IMPROVING PROTEIN DISORDER PREDICTION USING ENSEMBLE OF EMBEDDING, CONVOLUTIONAL AND RECURRENT LAYERS

A DISSERATATION

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

MASTER OF TECHNOLOGY
IN

BIOINFORMATICS

Submitted by:

SWATI SHARMA

(2K17/BIO/08)

Under the supervision of

DR. YASHA HASIJA

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

JUNE, 2019

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

CANDIDATE'S DECLARATION

I, Swati Sharma, 2K17/BIO/08 student of M. Tech Bioinformatics, hereby declare that the project entitled Dissertation titled "Improving protein disorder prediction using ensemble of embedding, convolutional and recurrent layers" which is submitted by me to Department of Biotechnology, Delhi Technological University, Delhi in the partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed for the basis for the award of any degree, Diploma Associate ship, Fellowship or other similar title or recognition.

Place: Delhi Swati Sharma

Date: 30th June 2019

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

CERTIFICATE

I hereby certify that the project dissertation titled "Improving protein disorder prediction using ensemble of embedding, convolutional and recurrent layers" which is submitted by Swati Sharma, 2K17/BIO/08, Department of Biotechnology, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master in Technology (Bioinformatics), is a record of a project work carried out by the student under my supervision. To the best of my knowledge this work has not been submitted in part or full for ant Degree or Diploma to this University or elsewhere.

Place: Delhi

Date: 30th June 2019

Dr. Yasha Hasija

(SUPERVISOR)

Associate Professor

Department of Biotechnology

Delhi Technological University

ABSTRACT

Intrinsically Disordered Regions (IDRs) are the regions in proteins that do not posses well organized two dimensional or three dimensional structures at their physiological conditions. These regions exist extravagantly in each domain and concerned with wide range of protein functions. Perceiving this far reaching presence of these regions in proteins, prodded the improvement of computational strategies to discover more of them. Every current procedure can be arranged into techniques depending on evolutionary profiles produced from multiple sequence alignment and those depending on sequence information. The techniques dependent on evolutionary sequence profiles are progressively more precise than single sequence methods due to the fact that the evolutionary sequence profiles contain significant information relating to the absence or presence of preserved residues due to their respective functional and structural roles. However, the tedious count of sequence profiles restricts the wide pertinence of profile dependent methods. Hence, study was proposed to hypothesize a strategy to reduce the performance gap between sequence dependent methods and profile dependent methods. Here we showed a model with enhanced accuracy utilizing an ensemble of embedding, convolutional and bi-directional LSTM layers with for predicting disordered regions in proteins in contrast to already existing state of art techniques. Successful prediction will help research community for both the onset of diseases and restorative treatment in context to intrinsically disordered regions or proteins.

ACKNOWLEDGEMENT

I am most thankful to my family for always promising me and giving time and unconditional help throughout this work.

I would likewise thank all the faculty members for giving their valuable time and giving me the opportunity to work and learn under Dr. Yasha Hasija.

I would like to covey my profound feeling of indebtedness and gratitude to Dr. Yasha Hasija, Associate Professor, Department of Biotechnology, DTU, for providing broad introduction to this topic through academics as well as through constructive and expert suggestions during this work. I feel having privilege to serve and expand my knowledge under her supervision. Her valuable suggestions and expert supervision fathered my work at each stage, without her warm love and consolation, the fulfilment of this work would have been difficult. I likewise want to expand my sincere appreciation towards all the members of the Department of Biotechnology for gratuitously supporting me while purposing this work. I would also like to convey my heartfelt gratitude towards my companions for their encouragement and good help during my work.

I likewise convey my genuine appreciation towards every one who have supported and given me with the facilities being required and their important suggestions and helpful discussions throughout the entire process of this work.

Swati Sharma

2K17/BIO/08

CONTENTS

CANDIDATE DECLARATION	**ii**
CERTIFICATE	**iii**
ABSTRACT	**iv**
ACKNOWLEDGEMENT	**v**
CONTENTS	**viii**
LIST OF FIGURES	**xi**
LIST OF TABLES	**xiii**
LIST OF ABBREVIATIONS	**xiv**
CHAPTER 1 INTRODUCTION	**1**
CHAPTER 2 REVIEW OF LITERATURE	**4**
2.1 Overview of disordered proteins	**4**
2.1.1 Will the domain be folded?	**4**
2.1.1.1 Sequence characteristics of disordered regions	**4**
2.1.1.2 Experimental techniques for characterization	**5**
2.1.1.3 General attributes of disordered regions	**5**
2.1.2 Functions of disordered regions	**6**
2.1.3 Roles of disordered regions	**7**
2.1.3.1 Recognition elements	**7**
2.1.3.2 Regulation through degradation	**7**
2.1.4 The natural ‘cost’ of disordered regions	**8**
2.1.5 Suggestions and future headings	**8**

2.2 Computational methods for IDPs prediction **9**

2.2.1 Physio-chemical techniques **9**

2.2.2 Template/homology based techniques **10**

2.2.3 Meta techniques **10**

2.2.4 Machine Learning techniques **11**

2.2.4.1 Classification models **11**

2.2.4.2 Sequence labeling models **12**

2.3 Machine Learning **18**

2.3.1 Artificial Neural Networks **23**

2.3.2 Convolutional Neural Networks **21**

2.3.3 Recurrent Neural Network **24**

2.3.3.1 Long-term dependencies **25**

2.3.3.2 Long Short Term Memory Networks **26**

2.3.3.3 Variations of LSTM Networks **28**

CHAPTER 3 MATERIALS AND METHODOLOGY **29**

3.1 Data Curation **29**

3.1.1 DisProt- Database of Protein Disorder **29**

3.1.2 MobiDB **30**

3.2 Data Preparation **30**

3.2.1 Preprocessing **30**

3.3 Building a model **31**

3.4 Developing a model **31**

3.4.1 Weight matrix through embedding layer **32**

3.4.2 Features extracting using layers **33**

3.5 Training a model **33**

3.5.1 Determining the quantities of neurons in a network **33**

3.5.2 Partitioning information into training and validation sets **33**

3.5.3 Learning rate and batch size estimate **34**

3.5.4 Avoid over-fitting **34**

CHAPTER 4 RESULTS **35**

4.1 Data preparation **35**

4.2 Model building **37**

4.3 Performance evaluation **38**

4.4 Validation **40**

CHAPTER 5 DISCUSSION AND CONCLUSION **43**

APPENDICES **45**

Appendix I: The python code for reading and preparing training data **45**

Appendix II: The python code for training a model **47**

Appendix III: Weight matrix representation of each amino acid **53**

REFERENCES **55**

LIST OF FIGURES

Figure 1:	**Couple binding and folding**	**6**
Figure 2:	**A typical machine learning workflow**	**17**
Figure 3:	**Artificial Neural Network**	**21**
Figure 4:	**Convolutional Neural Network**	**23**
Figure 5:	**Recurrent Neural Network with loops**	**25**
Figure 6:	**Repeating modules of RNN containing single tanh layer**	**26**
Figure 7:	**Repeating modules of LSTM containing four interacting layers**	**27**
Figure 8:	**DisProt- Database of Protein Disorder**	**29**
Figure 9:	**Screenshot of MobiDB**	**30**
Figure 10:	**Proposed model for prediction of DRs**	**31**
Figure 11:	**Features extraction using CNN layer**	**32**
Figure 12:	**Training a model**	**33**
Figure 13:	**Final model consists of embedding, conv1D, LSTM and dense layers**	**38**
Figure 14:	**Training accuracy and raining loss curves**	**39**

Figure 15:	**Q3 accuracy curve**	**39**
Figure 16:	**Validation accuracy and validation loss curves**	**40**
Figure 17:	**Training accuracy and validation accuracy**	**40**

LIST OF TABLES

Table 1:	**Summary of some DPs predictors**	**12**
Table 2:	**Sample of data retrieved from databases**	**35**
Table 3:	**Performance evaluation table for datasets**	**41**

LIST OF ABBREVIATIONS

IDPs	**Intrinsically Disorder Proteins**
IDRs	**Intrinsically Disorder Regions**
NMR	**Nuclear Magnetic Resonance**
CD	**Circular Dichroism**
KID	**Kinase-Inducible transcription activation Domain**
DFF	**DNA- Fragmentation Factor**
SS	**Secondary Structure**
CREB	**cAMP Response Element binding Protein**
RC	**Random Coil**
ML	**Machine Learning**
ROC	**Receiver Operating Curve**
AI	**Artificial Intelligence**
AUC	**Area Under ROC Curve**
PCA	**Principle Component Analysis**

ANN	**Artificial Neural Network**
CNN	**Convolutional Neural Network**
ReLU	**Rectified Linear Unit**
LSTM	**Long Short Term Memory Networks**
DisProt	**Database of Protein Disorder**
RNN	**Recurrent Neural Network**
CPU	**Control Processing Unit**
PDB	**Protein Data Bank**
SVM	**Support Vector Machine**
GPU	**Graphics Processing Unit**

CHAPTER 1

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 [1] - [2] and concerned with numerous protein functions [3] - [4]. 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 [5]. It has been observed that these disordered regions are accessory for biological activities carried out by most of the structured proteins [6]. 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 [7].

Multiple reports exist pointing towards the implications of disordered regions to various diseases including neurodegenerative diseases and cancer [6]. 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 [8]. 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 [7]. Moreover, health care has likewise been connected to disorder prediction in identifying risk and studying the progression of diseases in patients [9]. 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 x-ray crystallography experiments, where the missing electron density locales indicates the presence of disordered regions. On the other way, spectroscopic techniques (NMR) [44] - [45] with more advanced resolution and sensitivity reveal dynamics of sizeable disordered regions/proteins in solution and structural propensities.. Because of high cost of identifying disordered regions experimentally, it is essential to compute probable regions/proteins before conducting experimental studies [10].

It has been estimated that around 60 or more computational techniques have been developed so far [12] - [13] , many of these techniques utilize protein sequences [15] and information derived from them such as statistical potentials (FoldIndex) [18], physio-chemical properties (IUpred) [16], propensities (Globplot) [17] for analysis on protein. Based on the studies conducted it has been shown that these methods outperformed by sequence machine learning approaches [19] - [20] (CSpritz [23], DisEMBL [22], PONDR series [21]). However, these single-sequence methods are considered to be less accurate than sequence profile based machine-learning techniques obtained from multiple sequence alignment [25]. This is on the grounds that sequence profiles, for the most part made by programs, for example, PSI-Blast [25] and HHBlits [27], 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 [85], SPOT-Disorder [34] and AUCpred [28].

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 [35]. 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 [36] 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 [37]. 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 [40] 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 [30] - [33].

Present study was encouraged by recent progress in employing ensemble of Long Short Term Memory (LSTM) and Convolutional Neural Networks (CNN) [40]. 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.

CHAPTER 2

REVIEW OF LITERATURE

2.1 OVERVIEW OF DISORDERED PROTEINS

In eukaryotic genome numerous number of gene sequences encode whole proteins or portions of them that do not posses well-organized three-dimensional or two dimensional structures under physiological conditions [1] - [2]. These disordered regions are highly conserved with respect to their sequence and composition between species and, opposite to the traditional view, protein functions are highly dependent on stable three-dimensional structure, however disordered regions have significant role in crucial functional areas of proteins [3] - [4].

The presence of disordered regions of significant size (>50 amino acids) is prevalent in functional protein for instance [43], disordered regions in polypeptide hormone3 [45] have been known for many years. These disordered regions are liable for wide range of functions such as they are involved in cell signal transduction, transcriptional regulation and translational [46] - [47] The occurrence of these regions can be determined by X-ray crystallography experiments, where the missing electron density locales indicates the presence of disordered regions. On the other way, spectroscopic techniques (NMR) [44] - [45] with more advanced resolution and sensitivity reveal dynamics of sizeable disordered regions/proteins in solution and structural propensities.

2.1.1 Will the domain be unfolded or folded?

It remains a key for foreseeing the structures of globular proteins considering only sequence information, with the exception in certain circumstance where the structures of high homology sequences are defined. However, recognizing sequences which are going to be intrinsically disordered specifically, unable to spontaneously fold into well defined structures is nearly direct.

2.1.1.1 Sequence Characteristics of disordered regions

A probable disordered region mainly consists of amino-acid compositional bias, low complexity in sequence, lesser content of bulky hydrophobic residues which would generally be present in the core of proteins (Phe, Trp, Ile, Val and Tyr) and higher content of charged and polar residues mainly (Ser, Glu, Lys, Pro, Gln, Ser and, on occasion, Ala and Gly) [48] - [49].

Various computer programs are currently accessible that allows identification of disordered proteins or regions for instance FoldIndex [18], DisEMBL [22], PONDR [21], GLOBPLOT [17]. Investigations on complete genomes sequence information demonstrate that IDPs/IDRs are exceedingly common, and along with the complex nature of an organism the extent of disordered regions in proteins also increase exceptionally [52] - [55]. Database investigation shows that proteins related to diseases or engaged with eukaryotic cell signal transduction process possess greater tendency to undergo disordered transitions [50] - [51]. In excess of proteins have been shown tentatively to be either totally or mostly disordered (DisProt an experimentally annotated Database of Protein disorder) [56].

2.1.1.2 Experimental techniques for characterization

The most widely inspected experimental technique for determining disordered regions is Nuclear Magnetic Resonance (NMR) spectroscopy. However, alternative techniques for instance Fluorescence Spectroscopy, Hydrodynamic Measurements, Raman Spectroscopy, Vibrational CD Spectroscopy, Circular Dichorism are capable of providing additional information in context to disordered regions[57] - [58].

2.1.1.3 General attributes of disordered regions

Structural continuum for proteins, from proteins with firmly collapsed single domain, to proteins with multi-domain that may contain adaptable or unstructured regions, to yet compact but unstructured molten globules and, at long last, to exceedingly extended, heterogeneous disordered regions. This continuum has been translated in a group of three (ordered, molten globule and coil) or group of four in spite of the fact that there are diverse range of different structure types within every subdivision. In general, proteins characterized as intrinsically

disordered lack adequate hydrophobic core to spontaneously fold in well organized three dimensional structure. When all is
said in done, proteins with characteristically disordered regions can't cover adequate hydrophobic center to overlay immediately into the very sorted three-dimensional conformations that portray the proteins available within the Protein Data Bank.

2.1.2 Functions of disordered regions

New instances of functionality and involvement of intrinsically disordered regions in proteins are continuously been emerging. Some of these function include the cell signal transduction, regulation of translational process, regulation of transcription, protein phosphorylation, storing small molecules, regulation during assemblage of enormous multi-protein complexes for instance in bacterium flagellum or in assembly of ribosomes subunits. A recent review features the presence of disordered regions that are responsible for functions similar to that of chaperones [67]. This review also demonstrates that the disordered regions behave like recognizable elements and bind to mis-folded RNA and protein molecules and further support in unfolding and slackening of kinetically unfavorable intermediates. Large number of IDPs experience disordered to ordered transition and turn into a stable conformation upon recognizing and interacting with their target molecules in other words, they experience couple folding and binding process [66]. One of best known example is of activation domain in (CREB) [72] - [73]. The Kinase- Inducible transcriptional activation Domain (KID) is disordered, as a detached peptide as well as in full-length protein, however on recognition and interaction with the target molecule, it folds upon and ultimately leads to a formation of orthogonal helices. The intrinsic disordered regions in this domain can be anticipated from its sequence, that undergoes disordered to ordered transition [68].

Also, the presence of amphipathic elements that are anticipated to be disordered may give pieces of information with regard to the functional regions within the protein [69]. However, couple folding and binding may include only a couple of residues, as in KID of CREB or whole protein domain. In DNA- Fragmentation factor, the N-terminal domain (116 residues) though disordered in solution, yet creases to acquire a stable ordered globular conformation upon interacting with DFF40 [75] - [78] .

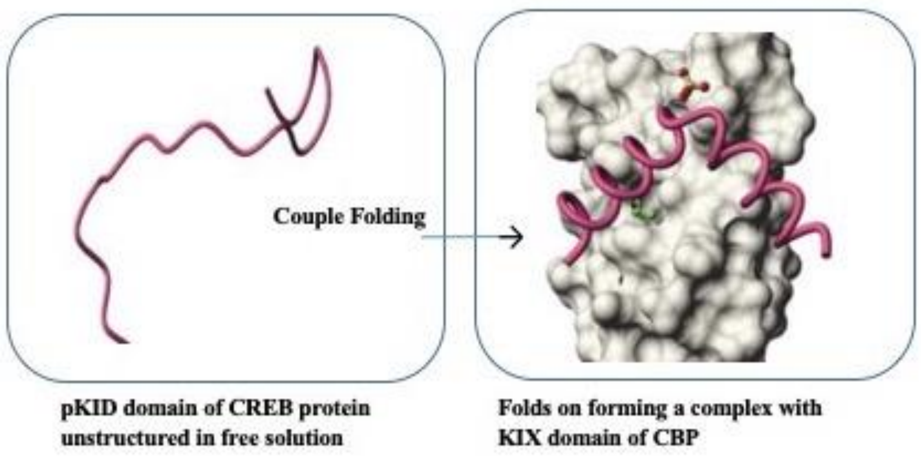

Couple Folding
pKID domain of CREB protein unstructured in free solution
Folds on forming a complex with KIX domain of CBP

Fig 1: Couple folding and binding

2.1.3 Roles of disordered regions

2.1.3.1 Recognition Elements- Nucleic acid- protein recognition

DNA-binding proteins appear to possess advanced techniques for managing the kinetic and thermodynamic difficulties of recognizing and interacting with specific DNA sequences, a considerable lot of them are partially folded or unfolded [70]. Induced couple folding plays an exceptional role in sequence-specific binding of proteins to DNA molecules, proposed over 10 years back by Spolar and Record, based on the enormous heat changes upon formation of DNA-protein complexes [71]. RNA-binding proteins additionally have disordered regions, and they are essentially organized in the complex as that in free state. 5S ribosomal RNA appears to bind to L5 ribosomal protein and form a complex by induced fit mechanism.

2.1.3.2 Regulation through degradation

The overall precariousness of the IDPs/IDRs that are associated with translation, transcription and cell signaling may further promote cell control through proteolytic cleavage and deterioration. For instance- Ubiquitin-proteasome complex system may possess a key role in activation of transcription through targeted deterioration of transcriptional-activation domains [79]. The stabilization of β-catenin and regulation of cadherin are also directed through targeted degradation. The entire cadherin cytoplasmic terminal domain is disordered and contains uncovered Pest-Sequence motifs. Pest motifs signal for degradation mediated through ubiquitin-proteasome system, but become inaccessible upon binding of cadherin to β-catenin.

The linker sequences should be extended and flexible, and have a moderately high stability in order to function correctly. The low hydrophobic residues content may be critical for function f linker sequences. In the cell mis-folded proteins are focused to undergo targeted degradation, and it is believed that, targeted degradation is mainly concerned with recognition of hydrophobic solvated residues by ubiquitin-proteasome complex system [80]. Linker sequences should be impervious to proteolysis, yet should essentially be unfolded. It is plausible that absence of hydrophobic amino acids in linker sequences is identified with characteristically unstructured fragment for moderately high stability. Besides, tentatively, it has been discovered that poly-glutamine residue repeats are impervious to deterioration by eukaryotic ubiquitin- proteasomes complex system. The specific amino-acid residues content in linker sequences additionally controls them to undergo folding in order to form structures that may associate non-specially with other proteins.

2.1.4 The natural ‘cost’ of disordered regions

Disordered regions are accessory for protein functions such as in case of transcriptional activators, cell signaling molecules and regulatory proteins. In any case, this function does not come without 'cost'. That disease related chromosomal translocations occur in disordered regions. For instance, translocation in the N-terminal unstructured regions of CBP/p300 or in linker between KIX and bromo- domains are related to human leukemia [84]. This translocation causes the segment causes the segments of CBP/p300 to get attached with MLL or MOZ regions (monocytic zinc-finger leukemia protein) or MLL are related with human leukemia [81]. These translocations lead to folded domains and in this way cause the proteins to have deviant functions [82]. On the other hand, translocation or truncations in genes that encode completely organized domains, would more likely prompt the generation of mis-folded proteins, which would be quickly destructed by cellular machinery and would along these lines not form any diseased phenotype [83].

2.1.5 Suggestions and future headings

Clearly, characterizations of functional disordered proteins are in recent stages of its success, therefore much work still be done in context to their characterization. Computational techniques to cover protein chains, and even the whole genome, for IDPs will without a doubt reveal a lot more proteins that have a place within this class. Corollary advancement in functional genomics propel our knowledge in context to functional properties of IDRs. The role of lower complexity in sequences is just at the recent stages to be addressed. Likewise, protein function that is reliable on the level of polypeptide mobility is as yet hazy. Our idea of function of proteins should subsequently advance our understanding from a static picture to an exceedingly powerful one, in that a few conformations which are reliable with different aspects of functions are represented.

2.2 COMPUTATIONAL METHOBS FOR IDPs PREDICTION

As the first computational prediction was made by Romero in 1997, during the previous 20 years, numerous predictors have been developed for distinguishing IDPs/IDRs [86] - [88]. Depending on various techniques, these strategies mainly sectioned into classes, such as physicochemical techniques, machine-learning techniques, template or homology techniques and meta strategy. These classes are summarized below: Physicochemical technique is dependent on chemical and physical properties of protein sequences, and machine learning techniques develops classification algorithms dependent on learning algorithms, homology or template based techniques depends on looking through known structures of proteins, meta technique incorporates the yields of various predictors. In any case, these four classifications are not carefully unique, predictors in any class may likewise utilize the procedures from different classes, for instance, a few of physical and chemical properties utilized by physicochemical techniques are also utilized as features to build the machine learning predictors as well as meta predictors.

2.2.1 Physicochemical-based techniques

These techniques predict IDPs/IDRs dependent on the chemical and physical properties, that influence binding and folding in proteins straightforwardly [90] . These chemical and physical properties incorporate the affinities of explicit residues, net charge, hydrophobicity, contact angle and so on. For instance, Uversky use the blend of low hydrophobicity and high overall charge to recognize IDPs [91]. Afterward, in view of this guideline, FoldIndex [18] is intended to recognize IDRs based on pre-characterized sliding window. However, a parameter P is utilized by GlobPlot [17] to foresee IDRs. The fundamental theory is that the probability of amino acid to be disordered can be described as P ¼ RC-SS, RC speak to specific residue propensity to exist in 'random coil' and SS to exist in 'secondary structure'. The fundamental calculation trailing GlobPlot [17] is a summation function of P, that is straightforward and quick. Physicochemical-based strategies are proficient with low computational expense. Moreover, their anticipated outcomes are easy to be deciphered [92]. Subsequently, extra valuable data given to the scientists who are keen on examining the disordered regions. These physical and chemical properties likewise utilized as features in machine learning techniques as well as in meta techniques.

2.2.2 Template/Homology based techniques

These techniques predict IDRs by utilizing proteins of known structures. These techniques first attempt to look known structures of homologous protein sequences (template), and after that examine the query and foresee the IDRs. For instance, in GSmetaDisorder3D and PrDOS [98], because of their diverse structure principles are utilized as component as opposed to independent predictors. The advantage of techniques is that the anticipated outcomes are easy to decipher. Be that as it may, although similar homologous sequences to target protein have various reliabilities, but at times, homologous proteins can't be identified [93].

2.2.3 Meta techniques

As examined above, for various computational predictors, they have their very own points of interest. In this way, a few meta techniques have been developed to consolidate different predictors into single model to further enhance prediction accuracy [95]. As indicated by various combination methodologies, these further can be separated into two classifications, that is the direct combination and the machine learning combination. Direct combination Straight joins the consequences of various techniques by a weighted casting a ballot technique, for example, PrDOS [98] , CSpritz [23], and MobiDB-light [97]. PrDOS [98] is a combination of two predictors, first is a SVM dependent, prepared with PSSMs and another is homology/template dependent predictor. The output depends on averaging the outcomes of both the predictors. Machine learning based combination contrasted with direct combination, machine learning combination utilizes the prediction consequences of different predictors as features to develop final model dependent on machine learning algorithm, for instance metaPrDOS, DISOPRED3, MD [94]. MetaPrDOS utilizes the outcomes of seven predictors to build a two-staged meta-predictor. The initial stage is to gather outcomes from seven predictors. The subsequent stage is to incorporate the gathered outcomes, and decide the probability of every residue to be disordered using a model planted on SVM [89]. MD is dependent on NN algorithm obtaining information from two sources. The principal source of information is the outcomes gathered from four predictors and secondary information is obtained from protein sequence properties. DISOPRED3 is based on DISOPRED2, a specific predictor of long disordered regions and closest neighbor, consists of two layer where the main layer contains three predictors comprising DISOPRED2, and the subsequent is a NN layer coordinating the outcomes of main layer. By joining various techniques, meta-predictors are known accomplish the cutting edge performance. however, because of their high computational cost, their applications have been restricted to limited number of proteins [96].

2.2.4 Machine-learning techniques

To defeat the inconveniences of others techniques, (physicochemical, template methods) techniques dependent on machine learning algorithms have been developed. Contrasted to other techniques, these techniques utilize negative and positive sets to recognize disorder regions, and also consolidate different features. As the most essential part of any in machine learning algorithm is feature extraction, in case of disordered feature extraction, it can be sectioned into the three classes. The main class is the sequence properties, for example, the amino acid residue propensity, composition, flexibility, hydrophobicity, low complexity. The subsequent class is the evolutionary relationship of profiles obtained from multiple sequence alignment. The last class is the structural data, for example, secondary structure, solvent accessibility, torsion angle. These techniques can be further subdivided into classification model or sequence labeling model.

2.2.4.1 Classification models

Traditional models were able to handle feature vectors of fixed length. Classification models are learnt in a directed way utilizing both the negative and positive datasets, and after that foresee the unseen data sample based on the trained model. Recognizing whether an amino acid is disordered or ordered is a problem of binary classification. The key to these techniques is the means by which proteins are changed into vectors of fixed length. Be that as it may, it is not a simple to convert them into fixed length vectors, since the proteins are of various lengths. In such a manner, sliding window method of fixed length incorporates amino acid residue information by taking into account its adjacent residues. A few algorithms to build such predictors, for example, Random Forest, Neural Network (NN), Support Vector Machine (SVM) [89]. PONDR [21] is one such primary predictor dependent on Neural Networks for various types of IDRs, comprising SDR (7–21), MDR (22–44), LDR (at least 45) and all lengths of IDRs.

2.2.4.2 Sequence labeling models

These are based on supervised learning by utilizing both the negative and positive datasets, where input to a model is a protein sequence, and final outcome is labeled sequences of disordered or ordered residues. Models based on algorithms as Conditional Random Field (CRF), Recurrent Neural Network (RNN), Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM), have been developed. One such example to this is SPOT-disorder [34], which is a deep-bidirectional model comprises a RNN layer (feed-forward) followed by two LSTM layers. It demonstrates that it has accomplished great performance for anticipating both LDRs and SDRs. This is because, LSTM eliminates long term dependencies problem, and the bidirectional network can catch the information from both forward and reverse directions. The performance of labeling methods is additionally improved on merging with deep learning methods. The reason is that LSTM and RNN can consequently perceive relevant data of the residues, and catch both the local and global logical data of proteins.

PREDICTOR	CATEGORY	CLASSIFIER	FEATURES	YEAR	AUC

GlobPlot	P	_	Amino acid propensity difference	2003	NA
IUPred	P	_	Amino acid composition	2005	0.66
FoldIndex	P	_	Net charge and hydrophobicity	2005	NA
DisEMBL	C	NN	Protein Sequence	2003	NA
PONDR VL3	C	NN	Residue frequency, flexibility and sequence composition	2001	0.69
Spritz	C	SVM	PSSM and secondary structure predictions	2006	NA
SPINE-D	C	NN	Residue and window level information from different parameters	2012	0.82
SLIDER	C	LR	Physicochemical properties, complexity of sequence and amino acid composition	2014	NA
DisPredict	C	SVM	Amino acids, physical-chemical properties, ASS	2015	NA
DISpro	L	RNN	PSSM, solvent accessibility, secondary structure	2005	NA
Espritz	L	BRNN	Sequence or add PSSM	2012	0.855
AUCpreD	L	CNN CRF	Residue-related features include identity, physical-chemical propensities	2016	0.88
SPOT-Disorder	L	LSTM RNN	SSM, Shannon entropy, physical-chemical properties, structural properties	2016	0.903
PrDOS	M	_	Combination of a SVM predictor and a homolgy predictor	2007	0.907
MD	M	_	Combination of DISOPRED2, PROFbval, IUPred and a few sequence properties	2009	0.849
PONDR-FIT	M	_	Combination of PONDR, PONDR VL3, IUPred, TopIDP and FoldIndex	2010	0.818
MetaDisorder	M	_	FloatCons: combination of 13 predictors	2012	0.753
DISOPRED3	M	_	Combination of	2015	NA

			DISOPRED2, LDRs and a nearest neighbor predictor		
MobiDB-lite	M	_	Combination of two variants of IUpred, and three variants of ESpritz, DisEMBL and GlobPlot	2017	NA
			P- Physio-chemical Method		
			C- Classification Models		
			L- Labeling Models		
			M- Meta Methods		

Table 1: Summary of some of IDPs/IDRs predictors

2.3 MACHINE LEARNING

At its very root, machine learning can be understood as programming of computers in order to optimize performance criteria with the help of example dataset and past experiences. This is extremely useful in cases where there exists no direct computer program to solve a given problem. Another scenario where machine learning comes in handy is when either the human expertise does not exist or when it is not explainable, one example of this is the conversion of acoustic speech signals into ASCII text for recognition of spoken speech, the task can be performed without any difficulty, but the explanation of how it is done is way more complicated. The same word is uttered by different individuals differently because of the variation in their age, gender, and/or accent. This conundrum is solved with the help of machine learning by using an approach within which we collect a large number of samples of utterances from varied people and learn to form these into words. Machine Learning can also have applied to cases where the problem under consideration changes with time or particular environment. The basic aim is to have robust multi-purpose systems that acclimatize to their circumstances, rather than systems which require different set of codes for different circumstances. An example here would be the routing packets over a computer network wherein the path that maintains the highest quality of service between the source and the destination changes with the change in network traffic. A machine learning routing program would acclimatize to the best path by monitoring the network traffic. Let’s take a look at yet another example, machine learning can be used for creation of an intelligent user interface wherein it adapts to the biometrics of the user, that is, his or her accent, handwriting, working habits, and so on. Many more such examples can be observed in everyday life in all sorts of domains, and commercial systems are available for recognizing speech and handwriting. Another commercial application can be observed in how online retail companies learn their customer's behavior by analyzing their past sales data in order to provide the customers a more customized experience and financial service providers use the customer's transaction history in order to predict the credit risks. For biotechnologists, machine learning has become an integral part of the bioinformatics studies. Since high throughput mechanisms are generating an enormous amount of data the storage and knowledge extraction from which requires the use of computers. Machine learning also finds application in the field of medicine for the purpose of medical diagnosis. Other future computer applications may include but are not limited to: self-driven cars which adapt to different roads and environmental conditions, phones that can translate to and from a foreign language in real-time, or robots that can adopt to new environments, for example, the surface of a yet unexplored planet.

With the modern technologies, numerous amount of information can be stored, processed and accessed from almost anywhere over a computer network. Majority of equipment these days acquire data digitally and reliably. Let us take a look at an example of a supermarket chain with thousands of products whose outlets are spread across the country and caters to the needs of millions of customers. They record each and every transaction at the point of sale, including details like, date, goods bought, total amount, the customer identification number, time, and so forth. This is an enormous amount of data which is practically of no use until it is analyzed and turned into useful information that can be used to understand a customer and make his future shopping experience much more customized. Simply put we have no idea about an individual's preferences, for example, the audience which would be interested in buying a particular product or which author books would interest a person who is deeply interested in Hemingway, if we knew this, then we would have simple written a code and everything would have been sorted. But since we don't, all we can rely is on the collected data and how well are able to analyze it to get the answers for the above mentioned and similar questions. And this has turned out to be really helpful because all though the details of underlying processes like consumer behavior, cannot be explained, but we know with certainty that it is not entirely random. When a person goes to a supermarket the shopping list is not entirely random, when he/she buys a soft drink or a beer there is usually chips accompanying it in the shopping list, ice-creams are common in summers whereas spices for butter chicken are common in winters. There is usually a pattern in the data, and using this pattern although we may not be able to determine the entire process with certainty but we surely can determine a useful and good approximation, patterns, or regularities. And the approximation may not be dead accurate but can still help explain majority of data. The key assumption here is that the future or at least the near future is not very different from the past. This is where the roots of machine learning lie. Data mining is when the principles of machine learning are applying to much larger datasets or databases. The name mining is derived from the analogy that when we extract a large amount of earth or raw material from mine and process it into smaller much more precious processed materials, similarly when we talk about data mining we are actually talking about using a numerous amount of information which is being collected and refined to have a more simplistic model with practical application.

In order to solve a problem on computer, one requires an algorithm. Algorithm is an orderly consortium of instructions that needs to be executed in order to convert input into output. The very first algorithm taught in earlier classes and with which students are well acquainted is that of sorting, wherein the input is a set of numbers and the outcome is a well ordered list of numbers. For performing sorting over a set of numbers there exist a number of different algorithms and our job is to find the one that requires least number of steps, memory or both or in other words the one which is the most efficient. But for problems like predicting consumer behavior we cannot design a standard consortium of instructions or an algorithm. Another example could be that of sorting the email into spam emails. Here the input is an email and the output is a yes/no answer, that is, whether the email is spam or not. But we have no way to obtain the output from the input because what is spam for a given person at a given time may not be spam for another individual or the same individual at another time. But what we can practice here is to collect a large amount of data with knowledge of which is spam and which is not. And from this sample we can make our machine "learn" what constitutes a spam. In other words, the computer itself devises an algorithm to perform the given task. In recent times biological sciences have made progress and high-throughput data collection technologies have been developed, simultaneously there has been development in computing, digital storage, and communication and information technologies, which has transformed life sciences into a data-rich science from data-poor branch. This advancement has led to transition of life sciences from hypothesis-driven approach to a data driven approach, that is we now have many answers to the same problem at any given time and we seek hypothesis that perfectly explains all of them rather than having one answer for one problem approach. This has led to the emergence of the field of bioinformatics, wherein we store, retrieve, analyze and assist in understanding the plethora of data acquired on daily basis. The same thing that led to its emergence is also the main hurdle to bioinformatics, biological information has to be extracted from the in-house and publicly stored data.

Machine learning plays key role in computational predictions in proteomics [37], metabolomics [38], genomics [36], sensitivity to compounds [38]. These applications delineated inside the standard machine learning work process, that incorporates five phases: data collection, data pre-processing and preparation, feature extraction, training and model building, model performance evaluation.

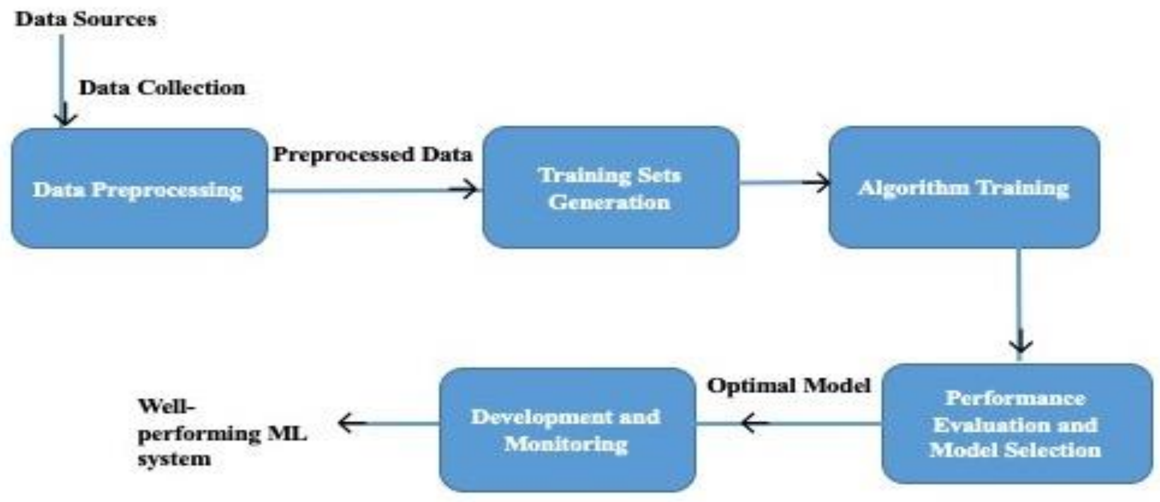


Fig 2: A typical machine learning workflow

Superficially speaking machine learning sectioned into unsupervised and supervised learning. In supervised learning we begin with a dataset that contains couplets of the input and the desired outcome, and the job is to develop a model that would precisely predict the desired outcome for future samples for which do not have the output values. The task is referred to as a classification task when the outcome is defined as finite set of discrete values and when it takes up continuous values it is referred to as a regression task.

In stark contrast with the supervised learning is the unsupervised learning. In this we begin with a dataset of training data that is supplied to the machine without any labels or without the output. The machine clusters or partitions the training examples into subsets, so that the examples within a cluster shows high degree of similarity or proximity. When a future case is presented to the machine the machine tries to classify it into one of the clusters.

Consider a typical example of life sciences [39], to foresee the viability of malignant cancerous cell line in response to a drug. Since, supervised machine learning directs to develop a model from given training sets to determine the future output. In this particular case, the input feature would include chemical composition of a drug, concentration of drug, cell line somatic sequence. These input features along with viability of cells in known cases (output label) utilized to train a model based on different algorithm for instance random forest, support vector machine to learn a functional relational between the input and the output labels, such that it could precisely predict the viability for unseen samples. As opposed to this is, unsupervised learning that intend to form clusters of sample by discovering patterns from them. some of common examples of unsupervised learning applied to biological data include clustering, outlier detection, PCA (Principle Component Analysis).

The final input labels, obtained from raw information describe the model, and their decision is exceptionally problem-specific. Inferring most useful features is crucial for model prediction accuracy, however, this task requires domain knowledge and can be labor-intensive. This is particularly difficult in cases where the data is of high dimensionality such that even computational strategies unable to assess exceptionally large number of possible couplets for feature extraction. A noteworthy ongoing development for automating this task by learning a reasonable portrayal of information with deep neural networks [40]. Briefly, neural networks get the raw information at input layers for feature selection and progressively transform them by combining outputs from previous layers in a data-driven way. Deep learning is recently a standout amongst the most other learning algorithms and has been appeared to exceptionally well in speech and image recognition [41], natural language processing and most currently, in computational biology.

2.3.1 Artificial Neural Networks

Neural networks also known as artificial neural networks, have been worked upon since it was recognized that the human brain computes in a way, totally distant to that of a conventional digital computer. Human brain could be understood like a complex, non-linear and a parallel processing machine. The structural components of the brain are known as neurons and the brain has the capability to organize them in order to execute various computations for instance motor control, perception, pattern recognition, and so on. And what is surprising is the fact that the human brain performs these computations a lot faster than any conventional digital computer in available today. When we take birth, the brain is well structured and possess the ability to make its own rules through experience, which builds up over time. Majority of brain development takes place in the neonatal stage, but the development continues well beyond this stage.

When it comes to the neurons it’s important to understand that they possess plasticity during development, that is, they adapt to the surrounding environment. Similarly, in the neural networks consisting of artificial neurons plasticity is important for information processing. Neural network can be understood as software on a machine or a digital computer that is designed to work in a similar way as that of human brain to carry out a certain function. Neural networks make use of colossal interconnection of simple computing cells often termed as neurons or processing units. Simon Haykin in his book Neural Networks: A comprehensive Foundation defines neural networks as “A neural network is massively parallel distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:

1. Knowledge is acquired by the network from its environment through a learning process.
2. Interneuron connection strengths, known as synaptic weights, are used to store the acquired knowledge.

Desired design objective is attained through orderly modification of the network's synaptic weights with the help of a learning algorithm. It is well established that neurons die and regrow to form new connections in synapse junctions inside the human body the same can be done with the neural networks as well.

Artificial neural network, at first influenced by brain's neural system, comprises interconnected layers of neurons. The depth is related to quantity of hidden layers in a network whereas greatest quantity of neurons in any layer of a network corresponds to width of the network. Since, it became conceivable to build network with numerous hidden layers, these are also referred to as "deep networks".

The below figure depicts the standard arrangement, where the input layer gets the information, while passing through multiple hidden layers it gets changed in a non-linear manner to yield a final output at output layer. All the previous layer neurons are associated to every neuron in preceding layer. To ascertain its output, every neuron uses a non-linear activation function and calculates a weighted summation of all the inputs. ReLU (Rectified Linear Unit) is well known activation function used in neural networks. This function goes positive signal while keeping a threshold of zero to negative signals. This function permits quicker learning contrasted with other activation functions such as tanh unit or sigmoid function.

The input information describes the model representation and the loads or weights are free parameters between any two neurons. Data samples at input and output layers are to optimize the weights and minimize the error function which estimates the fit between sample's true value and output of the model. This minimization is difficult to achieve as because of non convex and high dimensional data representation.

It took quite a few years before the backward propagation algorithm (via chain rule for derivatives) was used to calculate an error function, this ultimately empowers effective neural networks training utilizing stochastic gradient descent. While training a network, the sample's true label is compared with the anticipated label, to compute error function for current model loads. Then the computed error is reversed propagated along the network and the weights are updated accordingly. Gradient steepest-descent is basically used to optimize this error function. With the each progression, the present weight vector is moved in the direction of gradient steepest descent by learning rate η that is the vector length [31]. Since, learning of deep networks remnants a functioning area of research, programming packages have been available and can be utilized without knowledge of mathematical information involved.

For explicit applications, different designs of completely interconnected networks have been developed that vary in the manner in which the neurons are arranged. These incorporate recurrent neural network for sequential information convolutional neural network are broadly utilized in image processing, Boltzmann machines and auto-encoders for unsupervised learning. The specific objective of the problem and data will decide the design and different parameters of the network.

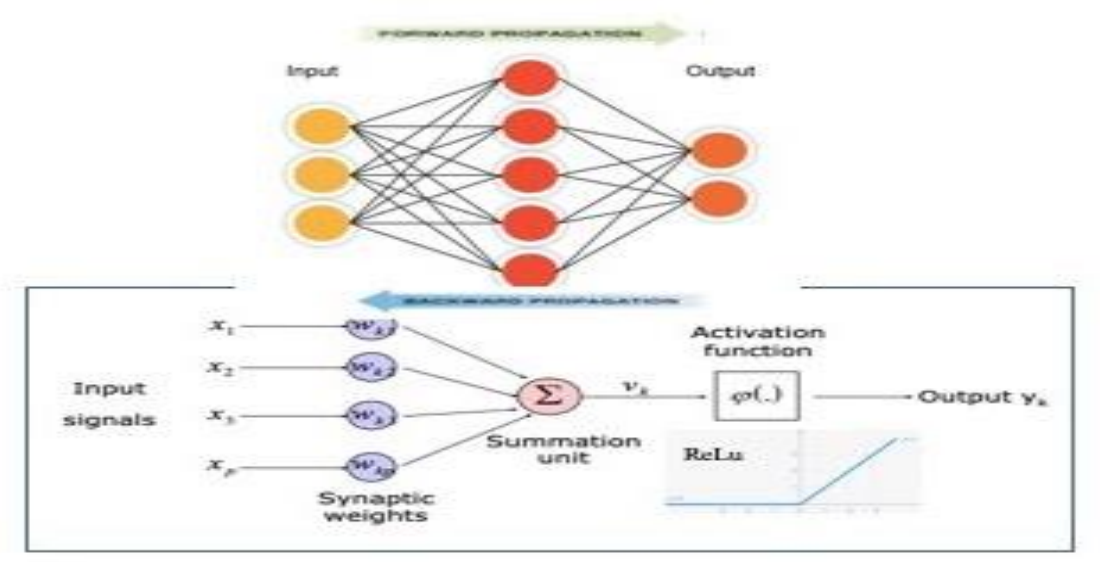


Fig 3: A typical Artificial Neural Network

2.3.2 Convolutional Neural Networks

A convolutional neural system is another category deep neural networks and widely used in for image processing. of profound neural systems, most regularly connected to examining visual symbolism.

Greatly influenced by biological processes, this category of deep networks resembles animal visual cortex in terms of organization and connection pattern of synaptic units (neurons) in a network. One advantage of CNN, they require comparatively lesser pre-processing of data in contrast to other algorithms. They are widely used in natural processing language, image classification, video and image recognition, medical analysis.

CNNs resemble standard neural networks with respect to their architecture that is, comprises multiple hidden layers in between input and output layers. Hidden layers in CNNs are referred as progression of convolutional layers such as fully connected layer, pooling layer, normalization layer, as their output and input are enacted by last convolution and ReLU function (commonly used activation function in CNNs).

A CNN design is framed by a heap of particular layers that change the input information to output information using a differentiable function. A couple different kinds of layers are commonly utilized in CNNs. These are additionally explained below.

The convolutional layer is the core component of CNN. Here, the parameters are learnable kernels of filters, having a small receptive field, that reach out through the full volume of input space. Each kernel convolved over height and width of input space and produce a 2D activation map by computing a dot product between input and kernel's entries. This allow network to learn these kernels that actuate on recognizing some particular sort of feature at any spatial position in the input. The full output space can be formed by stacking activation maps of all the kernels along the depth of convolutional layer. Therefore, each entry of output space can be seen as output of a neuron present in a small receptive field in input space and offers same parameters to that of neurons other neurons of same activation map.

When managing high-dimensional sources of information, for example, pictures, it is illogical to associate neurons to all neurons in the past volume as design of a network does not consider the spatial structure of the information. Convolutional systems exploit spatially relationship by implementing a sparse connectivity between neurons of adjoining layers: every neuron is associated with just a little area of the input space. The degree of this connectivity is a hyper-parameter that is receptive field of network unit. The associations are nearby in space yet dependably reach out along the whole depth of the input space Such a design guarantees, that the learnt kernels produce the most grounded reaction to a spatially nearby input design.

The idea behind the pooling layer is down-sampling of input data non-linearly. This pooling can be performed with number of different non-linear function, however max-pooling most frequently used in CNNs. It segments the input space into of non-overlapping rectangles and yield a maximum output from every sub section. Since, accurate position of feature is less significant with respective to the others, this forms the basic idea of adding a pooling layer in ConvNets. It dynamically lessens the spatial size, therefore diminish the quantity of parameters, number of computations, and subsequently avoid over-fitting. It is entirely expected to occasionally embed a pooling layer between progressive convolutional layers in a CNN architecture. The pooling task gives another type of interpretation invariance.

ReLU layer set zero to negative values and adequately expel them from activation map. It also expands the nonlinearity of a function and network without influencing the receptive fields of the convolution layer.

At last, after a few convolutional and max pooling layers, the abnormal state thinking in the neural system is done through completely associated layers. Neurons in a completely associated layer have associations with all initiations in the past layer, as found in standard (non-convolutional) fake neural systems. Their actuations would thus be able to be processed as a relative change, with grid duplication pursued by a predisposition balance (vector expansion of an educated or fixed inclination term)

At last, fully connected layer after a few pooling and convolutional layers. As found in the standard artificial neural network, the all the past layer activations are associated with all the neurons of full connected layer and output can be computed with matrix multiplication followed by bias offset.

The loss layer determines variation between the true and anticipated labels, typically the last layer of in any network. Depending on the task to be done suitable loss function can be used.

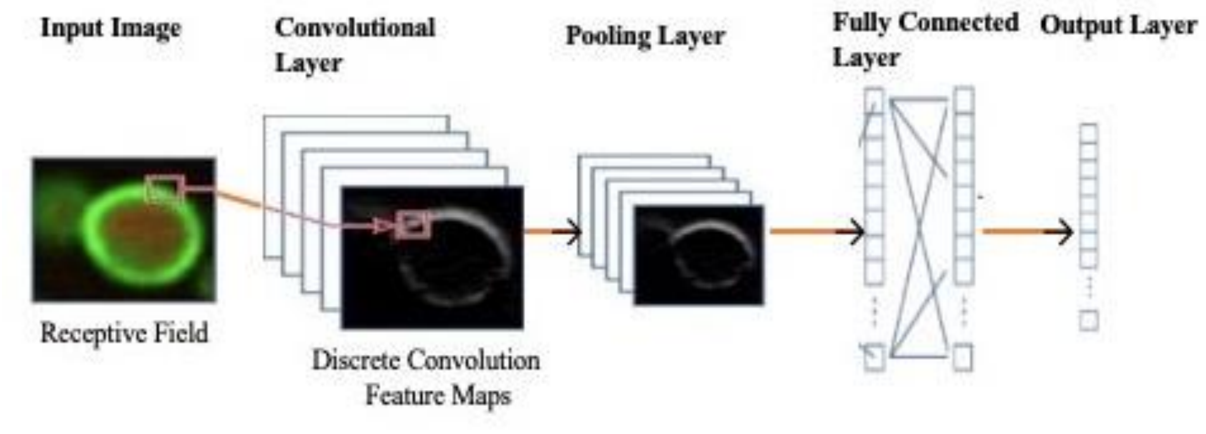


Fig 4: Working of convolutional neural network

2.3.3 Recurrent Neural Networks

People don't start their intuition from scratch consistently. Traditional neural networks unable to do this, and it appears to be their major disadvantage. For instance, suppose one needs to determine the sort of act occurring at every point in a motion picture. It's difficult for a traditional neural network to utilize its thinking about past act in the motion picture to advise upcoming act.

However, Recurrent Neural Networks (RNN) overcome this disadvantage of traditional networks. Since, these networks consist of loops which enable the data to endure for longer time. These networks work on the principle of keeping the output of a layer and providing this back to guide in anticipating the output. The input layer is framed as a feed-forward network. Every neuron in these network behaves like a memory cell where the information while passing through one neuron onto the next, remembers some information which are then utilized in performing calculations. In other words, neural network deals with front propagation and remembers some information it requires for later use.

In the below outline, piece of neural network, AA, some input information xt and produce an output ht, where a loop facilitates the information to be passed through the network from one stage then onto the next stage. with one stage of the network then onto the next. These loops cause recurrent neural network to be quite puzzling. For unknown reasons, they aren't too unique in relation to an ordinary neural network. A recurrent network can be seen as a combination of large number single network each passing an information to a successor.

The chain-like resemblance of these networks uncovers that they are identified for sequential data (genome and protein sequences) and lists. These are the networks to use such information with an ease.

Over the most recent couple of years, they have been successful applied in various applications such as in image processing, translation, language modeling, speech recognition, so on.

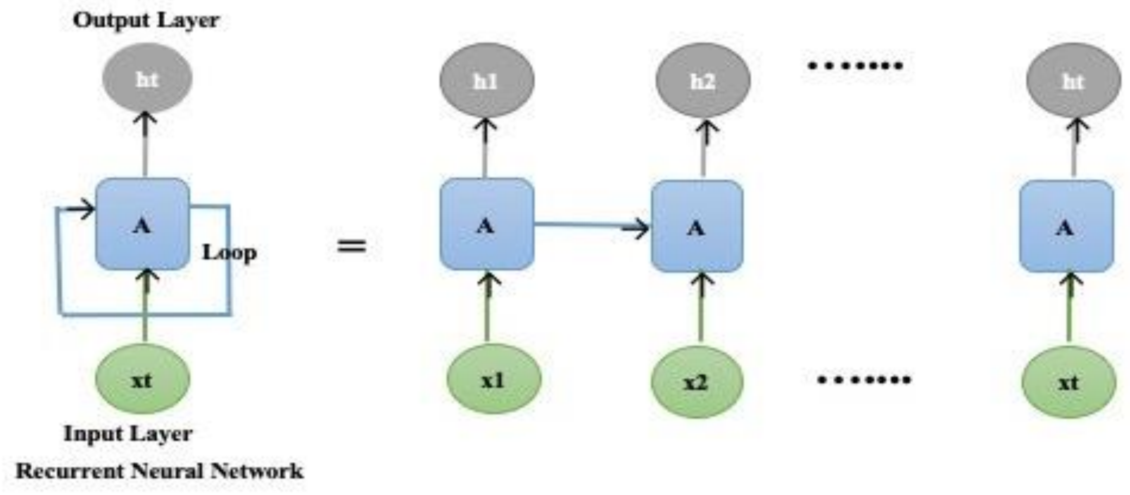


Fig 5: Recurrent neural networks with loops

Fundamental to these successes is the utilization of "LSTMs (Long Short Term Memory)" unique recurrent network, that works for varies tasks, a whole lot superior to the standard version. Practically all energizing outcomes dependent on recurrent networks are accomplished with them.

2.3.3.1 Long-Term Dependencies

One of the interests of RNNs is the possibility that they may most likely associate previous information to the current task, for instance, past video frame may comprehend regarding the current frame. In some cases, we just depend on recent data to perform the current task. In such situations, the gap between the relevant data and the point where it is required is little, RNNs able to figure out how to utilize the past data.

However, there are additional situations where it is feasible for the gap between the reliable data and the point where it is required turn out to be extremely huge. Unfortunately, as that gap becomes large enough, recurrent neural networks become unable to figure out how to connect this data.

In theory, recurrent neural networks are completely fit for taking care of such long term dependencies issues. In practice recurrent neural network don’t appear to have the option to learn them. LSTMs don't have this issue.

2.3.3.2 LSTM Networks

Long Short Term Memory networks [33] – typically referred to as "LSTMs" – an exceptional kind of RNN, fit for taking care of long-short term dependency conditions. They were originally proposed by Schmidhuber and Hochreiter. Many people promote and refine their work using LSTMs. They work enormously well for various different applications, and are recently being utilized more often. These networks are intended to circumvent long-term dependency problems. These networks have ability to retain information for longer period of time, is their default behavior not something they force to learn.

All RNNs have chains of repeating modules of NNs. In standard recurrent neural networks, the repeating modules have a straightforward architecture, for example a tanh layer.

Whereas, LSTMs additionally have chain like architecture. These modules in LSTM in spite of consisting single layer, they contains four or more layers, connecting in an extraordinary manner.

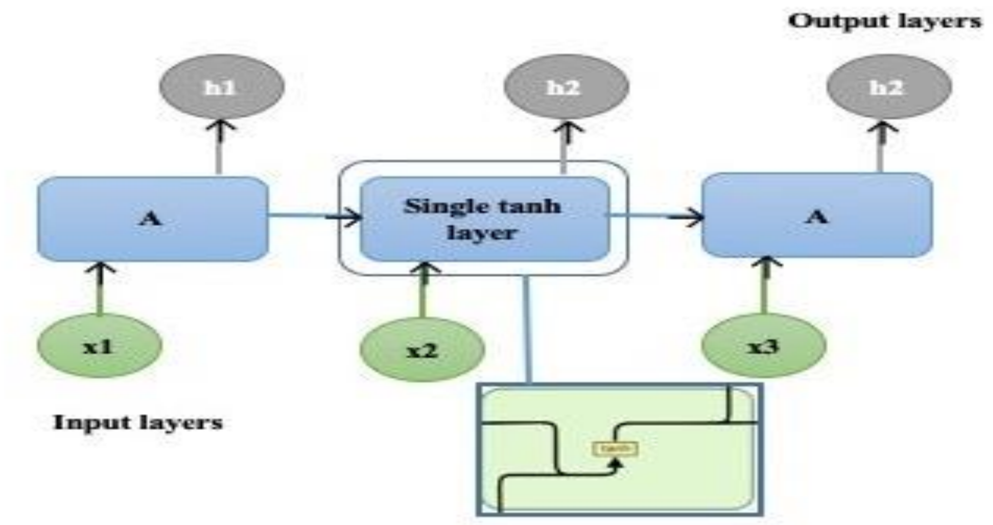


Fig 6: Repeating modules of RNN containing single tanh layer

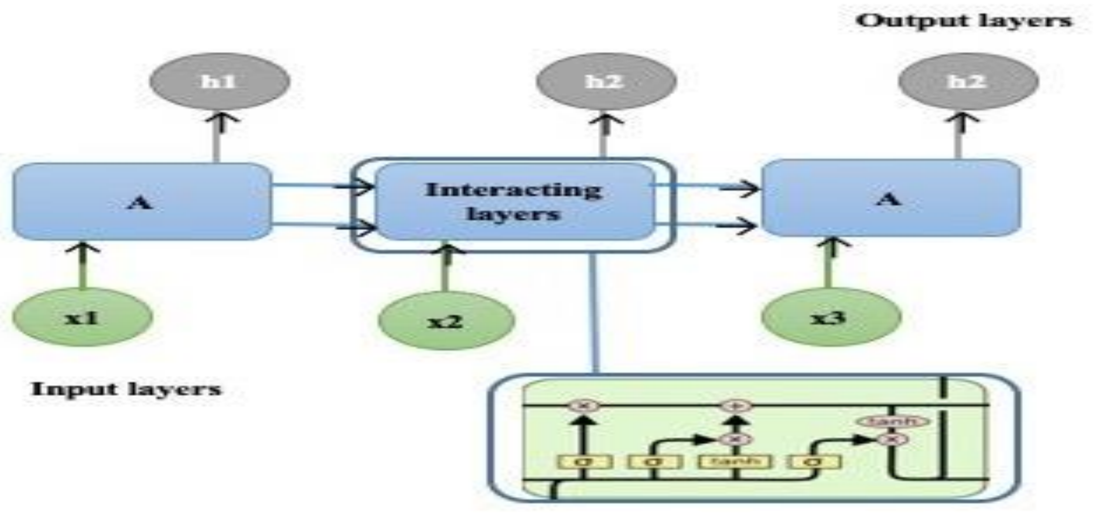


Fig 7: Repeating modules of LSTM containing four interacting layers

As depicted in a figure, four interacting layers consists of a ray which represents a vector transfer that point from output of one node to input of others. Point wise operations such as summation, deletion represented through circle. Learned neural layers are represented by boxes. Lines blending mean concatenation. Lines merging out represent data replications and replicates proceeding in opposite directions.

The core of LSTMs is the state of the cell that is the horizontal segment at the top of the network. Similar to conveyor belt, cell state runs the whole chain with limited interactions. The data flows through this segment without any transformation. However, LSTMs have the potential to transform this data by adding a new data or eliminating a part of it. This can be done through different structure known as gate of LSTM networks.

The sigmoid gate decides the amount of data to pass through depending on its output, which ranges from zero to one, where zero signifies “no data pass through” and one signifies “whole data pass through”.

A LSTM consists of three such gates in order to secure cell state. Process through LSTM interacting layer as follow:

The sigmoid function decides the data to be discarded from the cell state in the initial phase of a LSTMs. This is done by forget layer of sigmoid gate by considering h1 and x2t, and outputs some value for every data in previous cell state. Now, depending upon the output values decide the data to remember.

In the subsequent phase, LSTMs has to add the new data in the cell state, can be done in two steps. Initial step is carried out by input layer in sigmoid gate where the data values are updated. In the second step, vector of updated data values, $\tilde{C}2$ are added to the cell state by tanh layer. Further consolidate these step in the cell state to make a final update. finally, old cell states are updated to new one cell state following some calculations. Now, the updated cell state will decide the final output. In this, sigmoid gate decides the output data. Then cell state through tanh layer and multiply with the output of sigmoid gate, to yield a final output.

2.3.3.3 Variations of LSTM Network

Above explained LSTM is quite ordinary. Be that as it may, not all LSTMs are equivalent to the above mentioned. Indeed, it appears pretty much like every paper including LSTMs utilizes slightly distinct versions. The distinctions are minor, yet it is worth mentioning a few of them.

A most prominent LSTM variation, presented by Schmidhuber and Gers, including "peephole connections.". Another variety is to utilize coupled input and forget gates. Rather than independently choosing what data to forget and what new data to add, this variant make such choices altogether.

Gated Recurrent Unit (GRU) is the more progressively variant from LSTMs, presented by Cho. It consolidates input and forget gates into a particular "update gate." It likewise blends the hidden layers and cell, and rolls out a few different improvements. The subsequent model is less difficult than typical LSTM models, and has been becoming progressively well known. These are just a couple of the most prominent LSTM variations. Heaps of others, similar to Depth Gated RNNs by Yao are available. There are likewise totally extraordinary ways for dealing with long term dependencies similar to Clockwork RNNs by Koutnik.

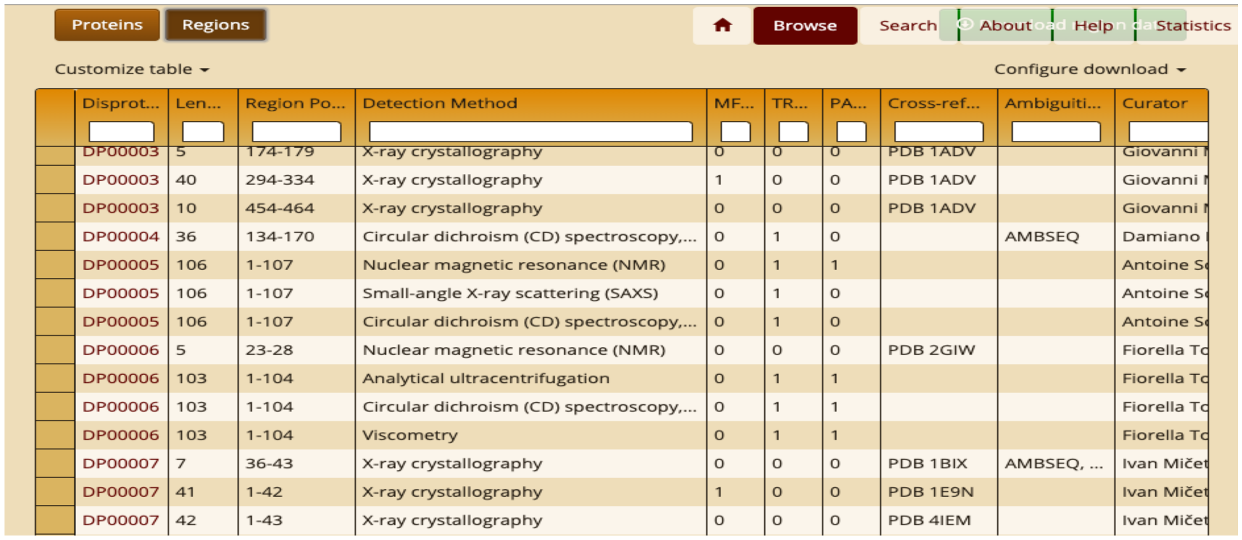
CHAPTER 3

MATERIALS AND METHODOLGY

3.1 DATA CURATION

In this study we intended to use an ensemble of long short term memory and convolutional networks to predict DRs. For this we required experimentally validated disordered protein sequences. These sequences were collected and from manually curated and annotated databases such as DisProt 7 [56] (v0.5 release 11-05-2017) and MobiDB [11] (release 24 October 2017).

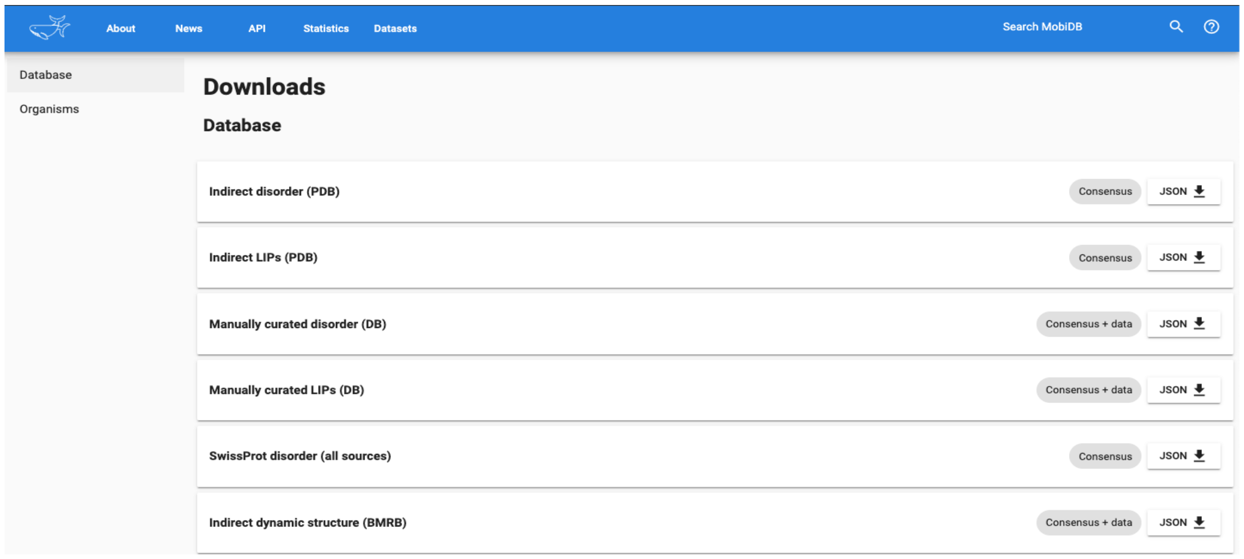
3.1.1 DisProt is a database comprising of experimentally validated and annotated information of disordered regions manually gathered from literature. Statistics includes 803 proteins and 2167 regions. Each evidence is recognized by at least one experiment. DisProt disorder region (DR) is unambiguously 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 I
DP00003	40	294-334	X-ray crystallography	1	0	0	PDB 1ADV		Giovanni I
DP00003	10	454-464	X-ray crystallography	0	0	0	PDB 1ADV		Giovanni I
DP00004	36	134-170	Circular dichroism (CD) spectroscopy,...	0	1	0		AMBSEQ	Damiano I
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

Fig 8: DisProt- Database of Protein Disorder

3.1.2 MobiDB 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

Fig 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.

3.2 DATA PREPARATION

3.2.1 Pre-processing

As all the data was available, it is imperative to get consistency in the length of the sequences so that our model could extract features considering patterns in the sequences but not the differences in the length of the sequences. Therefore, we chose to set some threshold value. The python code for reading and preparing the training data is given in Appendix I.

3.3 BUILDING A MODEL

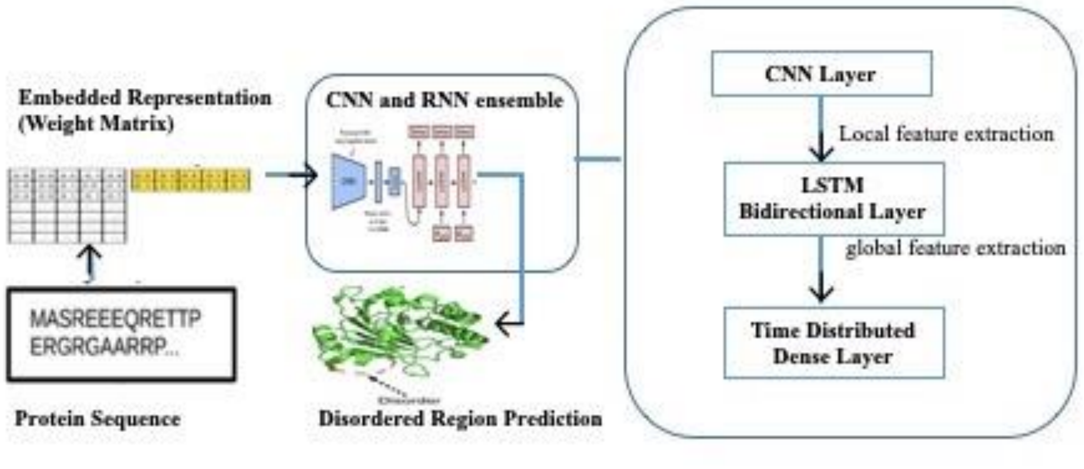

Embedded Representation (Weight Matrix)
CNN and RNN ensemble
CNN Layer
Local feature extraction
LSTM Bidirectional Layer
global feature extraction
Time Distributed Dense Layer
MASREEEQRETTP ERGRGAARRP...
Protein Sequence
Disordered Region Prediction

Fig 10: Proposed model for prediction of disordered regions

Above given figure 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. Our model was implemented in Tensorflow v1.4,51 allowing us to accelerate training up to 20 times faster.

3.4 DEVELOPING A MODEL

3.4.1 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 appendix III.

3.4.2 Feature extraction using layers

Next step was to extract feature from sequences using CNN layer. The one-dimensional (1D) convolution layers applied in our models utilize a filter depth of 40 at a kernel size of 64. Then we used a bidirectional LSTM layer 512 neurons for global feature extraction along the sequence. Then finally a time distributed dense layer of 552 neurons which makes use of softmax function to concatenate the outcomes from LSTM to produce the final output.

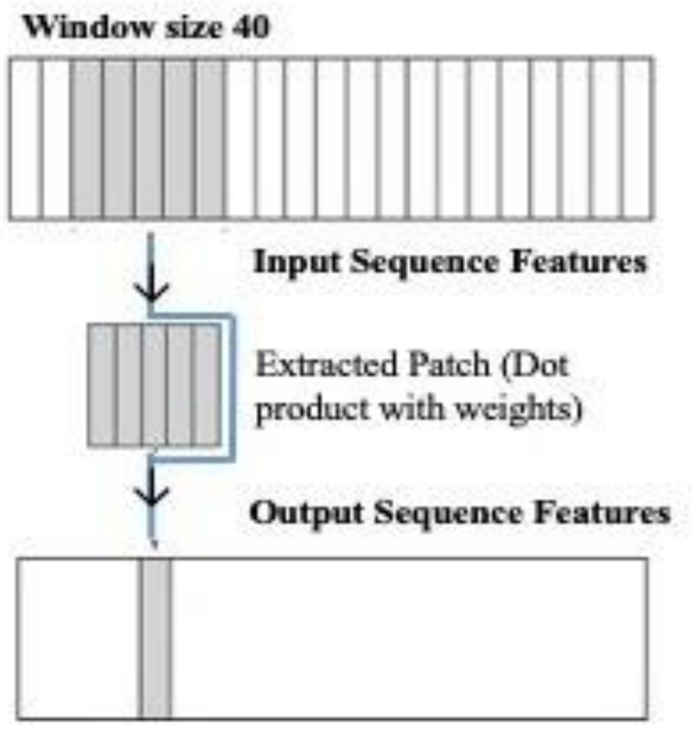


Fig 11: Feature extraction using CNN layer

3.5 TRAINING A MODEL

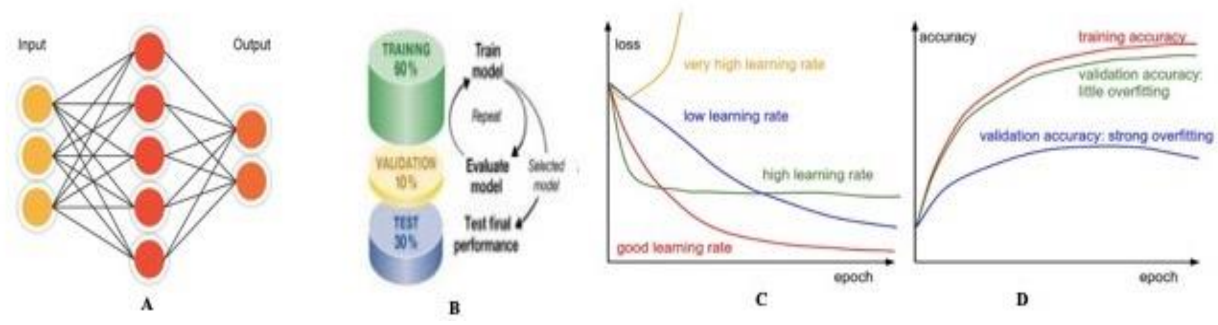


Fig 12: Training a model

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.

3.5.1 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.

3.5.2 Partitioning information 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.

3.5.3 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.

3.5.4 Avoid Over-Fitting

Neural Networks based models are challenging to train, data over-fitting is a vital test. Over-fitting comes about as a result of an excessively intricate model relative, making it difficult to the range of the training set, and would consequently have the option to be lessened by decreasing the model disperse quality, suchlike the quantity of hidden layers and neurons in a network, or by extending the proportion of training set by the method of data extension. We have played it safe for abstaining from over fitting: We applied a dropout rate of 0.2 in LSTM layer, softmax activation and L2 regularization penalty of 0.001 in dense layer.

Floyd hub cloud computing was used to train our model for 50 epochs having 32GB RAM, Intel Xeon 8 Cores CPU and11GB NVIDIA Tesla K80 GPU. The python code for training the model is given in appendix II.

CHAPTER 4

RESULTS

4.1 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.

In brief, 9604 manually curated consensus protein sequences were retrieved and then pre-processed to bring about the consistency in the length of sequences so that our model could extract features considering patterns in the sequences but not the difference in the length of the sequences. Therefore, we chose to set a threshold value of range of length of protein sequence to be 150-550 amino acid residues. So, the final dataset consists of 7008 protein chains which were then arbitrarily part into a training set of 5606 chains (80%) and a validation set of 1401 chains (20%). All these sequences have similarity in sequences of less than 25% as indicated by BlastClust.

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	SEVHPSRL QT	Native	19214187
DP00962	750	AF1 domain, Amino Terminal	NMR	720	TDNLLPMS PEEFDEVS RIVGSVEF	Native	19214187

		Domain (NTD)			DSMMNTV		
DP01091	81		XRAY	60	AEHQTAG RGRHGRG WAATARA Q	Native	20169168
DP01091	172		XRAY	159	VTQAPEEV DPDATS	Native	20169168
DP01099	10		XRAY	1	MASPPPFH SQ	Native	12923182
DP01099	350		XRAY	339	GQASETPH PRPS	Native	12923182
DP00981	173	C-terminal extension	PNMR	165	EKPSSAPSS	Native	1397302
DP00324	102		XRAY	92	REDSQRPG AHL	Native	11917013
DP00142	209		NMR	192	RAQIGGPE AGKSEQSG AK	Native	7649277
DP00142	209		FCD	192	RAQIGGPE AGKSEQSG AK	Native	10727931
DP00142	209		NCD	192	RAQIGGPE AGKSEQSG AK	Native	10727931
DP00023	199		XRAY	191	TAFMEKV LG	Native	9525918
DP00023	328		XRAY	289	PAKAEAG AEAGGGA GPGAEDEA GRGAVGD PELGDPPA APQ	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	SKQEMAS ASSSQRGR	Native	11917013
DP00322	31		XRAY	14	SALPDPAG APSRRQSR QR	Native	15525646
DP00733	378		XRAY	372	LLEEVSP	Native	15713488
DP00324	102		XRAY	92	REDSQRPG AHL	Native	11917013
DP00324	196		XRAY	182	SKQEMAS	Native	11917013

					ASSSQRGR		
DP00964	20		XRAY	1	MSKREETGLATSAGLIRYMD	Native	14661030
DP00962	750	AF1 domain, Amino Terminal Domain (NTD)	NMR	720	TDNLLPMSPEEFDEVSRIVGSVEFDSMMNTV	Native	19214187
DP00607	41		SDSPAGE	1	MWTLGRRAVAGLLASPSPAQAQTLTRVPRPAELAPLCGRRG	Native	17468497

Table 2: Sample of data retrieved from databases

4.2 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 figure13.

The 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. 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. Since, over-fitting is vital test in case of training a neural network because of its non-linear and non-convex nature consisting various parameters. We have played it safe for abstaining from over fitting. We have applied a dropout rate of 0.2 in LSTM layer and L2 regularization penalty of 0.001 in dense layer. We have used ReLU activation function for Conv1D layer and softmax for time distributed dense layer. The weights in the model were optimized utilizing Adam optimizer, in order to minimize categorical cross-entropy function, using default parameters.

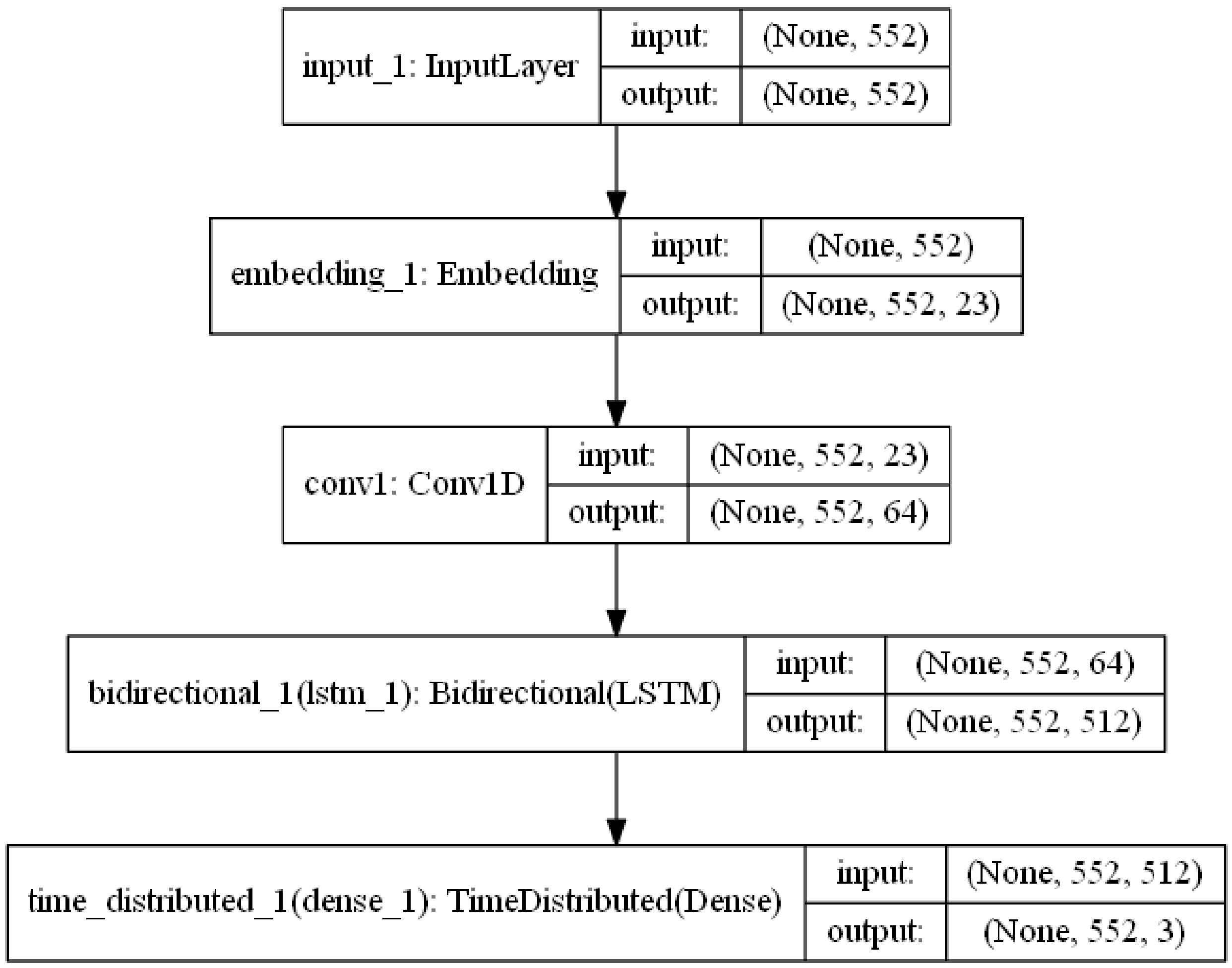


Figure 13 Final model consists of embedding, conv1D, LSTM and Time Distributed Dense layers.

4.3 PERFORMANCE EVALUTION

As our output is a singular node whose value has been compressed to ‘N’ for amino acid residues of structured regions whereas, ‘D’ for disordered regions. It is significant to consider skew-independent metrics for analyzing the performance of our model considering innately skewed distributions in datasets. The simplest metric is accuracy (it is no. of instances correctly predicted divided by total number of prediction made). Below figures depicts the accuracy and validation curves for training and validation sets.

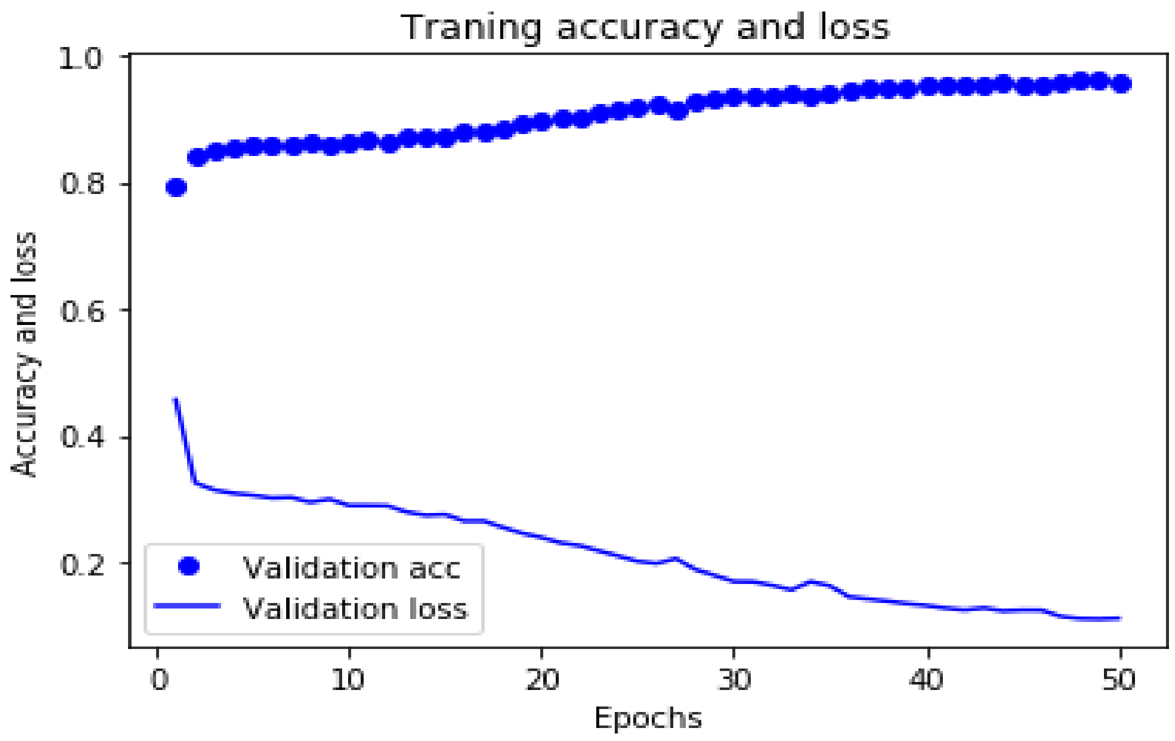


Figure 14: Training accuracy and Training loss curves

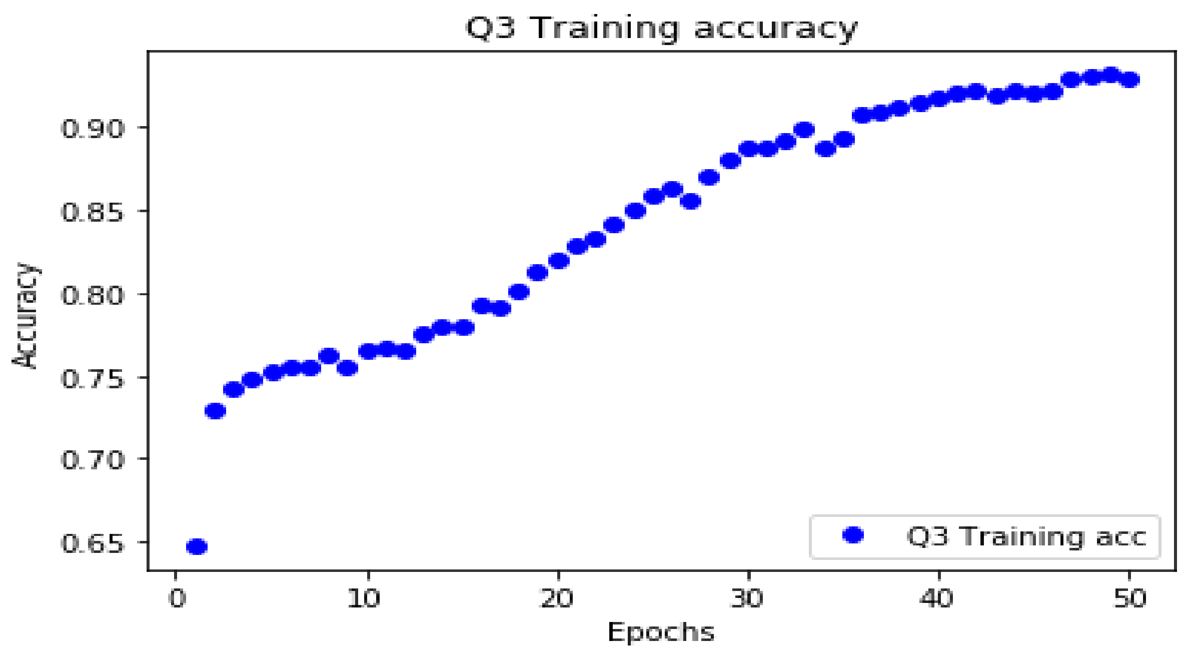


Figure 15: Q3 accuracy for training set

4.4 VALIDATION

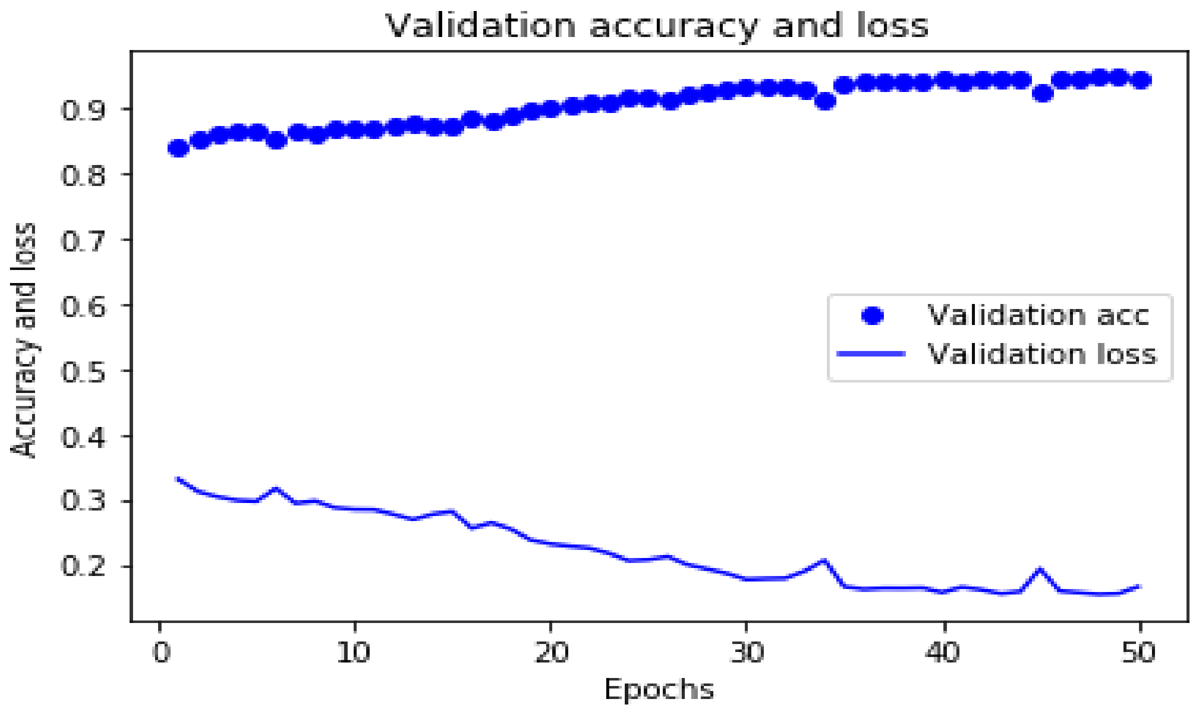


Figure 16: Validation accuracy and validation loss curves

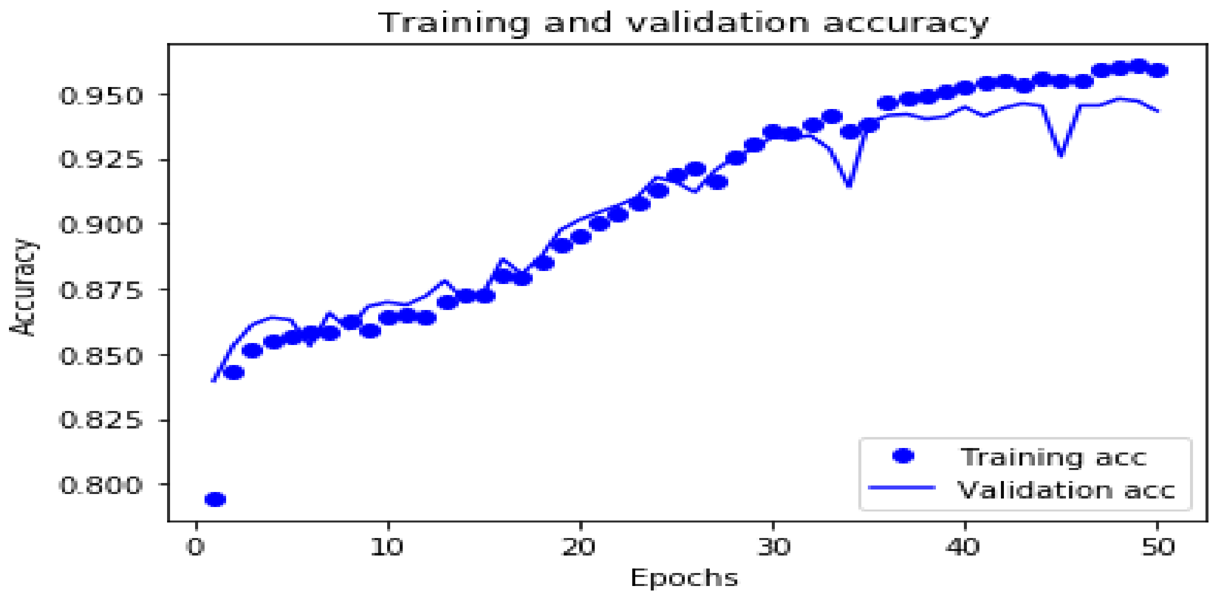


Figure 17: Accuracy curves for training and validation sets

Training Accuracy	**0.9467**
Training Loss	0.1385
Validation Accuracy	0.9370
Validation Loss	0.1706
Q3 Training Accuracy	0.9079
Q3 Validation Accuracy	0.8904

Table 3: Performance evaluation table on datasets

The results of model on datasets are presented on table 3. The datasets have substantially different ratios of ordered to disordered residues. This is largely due to different numbers of fully disordered proteins included in each set. Thus, it is not surprising that the performance varies across different datasets. Except for a few methods, the majority has a similar trend: the medium size around 30 residues has the best discrimination between unstructured and structured regions. This model is comparable to a few methods in short IDRs but are more accurate than all sequence methods in long-disordered regions. After training for 50 epochs we found that the training accuracy of our model for predicting disordered regions was around 94.67%, while in validation accuracy was around 93.70. This is the highest accuracy attained for prediction model in context to disordered regions prediction. Q3 accuracy is a 3 state per residue accuracy measures the percentage of correctly predicted residues in all the three classes and is given by Q3= 100. Sum Ci/ N where, Ci is the number of correctly predicted residues in a class and N is the number of residues. So, Q3 accuracy for training set 0.9079 and for validation 0.8904 were observed. As training accuracy was somewhat similar to validation accuracy we can state that out model was not over-fitted. Loss of 0.1385 and 0.1706 were observed in training and validation sets respectively.

After training for 50 epochs we found that the training accuracy of our model for predicting disordered regions was around 94.67%, while in validation accuracy was around 93.70. This is the highest accuracy attained for prediction model in context to disordered regions prediction. Q3 accuracy is a 3 state per residue accuracy measures the percentage of correctly predicted residues in all the three classes and is given by Q3= 100. Sum Ci/ N where, Ci is the number of correctly predicted residues in a class and N is the number of residues. So, Q3 accuracy for training set 0.9079 and for validation 0.8904 were observed. As training accuracy was somewhat similar to validation accuracy we can state that out model was not over-fitted. Loss of 0.1385 and 0.1706 were observed in training and validation sets respectively.

CHAPTER 5

DISCUSSION

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 on 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.

For developing an improved algorithm for prediction of intrinsic disorder regions considering only amino acid sequence information. We have demonstrated that utilizing the ensemble of embedding, convolutional and LSTM layers enables the technique to improve over other techniques in terms of their capability to isolate disordered regions from structured regions in amino acid chains. We trained this ensemble network on experimentally validated and disordered region sequences (7008) from DisProt and MobiDB and implementation was carried on Floyd hub cloud computing example having 32GB RAM, 11GB NVIDIA Tesla K80 GPU and Intel Xeon 8 Cores CPU for 50 epochs. The final model achieved an accuracy of 94.67%, which is considered to be the highest accuracy achieved so far, in context to disordered region prediction in proteins. Moreover, this model permits quick genome-scale prediction that is too tedious for profile-based methods. So, we finally demonstrated that this sequence based model is more precise that already existing sequence and profile based methods, for few homologous sequences.

Present study mainly focusses on identifying disordered region prediction by employing ensemble of embedding, convolutional and LSTM layers. However, subsequent work on determining the function of these regions can further help research community in the field.

APPENDICES

APPENDIX I

The python code for reading and preparing training data.

```
dataset = []
for item in data:
    dataset.append(item['sequence'])

length_list=[len(seq) for seq in dataset]

filtre_data= []

for item in data:
    if len(item['sequence'])>150 and len(item['sequence']) < 550:
        filtre_data.append(item)

with open('datset.pkl','wb') as dataset:
    pickle.dump(filtre_data,dataset)
```

```
import matplotlib.pyplot as plt

# An "interface" to matplotlib.axes.Axes.hist() method
n, bins, patches = plt.hist(x=length_list, bins='auto', color='#0504aa',
                            alpha=0.7, rwidth=0.85)

plt.boxplot(x=length_list)

import numpy as np

seq_len = np.array(length_list)

#==============================================================================
========================================
def diorder_seq(length,list_indeces):
    seq = []
    for i in range(length):
        seq.append('N')
    for items in list_indeces:
        try:
            for i in range(items[0], items[1]+1):
                seq[i]='D'
        except:
            for i in range(items[0], items[1]):
                seq[i]='D'
    return ''.join(seq)
#==============================================================================
==========================================#
with open('final_dataset.pkl','rb') as D:
    data = pickle.load(D)
```

```
sequence_data = []
for items in data:
    seq_data=[items['sequence']]
    temp_l = items['mobidb_consensus']['disorder']['db'][0]['regions']
    seq_data.append(diorder_seq(len(items['sequence']),temp_l))
    sequence_data.append(seq_data)



with open('formated_1_dataset.pkl','wb') as data_set:
    pickle.dump(sequence_data,data_set)
```

APPENDIX II

The python code for training a model.

```
# -*- coding: utf-8 -*-


# -*- coding: utf-8 -*-


# -*- coding: utf-8 -*-


# -*- coding: utf-8 -*-


# -*- coding: utf-8 -*-

import pickle
from sklearn.model_selection import train_test_split
import keras

from keras.models import Model, Input,Sequential
from keras.layers import LSTM, Embedding, Dense, TimeDistributed, Bidirectional,concatenate
from keras.metrics import categorical_accuracy
from keras.layers.convolutional import Convolution1D
from keras import backend as K
import tensorflow as tf
from keras.preprocessing import text, sequence
from keras.preprocessing.text import Tokenizer
from keras.utils import to_categorical, plot_model
from keras.layers.core import Flatten, Reshape

#tf.keras.backend.clear_session()

#==========================================================================
=========
#processing protein sequences
with open ('formated_1_dataset.pkl','rb') as data:
    data_set=pickle.load(data)
records = []
str_data =[]
for i in data_set:
    records.append(i[0])
    str_data.append(i[1])




characters=set()
for record in records:
    for char in record:
        characters.add(char)
print(characters)
print(len(characters))
count = -1
```

```
AA_list = list(characters)
aa_index = {char:AA_list.index(char) for char in AA_list}

# The first indices are reserved
aa_index = {k:(v+2) for k,v in aa_index.items()}
aa_index["<PAD>"] = 0
aa_index["<START>"] = 1
reverse_aa_index = dict([(value, key) for (key, value) in aa_index.items()])
def aa_review(text):
    return [aa_index.get(i,'?') for i in text]



seq_data = [aa_review(record) for record in records]

records = 0
#==================================================================
#==================================================================
#processing str data
#
==================================================================
========
# records = list(SeqIO.parse("proten_len_200_250_secstr.fasta", "fasta"))
# characters=set()
# recorda=0
tokenizer_decoder = Tokenizer(char_level=True)
tokenizer_decoder.fit_on_texts(str_data)
target_data = tokenizer_decoder.texts_to_sequences(str_data)
target_data = sequence.pad_sequences(target_data, maxlen=552, padding='post')
str_data = to_categorical(target_data)

# target_seqs=0
# records = 0
#==========================================================================
#==========================================================================
train_data, test_data, train_label, test_label = train_test_split(seq_data, str_data, test_size = 0.2,
random_state = 0)
seq_data = 0
str_data=0
def decode_aa_review(t):
    return ' '.join([reverse_aa_index.get(i,'?') for i in text])
maxlen = 552
train_data = keras.preprocessing.sequence.pad_sequences(train_data,
                                                        value=aa_index["<PAD>"],
                                                        padding='post',
                                                        maxlen=maxlen)

test_data = keras.preprocessing.sequence.pad_sequences(test_data,
                                                       value=aa_index["<PAD>"],
                                                       padding='post',
                                                       maxlen=maxlen)
embedding_dim=64
#model = keras.Sequential([
#  layers.Embedding(25, embedding_dim, input_length=maxlen),
#  #layers.GlobalAveragePooling1D(),
#  layers.CuDNNGRU(512),
#  layers.Dense(1024, activation='relu'),
#  layers.Dropout(0.2),
#  layers.Dense(252, activation='softmax')
#])

input = Input(shape=(maxlen,))
```

```
x = Embedding(input_dim=23, output_dim=23, input_length=maxlen)(input)
x = Convolution1D(filters=64,kernel_size=40,
                        padding='same', activation='relu', name='conv1')(x)
x = Bidirectional(LSTM(units=256, return_sequences=True, recurrent_dropout=0.2))(x)
#conca_output = concatenate([conv, x])
y = TimeDistributed(Dense(3, activation="softmax"))(x)
model = Model(input, y)
model.summary()
plot_model(model,show_shapes=True, to_file = 'model_3.png')
#model.compile(optimizer='adam',
#               loss='categorical_crossentropy',
#               metrics=['mae', 'acc'])

def q3_acc(y_true, y_pred):
    y = tf.argmax(y_true, axis=-1)
    y_ = tf.argmax(y_pred, axis=-1)
    mask = tf.greater(y, 0)
    return K.cast(K.equal(tf.boolean_mask(y, mask), tf.boolean_mask(y_, mask)), K.floatx())

model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=["accuracy",
q3_acc])



history = model.fit(train_data, train_label, batch_size=128, epochs=50,
          validation_data=(test_data, test_label ), verbose=1)
model.save('model_3 embeding.h5')

import matplotlib.pyplot as plt

acc = history.history['acc']
```

```
val_acc = history.history['val_acc']

epochs = range(1, len(acc) + 1)

plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
fig = plt.figure(figsize=(64,36))



plt.show()
plt.savefig('graph.png')

e = model.layers[1]
weights = e.get_weights()[0]
print(weights.shape) # shape: (vocab_size, embedding_dim)



out_v = open('vecs.tsv', 'w')
out_m = open('meta.tsv', 'w')
for word_num in range(23):
  word = reverse_aa_index[word_num]
  embeddings = weights[word_num]
  out_m.write(word + "\n")
  out_v.write('\t'.join([str(x) for x in embeddings]) + "\n")
out_v.close()
out_m.close()
```

APPENDIX III

Weight matrix representation for each amino acid.

	1	2	3	4	5	6
1	0.025021	-0.036475	0.061146	-0.027819	-0.220590	-0.025767
2	0.025021	0.012893	0.133511	-0.089925	0.039298	0.034693
3	0.025021	0.028937	0.009690	0.021796	0.084536	-0.01832
4	0.025021	-0.00754	0.003587	-0.09704	0.230150	-0.02719
5	0.025021	-0.02647	-0.13915	0.115094	0.081384	-0.03878
6	0.025021	-0.00863	-0.02195	-0.15833	0.137588	-0.04402
7	0.025021	0.03099	-0.025173	-0.01422	0.214023	-0.04926
8	0.025021	0.04688	0.021834	-0.16764	0.071661	0.088747
9	0.025021	0.03941	0.005920	0.094814	0.036883	-0.09925
10	0.025021	0.03170	-0.01647	-0.25267	-0.09079	0.008686
11	0.025021	-0.00048	-0.02968	0.102391	-0.06229	-0.13479
12	0.025021	-0.01187	-0.018414	0.005399	0.029898	-0.09448
13	0.025021	-0.02935	-0.036693	0.172128	0.041798	-0.05118
14	0.025021	-0.01545	0.014891	0.000304	0.024419	-0.02863
15	0.025021	-0.02666	0.033581	-0.08667	-0.312230	-0.12138
16	0.025021	-0.01517	-0.024412	0.109748	0.114898	-0.07381
17	0.025021	0.032637	0.0681441	-0.004931	0.057595	-0.06326
18	0.025021	-0.03690	-0.047979	-0.15485	0.192817	0.054848
19	0.025021	0.003111	0.150229	-0.00490	-0.04933	0.062324
20	0.025021	-0.03795	-0.029628	0.026925	0.000760	0.028242
21	0.025021	0.027033	0.071007	-0.07000	-0.11766	-0.09497
22	0.025021	-0.00915	0.043014	0.038571	0.220441	-0.15243
23	0.025021	0.048732	-0.04808	0.227790	-0.05666	-0.05622

19	18	17	16	15	14	13	12	11	10	9	8	7
-0.01845	-0.06395	0.037704	-0.042045	-0.083342	-0.10467	-0.09106	-0.08585	-0.092549	-0.052370	0.004707	-0.052807	-0.095897
0.039308	0.152532	0.004083	0.033026	-0.14215	0.037764	0.072685	0.076394	-0.040979	-0.335798	0.140524	-0.019813	0.093197
0.0840336	0.0813148	-0.065897	0.008008	-0.088425	-0.072632	0.159897	-0.01072	0.053556	0.224473	-0.11482	0.069898	-0.120718
0.011346	-0.043286	0.054116	-0.019199	0.106164	0.042912	-0.06590	-0.02760	0.081795	-0.10353	-0.11056	0.085711	0.024080
0.0942608	-0.03202	0.014990	0.054643	-0.03732	0.086834	0.072136	-0.03396	-0.16265	0.102720	0.036869	0.005357	0.078406
0.1584819	0.019868	-0.103892	0.006118	-0.14105	0.022518	0.00898	-0.00256	0.096459	0.013369	0.189452	-0.03645	-0.09229
0.063055	0.035782	0.085890	0.049960	0.125417	0.075906	0.109538	-0.05149	0.021121	-0.17423	-0.08519	0.075287	0.082806
-0.06231	0.073879	0.023988	-0.03485	-0.12409	0.053417	-0.04668	-0.03837	-0.04517	-0.01168	-0.09719	-0.05003	-0.03683
0.036788	-0.042228	0.043295	-0.06806	0.099812	0.079897	0.029064	0.066902	0.044789	0.089849	-0.05639	-0.09345	0.051829
0.031592	0.0635706	0.130286	0.108145	-0.01680	-0.127746	-0.05533	-0.02786	-0.110731	0.099581	0.211781	-0.03445	-0.152063
-0.035242	-0.055813	0.086673	-0.049247	-0.08393	0.112144	-0.06055	0.130358	0.033888	0.060588	-0.02382	0.047267	-0.052783
0.005345	0.116522	-0.243367	-0.028476	0.153057	0.143828	0.148212	0.157941	-0.02712	-0.00159	-0.00317	-0.05195	0.0659365
0.0807779	0.0210905	-0.008785	0.0430721	-0.063452	-0.046653	-0.010047	-0.022264	0.021869	0.130828	0.030222	-0.03313	0.0999959
0.0574995	0.051017	0.0033423	0.0160423	0.060379	0.112299	0.070806	-0.030534	-0.05726	0.176448	0.042278	-0.06124	-0.031587
-0.196096	0.0043363	-0.020154	-0.092898	-0.028568	-0.079685	0.0788731	0.0942997	0.055702	0.246467	0.045565	0.048606	0.0432324
0.0899031	-0.090445	-0.008971	-0.077013	-0.096864	-0.06124	-0.074294	0.054966	-0.11904	0.025186	0.098043	0.153976	-0.104602
0.059584	-0.04214	0.0976226	-0.150650	0.047377	0.067196	-0.02295	-2.06700	0.028004	0.034007	0.002242	-0.00651	0.0900318
0.116993	0.093722	0.029845	-0.02069	0.044344	-0.03045	-0.03064	0.118536	-0.10063	0.257155	0.009728	-0.05530	-0.122121
-0.06891	0.035078	0.009488	0.054965	0.055831	-0.034983	-0.08272	0.07983	0.088158	-0.09873	0.027862	-0.01865	-0.083930
-0.074982	0.0139947	0.137632	0.229791	-0.108774	-0.038539	0.083703	0.041534	0.070487	-0.24237	0.023281	0.006628	-0.042916
0.106010	-0.02052	-0.054374	-0.006126	-0.00441	-0.003738	-0.09441	-0.062519	0.118750	-0.09772	-0.02639	-0.01460	-0.093826
-0.01240	-0.08052	0.130577	0.090208	-0.01849	-0.02103	-0.009997	-0.02542	-0.04270	0.046312	-0.24773	-0.11939	-0.007711
-0.163976	-0.006119	0.110932	-0.147681	0.118759	0.144508	0.242654	-0.03798	-0.16801	0.014559	0.095847	-0.05817	-0.007892

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
20	0.077032	-0.106344	-0.035114	-0.008172	0.0432548	0.068955	0.070649	0.063123	-0.010556	0.091621	0.127482	-0.102046	0.095846	0.0777998	0.1410488	-0.040868	0.031778	-0.015942	-0.08856	-0.020348	-0.02657	0.085550	-0.197067
21	-0.04075	-0.08740	0.061003	0.240965	0.0543927	-0.128874	0.037664	-0.071908	0.156007	0.152467	-0.007902	0.075049	-0.06801	-0.005301	0.2219452	-0.115558	0.001889	-0.117522	-0.13029	-0.090724	0.058020	-0.15595	-0.161557
22	0.011735	-0.10778	0.0200452	-0.080512	0.062375	0.217763	-0.088460	-0.010975	0.085734	-0.03074	0.059862	-0.170798	-0.048157	-0.054328	-0.008949	-0.039647	0.07782	-0.042409	-0.053301	-0.100945	-0.08401	0.018063	0.047764
23	0.001438	0.072613	-0.044831	0.156198	0.1439282	-0.09324	0.129185	-0.095890	-0.017582	-0.27177	0.128796	0.109456	0.167153	0.10186	0.1940716	0.0020433	-0.025462	0.099935	-0.112447	-0.130904	-0.08382	0.156034	0.09244

REFERENCES

[1] 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.

[2] Z. Peng *et al.*, “Exceptionally abundant exceptions: Comprehensive characterization of intrinsic disorder in all domains of life,” *Cell. Mol. Life Sci.*, 2014.

[3] H. J. Dyson and P. E. Wright, “Intrinsically unstructured proteins and their functions,” *Nature Reviews Molecular Cell Biology*. 2005.

[4] 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.

[5] 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.

[6] Y. Shigemitsu and H. Hiroaki, “Common molecular pathogenesis of disease-related intrinsically disordered proteins revealed by NMR analysis,” *Journal of Biochemistry*. 2018.

[7] V. Receveur-Bréhot, J. M. Bourhis, V. N. Uversky, B. Canard, and S. Longhi, “Assessing protein disorder and induced folding,” *Proteins Struct. Funct. Genet.*, 2006.

[8] V. N. Uversky, “Functions of short lifetime biological structures at large: the case of intrinsically disordered proteins,” *Brief. Funct. Genomics*, 2018.

[9] S. DeForte and V. N. Uversky, "Resolving the ambiguity: Making sense of intrinsic disorder when PDB structures disagree," *Protein Sci.*, 2016.

[10] R. Konrat, "NMR contributions to structural dynamics studies of intrinsically disordered proteins," *J. Magn. Reson.*, 2014.

[11] D. Piovesan *et al.*, "MobiDB 3.0: More annotations for intrinsic disorder, conformational diversity and interactions in proteins," *Nucleic Acids Res.*, 2018.

[12] 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.

[13] 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.

[14] R. J. P. WILLIAMS, "THE CONFORMATION PROPERTIES OF PROTEINS IN SOLUTION," *Biol. Rev.*, 1979.

[15] 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.

[16] 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.

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

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

[19] 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.

[20] 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.

[21] P. Romero, Z. Obradovic, X. Li, E. C. Garner, C. J. Brown, and A. K. Dunker, "Sequence complexity of disordered protein," *Proteins Struct. Funct. Genet.*, 2001.

[22] 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.

[23] 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.

[24] I. Walsh, A. J. M. Martin, T. Di domenico, and S. C. E. Tosatto, “Espritz: Accurate and fast prediction of protein disorder,” *Bioinformatics*, 2012.

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

[26] M. S. Klausen *et al.*, “NetSurfP-2.0: Improved prediction of protein structural features by integrated deep learning,” *Proteins Struct. Funct. Bioinforma.*, 2019.

[27] 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.

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

[29] S. Ovchinnikov *et al.*, “Protein structure determination using metagenome sequence data,” *Science (80-.).*, 2017.

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

[31] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning Internal Representations by Error Propagation,” in *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, 2013.

[32] K. He, X. Zhang, S. Ren, and J. Sun, “Identity mappings in deep residual networks,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016.

[33] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, 1997.

[34] 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.

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

[36] 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.

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

[38] D. B. Kell, “Metabolomics, machine learning and modelling: towards an understanding of the language of cells,” *Biochem. Soc. Trans.*, 2006.

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

[40] J. Schmidhuber, “Deep Learning in neural networks: An overview,” *Neural Networks*. 2015.

[41] G. Hinton *et al.*, “Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups,” *IEEE Signal Process. Mag.*, 2012.

[42] A. K. Dunker, C. J. Brown, J. D. Lawson, L. M. Iakoucheva, and Z. Obradović, “ Intrinsic Disorder and Protein Function † ,” *Biochemistry,* 2002.

[43] V. N. Uversky, “Natively unfolded proteins: A point where biology waits for physics,” *Protein Sci.,* 2002.

[44] C. BOESCH, A. BUNDI, M. OPPLIGER, and K. WüTHRICH, “1H Nuclear-Magnetic-Resonance Studies of the Molecular Conformation of Monomeric Glucagon in Aqueous Solution,” *Eur. J. Biochem.,* 1978.

[45] A. J. Daniels, R. J. P. Williams, and P. E. Wright, “The character of the stored molecules in chromaffin granules of the adrenal medulla: A nuclear magnetic resonance study,” *Neuroscience,* 1978.

[46] P. E. Wright and H. J. Dyson, “Intrinsically unstructured proteins: Re-assessing the protein structure-function paradigm,” *J. Mol. Biol.,* 1999.

[47] L. M. Iakoucheva, C. J. Brown, J. D. Lawson, Z. Obradović, and A. K. Dunker, “Intrinsic disorder in cell-signaling and cancer-associated proteins.,” *J. Mol. Biol.,* 2002.

[48] P. Tompa, “Intrinsically unstructured proteins.,” *Trends Biochem. Sci.,* 2002.

[49] S. Vucetic, C. J. Brown, A. K. Dunker, and Z. Obradovic, “Flavors of protein disorder,” *Proteins Struct. Funct. Genet.,* 2003.

[50] P. J. Mitchell and R. Tjian, “Transcriptional regulation in mammalian cells by sequence-specific DNA binding proteins,” *Science (80-.).,* 1989.

[51] P. O’hare and G. Williams, “Structural Studies of the Acidic Transactivation Domain of the Vmw65 Protein of Herpes Simplex Virus Using 1H NMR,” *Biochemistry,* 1992.

[52] S. Longhi *et al.*, “The C-terminal domain of the measles virus nucleoprotein is intrinsically disordered and folds upon binding to the C-terminal moiety of the phosphoprotein,” *J. Biol. Chem.*, 2003.

[53] V. N. Uversky, J. R. Gillespie, and A. L. Fink, “Why are ‘natively unfolded’ proteins unstructured under physiologic conditions?,” *Proteins Struct. Funct. Genet.,* 2000.

[54] J. J. Ward, J. S. Sodhi, L. J. McGuffin, B. F. Buxton, and D. T. Jones, “Prediction and functional analysis of native disorder in proteins from the three kingdoms of life.,” *J. Mol. Biol.,* 2004.

[55] E. A. Weathers, M. E. Paulaitis, T. B. Woolf, and J. H. Hoh, “Reduced amino acid alphabet is sufficient to accurately recognize intrinsically disordered protein,” *FEBS Lett.*, 2004.

[56] S. Vucetic *et al.*, "DisProt: A database of protein disorder," *Bioinformatics,* 2005.

[57] H. J. Dyson and P. E. Wright, "Nuclear magnetic resonance methods for elucidation of structure and dynamics in disordered states," in *Methods in Enzymology,* 2001.

[58] H. J. Dyson and P. E. Wright, "Unfolded Proteins and Protein Folding Studied by NMR," *Chem. Rev.,* 2004.

[59] H. J. Dyson and P. E. Wright, "Insights into the structure and dynamics of unfolded proteins from nuclear magnetic resonance," *Advances in Protein Chemistry*. 2002.

[60] A. L. Rucker, C. T. Pager, M. N. Campbell, J. E. Qualls, and T. P. Creamer, "Host-guest scale of left-handed polyproline II helix formation," *Proteins Struct. Funct. Genet.,* 2003.

[61] Z. Shi, C. A. Olson, G. D. Rose, R. L. Baldwin, and N. R. Kallenbach, "Polyproline II structure in a sequence of seven alanine residues," *Proc. Natl. Acad. Sci.,* 2002.

[62] A. K. Dunker, Z. Obradovic, P. Romero, E. C. Garner, and C. J. Brown, "Intrinsic protein disorder in complete genomes.," *Genome Inform. Ser. Workshop Genome Inform.,* 2000.

[63] J. C. Wootton and M. H. Drummond, "The q-linker: A class of interdomain sequences found in bacterial multidomain regulatory proteins," *Protein Eng. Des. Sel.,* 1989.

[64] H. X. Zhou, "Quantitative account of the enhanced affinity of two linked scFvs specific for different epitopes on the same antigen," *J. Mol. Biol.,* 2003.

[65] J. H. Laity, H. J. Dyson, and P. E. Wright, "DNA-induced α-helix capping in conserved linker sequences is a determinant of binding affinity in Cys2-His2 zinc fingers," *J. Mol. Biol.,* 2000.

[66] M. A. Young, S. Gonfloni, G. Superti-Furga, B. Roux, and J. Kuriyan, "Dynamic coupling between the SH2 and SH3 domains of c-Src and Hck underlies their inactivation by C-Terminal tyrosine phosphorylation," *Cell,* 2001.

[67] J. H. Laity, H. J. Dyson, and P. E. Wright, "Molecular basis for modulation of biological function by alternate splicing of the Wilms' tumor suppressor protein," *Proc. Natl. Acad. Sci.,* 2002.

[68] K. Namba, "Roles of partly unfolded conformations in macromolecular self-assembly," *Genes to Cells*. 2001.

[69] P. TOMPA and P. CSERMELY, "The role of structural disorder in the function of RNA and protein chaperones," *FASEB J.,* 2004.

[70] H. J. Dyson and P. E. Wright, "Coupling of folding and binding for unstructured proteins.," *Curr. Opin. Struct. Biol.,* 2002.

[71] A. P. Demchenko, "Recognition between flexible protein molecules: Induced and assisted folding," *Journal of Molecular Recognition*. 2001.

[72] I. Radhakrishnan, G. C. Pérez-Alvarado, H. J. Dyson, and P. E. Wright, "Conformational

preferences in the Ser133-phosphorylated and non- phosphorylated forms of the kinase inducible transactivation domain of CREB," *FEBS Lett.*, 1998.

[73] J. P. Richards, H. P. Bächinger, R. H. Goodman, and R. G. Brennan, "Analysis of the structural properties of cAMP-responsive element-binding protein (CREB) and phosphorylated CREB," *J. Biol. Chem.*, 1996.

[74] I. Radhakrishnan, G. C. Pérez-Alvarado, D. Parker, H. J. Dyson, M. R. Montminy, and P. E. Wright, "Solution structure of the KIX domain of CBP bound to the transactivation domain of CREB: A model for activator:coactivator interactions," *Cell*, 1997.

[75] P. Zhou, A. A. Lugovskoy, J. S. McCarty, P. Li, and G. Wagner, "Solution structure of DFF40 and DFF45 N-terminal domain complex and mutual chaperone activity of DFF40 and DFF45," *Proc. Natl. Acad. Sci.*, 2002.

[76] R. S. Spolar and M. T. Record, "Coupling of local folding to site-specific binding of proteins to DNA," *Science (80-.).*, 1994.

[77] L. Patel, C. Abate, and T. Curran, "Altered protein conformation on DNA binding by Fos and Jun," *Nature*, 1990.

[78] J. P. DiNitto and P. W. Huber, "Mutual induced fit binding of Xenopus ribosomal protein L5 to 5 S rRNA," *J. Mol. Biol.*, 2003.

[79] S. E. Salghetti, A. A. Caudy, J. G. Chenoweth, and W. P. Tansey, "Regulation of transcriptional activation domain function by ubiquitin," *Science (80-.).*, 2001.

[80] P. Venkatraman, R. Wetzel, M. Tanaka, N. Nukina, and A. L. Goldberg, "Eukaryotic proteasomes cannot digest polyglutamine sequences and release them during degradation of polyglutamine-containing proteins," *Mol. Cell*, 2004.

[81] X. J. Yang, "The diverse superfamily of lysine acetyltransferases and their roles in leukemia and other diseases," *Nucleic Acids Research*. 2004.

[82] A. McCampbell and K. H. Fischbeck, "Polyglutamine and CBP: Fatal attraction?," *Nature Medicine*. 2001.

[83] J. Nucifora *et al.*, "Interference by huntingtin and atrophin-1 with CBP-mediated transcription leading to cellular toxicity," *Science (80-.).*, 2001.

[84] S. Karlin, L. Brocchieri, A. Bergman, J. Mrazek, and A. J. Gentles, "Amino acid runs in eukaryotic proteomes and disease associations," *Proc. Natl. Acad. Sci.*, 2002.

[85] 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.

[86] J. Cheng, M. J. Sweredoski, and P. Baldi, "Accurate prediction of protein disordered regions by mining protein structure data," *Data Min. Knowl. Discov.*, 2005.

[87] B. Liu, F. Liu, X. Wang, J. Chen, L. Fang, and K. C. Chou, "Pse-in-One: A web server for

generating various modes of pseudo components of DNA, RNA, and protein sequences," *Nucleic Acids Res.*, 2015.

[88] L. Wei and Q. Zou, "Recent progress in machine learning-based methods for protein fold recognition," *International Journal of Molecular Sciences*. 2016.

[89] D. Li, Y. Ju, and Q. Zou, "Protein Folds Prediction with Hierarchical Structured SVM," *Curr. Proteomics*, 2016.

[90] H. Lin, Z. Y. Liang, H. Tang, and W. Chen, "Identifying sigma70 promoters with novel pseudo nucleotide composition," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2017.

[91] C.-J. Zhang, H. Tang, W.-C. Li, H. Lin, W. Chen, and K.-C. Chou, "iOri-Human: identify human origin of replication by incorporating dinucleotide physicochemical properties into pseudo nucleotide composition," *Oncotarget*, 2016.

[92] H. Yang *et al.*, " Identification of Secretory Proteins in Mycobacterium tuberculosis Using Pseudo Amino Acid Composition ," *Biomed Res. Int.*, 2016.

[93] B. Liu, J. Chen, and X. Wang, "Application of learning to rank to protein remote homology detection," *Bioinformatics*, 2015.

[94] L. P. Kozlowski and J. M. Bujnicki, "MetaDisorder: a meta-server for the prediction of intrinsic disorder in proteins," *BMC Bioinformatics*, 2012.

[95] T. Ishida and K. Kinoshita, "Prediction of disordered regions in proteins based on the meta approach," *Bioinformatics*, 2008.

[96] A. Schlessinger, M. Punta, G. Yachdav, L. Kajan, and B. Rost, "Improved disorder prediction by combination of orthogonal approaches," *PLoS One*, 2009.

[97] M. Necci, D. Piovesan, Z. Dosztanyi, and S. C. E. Tosatto, "MobiDB-lite: Fast and highly specific consensus prediction of intrinsic disorder in proteins," *Bioinformatics*, 2017.

[98] T. Ishida and K. Kinoshita, "PrDOS: Prediction of disordered protein regions from amino acid sequence," *Nucleic Acids Res.*, 2007.