21690707.2016.1259708*, vol. 4, no. 1, p. e1259708, Jan. 2016, doi: 10.1080/21690707.2016.1259708.

[286] D. Kovacs, B. Szabo, R. Pancsa, and P. Tompa, "Intrinsically disordered proteins undergo and assist folding transitions in the proteome," *Arch Biochem Biophys*, vol. 531, no. 1–2, pp. 80–89, Mar. 2013, doi: 10.1016/J.ABB.2012.09.010.

[287] P. Tompa, E. Schad, A. Tantos, and L. Kalmar, “Intrinsically disordered proteins: emerging interaction specialists,” *Curr Opin Struct Biol*, vol. 35, pp. 49–59, Dec. 2015, doi: 10.1016/J.SBI.2015.08.009.

[288] V. N. Uversky, “Unusual biophysics of intrinsically disordered proteins,” *Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics*, vol. 1834, no. 5, pp. 932–951, May 2013, doi: 10.1016/J.BBAPAP.2012.12.008.

[289] A. L. Darling and V. N. Uversky, “Intrinsic disorder and posttranslational modifications: The darker side of the biological dark matter,” *Front Genet*, vol. 9, no. MAY, p. 158, May 2018, doi: 10.3389/FGENE.2018.00158/BIBTEX.

[290] K. Moritsugu, T. Terada, and A. Kidera, “Disorder-to-order transition of an intrinsically disordered region of sortase revealed by multiscale enhanced sampling,” *J Am Chem Soc*, vol. 134, no. 16, pp. 7094–7101, Apr. 2012, doi: 10.1021/JA3008402/SUPPL_FILE/JA3008402_SI_001.PDF.

[291] H. J. Dyson and P. E. Wright, “Role of intrinsic protein disorder in the function and interactions of the transcriptional coactivators CREB-binding Protein (CBP) and p300,” *Journal of Biological Chemistry*, vol. 291, no. 13, pp. 6714–6722, Mar. 2016, doi: 10.1074/JBC.R115.692020/ATTACHMENT/F7943B30-EF43-45CF-AF70-BF900A72D942/MMC3.PDF.

[292] M. Arai, K. Sugase, H. J. Dyson, and P. E. Wright, “Conformational propensities of intrinsically disordered proteins influence the mechanism of binding and folding,” *Proc Natl Acad Sci U S A*, vol. 112, no. 31, pp. 9614–9619, Aug. 2015, doi: 10.1073/PNAS.1512799112/SUPPL_FILE/PNAS.201512799SI.PDF.

[293] V. N. Uversky, “Intrinsically disordered proteins and novel strategies for drug discovery,” *http://dx.doi.org/10.1517/17460441.2012.686489*, vol. 7, no. 6, pp. 475–488, Jun. 2012, doi: 10.1517/17460441.2012.686489.

[294] S. E. Bondos, A. K. Dunker, and V. N. Uversky, “Intrinsically disordered proteins play diverse roles in cell signaling,” *Cell Communication and Signaling*, vol. 20, no. 1, pp. 1–26, Dec. 2022, doi: 10.1186/S12964-022-00821-7/TABLES/1.

[295] M. Varadi, F. Zsolyomi, M. Guharoy, P. Tompa, and Y. K. Levy, “Functional Advantages of Conserved Intrinsic Disorder in RNA-Binding Proteins,” *PLoS One*, vol. 10, no. 10, p. e0139731, Oct. 2015, doi: 10.1371/JOURNAL.PONE.0139731.

[296] L. Rajkowitsch *et al.*, “RNA Chaperones, RNA Annealers and RNA Helicases,” *http://dx.doi.org/10.4161/rna.4.3.5445*, vol. 4, no. 3, pp. 118–130, 2007, doi: 10.4161/RNA.4.3.5445.

[297] S. Calabretta and S. Richard, “Emerging Roles of Disordered Sequences in RNA-Binding Proteins,” *Trends Biochem Sci*, vol. 40, no. 11, pp. 662–672, Nov. 2015, doi: 10.1016/J.TIBS.2015.08.012.

[298] A. H. Huber and W. I. Weis, “The Structure of the β-Catenin/E-Cadherin Complex and the Molecular Basis of Diverse Ligand Recognition by β-Catenin,” *Cell*, vol. 105, no. 3, pp. 391–402, May 2001, doi: 10.1016/S0092-8674(01)00330-0.

[299] M. Guharoy, P. Bhowmick, and P. Tompa, “Design Principles Involving Protein Disorder Facilitate Specific Substrate Selection and Degradation by the Ubiquitin-Proteasome System *,” *Journal of Biological Chemistry*, vol. 291, no. 13, pp. 6723–6731, Mar. 2016, doi: 10.1074/JBC.R115.692665.

[300] I. Amm, T. Sommer, and D. H. Wolf, “Protein quality control and elimination of protein waste: The role of the ubiquitin–proteasome system,” *Biochimica et Biophysica Acta (BBA) - Molecular Cell Research*, vol. 1843, no. 1, pp. 182–196, Jan. 2014, doi: 10.1016/J.BBAMCR.2013.06.031.

[301] P. Hänzelmann, C. Galgenmüller, and H. Schindelin, “Structure and Function of the AAA+ ATPase p97, a Key Player in Protein Homeostasis,” *Subcell Biochem*, vol. 93,

pp. 221–272, 2019, doi: 10.1007/978-3-030-28151-9_7.

[302] B. M. Dancy and P. A. Cole, "Protein lysine acetylation by p300/CBP," *Chem Rev*, vol. 115, no. 6, pp. 2419–2452, Mar. 2015, doi: 10.1021/CR500452K/ASSET/IMAGES/LARGE/CR-2014-00452K_0021.JPEG.

[303] F. Wang, C. B. Marshall, and M. Ikura, "Transcriptional/epigenetic regulator CBP/p300 in tumorigenesis: Structural and functional versatility in target recognition," *Cellular and Molecular Life Sciences*, vol. 70, no. 21, pp. 3989–4008, Nov. 2013, doi: 10.1007/S00018-012-1254-4/FIGURES/3.

[304] X. J. Yang and M. Ullah, "MOZ and MORF, two large MYSTic HATs in normal and cancer stem cells," *Oncogene 2007 26:37*, vol. 26, no. 37, pp. 5408–5419, Aug. 2007, doi: 10.1038/sj.onc.1210609.

[305] Y. Liu, X. Wang, and B. Liu, "A comprehensive review and comparison of existing computational methods for intrinsically disordered protein and region prediction," *Brief Bioinform*, vol. 20, no. 1, pp. 330–346, Jan. 2019, doi: 10.1093/BIB/BBX126.

[306] L. P. Kozlowski and J. M. Bujnicki, "MetaDisorder: a meta-server for the prediction of intrinsic disorder in proteins," *BMC Bioinformatics*, vol. 13, no. 1, pp. 1–11, May 2012, doi: 10.1186/1471-2105-13-111/TABLES/7.

[307] F. X. Theillet *et al.*, "Physicochemical properties of cells and their effects on intrinsically disordered proteins (IDPs)," *Chem Rev*, vol. 114, no. 13, pp. 6661–6714, Jul. 2014, doi: 10.1021/CR400695P/ASSET/IMAGES/LARGE/CR-2013-00695P_0014.JPEG.

[308] S. Banchenko *et al.*, "Common Mode of Remodeling AAA ATPases p97/CDC48 by Their Disassembling Cofactors ASPL/PUX1," *Structure*, vol. 27, no. 12, pp. 1830-1841.e3, Dec. 2019, doi: 10.1016/j.str.2019.10.001.

[309] L. P. Kozlowski and J. M. Bujnicki, "MetaDisorder: a meta-server for the prediction of

intrinsic disorder in proteins," *BMC Bioinformatics*, vol. 13, no. 1, pp. 1–11, May 2012, doi: 10.1186/1471-2105-13-111/TABLES/7.

[310] T. Ishida and K. Kinoshita, "PrDOS: prediction of disordered protein regions from amino acid sequence," *Nucleic Acids Res*, vol. 35, no. Web Server issue, Jul. 2007, doi: 10.1093/NAR/GKM363.

[311] I. Walsh, A. J. M. Martin, T. di Domenico, A. Vullo, G. Pollastri, and S. C. E. Tosatto, "CSpritz: accurate prediction of protein disorder segments with annotation for homology, secondary structure and linear motifs," *Nucleic Acids Res*, vol. 39, no. suppl_2, pp. W190–W196, Jul. 2011, doi: 10.1093/NAR/GKR411.

[312] D. Piovesan *et al.*, "MobiDB: intrinsically disordered proteins in 2021," *Nucleic Acids Res*, vol. 49, no. D1, pp. D361–D367, Jan. 2021, doi: 10.1093/NAR/GKAA1058.

[313] D. T. Jones and D. Cozzetto, "DISOPRED3: precise disordered region predictions with annotated protein-binding activity," *Bioinformatics*, vol. 31, no. 6, pp. 857–863, Mar. 2015, doi: 10.1093/BIOINFORMATICS/BTU744.

[314] T. Ishida and K. Kinoshita, "Prediction of disordered regions in proteins based on the meta approach," *Bioinformatics*, vol. 24, no. 11, pp. 1344–1348, Jun. 2008, doi: 10.1093/BIOINFORMATICS/BTN195.

[315] B. Xue, R. L. Dunbrack, R. W. Williams, A. K. Dunker, and V. N. Uversky, "PONDR-FIT: A meta-predictor of intrinsically disordered amino acids," *Biochimica et Biophysica Acta (BBA) - Proteins and Proteomics*, vol. 1804, no. 4, pp. 996–1010, Apr. 2010, doi: 10.1016/J.BBAPAP.2010.01.011.

[316] J. Cheng, M. J. Sweredoski, and P. Baldi, "Accurate prediction of protein disordered regions by mining protein structure data," *Data Min Knowl Discov*, vol. 11, no. 3, pp. 213–222, Nov. 2005, doi: 10.1007/S10618-005-0001-Y/TABLES/2.

[317] Z. D.-P. Science and undefined 2018, "Prediction of protein disorder based on IUPred,"

Wiley Online Library, vol. 27, no. 1, pp. 331–340, Jan. 2017, doi: 10.1002/pro.3334.

[318] B. Mészáros, G. Erdös, and Z. Dosztányi, "IUPred2A: Context-Dependent Prediction of Protein Disorder as a Function of Redox State and Protein Binding," *Nucleic Acids Res.*, vol. 46, p. W329, 2018.

[319] D. Konings, L. van Duijn, H. Voorma, and P. Hogeweg, "Minimal energy foldings of eukaryotic mRNAs form a separate leader domain.," *J Theor Biol*, vol. 127, 1987.

[320] J. Prilusky *et al.*, "FoldIndex© A Simple Tool to Predict Whether a Given Protein Sequence Is Intrinsically Unfolded," *Bioinformatics*, vol. 21, p. 3435, 2005.

[321] A. Vullo, O. Bortolamil, G. Pollastri, and S. C. E. Tosatto, "Spritz: a server for the prediction of intrinsically disordered regions in protein sequences using kernel machines," *Nucleic Acids Res*, vol. 34, no. suppl_2, pp. W164–W168, Jul. 2006, doi: 10.1093/NAR/GKL166.

[322] T. Zhang, E. Faraggi, B. Xue, A. K. Dunker, V. N. Uversky, and Y. Zhou, "SPINE-D: Accurate Prediction of Short and Long Disordered Regions by a Single Neural-Network Based Method," *http://dx.doi.org/10.1080/073911012010525022*, vol. 29, no. 4, pp. 799–813, 2012, doi: 10.1080/073911012010525022.

[323] Z. Peng, M. J. Mizianty, and L. Kurgan, "Genome-scale prediction of proteins with long intrinsically disordered regions," *Proteins: Structure, Function, and Bioinformatics*, vol. 82, no. 1, pp. 145–158, Jan. 2014, doi: 10.1002/PROT.24348.

[324] L. J. Mcguffin, "Intrinsic disorder prediction from the analysis of multiple protein fold recognition models," *Bioinformatics*, vol. 24, no. 16, pp. 1798–1804, Aug. 2008, doi: 10.1093/BIOINFORMATICS/BTN326.

[325] I. Walsh, A. J. M. Martin, T. di domenico, and S. C. E. Tosatto, "ESpritz: accurate and fast prediction of protein disorder," *Bioinformatics*, vol. 28, no. 4, pp. 503–509, Feb. 2012, doi: 10.1093/BIOINFORMATICS/BTR682.

[326] J. Hanson, K. K. Paliwal, T. Litfin, and Y. Zhou, "SPOT-Disorder2: Improved Protein Intrinsic Disorder Prediction by Ensembled Deep Learning," *Genomics Proteomics Bioinformatics*, vol. 17, no. 6, pp. 645–656, Dec. 2019, doi: 10.1016/J.GPB.2019.01.004.

[327] T. Ishida and K. Kinoshita, "PrDOS: prediction of disordered protein regions from amino acid sequence," *Nucleic Acids Res*, vol. 35, no. suppl_2, pp. W460–W464, Jul. 2007, doi: 10.1093/NAR/GKM363.

[328] J. J. Ward, L. J. McGuffin, K. Bryson, B. F. Buxton, and D. T. Jones, "The DISOPRED server for the prediction of protein disorder," *Bioinformatics*, 2004, doi: 10.1093/bioinformatics/bth195.

[329] M. Necci, D. Piovesan, Z. Dosztányi, and S. C. Tosatto, "MobiDB-lite: Fast and Highly Specific Consensus Prediction of Intrinsic Disorder in Proteins," *Bioinformatics*, vol. 33, p. 1402, 2017.

[330] S. Vucetic *et al.*, "DisProt: A database of protein disorder," *Bioinformatics*, 2005, doi: 10.1093/bioinformatics/bth476.

[331] D. Piovesan *et al.*, "MobiDB 3.0: More annotations for intrinsic disorder, conformational diversity and interactions in proteins," *Nucleic Acids Res*, 2018, doi: 10.1093/nar/gkx1071.

[332] V. Vacic, V. N. Uversky, A. K. Dunker, and S. Lonardi, "Composition Profiler: A tool for discovery and visualization of amino acid composition differences," *BMC Bioinformatics*, 2007, doi: 10.1186/1471-2105-8-211.

[333] Y. Huang, B. Niu, Y. Gao, L. Fu, and W. Li, "CD-HIT Suite: A web server for clustering and comparing biological sequences," *Bioinformatics*, 2010, doi: 10.1093/bioinformatics/btq003.

[334] F. L. Sirota, H. S. Ooi, T. Gattermayer, G. Schneider, F. Eisenhaber, and S. Maurer-

Stroh, "Parameterization of disorder predictors for large-scale applications requiring high specificity by using an extended benchmark dataset," *BMC Genomics*, 2010, doi: 10.1186/1471-2164-11-S1-S15.

[335] T. Zhang, E. Faraggi, B. Xue, A. K. Dunker, V. N. Uversky, and Y. Zhou, "Spine-d: Accurate prediction of short and long disordered regions by a single neural-network based method," *J Biomol Struct Dyn*, 2012, doi: 10.1080/073911012010525022.

[336] M. J. Mizianty, Z. Peng, and L. Kurgan, "MFDp2," *Intrinsically Disord Proteins*, 2013, doi: 10.4161/idp.24428.

[337] L. P. Kozlowski and J. M. Bujnicki, "MetaDisorder: a meta-server for the prediction of intrinsic disorder in proteins," *BMC Bioinformatics*, 2012, doi: 10.1186/1471-2105-13-111.

[338] L. J. Mcguffin, "Intrinsic disorder prediction from the analysis of multiple protein fold recognition models," in *Bioinformatics*, 2008. doi: 10.1093/bioinformatics/btn326.

[339] B. Xue, R. L. Dunbrack, R. W. Williams, A. K. Dunker, and V. N. Uversky, "PONDR-FIT: A meta-predictor of intrinsically disordered amino acids," *Biochim Biophys Acta Proteins Proteom*, 2010, doi: 10.1016/j.bbapap.2010.01.011.

[340] J. Cheng, M. J. Sweredoski, and P. Baldi, "Accurate prediction of protein disordered regions by mining protein structure data," *Data Min Knowl Discov*, 2005, doi: 10.1007/s10618-005-0001-y.

[341] J. Hanson, K. Paliwal, and Y. Zhou, "Accurate Single-Sequence Prediction of Protein Intrinsic Disorder by an Ensemble of Deep Recurrent and Convolutional Architectures," *J Chem Inf Model*, 2018, doi: 10.1021/acs.jcim.8b00636.

[342] J. Liu and B. Rost, "NORSp: Predictions of long regions without regular secondary structure," *Nucleic Acids Res*, 2003, doi: 10.1093/nar/gkg515.

[343] K. Peng, S. Vucetic, P. Radivojac, C. J. Brown, A. K. Dunker, and Z. Obradovic,

"Optimizing long intrinsic disorder predictors with protein evolutionary information," *J Bioinform Comput Biol*, 2005, doi: 10.1142/S0219720005000886.

[344] T. Young, D. Hazarika, S. Poria, and E. Cambria, "Recent trends in deep learning based natural language processing [Review Article]," *IEEE Computational Intelligence Magazine*. 2018. doi: 10.1109/MCI.2018.2840738.

[345] X. Jing, Q. Dong, D. HONG, and R. Lu, "Amino acid encoding methods for protein sequences: a comprehensive review and assessment," *IEEE/ACM Trans Comput Biol Bioinform*, 2019, doi: 10.1109/tcbb.2019.2911677.

[346] A. Katuwawala, C. J. Oldfield, and L. Kurgan, "Accuracy of protein-level disorder predictions," *Brief Bioinform*, 2019, doi: 10.1093/bib/bbz100.

[347] S. K. Burley and G. A. Petsko, "Aromatic-aromatic interaction: A mechanism of protein structure stabilization," *Science (1979)*, 1985, doi: 10.1126/science.3892686.

[348] C. J. Oldfield and A. K. Dunker, "Intrinsically Disordered Proteins and Intrinsically Disordered Protein Regions," *Annu Rev Biochem*, 2014, doi: 10.1146/annurev-biochem-072711-164947.

[349] J. A. Rodriguez-Serrano, A. Gordo, and F. Perronnin, "Label Embedding: A Frugal Baseline for Text Recognition," *Int J Comput Vis*, 2015, doi: 10.1007/s11263-014-0793-6.

[350] M. Dwarampudi and N. V Reddy, "Effects of padding on LSTMs and CNNs," *arXiv preprint arXiv:1903.07288*, 2019.

[351] C. G. Wermuth, "Similarity in drugs: reflections on analogue design," *Drug Discov Today*, vol. 11, no. 7–8, pp. 348–354, Apr. 2006, doi: 10.1016/J.DRUDIS.2006.02.006.

[352] R. Bade, H. F. Chan, and J. Reynisson, "Characteristics of known drug space. Natural products, their derivatives and synthetic drugs," *Eur J Med Chem*, vol. 45, no. 12, pp. 5646–5652, Dec. 2010, doi: 10.1016/J.EJMECH.2010.09.018.

[353] T. Sato *et al.*, “Structure-based design of selective, orally available salt-inducible kinase inhibitors that stimulate bone formation in mice,” *Proc Natl Acad Sci U S A*, vol. 119, no. 50, p. e2214396119, Dec. 2022, doi: 10.1073/PNAS.2214396119/SUPPL_FILE/PNAS.2214396119.SD06.XLSX.

[354] M. Yu, Y. Yang, M. Sykes, and S. Wang, “Small-Molecule Inhibitors of Tankyrases as Prospective Therapeutics for Cancer,” *J Med Chem*, vol. 65, no. 7, pp. 5244–5273, Apr. 2022, doi: 10.1021/ACS.JMEDCHEM.1C02139/SUPPL_FILE/JM1C02139_SI_001.XLSX.

[355] L. M. Lima, B. N. M. da Silva, G. Barbosa, and E. J. Barreiro, “β-lactam antibiotics: An overview from a medicinal chemistry perspective,” *Eur J Med Chem*, vol. 208, p. 112829, Dec. 2020, doi: 10.1016/J.EJMECH.2020.112829.

[356] J. Fischer and C. Robin Ganellin, “Analogue-based Drug Discovery,” *Analogue-based Drug Discovery*, pp. 1–575, May 2006, doi: 10.1002/3527608001.

[357] J. Fischer and C. Robin Ganellin, “Analogue-based Drug Discovery,” *Analogue-based Drug Discovery*, pp. 1–575, May 2006, doi: 10.1002/3527608001.

[358] J. G. Cumming, A. M. Davis, S. Muresan, M. Haeberlein, and H. Chen, “Chemical predictive modelling to improve compound quality,” *Nature Reviews Drug Discovery 2013 12:12*, vol. 12, no. 12, pp. 948–962, Nov. 2013, doi: 10.1038/nrd4128.

[359] Y. Wang, J. Xiao, T. O. Suzek, J. Zhang, J. Wang, and S. H. Bryant, “PubChem: a public information system for analyzing bioactivities of small molecules,” *Nucleic Acids Res*, vol. 37, no. suppl_2, pp. W623–W633, Jul. 2009, doi: 10.1093/NAR/GKP456.

[360] D. Mendez *et al.*, “ChEMBL: towards direct deposition of bioassay data,” *Nucleic Acids Res*, vol. 47, no. D1, pp. D930–D940, Jan. 2019, doi: 10.1093/NAR/GKY1075.

[361] J. J. Irwin, T. Sterling, M. M. Mysinger, E. S. Bolstad, and R. G. Coleman, “ZINC: A free tool to discover chemistry for biology,” *J Chem Inf Model*, vol. 52, no. 7, pp. 1757–

1768, Jul. 2012, doi: 10.1021/CI3001277/SUPPL_FILE/CI3001277_SI_001.PDF.

[362] P. G. Polishchuk, T. I. Madzhidov, and A. Varnek, “Estimation of the size of drug-like chemical space based on GDB-17 data,” *J Comput Aided Mol Des*, vol. 27, no. 8, pp. 675–679, Aug. 2013, doi: 10.1007/S10822-013-9672-4/FIGURES/2.

[363] A. Gupta, A. T. Müller, B. J. H. Huisman, J. A. Fuchs, P. Schneider, and G. Schneider, “Generative Recurrent Networks for De Novo Drug Design,” *Mol Inform*, vol. 37, no. 1, Jan. 2018, doi: 10.1002/MINF.201700111.

[364] M. H. S. Segler, T. Kogej, C. Tyrchan, and M. P. Waller, “Generating focused molecule libraries for drug discovery with recurrent neural networks,” *ACS Cent Sci*, vol. 4, no. 1, pp. 120–131, Jan. 2018, doi: 10.1021/ACSCENTSCI.7B00512.

[365] D. Polykovskiy *et al.*, “Entangled Conditional Adversarial Autoencoder for de Novo Drug Discovery,” *Mol Pharm*, vol. 15, no. 10, pp. 4398–4405, Oct. 2018, doi: 10.1021/ACS.MOLPHARMACEUT.8B00839.

[366] A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper, and A. Zhavoronkov, “DruGAN: An Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico,” *Mol Pharm*, vol. 14, no. 9, pp. 3098–3104, Sep. 2017, doi: 10.1021/ACS.MOLPHARMACEUT.7B00346.

[367] M. Skalic, D. Sabbadin, B. Sattarov, S. Sciabola, and G. de Fabritiis, “From Target to Drug: Generative Modeling for the Multimodal Structure-Based Ligand Design,” *Mol Pharm*, vol. 16, no. 10, pp. 4282–4291, Oct. 2019, doi: 10.1021/ACS.MOLPHARMACEUT.9B00634.

[368] M. Skalic, J. Jiménez, D. Sabbadin, and G. de Fabritiis, “Shape-Based Generative Modeling for de Novo Drug Design,” *J Chem Inf Model*, vol. 59, no. 3, pp. 1205–1214, Mar. 2019, doi: 10.1021/ACS.JCIM.8B00706.

[369] N. Ståhl, G. Falkman, A. Karlsson, G. Mathiason, and J. Boström, “Deep Reinforcement

Learning for Multiparameter Optimization in de novo Drug Design," *J Chem Inf Model*, vol. 59, no. 7, pp. 3166–3176, Jun. 2019, doi: 10.1021/ACS.JCIM.9B00325.

[370] L. C. Blum and J. L. Reymond, "970 Million druglike small molecules for virtual screening in the chemical universe database GDB-13," *J Am Chem Soc*, vol. 131, no. 25, pp. 8732–8733, Jul. 2009, doi: 10.1021/JA902302H.

[371] L. Ruddigkeit, R. van Deursen, L. C. Blum, and J. L. Reymond, "Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17," *J Chem Inf Model*, vol. 52, no. 11, pp. 2864–2875, Nov. 2012, doi: 10.1021/CI300415D.

[372] R. Gómez-Bombarelli *et al.*, "Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules," *ACS Cent Sci*, vol. 4, no. 2, pp. 268–276, Feb. 2018, doi: 10.1021/ACSCENTSCI.7B00572.

[373] P. Pogány, N. Arad, … S. G.-J. of chemical, and undefined 2018, "De novo molecule design by translating from reduced graphs to SMILES," *ACS Publications*, vol. 59, no. 3, pp. 1136–1146, Mar. 2018, doi: 10.1021/acs.jcim.8b00626.

[374] Y. Li, L. Zhang, and Z. Liu, "Multi-objective de novo drug design with conditional graph generative model," *J Cheminform*, vol. 10, no. 1, Dec. 2018, doi: 10.1186/S13321-018-0287-6.

[375] P. C. Kotsias, J. Arús-Pous, H. Chen, O. Engkvist, C. Tyrchan, and E. J. Bjerrum, "Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks," *Nat Mach Intell*, vol. 2, no. 5, pp. 254–265, May 2020, doi: 10.1038/S42256-020-0174-5.

[376] M. Skalic, D. Sabbadin, B. Sattarov, S. Sciabola, and G. de Fabritiis, "From Target to Drug: Generative Modeling for the Multimodal Structure-Based Ligand Design," *Mol Pharm*, vol. 16, no. 10, pp. 4282–4291, Oct. 2019, doi: 10.1021/ACS.MOLPHARMACEUT.9B00634.

[377] G. T. N. Ton, M. E. Banaszynski, and J. M. Kolesar, “Vandetanib: a novel targeted therapy for the treatment of metastatic or locally advanced medullary thyroid cancer.,” *Am J Health Syst Pharm*, vol. 70, no. 10, pp. 849–855, 2013, doi: 10.2146/ajhp120253.

[378] J. F. Prescott, “Beta-lactam Antibiotics,” *Antimicrobial Therapy in Veterinary Medicine*, pp. 133–152, Sep. 2013, doi: 10.1002/9781118675014.CH8.

[379] T. D. Gootz, “Discovery and development of new antimicrobial agents,” *Clin Microbiol Rev*, vol. 3, no. 1, pp. 13–31, 1990, doi: 10.1128/CMR.3.1.13.

[380] F. E. Koehn and G. T. Carter, “The evolving role of natural products in drug discovery,” *Nature Reviews Drug Discovery 2005 4:3*, vol. 4, no. 3, pp. 206–220, Feb. 2005, doi: 10.1038/nrd1657.

[381] G. A. Showell and J. S. Mills, “Chemistry challenges in lead optimization: silicon isosteres in drug discovery,” *Drug Discov Today*, vol. 8, no. 12, pp. 551–556, Jun. 2003, doi: 10.1016/S1359-6446(03)02726-0.

[382] D. J. Newman, G. M. Cragg, and K. M. Snader, “Natural Products as Sources of New Drugs over the Period 1981−2002,” *J Nat Prod*, vol. 66, no. 7, pp. 1022–1037, Jul. 2003, doi: 10.1021/NP030096L.

[383] E. Serrao, S. Odde, K. Ramkumar, and N. Neamati, “Raltegravir, elvitegravir, and metoogravir: The birth of ‘me-too’ HIV-1 integrase inhibitors,” *Retrovirology*, vol. 6, no. 1, pp. 1–14, Mar. 2009, doi: 10.1186/1742-4690-6-25/TABLES/1.

[384] Y. L. Bennani, “Drug discovery in the next decade: innovation needed ASAP,” *Drug Discov Today*, vol. 16, no. 17–18, pp. 779–792, Sep. 2011, doi: 10.1016/J.DRUDIS.2011.06.004.

[385] S. H. Preskorn, “The evolution of antipsychotic drug therapy: Reserpine, chlorpromazine, and haloperidol,” *J Psychiatr Pract*, vol. 13, no. 4, pp. 253–257, Jul. 2007, doi: 10.1097/01.PRA.0000281486.34817.8B.

[386] C. R. Wilson, L. Sherritt, E. Gates, and J. R. Knight, “Are clinical impressions of adolescent substance use accurate?,” *Pediatrics*, vol. 114, no. 5, Nov. 2004, doi: 10.1542/peds.2004-0098.

[387] J. C. Blanchard and L. Julou, “Receptor binding methods: application to the study of cyclopyrrolones, a new family of minor tranquillizers,” *Veterinary Pharmacology and Toxicology*, pp. 437–449, 1983, doi: 10.1007/978-94-009-6604-8_44.

[388] A. M. Clark, “Natural products as a resource for new drugs,” *Pharm Res*, vol. 13, no. 8, pp. 1133–1141, 1996, doi: 10.1023/A:1016091631721/METRICS.

[389] G. Li and H. X. Lou, “Strategies to diversify natural products for drug discovery,” *Med Res Rev*, vol. 38, no. 4, pp. 1255–1294, Jul. 2018, doi: 10.1002/MED.21474.

[390] A. Fura, Y. Z. Shu, M. Zhu, R. L. Hanson, V. Roongta, and W. G. Humphreys, “Discovering drugs through biological transformation: Role of pharmacologically active metabolites in drug discovery,” *J Med Chem*, vol. 47, no. 18, pp. 4339–4351, Aug. 2004, doi: 10.1021/JM040066V/ASSET/JM040066V.FP.PNG_V03.

[391] H. van de Waterbeemd and E. Gifford, “ADMET in silico modelling: towards prediction paradise?,” *Nature Reviews Drug Discovery 2003 2:3*, vol. 2, no. 3, pp. 192–204, Mar. 2003, doi: 10.1038/nrd1032.

[392] A. F. Nassar, “Role of Structural Modifications of Drug Candidates to Enhance Metabolic Stability,” *Drug Metabolism Handbook*, pp. 303–322, Dec. 2022, doi: 10.1002/9781119851042.CH9.

[393] J. A. Castillo-Garit, Y. Cañizares-Carmenate, H. Pham-The, V. Pérez-Doñate, F. Torrens, and F. Pérez-Giménez, “A Review of Computational Approaches Targeting SARS-CoV-2 Main Protease to the Discovery of New Potential Antiviral Compounds,” *Curr Top Med Chem*, vol. 22, Apr. 2022, doi: 10.2174/2667387816666220426133555.

[394] M. Kumari *et al.*, “A critical overview of current progress for COVID-19: development of vaccines, antiviral drugs, and therapeutic antibodies,” *Journal of Biomedical Science*

2022 29:1, vol. 29, no. 1, pp. 1–36, Sep. 2022, doi: 10.1186/S12929-022-00852-9.

[395] S. Ghahremanian, M. M. Rashidi, K. Raeisi, and D. Toghraie, “Molecular dynamics simulation approach for discovering potential inhibitors against SARS-CoV-2: A structural review,” *J Mol Liq*, vol. 354, p. 118901, May 2022, doi: 10.1016/J.MOLLIQ.2022.118901.

[396] P. G. Polishchuk, T. I. Madzhidov, and A. Varnek, “Estimation of the size of drug-like chemical space based on GDB-17 data,” *J Comput Aided Mol Des*, vol. 27, no. 8, pp. 675–679, 2013, doi: 10.1007/S10822-013-9672-4.

[397] L. Ruddigkeit, L. C. Blum, and J. L. Reymond, “Visualization and virtual screening of the chemical universe database GDB-17,” *J Chem Inf Model*, vol. 53, no. 1, pp. 56–65, Jan. 2013, doi: 10.1021/ci300535x.

[398] T. Miyao, H. Kaneko, and K. Funatsu, “Inverse QSPR/QSAR Analysis for Chemical Structure Generation (from y to x),” *J Chem Inf Model*, vol. 56, no. 2, pp. 286–299, Feb. 2016, doi: 10.1021/ACS.JCIM.5B00628.

[399] M. Wang *et al.*, “Deep learning approaches for de novo drug design: An overview,” *Curr Opin Struct Biol*, vol. 72, pp. 135–144, Feb. 2022, doi: 10.1016/J.SBI.2021.10.001.

[400] “2020 | MIT Technology Review.” https://www.technologyreview.com/10-breakthrough-technologies/2020/ (accessed Jan. 04, 2023).

[401] P. Pogány, N. Arad, S. Genway, and S. D. Pickett, “De Novo Molecule Design by Translating from Reduced Graphs to SMILES,” *J Chem Inf Model*, vol. 59, no. 3, pp. 1136–1146, Mar. 2019, doi: 10.1021/ACS.JCIM.8B00626.

[402] X. Yang, J. Zhang, K. Yoshizoe, K. Terayama, and K. Tsuda, “ChemTS: an efficient python library for de novo molecular generation,” *Sci Technol Adv Mater*, vol. 18, no. 1, pp. 972–976, Dec. 2017, doi: 10.1080/14686996.2017.1401424.

[403] A. Gupta, A. T. Müller, B. J. H. Huisman, J. A. Fuchs, P. Schneider, and G. Schneider,

"Generative Recurrent Networks for De Novo Drug Design," *Mol Inform*, vol. 37, no. 1, Jan. 2018, doi: 10.1002/MINF.201700111.

[404] P. S. Kutchukian and E. I. Shakhnovich, "De novo design: Balancing novelty and confined chemical space," *Expert Opin Drug Discov*, vol. 5, no. 8, pp. 789–812, Aug. 2010, doi: 10.1517/17460441.2010.497534.

[405] Y. Zhang, M. Luo, P. Wu, S. Wu, T.-Y. Lee, and C. Bai, "Application of Computational Biology and Artificial Intelligence in Drug Design," *Int J Mol Sci*, vol. 23, no. 21, p. 13568, Nov. 2022, doi: 10.3390/IJMS232113568.

[406] M. H. S. Segler, T. Kogej, C. Tyrchan, and M. P. Waller, "Generating focused molecule libraries for drug discovery with recurrent neural networks," *ACS Cent Sci*, vol. 4, no. 1, pp. 120–131, Jan. 2018, doi: 10.1021/ACSCENTSCI.7B00512.

[407] M. Xu, T. Ran, and H. Chen, "De Novo Molecule Design through the Molecular Generative Model Conditioned by 3D Information of Protein Binding Sites," *J Chem Inf Model*, vol. 61, no. 7, pp. 3240–3254, Jul. 2021, doi: 10.1021/ACS.JCIM.0C01494.

[408] M. Skalic, J. Jiménez, D. Sabbadin, and G. de Fabritiis, "Shape-Based Generative Modeling for de Novo Drug Design," *J Chem Inf Model*, vol. 59, no. 3, pp. 1205–1214, Mar. 2019, doi: 10.1021/ACS.JCIM.8B00706.

[409] Y. Li *et al.*, "Highly efficient actively Q-switched Yb:LGGG laser generating 3.26 mJ of pulse energy," *Opt Mater (Amst)*, vol. 79, pp. 33–37, May 2018, doi: 10.1016/j.optmat.2018.03.022.

[410] M. Skalic, D. Sabbadin, B. Sattarov, S. Sciabola, and G. de Fabritiis, "From Target to Drug: Generative Modeling for the Multimodal Structure-Based Ligand Design," *Mol Pharm*, vol. 16, no. 10, pp. 4282–4291, Oct. 2019, doi: 10.1021/ACS.MOLPHARMACEUT.9B00634.

[411] A. Kadurin, S. Nikolenko, K. Khrabrov, A. Aliper, and A. Zhavoronkov, "DruGAN: An

Advanced Generative Adversarial Autoencoder Model for de Novo Generation of New Molecules with Desired Molecular Properties in Silico," *Mol Pharm*, vol. 14, no. 9, pp. 3098–3104, Sep. 2017, doi: 10.1021/ACS.MOLPHARMACEUT.7B00346.

[412] N. Brown, M. Fiscato, M. H. S. Segler, and A. C. Vaucher, "GuacaMol: Benchmarking Models for de Novo Molecular Design," *J Chem Inf Model*, vol. 59, no. 3, pp. 1096–1108, Mar. 2019, doi: 10.1021/ACS.JCIM.8B00839.

[413] N. Ståhl, G. Falkman, A. Karlsson, G. Mathiason, and J. Boström, "Deep Reinforcement Learning for Multiparameter Optimization in de novo Drug Design," *J Chem Inf Model*, vol. 59, no. 7, pp. 3166–3176, Jun. 2019, doi: 10.1021/ACS.JCIM.9B00325.

[414] F. Cheng and N. Tuncbag, "Editorial overview: Artificial intelligence (AI) methodologies in structural biology," *Curr Opin Struct Biol*, vol. 74, Jun. 2022, doi: 10.1016/j.sbi.2022.102387.

[415] R. Gómez-Bombarelli *et al.*, "Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules," *ACS Cent Sci*, vol. 4, no. 2, pp. 268–276, Feb. 2018, doi: 10.1021/ACSCENTSCI.7B00572.

[416] Y. Qian Zhao, X. Hong Wang, X. Fang Wang, and F. Y. Shih, "Retinal vessels segmentation based on level set and region growing," *Pattern Recognit*, vol. 47, no. 7, pp. 2437–2446, 2014, doi: 10.1016/j.patcog.2014.01.006.

[417] P. C. Kotsias, J. Arús-Pous, H. Chen, O. Engkvist, C. Tyrchan, and E. J. Bjerrum, "Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks," *Nat Mach Intell*, vol. 2, no. 5, pp. 254–265, May 2020, doi: 10.1038/S42256-020-0174-5.

[418] D. Polykovskiy *et al.*, "Entangled Conditional Adversarial Autoencoder for de Novo Drug Discovery," *Mol Pharm*, vol. 15, no. 10, pp. 4398–4405, Oct. 2018, doi: 10.1021/ACS.MOLPHARMACEUT.8B00839.

[419] H. Chen, O. Engkvist, Y. Wang, M. Olivecrona, and T. Blaschke, “The rise of deep learning in drug discovery,” *Drug Discov Today*, vol. 23, no. 6, pp. 1241–1250, Jun. 2018, doi: 10.1016/j.drudis.2018.01.039.

[420] H. Yong, J. Huang, D. Meng, X. Hua, and L. Zhang, “Momentum Batch Normalization for Deep Learning with Small Batch Size,” *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12357 LNCS, pp. 224–240, 2020, doi: 10.1007/978-3-030-58610-2_14/COVER.

[421] S.-H. Gao, Q. Han, D. Li, M.-M. Cheng, P. Peng, and N. University, “Representative Batch Normalization With Feature Calibration.” pp. 8669–8679, 2021. Accessed: Dec. 27, 2022. [Online]. Available: http://mmcheng.net/rbn

[422] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, “Understanding Batch Normalization,” *Adv Neural Inf Process Syst*, vol. 31, 2018.

[423] S. Santurkar, D. Tsipras, A. Ilyas, and A. M. ¸ A. Mit, “How Does Batch Normalization Help Optimization?,” *Adv Neural Inf Process Syst*, vol. 31, 2018.

[424] G. van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009.

[425] G. Landrum *et al.*, “rdkit/rdkit: 2022_09_3 (Q3 2022) Release,” Dec. 2022, doi: 10.5281/ZENODO.7415128.

[426] N. M. O’Boyle, M. Banck, C. A. James, C. Morley, T. Vandermeersch, and G. R. Hutchison, “Open Babel: An Open chemical toolbox,” *J Cheminform*, vol. 3, no. 10, pp. 1–14, Oct. 2011, doi: 10.1186/1758-2946-3-33/TABLES/2.

[427] P. P. Knowles *et al.*, “Structure and Chemical Inhibition of the RET Tyrosine Kinase Domain,” *Journal of Biological Chemistry*, vol. 281, no. 44, pp. 33577–33587, Nov. 2006, doi: 10.1074/JBC.M605604200.

[428] G. M. Morris *et al.*, "AutoDock4 and AutoDockTools4: Automated docking with selective receptor flexibility," *J Comput Chem*, vol. 30, no. 16, pp. 2785–2791, Dec. 2009, doi: 10.1002/JCC.21256.

[429] A. ElTijani, M. Y. Alsafi, and A. F. Ahmed, "EasyDockVina: Graphical Interface for Ligand Optimization and High Throughput Virtual Screening with Vina," Sep. 2019, doi: 10.5281/ZENODO.3732170.

[430] J. Lee *et al.*, "CHARMM-GUI Input Generator for NAMD, GROMACS, AMBER, OpenMM, and CHARMM/OpenMM Simulations Using the CHARMM36 Additive Force Field," *J Chem Theory Comput*, vol. 12, no. 1, pp. 405–413, Jan. 2016, doi: 10.1021/ACS.JCTC.5B00935/ASSET/IMAGES/LARGE/CT-2015-00935E_0005.JPEG.

[431] K. Vanommeslaeghe *et al.*, "CHARMM general force field: A force field for drug-like molecules compatible with the CHARMM all-atom additive biological force fields," *J Comput Chem*, vol. 31, no. 4, pp. 671–690, Mar. 2010, doi: 10.1002/JCC.21367.

[432] M. J. Abraham *et al.*, "GROMACS: High performance molecular simulations through multi-level parallelism from laptops to supercomputers," *SoftwareX*, vol. 1–2, pp. 19–25, Sep. 2015, doi: 10.1016/J.SOFTX.2015.06.001.

[433] N. Homeyer and H. Gohlke, "Free Energy Calculations by the Molecular Mechanics Poisson−Boltzmann Surface Area Method," *Mol Inform*, vol. 31, no. 2, pp. 114–122, Feb. 2012, doi: 10.1002/MINF.201100135.

[434] M. S. Valdés-Tresanco, M. E. Valdés-Tresanco, P. A. Valiente, and E. Moreno, "Gmx_MMPBSA: A New Tool to Perform End-State Free Energy Calculations with GROMACS," *J Chem Theory Comput*, vol. 17, no. 10, pp. 6281–6291, Oct. 2021, doi: 10.1021/ACS.JCTC.1C00645/ASSET/IMAGES/LARGE/CT1C00645_0005.JPEG.

[435] S. Genheden and U. Ryde, "The MM/PBSA and MM/GBSA methods to estimate

ligand-binding affinities," *http://dx.doi.org/10.1517/17460441.2015.1032936*, vol. 10, no. 5, pp. 449–461, May 2015, doi: 10.1517/17460441.2015.1032936.

[436] N. Brown, M. Fiscato, M. H. S. Segler, and A. C. Vaucher, "GuacaMol: Benchmarking Models for de Novo Molecular Design," *J Chem Inf Model*, vol. 59, no. 3, pp. 1096–1108, Mar. 2019, doi: 10.1021/ACS.JCIM.8B00839.

[437] D. Polykovskiy *et al.*, "Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models," *Front Pharmacol*, vol. 11, p. 1931, Dec. 2020, doi: 10.3389/FPHAR.2020.565644/BIBTEX.

[438] F. Urbina, C. T. Lowden, J. C. Culberson, and S. Ekins, "MegaSyn: Integrating Generative Molecular Design, Automated Analog Designer, and Synthetic Viability Prediction," *ACS Omega*, vol. 7, no. 22, pp. 18699–18713, Jun. 2022, doi: 10.1021/ACSOMEGA.2C01404/SUPPL_FILE/AO2C01404_SI_002.ZIP.

[439] R. A. Laskowski and M. B. Swindells, "LigPlot+: multiple ligand-protein interaction diagrams for drug discovery," *J Chem Inf Model*, vol. 51, no. 10, pp. 2778–2786, Oct. 2011, doi: 10.1021/CI200227U.

[440] M. Hernández-Rodríguez, M. C. Rosales-Hernández, J. E. Mendieta-Wejebe, M. Martínez-Archundia, and J. Correa Basurto, "Current Tools and Methods in Molecular Dynamics (MD) Simulations for Drug Design," *Curr Med Chem*, vol. 23, no. 34, pp. 3909–3924, Jun. 2016, doi: 10.2174/0929867323666160530144742.

[441] X. Yang, J. Zhang, K. Yoshizoe, K. Terayama, and K. Tsuda, "ChemTS: an efficient python library for de novo molecular generation," *Sci Technol Adv Mater*, vol. 18, no. 1, pp. 972–976, Dec. 2017, doi: 10.1080/14686996.2017.1401424.

[442] M. Wang *et al.*, "Deep learning approaches for de novo drug design: An overview," *Curr Opin Struct Biol*, vol. 72, pp. 135–144, Feb. 2022, doi: 10.1016/J.SBI.2021.10.001.

Chapter 8. List of publications

Publications from thesis

1. **Chakraborty R**, Hasija Y. Predicting microRNA sequence using CNN and LSTM stacked in Seq2Seq architecture. *IEEE/ACM transactions on computational biology and bioinformatics*. 2019 Aug 20;17(6):2183-8.
2. **Chakraborty R**, Hasija Y. Predicting protein intrinsically disordered regions by applying natural language processing practices. *Soft Computing*. 2022 May 5:1-1.
3. **Chakraborty R**, Hasija Y. Utilizing Deep Learning to Explore Chemical Space for Drug Analogues Generation. Expert Systems with Applications. 2023 Jun 1:120592.

Other publications

1. Rangra S , **Chakraborty R**, Hasija Y and Aggarwal K.K. A cystatin C similar protein from Musa acuminata that inhibits cathepsin B involved in rheumatoid arthritis using in-silico approach and in vitro cathepsin B inhibition by protein extract. Journal of Biomolecular Structure & Dynamics. 2022 December.(Accepted)
2. **Chakraborty R**, Bhattacharje G, Baral J, Manna B, Mullick J, Mathapati BS, Abraham P, Madhumathi J, Hasija Y, Ghosh A, Das AK. In-silico screening and in-vitro assay show the antiviral effect of Indomethacin against SARS-CoV-2. *Computers in Biology and Medicine*. 2022 Jun 30:105788.
3. Chatterjee S, **Chakraborty R**, Hasija Y. Polymorphisms at site 469 of B-RAF protein associated with skin melanoma may be correlated with dabrafenib resistance: An in silico study. *Journal of Biomolecular Structure and Dynamics*. 2021 Jul 1:1-6.
4. **Chakraborty R**, Gupta H, Rahman R, Hasija Y. In silico analysis of nsSNPs in ABCB1 gene affecting breast cancer associated protein P-glycoprotein (P-gp). *Computational Biology and Chemistry*. 2018 Dec 1; 77:430-41.

Book

1. Hasija Y, **Chakraborty R**. Hands on Data Science for Biologists Using Python. CRC Press; 2021 Apr 8.

Book chapters

1. Nayak R, **Chakraborty R**, Hasija Y. System biology and synthetic biology. Translational Biotechnology. 2021 Jan 1:329-44.

Conferences and presentations

1. Shiva M, **Chakraborty R**, Sharma I, Hasija Y. DermaMeth: Human DNA Methylation Database for Genes Associated with Dermatological Disorders. In 2019 International Conference on Computing, Power and Communication Technologies (GUCON) 2019 Sep 27 (pp. 1-7). IEEE.
2. **Chakraborty R**, Kedia S, Hasija Y. Integrated transcriptional meta-analysis of pigmentation disorders. InBiotechnology and Biological Sciences 2019 Nov 20 (pp. 159-164). CRC Press.
3. **Chakraborty R**, Hasija Y. miDerma: An Integrated Database and Tool for Analysis of miRNAs associated with Dermatological Disorders. In2018 International Conference on Bioinformatics and Systems Biology (BSB) 2018 Oct 26 (pp. 170-173). IEEE.

Predicting MicroRNA Sequence Using CNN and LSTM Stacked in Seq2Seq Architecture

Rajkumar Chakraborty and Yasha Hasija

Abstract—CNN and LSTM have proven their ability in feature extraction and natural language processing, respectively. So, we tried to use their ability to process the language of RNAs, i.e., predicting sequence of microRNAs using the sequence of mRNA. The idea is to extract the features from sequence of mRNA using CNN and use LSTM network for prediction of miRNA. The model has learned the basic features such as seed match at first 2–8 nucleotides starting at the $5'$ end and counting toward the $3'$ end. Also, it was able to predict G-U wobble base pair in seed region. While validating on experimentally validated data, the model was able to predict on average 72 percent of miRNAs for specific mRNA and shows highest positive expression fold change of predicted targets on a microarray data generated using anti 25 miRNAs compare to other predicted tools. Codes are available at https://github.com/rajkumar1501/sequence-prediction-using-CNN-and-LSTMs

Index Terms—microRNA Target prediction, CNNs, LSTMs, seq2seq architecture, neural networks

1 Introduction

Deep neural networks are machine learning algorithms which are inspired by biological neural networks, i.e., how our brain learns [1]. Since their development in mid of 20th century, they did not get much application as they are computationally intensive. The inception of modern hardware systems, especially the GPUs (Graphical Processing Units) with more excellent computational capability, has allowed neural networks to regain popularity and applications. Additionally, particular types of neural networks, such as convolutional neural networks(CNNs) and recurrent neural networks(RNNs) with long short-term memory cells(LSTM) find their applications in various fields like image processing, speech recognition and natural language processing. ANNs, CNNs and LSTM have also been used in solving various biological problems including prediction of protein secondary structure, protein sub-cellular localisation, peptide binding to MHC-II molecules, prediction of methyladenosine sites in mRNA, image recognition of skin disorders etc. [2], [3], [4], [5].

A typical neural network learns by adjusting its weights or priority of any given features for calculating the values close to the given output, i.e., values with minimum error. The weights are calculated according to the difference between calculated output and given output using an algorithm known as Gradient descent [6]. Neural networks require features as input and providing meaningful features, or feature selection is a tedious task. Various robust feature selection techniques are available such as LibD3C, MRMD and CNNs [7], [8], [9]. For ease of integration with ANNs, CNNs are preferred over other techniques. CNNs are a type of neural networks which are used for feature extraction when specific features or patterns cannot be determined, such as in image and sequence data. CNNs slide a grid, also known as filters (a set of weights), over the input data which are fed into different neurons of imiditade hidden layer every time the filter is moved. These filters are not interconnected, and while moving through the data, it can extract patterns or features from the input irrespective of the position where they are found [10].

RNNs are a kind of neural network architecture which can deal with sequential data such as time series data and text data. They have typical feedforward connections, but also, the neurons of hidden layers are connected with a time-delayed connection for retaining the weights of previous time-step [11]. In this way, RNNs are capable of learning through sequence data by storing instincts from previous elements and analysing the present element in context of the previous one. Long short-term memory (LSTM) are a special kind of RNNs where the simple matrix of hidden neuron is succeeded with the LSTM memory block which minimises the vanishing gradient problem [12]. LSTMs have a memory block cell where context-dependent weights are achieved. This block is further controlled by input, output and forget control gates so that it can read the input sequence and decide the elements it should keep in the cell for each time-step. So, it is easier for LSTMs to save a given input feature over many time frames which is its advantage over RNNs [13]. Previously, LSTMs have been used in building chatbots, language translation, image captioning etc. [14].

The primary purpose of this work is to utilise above-stated speciality of CNN and LSTM architecture in prediction of miRNA sequences based on target mRNA segment where they bound. The problem stated is a typical sequence to sequence prediction problem where LSTMs are most suitable. The human genome encodes for over 2200 microRNAs, which are mostly 28bp long, non-coding RNA molecules. These molecules play an essential role in regulating post-transcriptional gene expression [15]. Since one microRNA can target multiple gene transcripts, microRNAs are known to be involves in regulating gene expression and mRNA translation mechanisms [15]. Recently, there have been many evidences, which suggests the importance of miRNAs in human diseases [16]. Studies have shown that mutations, dysregulations or even dysfunction of miRNA biogenesis and their targets lead to the obstruction in physiological and biochemical pathways which cause various diseases in human [17]. Considering the massive presence of these regulatory RNAs, their diverse expression along with significant number of mRNA targets, it is not surprising that miRNAs have been playing crucial role in a broad range of diseases including immunological diseases [18], cancers [19], and various skin diseases [20]. The connection between miRNA and mRNA can be resolved experimentally utilising different techniques, for example, HITS-CLIP [21], PAR-CLIP [22], CLASH [23] etc. However, these methods are cumbersome and occasionally require large number of cells for library preparation. Computational prediction of these associations may aid the scientific community by prioritising microRNAs that could be used as markers for change in mRNA expression.

In-silico prediction of microRNA targets, which has been a mind storming challenge for scientific community for decades, is a basic fundamental step in finding microRNA-mRNA target association. The current methods for microRNA target predictions incorporate various computational methodologies, from the demonstration of physical association algorithms to the application of machine learning algorithms [24]. Algorithms have been developed using consensus properties which have been identified such as, seed site complementary binding, homology of miRNAs etc. As every properties are not universal for every miRNA and RNA interactions so machine learning base tools are also developed using these properties as features like site accessibility, evolutionary conservation, free energy etc. RNAhybrid [25], PicTar [26], TargetScan [27], PITA

- *The authors are with Delhi Technological University, Delhi 110042, India. E-mail: raj150194@gmail.com, yashahasija@dtu.ac.in.*

Manuscript received 14 May 2019; revised 24 July 2019; accepted 13 Aug. 2019. Date of publication 20 Aug. 2019; date of current version 8 Dec. 2020.
(Corresponding author: Yasha Hasija.)
Digital Object Identifier no. 10.1109/TCBB.2019.2936186

Soft Computing (2022) 26:12343–12353
https://doi.org/10.1007/s00500-022-07085-w

APPLICATION OF SOFT COMPUTING

Predicting protein intrinsically disordered regions by applying natural language processing practices

Rajkumar Chakraborty[1] · Yasha Hasija[1]

Accepted: 30 March 2022 / Published online: 5 May 2022

Abstract

Intrinsically disordered regions (IDRs) in proteins are the regions that lack a stable two-dimensional or three-dimensional structure. Due to their high degree of flexibility, these regions are required for a variety of cellular functions. The IDRs are determined experimentally using X-ray crystallography and NMR. Numerous computational techniques for IDR prediction have been developed, but due to the certain unknown amino acid properties and interactions, these techniques have a low predictive rate. IDR-amino acid enrichment analyses on protein chains with varying structural and physicochemical properties have shed light on a variety of features of IDRs. Additionally, repetitions of certain specific amino acids that appear to be preferentially present in the IDRs were observed. Following that, a deep neural network model inspired by natural language processing techniques was trained to learn amino acid features for the classification of IDRs and non-IDRs. We portrayed amino acids as single letters that comprise the proteome's language. Our method outperformed other IDR prediction tools currently available. This study can assist researchers in comprehending IDR constituents and utilizing NLP techniques for processing genomic or proteomic language in order to gain additional insight from them. The executable package and the dataset can be found at https://doi.org/10.24433/CO.3457808.v1.

Keywords IDR prediction · Protein intrinsic disorders prediction · IDRs properties · CNNs · LSTMs · Neural networks · Natural language processing

1 Introduction

In the early days, proteins were considered to have stable tertiary structures for performing molecular functions. In 1998, Romero et al. (Romero et al. 1998) proposed that around 15,000 protein sequences stored in SwissProt have regions which do not have well-defined 3D structures. Researchers then had to reassess the proteins' stable 3D structure theory. The proteins or regions of a protein that do not have a stable tertiary structure are termed as "intrinsically disordered proteins" (IDPs) or "intrinsically disordered regions" (IDRs), respectively (Dunker et al. 2013). IDRs can be short (< 30 residues) or long (> 30 residues), or the whole protein can be disordered. These regions are critical for cellular functions where in IDRs are used to distinguish between unfolded and partially folded protein structures (Reichmann et al. 2012). IDRs are present in proteins' binding motifs, assisting them in binding to other proteins (Neduva et al. 2005), allowing for the flow of information via cell signaling (Uversky et al. 2005). The flexibility of IDRs enables variation in binding sites and binding partners, providing new insights into molecular recognition mechanisms(Dyson and Wright 2002). Due to the crucial role IDRs play in the function of living cells, variations and mutations in these regions are frequently associated with a variety of diseases such as cancer, neurodegenerative, and cardiovascular diseases and are also considered putative drug targets(Cheng et al. 2006). IDPs and IDRs are present in almost every life form in significant numbers, which shows their importance in the living systems (Xue et al. 2012).

Experimental methods such as X-ray crystallography (Tompa 2012) and NMR (Ota et al. 2013) can be used to characterize these disordered regions in protein structures.

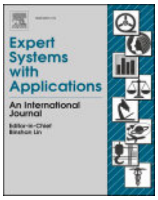
✉ Yasha Hasija
yashahasija06@gmail.com

[1] Department of Biotechnology, Delhi Technological University, Main Bawana Road, Shahbad, Daulatpur, New Delhi 110042, India

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Utilizing deep learning to explore chemical space for drug lead optimization

Rajkumar Chakraborty, Yasha Hasija *

Department of Biotechnology, Delhi Technological University, Delhi 110042, India

ARTICLE INFO

Keywords:
Drug lead optimization
Autoencoders
Drug design
MD simulations

ABSTRACT

The goal of medicinal chemistry is to improve on existing drug molecules or to create new ones for use in medicine. This is frequently accomplished by lead optimization, which entails creating similar but slightly modified versions of existing molecules. Generative models that use various representations of molecules, such as SMILES codes and molecular graphs, have been developed to aid in the search for hits in the unexplored chemical space. In this study, an autoencoder architecture was trained on chemical SMILES from the ChEMBL database to generate 157 analogues of Vandetanib by introducing noise to its latent representation. The distribution of the autoencoder's latent space was controlled by varying batch sizes during the reconstruction of chemical SMILES. Virtual screening and molecular dynamics simulations were conducted, and it was found that at least two analogues had a higher binding affinity than the control compound, demonstrating the potential of this approach for lead optimization. This architecture has a small number of parameters and has the potential to generate a wide variety of molecules. The model is implemented in Google Colaboratory notebook to be explored by scientific community via https://colab.research.google.com/drive/1BPhw7_-_VV11dbk6s9JGE0bSX0K_-qIh?usp=sharing.

1. Introduction

Medicinal chemists play an important role to enhance an existing active molecule, whether it be natural or synthetic (i.e., through the practise of lead optimization) (Wermuth, 2006). Drug lead optimization is a critical step in the drug discovery and development process where a lead compound with potential as a drug candidate is modified and improved to enhance its efficacy, safety, pharmacokinetic properties and drug-like characteristics. The use of lead optimization has been successful, simple to manufacture, and widely accepted in pharmaceutical research since its inception. According to a review, 10% of pharmaceuticals on the market are unaltered compound which exists in nature, 29% are their derivatives (semi-synthetics), and the rest 61% are synthetic (Bade, Chan, & Reynisson, 2010). Using commercially accessible drug structures as a starting point for research (i.e., Analogues-based drug design), results in iterative adjustments that improve therapeutic molecules' efficacy and safety (Sato et al., 2022; Yu, Yang, Sykes, & Wang, 2022). For example, the 55 years of Analogues design research that followed the creation of the historical antibiotic penicillin G allowed for the development of broad-spectrum, modern, orally active-lactam antibiotics such as ampicillin and amoxicillin(Lima, da Silva, Barbosa, & Barreiro, 2020). Neuroleptics, antidepressants, and antihypertensive medications all underwent similar changes (Fischer & Robin Ganellin, 2006).

In lead optimization, Analogues which are similar to a known lead molecule, are usually created through small molecular changes such as the formation of homologues, vinylogues, isosteres, positional isomers, optical isomers, and altered ring systems(Fischer & Robin Ganellin, 2006). The basic structure of the molecule is usually preserved or only slightly modified. Substituent effects, which can be used to fine-tune the molecule, can be analysed using techniques like QSAR (Cumming et al., 2013). Given that the Analogues are derived from a known therapeutic molecule as a starting point, drug-like properties are not a major concern. When necessary, filters are used to remove reactive or toxic groups and consider qualities related to absorption, distribution, metabolism, and excretion (ADME). Drug lead optimization plays a crucial role in reducing the time and costs associated with traditional drug discovery and helps to identify promising drug candidates for further development. This process often involves various stages such as hit-to-lead and lead optimization and utilizes a range of techniques

* Corresponding author.
E-mail addresses: yashahasija6oct@gmail.com, yashahasija@dtu.ac.in (Y. Hasija).

https://doi.org/10.1016/j.eswa.2023.120592
Received 17 February 2023; Received in revised form 21 April 2023; Accepted 27 May 2023
Available online 1 June 2023

RAJKUMAR CHAKRABORTY

PROFILE

Budding bioinformatics researcher seeking challenging opportunity in Genomics, Proteomics, and Drug Designing domain for utilizing Natural Language Processing, Generative Neural Network Models, protein modeling, and bioinformatics data analysis skills. Ph.D. was focused on Biological Language Model Development. Have strong analytical and problem-solving aptitude.

CONTACT

PHONE:
8436374454
EMAIL:
raj150194@gmail.com
Address:
VILL AND POST - JORDA, P.S. - INDPUR
BANKURA- 722173, W.B
India

PUBLICATIONS

Journal Publications:	7
Book:	1
Book Chapter:	1
Conference publications:	3
Citations:	36
h-index:	3

Web of Science ID: **U-1802-2019**
ORCiD: **0000-0002-7091-3935**

HOBBIES

Travelling
Cooking

PERSONAL DETAILS

Nationality	Indian
Language proficiency	English, Hindi, Bengali
Date of Birth	15th January, 1994
Marital status	Unmarried

WORK EXPERIENCE

Aganitha AI Inc.
March 2023 – Ongoing
Junior Scientist-2

Delhi Technological University- Guest faculty
August 2022 – December 2022
Teaching "Fundamentals of Computational Biology" Course to B. Tech 7th Sem Students.

EDUCATION

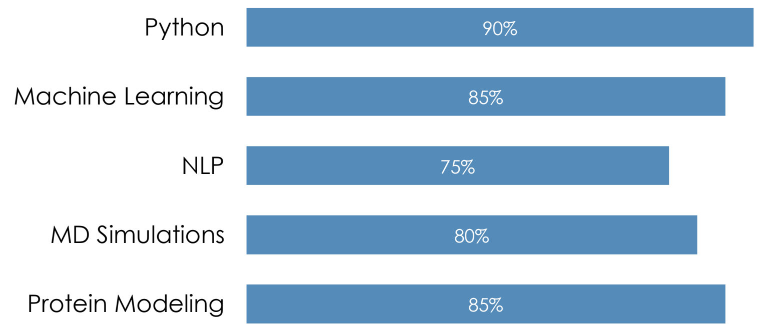
Ph.D. – Delhi Technological University
August 2018–Present (Defense scheduled on 15th June, 2023)

- Worked on thesis titled "Genomic Language Processing Using Machine Learning"
- Demonstrated a strong commitment to research and made significant contributions to the development of **Biological Language models** for **genomics, proteomics and drug development using deep learning techniques**. Gained extensive knowledge with strong understanding of the underlying concepts and methods in techniques like **protein modelling, docking, and molecular dynamics simulations**.

M.Tech (Bioinformatics) – Delhi Technological University
August 2016–July 2018
Worked on project titled "mirBoT: A mircoRNA sequence prediction from RNA sequence tool base on LSTMs and seq2seq architecture."
Developed miDerma database.

B.Tech (Biotechnology) – Bengal College of Engineering and Technology
August 2011–July 2017
Project title – "Quantifying phosphate solubilizing ability of bacteria in different temperature regimes"

SKILLS


Python
Machine Learning
NLP
MD Simulations
Protein Modeling