IDRPRED - A COMPUTATIONAL PROTOCOL FOR ENHANCEMENT OF ACCURACY IN IDENTIFICATION OF INTRINSICALLY DISORDERED REGIONS OF PROTEINS

Submitted in partial fulfillment of the requirements for the award of the degree of
Master of Technology
in
Bioinformatics

Submitted by
Soumak Dalapati
MBI2021017

Under the
Supervision
of

Dr. Tapobrata Lahiri

Department of Applied Sciences
INDIAN INSTITUTE OF INFORMATION TECHNOLOGY
ALLAHABAD
Prayagraj -211015

May, 2023

CANDIDATE DECLARATION

I hereby declare that the work presented in this report entitled “IDRPred - A Computational Protocol for enhancement of accuracy in the identification of IDR of proteins”, submitted towards fulfillment of MASTER’S THESIS report of **Bioinformatics** at Indian Institute of Information Technology Allahabad, is an authenticated record of our original work carried out under the guidance of **Prof./Dr. Tapobrata Lahiri.** Due acknowledgements have been made in the text to all other material used. The project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Soumak Dalapati
MBI2021017
Dept. of Applied Sciences

CERTIFICATE FROM SUPERVISOR

This is to certify that the statement made by the candidate is correct to the best of my knowledge and belief. The project titled "IDRPred - A Computational Protocol for enhancement of accuracy in the identification of IDR of proteins" is a record of candidates' work carried out by him under my guidance and supervision. I do hereby recommend that it should be accepted in the fulfillment of the requirements of the Master's thesis at IIIT Allahabad.

Dr. Tapobrata Lahiri

CERTIFICATE OF APPROVAL

The forgoing thesis is hereby approved as a creditable study carried out in the area of Applied Sciences and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the thesis only for the purpose for which it is submitted.

Committee on final examination for the evaluation of thesis:

1. ____

2. ____

3. ____

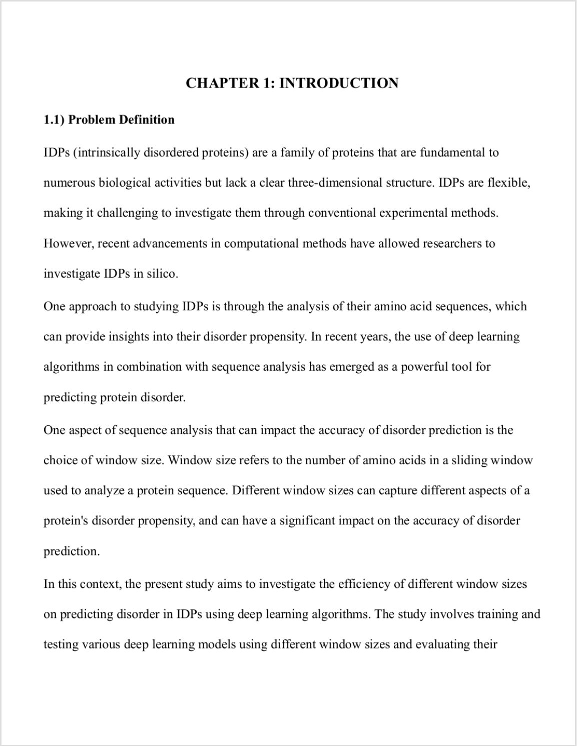
Dean(A & R)

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Soumak Dalpati
Assignment title:	thests
Submission title:	thesis
File name:	thesis2.pdf
File size:	2.51M
Page count:	48
Word count:	7,789
Character count:	42,073
Submission date:	17-May-2023 04:58PM (UTC+0530)
Submission ID:	2095369225



CHAPTER 1: INTRODUCTION

1.1) Problem Definition

IDPs (intrinsically disordered proteins) are a family of proteins that are fundamental to numerous biological activities but lack a clear three-dimensional structure. IDPs are flexible, making it challenging to investigate them through conventional experimental methods. However, recent advancements in computational methods have allowed researchers to investigate IDPs in silico.

One approach to studying IDPs is through the analysis of their amino acid sequences, which can provide insights into their disorder propensity. In recent years, the use of deep learning algorithms in combination with sequence analysis has emerged as a powerful tool for predicting protein disorder.

One aspect of sequence analysis that can impact the accuracy of disorder prediction is the choice of window size. Window size refers to the number of amino acids in a sliding window used to analyze a protein sequence. Different window sizes can capture different aspects of a protein's disorder propensity, and can have a significant impact on the accuracy of disorder prediction.

In this context, the present study aims to investigate the efficiency of different window sizes on predicting disorder in IDPs using deep learning algorithms. The study involves training and testing various deep learning models using different window sizes and evaluating their

thesis

ORIGINALITY REPORT

4%	2%	4%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	dokumen.pub Internet Source	1%
2	Mohammad Shahin, F. Frank Chen, Ali Hosseinzadeh, Mazdak Maghanaki. "Waste Reduction via Computer Vision-based Inspection: Towards Lean Systems in Metal Production", Research Square Platform LLC, 2023 Publication	1%
3	"Medical Biometrics", Springer Nature, 2010 Publication	1%
4	M. Arif Wani, Farooq Ahmad Bhat, Saduf Afzal, Asif Iqbal Khan. "Advances in Deep Learning", Springer Science and Business Media LLC, 2020 Publication	1%
5	"Advanced Computing Technologies and Applications", Springer Science and Business Media LLC, 2020 Publication	1%

ACKNOWLEDGMENT

First and foremost, I want to convey my thanks to IIIT-ALLAHABAD. IIIT-A has given me the opportunity to acquire new approaches and concepts in order to improve my skills. Being a member of the IIIT-A family is the best asset I believe I have.

My mentor, Prof. Tapobrata Lahiri, was the most noteworthy source of advice. I owe him a debt of gratitude for giving me a novel concept. He has been my mentor throughout the thesis process. I'd like to express my gratitude for his insightful advice and support.

I'd like to offer my heartfelt gratitude and appreciation to Mr. Deepak Chaurasiya and Miss Asmita Tripathi for their invaluable assistance with my thesis.

I'd like to express my gratitude to my students and friends for their participation. Many people assisted and supported me during my thesis study, and I am grateful to them all.

This recognition would be inadequate without expressing my heartfelt gratitude to my family, without whom this thesis would not have been possible.

Soumak Dalapati
MBI2021017
May, 2023

ABSTRACT

Intrinsically disordered proteins (IDPs) are a fascinating class of proteins that lack a fixed three-dimensional structure. The study of IDPs has uncovered their pivotal roles in numerous cellular processes, including signaling, regulation, and protein-protein interactions. One of the key approaches in investigating IDPs involves window-based analysis, where different window sizes are used to study the protein's flexibility and functionality.

The analysis begins by introducing the concept of window-based analysis and its application in studying IDPs. By considering different window sizes, researchers gain insights into various regions of the protein, revealing the contributions of specific segments to its overall flexibility and functional properties. This approach, combined with deep learning methods, enables the development of efficient models for predicting IDP behavior and its relationship to disease.

In summary, this review underscores the importance of window-based analysis in investigating the link between disorder and disease in IDPs. By considering different window sizes, researchers can gain crucial insights into the structural dynamics and functional implications of IDPs. This approach offers a promising avenue for unraveling the molecular basis of various diseases and developing targeted therapeutic interventions.

TABLE OF CONTENTS

THESIS OUTLINES

Below mentioned are the outlines of my thesis chapter wise:

Chapter 1: The first chapter gives an overview of disordered proteins, as well as the goal, purpose, and problem definition.

Chapter 2: The second chapter offers a summary of the literature review on IDPs and IDRs.

Chapter 3: This chapter contains information about the DisProt database, which is used to obtain the sequences for Training,Testing and Validation and about CIDER which is used to obtain features for these proteins.

Chapter 4: This chapter contains the discussion on the CNN algorithm which I have applied to train and test the model and predict various accuracy scores and brief introduction on the Neural Networks.

Chapter 5: This chapter discusses the procedures I used to complete this thesis.

Chapter 6: This chapter contains the findings of my approaches.

Chapter 7: This chapter includes the conclusion as well as potential future work.

CHAPTER 1: INTRODUCTION

1.1) Problem Definition

IDPs (intrinsically disordered proteins) are a family of proteins that are fundamental to numerous biological activities but lack a clear three-dimensional structure. IDPs are flexible, making it challenging to investigate them through conventional experimental methods. However, recent advancements in computational methods have allowed researchers to investigate IDPs in silico.

One approach to studying IDPs is through the analysis of their amino acid sequences, which can provide insights into their disorder propensity. In recent years, the use of deep learning algorithms in combination with sequence analysis has emerged as a powerful tool for predicting protein disorder.

One aspect of sequence analysis that can impact the accuracy of disorder prediction is the choice of window size. Window size refers to the number of amino acids in a sliding window used to analyze a protein sequence. Different window sizes can capture different aspects of a protein's disorder propensity, and can have a significant impact on the accuracy of disorder prediction.

In this context, the present study aims to investigate the efficiency of different window sizes on predicting disorder in IDPs using deep learning algorithms. The study involves training and testing various deep learning models using different window sizes and evaluating their performance. The results of this study can help researchers build more precise methods for researching IDPs and shed light on the ideal window size for anticipating disorder in IDPs.

1.2) Motivation

The analysis of IDP proteins at different window sizes and their efficiency using deep learning has several motivating factors. First off, IDPs are essential for a number of cellular processes, including transcription, regulation, and signaling. Understanding the structural dynamics of IDPs is therefore essential for elucidating their functional mechanisms.

Second, due to IDPs' inherent flexibility and lack of a well defined structure, conventional experimental methods like X-ray crystallography and nuclear magnetic resonance spectroscopy are limited in their ability to examine them. Therefore, computational methods such as analyzing IDPs at different window sizes and using deep learning models are essential for predicting and understanding their structural dynamics.

Thirdly, the development of new therapeutics that target IDPs has gained considerable attention in recent years. IDPs are involved in numerous diseases, including cancer, neurodegeneration, and viral infections. By understanding the structural dynamics of IDPs, researchers can develop new drugs that target specific regions of the protein, leading to more effective treatments with fewer side effects.

Finally, the field of protein structure prediction and analysis is rapidly advancing, with new computational techniques and algorithms being developed continuously. The analysis of IDPs at different window sizes and their efficiency using deep learning is a promising approach that has the potential to yield novel insights into the structure and function of these proteins. As such, the motivation for such analysis is to gain a better understanding of the complex and dynamic nature of IDPs, leading to new discoveries in the field of biochemistry and medicine.

1.3) Methods of Prediction

Predicting the location of these IDRs is an essential step towards understanding the functional mechanisms of these proteins. Several computational methods have been developed to predict IDRs, each with its own strengths and limitations.

One approach involves analyzing protein sequences using machine learning algorithms. These algorithms are trained on large datasets of proteins with experimentally validated IDRs, allowing them to predict IDRs based on specific sequence features such as amino acid composition, hydrophobicity, and charge distribution. Machine learning algorithms such as support vector machines (SVM), artificial neural networks (ANN), and hidden Markov models (HMM) have been widely used for IDR prediction.

Another approach involves using disorder prediction algorithms that are based on the principles of statistical mechanics and machine learning. These algorithms use sequence information and predicted protein properties, such as secondary structure and solvent accessibility, to predict the propensity of a given residue to be disordered. Examples of widely used disorder prediction algorithms include DISOPRED, DISpro, and IUPred. Additionally, several databases have been developed to compile experimentally validated IDRs. These databases provide a resource for researchers to study and analyze the properties and functions of IDRs. Examples of such databases include DisProt, MobiDB, and D2P2.

1.4) Disease and Disorder

Intrinsically disordered proteins (IDPs) are involved in a wide range of biological functions, including signal transduction, transcriptional control, and protein-protein interactions, according to research on IDPs. IDPs' diminished functionality or development of harmful functions, however, have also been linked to a number of illnesses.

For example, mutations in the IDP α-synuclein have been associated with Parkinson's disease, a neurodegenerative disorder. It has been suggested that the accumulation of α-synuclein in neuronal cells leads to the formation of toxic aggregates that disrupt normal cellular function, ultimately leading to cell death.

Similarly, the IDP p53, a tumor suppressor protein, is frequently mutated in various cancers. Mutations in p53 often result in a loss of function, which can lead to the uncontrolled growth of cells and the development of cancer.

IDPs have also been linked to prion disorders, ALS, and amyotrophic lateral sclerosis (ALS), among other illnesses. The accumulation of IDPs is believed to be a critical factor in the pathophysiology of each of these disorders.

Overall, the understanding of the molecular pathways underlying numerous diseases has improved as a result of the research of IDPs and their function in disease. The development of therapeutic strategies targeting IDPs has emerged as a promising approach for the treatment of these diseases.

CHAPTER 2: LITERATURE REVIEW

2.1) IDPs, IDRs and their Properties

IDPs (intrinsically disordered proteins) are a type of proteins that function normally but lack a clear three-dimensional structure. These proteins frequently have adaptable, dynamic regions of intrinsic disorder known as intrinsically disordered regions (IDRs).

IDRs can possess a range of characteristics, including as a high net charge, a low hydrophobicity, and a high degree of flexibility. These characteristics enable IDRs to interact with a variety of binding partners and take part in a number of biological activities, including protein-protein interactions, signal transduction, and transcriptional control.

IDPs don't have a well defined structure because they lack a stable hydrophobic core and have a lot of charged and polar residues. IDPs can undergo conformational changes and take on various forms while engaging with various binding partners because they lack a fixed structure.

IDRs can also be post-translationally modified, such as through phosphorylation or acetylation, which can modulate their interaction with other proteins and cellular processes.

IDRs don't have a clear structure, but they are essential to many biological functions and are frequently linked to the onset of diseases like cancer and neurodegenerative disorders. Understanding the properties of IDRs and their interaction with binding partners can provide insights into the molecular mechanisms underlying these diseases and potentially lead to the development of novel therapeutic strategies.

2.2) IDPs and their "Mysterious Physics"

Research on the physics of intrinsically disordered proteins (IDPs) is progressing, however it is yet unclear. IDPs are extremely flexible and lack a stable structure, in contrast to folded proteins, which have a clear three-dimensional shape.

One of the key factors contributing to the mysterious physics of IDPs is the high level of disorder and flexibility within these proteins. IDPs often contain regions of intrinsic disorder, or intrinsically disordered regions (IDRs), which are highly dynamic and can adopt different conformations. Research on the physics of intrinsically disordered proteins (IDPs) is progressing, however it is yet unclear. IDPs are extremely flexible and lack a stable structure, in contrast to folded proteins, which have a clear three-dimensional shape.

Another factor contributing to the mysterious physics of IDPs is the lack of understanding of the thermodynamics and kinetics of their interactions with binding partners. IDPs are thought to undergo conformational changes upon binding to other proteins or molecules, but the underlying mechanisms of these interactions are not yet fully understood.

Despite these challenges, research on the physics of IDPs is advancing rapidly. Thanks to advancements in experimental methods like nuclear magnetic resonance spectroscopy and single-molecule fluorescence spectroscopy, researchers have been able to investigate the dynamics and interactions of IDPs in greater detail. Additionally, computational methods, including molecular dynamics simulations, have provided insights into the behavior of IDPs at the atomic level.

Overall, the mysterious physics of IDPs presents a fascinating challenge for researchers in the field, and continued investigation into these proteins promises to provide important insights into the fundamental principles of protein dynamics and interactions.

2.3) Review and comparison of IDPs and region of forecast methodologies currently in use

There are several existing methods for predicting intrinsically disordered proteins (IDPs) and their regions, and these methods differ in their approaches and underlying algorithms. In this review, we will compare and contrast some of the most widely used methods for IDP prediction.

One common approach for IDP prediction is based on amino acid sequence analysis. These methods utilize sequence-based features, such as predicted disorder propensity and hydrophobicity, to identify IDPs and their regions. Examples of sequence-based methods include DISOPRED, IUPred, and PONDR.

Another approach for IDP prediction is based on experimental techniques, such as nuclear magnetic resonance spectroscopy and circular dichroism spectroscopy. These methods measure the dynamics and conformation of proteins under different conditions, and can provide valuable insights into the structure and function of IDPs.

For IDP prediction, machine learning techniques such as neural networks and support vector machines have also been developed. These methods utilize large datasets of known IDPs and their sequences to develop predictive models that can identify IDPs and their regions.

Convolutional and recurrent neural networks, among other deep learning techniques, have recently become effective tools for IDP prediction. These methods can process large amounts of sequence data and learn complex features, allowing for more accurate and efficient IDP prediction.

Another approach for predicting intrinsically disordered regions (IDRs) within intrinsically disordered proteins (IDPs) is the use of window-based methods. These methods involve breaking up the protein sequence into shorter segments, or windows, and predicting the disorder propensity of each window.

Window-based approaches can be based on a variety of properties, including projected solvent accessibility, predicted secondary structure, and predicted amino acid composition. These features are calculated for each window and used to predict the disorder propensity of that window. By sliding the window along the protein sequence, a prediction can be made for each position within the protein.

One advantage of window-based methods is that they can capture local variations in disorder propensity within the protein. This is particularly important for IDPs, which can have regions of disorder and order within the same protein. By predicting disorder propensity at the local level, window-based methods can provide more accurate predictions of IDRs.

However, window-based methods can also be sensitive to the choice of window size and the overlap between windows. A smaller window size may capture local variations more accurately but may miss larger-scale patterns of disorder, while a larger window size may miss local variations in disorder. The amount of overlap between windows can also affect the accuracy of predictions, with greater overlap potentially leading to more accurate predictions but slower computational times.

Window-based methods offer a useful approach for predicting IDRs within IDPs, but careful consideration must be given to the choice of window size and overlap. Additionally, other factors, such as the choice of features and machine learning algorithms, can also affect the accuracy of predictions.

Overall, the choice of method for IDP prediction depends on the specific research question and available data. Sequence-based methods are useful for predicting IDPs from amino acid sequences, while experimental techniques provide direct measurements of protein conformation and dynamics. Machine learning and deep learning approaches offer powerful tools for IDP prediction from large datasets, and can provide insights into the underlying physical and biochemical principles governing IDP behavior.

2.4) Studying the Efficiency using Window-based Techniques

The effectiveness of prediction approaches can be greatly improved by using window-based techniques to forecast intrinsically disordered regions (IDRs) within intrinsically disordered proteins (IDPs). By varying the window size and overlap between windows, the accuracy of predictions can be assessed for different types of IDPs and under different conditions.

For example, larger window sizes may be more effective for predicting IDRs in IDPs that have longer stretches of disorder, while smaller window sizes may be more effective for predicting IDRs in IDPs that have shorter stretches of disorder. Similarly, greater overlap between windows may be more effective for predicting IDRs in IDPs with more complex disorder patterns, while less overlap may be more effective for predicting IDRs in IDPs with simpler disorder patterns.

By comparing the accuracy of predictions across different window sizes and overlaps, researchers can identify the optimal window size and overlap for predicting IDRs in specific types of IDPs. This data can be utilized to increase the precision of forecasts and to direct the creation of brand-new prediction techniques.

In addition, studying the efficiency of window-based techniques for predicting IDRs can provide insights into the underlying physical and biochemical principles that govern IDP behavior. By analyzing the features that contribute to accurate predictions, researchers can identify key factors that contribute to disorder and binding interactions within IDPs.

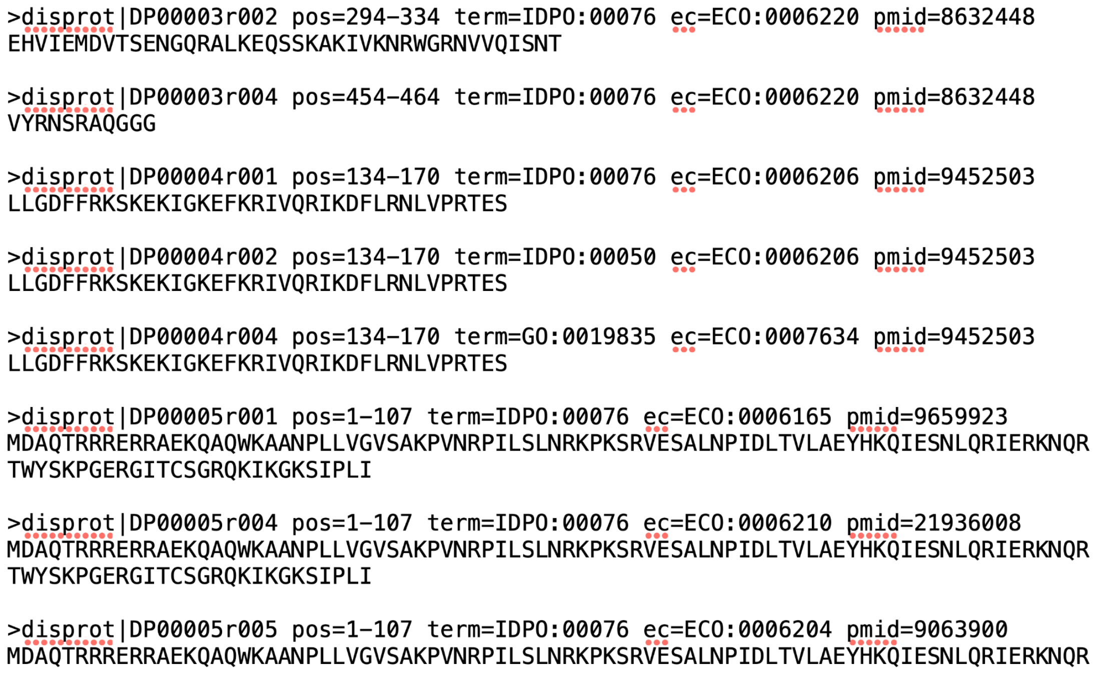
Overall, studying the efficiency of window-based techniques for predicting IDRs within IDPs can provide valuable insights into the behavior of these proteins and can guide the development of new prediction methods that more accurately capture the complexity of IDP behavior.

CHAPTER 3: FEATURE EXTRACTION ANALYSIS

3.1) IDR sequences from DisProt Database:

A large collection of experimentally validated intrinsically disordered protein (IDP) sequences can be found in the DisProt database. It has been discovered that proteins' disordered regions are crucial for a number of biological processes, including protein-protein interactions, signaling cascades, and transcriptional control. DisProt contains a wealth of information on disordered regions, including their location, length, and functional properties.

DisProt sequences are classified into two categories: "fully disordered" and "disorder in part of the protein". The fully disordered category contains sequences where the entire protein is disordered, while the "disorder in part of the protein" category contains sequences where only part of the protein is disordered. DisProt also provides additional annotations for each sequence, including functional annotations and information on experimental techniques used to verify disorder. A sample screenshot of IDR sequences from Disprot in given below



```
>disprot|DP00003r002 pos=294-334 term=IDPO:00076 ec=ECO:0006220 pmid=8632448
EHVIEMDVTSENGQRALKEQSSKAKIVKNRWGRNVVQISNT

>disprot|DP00003r004 pos=454-464 term=IDPO:00076 ec=ECO:0006220 pmid=8632448
VYRNSRAQGGG

>disprot|DP00004r001 pos=134-170 term=IDPO:00076 ec=ECO:0006206 pmid=9452503
LLGDFFRKSKEKIGKEFKRIVQRIKDFLRNLVPRTES

>disprot|DP00004r002 pos=134-170 term=IDPO:00050 ec=ECO:0006206 pmid=9452503
LLGDFFRKSKEKIGKEFKRIVQRIKDFLRNLVPRTES

>disprot|DP00004r004 pos=134-170 term=GO:0019835 ec=ECO:0007634 pmid=9452503
LLGDFFRKSKEKIGKEFKRIVQRIKDFLRNLVPRTES

>disprot|DP00005r001 pos=1-107 term=IDPO:00076 ec=ECO:0006165 pmid=9659923
MDAQTRRRERRAEKQAQWKAANPLLVGVSAKPVNRPILSLNRKPKSRVESALNPIDLTVLAEYHKQIESNLQRIERKNQR
TWYSKPGERGITCSGRQKIKGKSIPLI

>disprot|DP00005r004 pos=1-107 term=IDPO:00076 ec=ECO:0006210 pmid=21936008
MDAQTRRRERRAEKQAQWKAANPLLVGVSAKPVNRPILSLNRKPKSRVESALNPIDLTVLAEYHKQIESNLQRIERKNQR
TWYSKPGERGITCSGRQKIKGKSIPLI

>disprot|DP00005r005 pos=1-107 term=IDPO:00076 ec=ECO:0006204 pmid=9063900
MDAQTRRRERRAEKQAQWKAANPLLVGVSAKPVNRPILSLNRKPKSRVESALNPIDLTVLAEYHKQIESNLQRIERKNQR
```

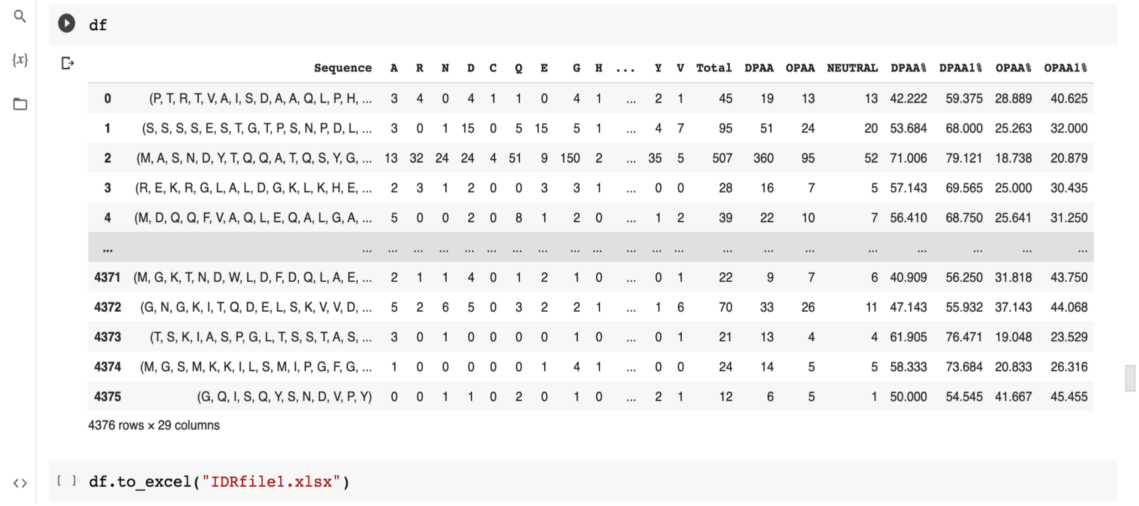
3.1.1) Analysis of IDR Sequences from DisProt Database:

Initially, all the sequences were extracted from the DisProt.txt file. The total number of sequences was 10,337. All the unique IDR sequences were filtered out and they were 4790 in number. From this set of 4790 sequences, only those sequences were taken for analysis whose length were greater than 10. After this filter, there were 4376 sequences in number. Now, there is something called Disordered Promoting Amino Acids(DPAAs) and Ordered Promoting Amino Acids(OPAAs). Studying DPAAs and OPAAs can provide important insights into the mechanisms and functions of IDPs, as well as aid in the prediction and characterization of IDRs.

DPAAs are a group of amino acids that are more frequently found in disordered regions than in structured regions, and include residues such as serine, glycine, and glutamine.

Conversely, OPAAs are amino acids that are more commonly found in structured regions and include residues such as cysteine, phenylalanine, and tyrosine.

Analyzing the DPAAs and OPAAs percentages in ordered and disordered regions of a protein sequence can reveal important information about the sequence-structure relationships and functional roles of IDRs. For example, high levels of DPAAs in certain regions of a protein sequence may indicate a propensity for disorder and flexibility, which can enable binding to multiple partners and promote signaling or regulatory functions. Conversely, high levels of OPAAs in certain regions may indicate a strong tendency for structure and stability, which can enable specific binding and catalytic functions. I tried analyzing the percentage of DPAAs and OPAAs and here is the snapshot of the results:



```
df
```

	Sequence	A	R	N	D	C	Q	E	G	H	...	Y	V	Total	DPAA	OPAA	NEUTRAL	DPAA%	DPAA1%	OPAA%	OPAA1%
0	(P, T, R, T, V, A, I, S, D, A, A, Q, L, P, H, ...	3	4	0	4	1	1	0	4	1	...	2	1	45	19	13	13	42.222	59.375	28.889	40.625
1	(S, S, S, S, E, S, T, G, T, P, S, N, P, D, L, ...	3	0	1	15	0	5	15	5	1	...	4	7	95	51	24	20	53.684	68.000	25.263	32.000
2	(M, A, S, N, D, Y, T, Q, Q, A, T, Q, S, Y, G, ...	13	32	24	24	4	51	9	150	2	...	35	5	507	360	95	52	71.006	79.121	18.738	20.879
3	(R, E, K, R, G, L, A, L, D, G, K, L, K, H, E, ...	2	3	1	2	0	0	3	3	1	...	0	0	28	16	7	5	57.143	69.565	25.000	30.435
4	(M, D, Q, Q, F, V, A, Q, L, E, Q, A, L, G, A, ...	5	0	0	2	0	8	1	2	0	...	1	2	39	22	10	7	56.410	68.750	25.641	31.250
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4371	(M, G, K, T, N, D, W, L, D, F, D, Q, L, A, E, ...	2	1	1	4	0	1	2	1	0	...	0	1	22	9	7	6	40.909	56.250	31.818	43.750
4372	(G, N, G, K, I, T, Q, D, E, L, S, K, V, V, D, ...	5	2	6	5	0	3	2	2	1	...	1	6	70	33	26	11	47.143	55.932	37.143	44.068
4373	(T, S, K, I, A, S, P, G, L, T, S, S, T, A, S, ...	3	0	1	0	0	0	0	1	0	...	0	1	21	13	4	4	61.905	76.471	19.048	23.529
4374	(M, G, S, M, K, K, I, L, S, M, I, P, G, F, G, ...	1	0	0	0	0	0	1	4	1	...	0	0	24	14	5	5	58.333	73.684	20.833	26.316
4375	(G, Q, I, S, Q, Y, S, N, D, V, P, Y)	0	0	1	1	0	2	0	1	0	...	2	1	12	6	5	1	50.000	54.545	41.667	45.455

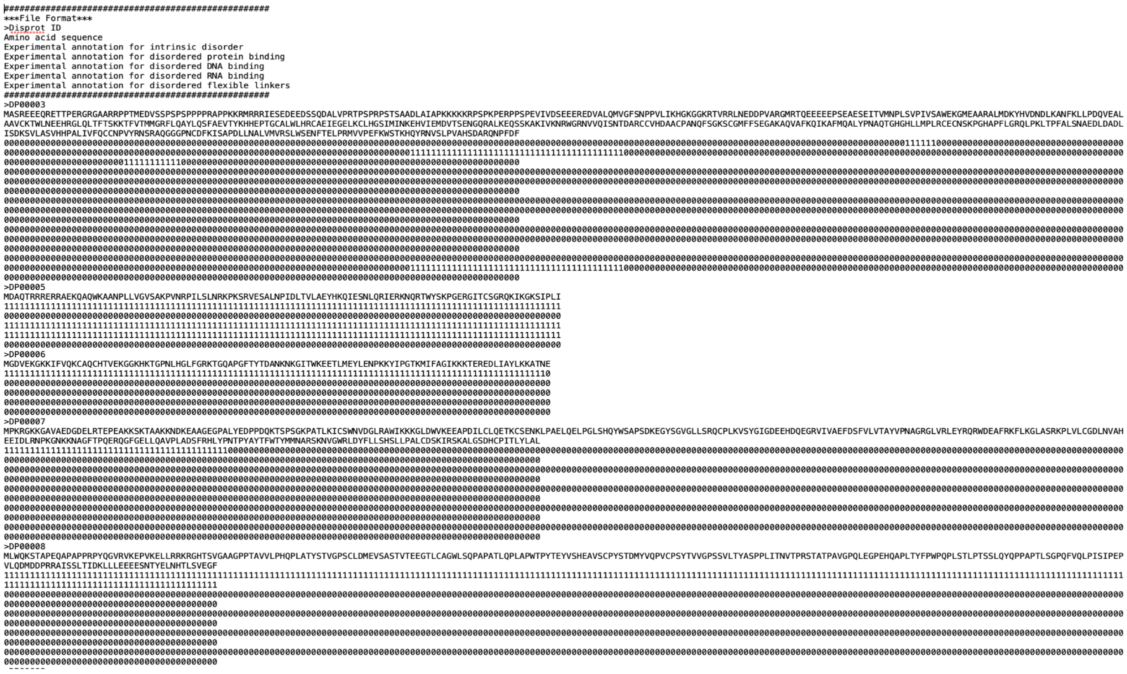
4376 rows × 29 columns

```
[ ] df.to_excel("IDRfile1.xlsx")
```

By incorporating DPAAs and OPAAs percentages as additional features in the prediction and classification of IDRs using machine learning models such as CNNs, one can potentially improve the accuracy and interpretability of the model, as well as gain new insights into the sequence-structure-function relationships of IDPs.

3.2) IDR sequences from FIDPNN Database:

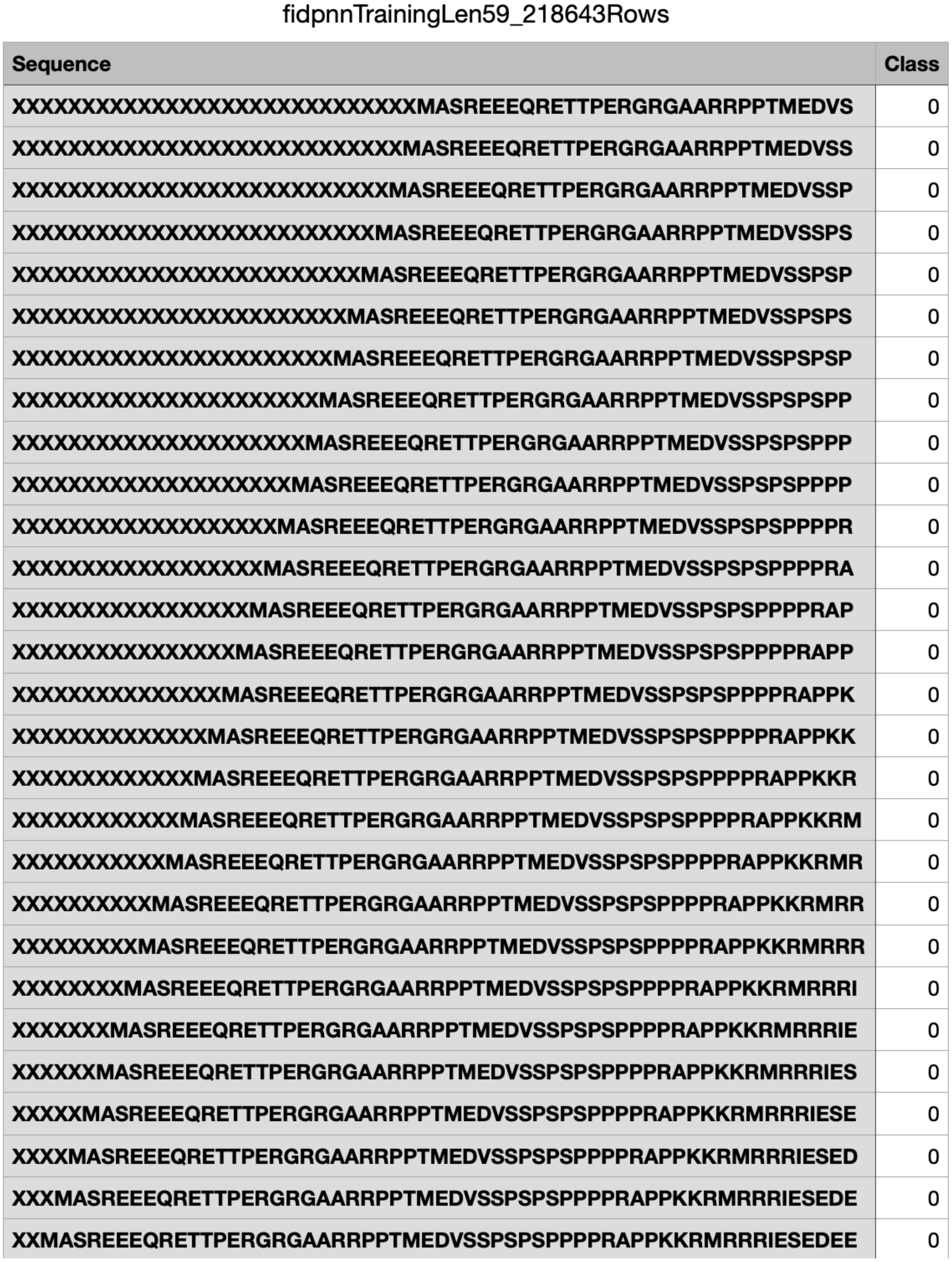
The fIDP-NN database is a curated dataset of intrinsically disordered regions (IDR) sequences, which have been collected from various sources such as the DisProt database, the Protein Data Bank (PDB) and other literature resources. The fIDP-NN database contains a diverse collection of IDR sequences, which have been classified into different categories based on their length, function and other characteristics. Below figures show the train dataset. Similarly, we also have validation and test datasets.



```
###################################################
***File Format***
>Disprot ID
Amino acid sequence
Experimental annotation for intrinsic disorder
Experimental annotation for disordered protein binding
Experimental annotation for disordered DNA binding
Experimental annotation for disordered RNA binding
Experimental annotation for disordered flexible linkers
###################################################
>DP00003
MASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIESEDEEDSSQDALVPRTPSPRPSTSAADLAIAPKKKKKRPSPKPERPPSPEVIVDSEEEREDVALQMVGFSNPPVLIKHGKGGKRTVRRLNEDDPVARGMRTQEEEEEPSEAESEITVMNPLSVPIVSAWEKGMEAARALMDKYHVDNDLKANFKLLPDQVEAL
AAVCKTWLNEEHRGLQLTFTSKKTFVTMMGRFLQAYLQSFAEVTYKHHEPTGCALWLHRCAEIEGELKCLHGSIMINKEHVIEMDVTSENGQRALKEQSSKAKIVKNRWGRNVVQISNTDARCCVHDAACPANQFSGKSCGMFFSEGAKAQVAFKQIKAFMQALYPNAQTGHGHLLMPLRCECNSKPGHAPFLGRQLPKLTPFALSNAEDLDADL
ISDKSVLASVHHPALIVFQCCNPVYRNSRAQGGGPNCDFKISAPDLLNALVMVRSLWSENFTELPRMVVPEFKWSTKHQYRNVSLPVAHSDARQNPFDF
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000001111111111000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000001111111111111111111111111111111111111111100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
>DP00005
MDAQTRRRERRAEKQAQWKAANPLLVGVSAKPVNRPILSLNRKPKSRVESALNPIDLTVLAEYHKQIESNLQRIERKNQRTWYSKPGERGITCSGRQKIKGKSIPLI
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
>DP00006
MGDVEKGKKIFVQKCAQCHTVEKGGKHKTGPNLHGLFGRKTGQAPGFTYTDANKNGITWKEETLMEYLENPKKYIPGTKMIFAGIKKKTEREDLIAYLKKATNE
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111110
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
>DP00007
MPKRGKKGAVAEDGDELRTEPEAKKSKTAAKKNDKEAAGEGPALYEDPPDQKTSPSGKPATLKICSWNVDGLRAWIKKKGLDWVKEEAPDILCLQETKCSENKLPAELQELPGLSHQYWSAPSDKEGYSGVGLLSRQCPLKVSYGIGDEEHDQEGRVIVAEFDSFVLVTAYVPNAGRGLVRLEYRQRWDEAFRKFLKGLASRKPLVLCGDLNVAH
EEIDLRNPKGNKKNAGFTPQERQGFGELLQAVPLADSFRHLYPNTPYAYTFWTYMMNARSKNVGWRLDYFLLSHSLLPALCDSKIRSKALGSDHCPITLYLAL
1111111111111111111111111111111111111111111000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
>DP00008
MLWQKSTAPEQAPAPPRPYQGVRVKEPVKELLRRKRGHTSVGAAGPPTAVVLPHQPLATYSTVGPSCLDMEVSASTVTEEGTLCAGWLSQPAPATLQPLAPWTPYTEYVSHEAVSCPYSTDMYVQPVCPSYTVVGPSSVLTYASPPLITNVTPRSTATPAVGPQLEGPEHQAPLTYFPWPQPLSTLPTSSLQYQPPAPTLSGPQFVQLPISIPEP
VLQDMDDPRRAISSLTIDKLLLEEEESNTYELNHTLSVEGF
11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
11111111111111111111111111111111111111111
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000
>DP00009
```

FIGURE : Train Dataset

From the above set of data, we extracted the sequences and their intrinsic order for all of the training, testing and validation data. For window-based analysis of IDR, we divided the entire sequence into separate window sizes of subsequences. A sample screenshot is shown below:



fidpnnTrainingLen59_218643Rows

Sequence	Class
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVS	0
XXXXXXXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSS	0
XXXXXXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSP	0
XXXXXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPS	0
XXXXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSP	0
XXXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPS	0
XXXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSP	0
XXXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPP	0
XXXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPP	0
XXXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPP	0
XXXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPR	0
XXXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRA	0
XXXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAP	0
XXXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPP	0
XXXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPK	0
XXXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKK	0
XXXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKR	0
XXXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRM	0
XXXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMR	0
XXXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRR	0
XXXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRR	0
XXXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRI	0
XXXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIE	0
XXXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIES	0
XXXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIESE	0
XXXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIESED	0
XXXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIESEDE	0
XXMASREEEQRETTPERGRGAARRPPTMEDVSSPSPSPPPPRAPPKKRMRRRIESEDEE	0

3.3) Feature Extraction From CNN:

This part is explained in the Algorithms chapter.

CHAPTER 4: ALGORITHMS USED

4.1) Convolution Neural Networks

With the help of a window-based strategy, convolutional neural networks (CNNs) have been effectively used to predict intrinsically disordered regions (IDRs) within intrinsically disordered proteins (IDPs). Among deep learning algorithms, CNNs are capable of accurately capturing intricate correlations between information and predicting the outcome.

Convolutional, pooling, and fully connected layers are only a few of the many layers that make up CNN's architecture. The input to the CNN is a sequence of amino acids for predicting IDRs inside IDPs, and the output is a binary prediction of whether or not each amino acid belongs to a disordered area.

Convolutional layers perform feature extraction by sliding a filter or kernel across the input sequence, extracting local features such as amino acid composition and physicochemical properties. Pooling layers reduce the dimensionality of the extracted features by summarizing the output of the convolutional layer in a small neighborhood. Finally, fully connected layers take the output of the previous layers and produce the final prediction.

The use of CNNs for predicting IDRs within IDPs has shown promising results, with some studies reporting higher accuracy compared to traditional machine learning methods. The effectiveness of CNNs can be attributed to their ability to learn complex patterns within the input sequence, including local variations in disorder propensity and longer-range interactions between residues.

One of the main advantages of using a CNN architecture is that it can effectively capture local features within the amino acid sequence, such as the distribution of amino acids, sequence motifs, and physicochemical properties. Convolutional layers perform a series of convolutions on the input sequence using a set of filters or kernels, which detect specific features at different locations in the sequence. The output of the convolutional layer is a set of feature maps, which represent the presence or absence of the detected features.

Pooling layers are used to reduce the dimensionality of the feature maps by summarizing the output of the convolutional layer in a small neighborhood. Max pooling is a commonly used pooling operation, which selects the maximum value within a sliding window. Other types of pooling operations include average pooling and global pooling, which summarize the entire feature map.

One or more fully connected layers receive the output from the convolutional and pooling layers, which results in the final prediction of disorder or order for each amino acid in the input sequence. The completely connected layer's output is often subjected to a sigmoid activation function in the final layer, which generates a probability score between 0 and 1 for each amino acid that indicates the possibility of disorder.

For each amino acid in the input sequence, the CNN model must be trained by minimizing a loss function that evaluates the discrepancy between the predicted probability scores and the actual labels of disorder or order. For binary classification tasks like forecasting disorder or order, cross-entropy loss is a frequently employed loss function.

To prevent overfitting of the CNN model, various regularization techniques can be applied, such as dropout, weight decay, and early stopping. Dropout randomly drops out a fraction of neurons during training, which prevents the model from relying too heavily on a small set of features. Weight decay penalizes large weights in the model, which encourages simpler models that generalize better to new data. Early stopping stops training when the performance on a validation set starts to decrease, which prevents the model from overfitting to the training set.

However, the performance of CNNs for predicting IDRs within IDPs can be affected by various factors, such as the choice of window size, the overlap between windows, and the selection of features. Therefore, careful consideration must be given to these factors when designing a CNN architecture for predicting IDRs within IDPs.

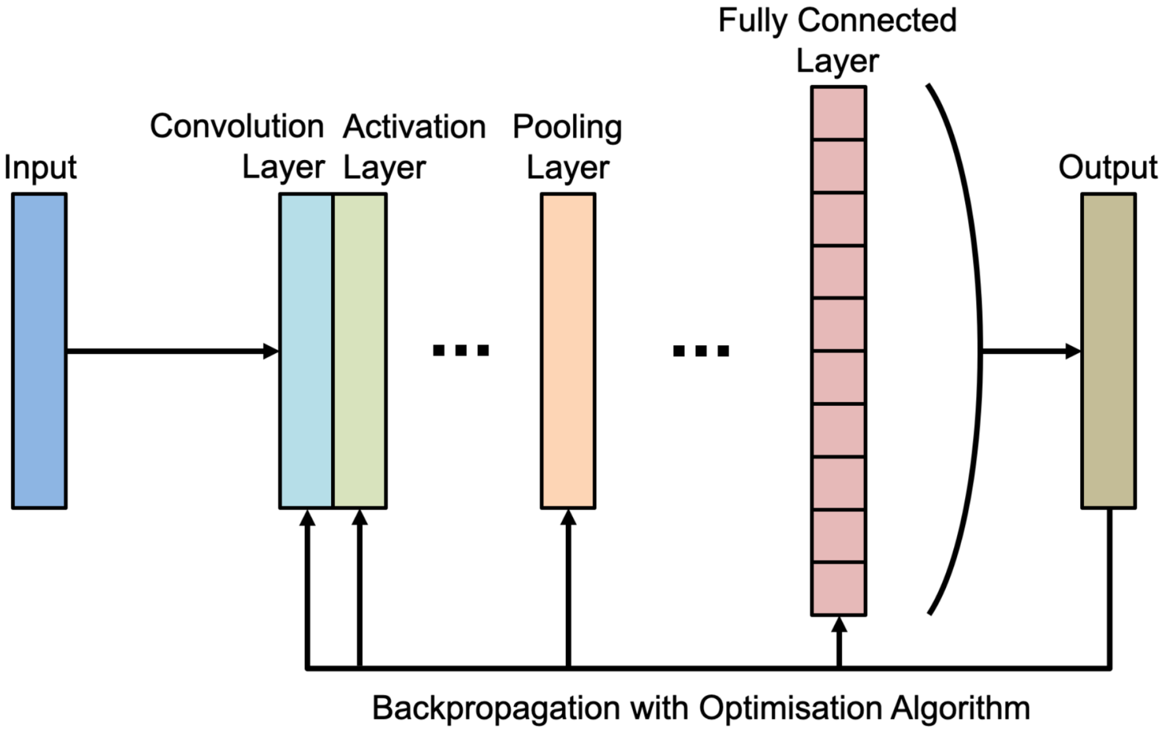


Figure 1. Basic structure of a typical Convolutional Neural Network (CNN)

Reference - https://www.mdpi.com/2076-3417/11/2/744

4.2) CNN and its role in analyzing IDRs

The window-based approach used in CNN architecture is particularly well-suited for analyzing IDPs because it allows the model to capture the local context of each amino acid in the sequence. The size of the window is typically optimized based on the length of the IDP and the desired level of resolution in the prediction. For example, a larger window size may be used for longer IDPs or for regions with more complex disorder patterns, while a smaller window size may be used for shorter IDPs or for regions with more localized disorder.

The overlap between windows is another important parameter to consider in CNN architecture. Overlapping windows can provide more continuous coverage of the amino acid sequence and capture complex patterns of disorder. However, they can also increase the redundancy of information and introduce biases into the model. Non-overlapping windows may provide a more independent representation of the amino acid sequence but may miss important local features.

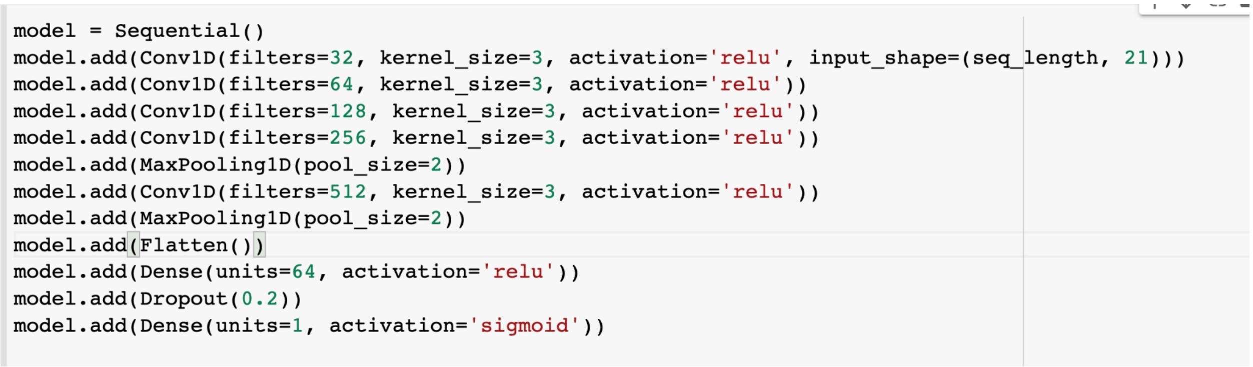
Another advantage of the CNN architecture is its ability to incorporate multiple sources of information into the model. For example, the amino acid composition, physicochemical properties, predicted secondary structure, and evolutionary conservation of each window can all be used as input features to the model. The choice of input features can greatly affect the accuracy of the predictions and should be carefully selected based on the specific problem being addressed.

Other machine learning techniques, such as recurrent neural networks (RNNs), support vector machines (SVMs), and random forests, have also been used to the challenge of forecasting IDRs within IDPs in addition to the CNN design. Each algorithm has advantages and disadvantages of its own, and the selection of an algorithm depends on the particular issue being solved and the data at hand.

The training process is also crucial for achieving accurate predictions. The choice of loss function, optimization algorithm, and regularization methods can greatly affect the performance of the model. To assess the effectiveness of the CNN and to spot probable sources of inaccuracy, cross-validation and independent testing can be performed.

Overall, the CNN architecture for predicting IDRs within IDPs is a powerful tool that can capture local features within the amino acid sequence and effectively distinguish IDRs from structured regions. The optimization of window size, overlap, and input features is critical to achieving accurate predictions, and various regularization techniques can be applied to prevent overfitting of the model.

4.3) Flowchart of CNN used for analyzing the IDR dataset



```
model = Sequential()
model.add(Conv1D(filters=32, kernel_size=3, activation='relu', input_shape=(seq_length, 21)))
model.add(Conv1D(filters=64, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=128, kernel_size=3, activation='relu'))
model.add(Conv1D(filters=256, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Conv1D(filters=512, kernel_size=3, activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Flatten())
model.add(Dense(units=64, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(units=1, activation='sigmoid'))
```

Figure 2. CNN Model Architecture code

The above code goes through the following steps:

-The input is a 21 x 21(let's suppose window size chosen is 21) image with 21 channels (possibly RGB values).

-The first convolution layer applies 32 filters of size 3 x 3 x 21 with a rectified linear unit (ReLU) activation function to the input image. This produces 32 feature maps of size 19 x 19 x 1.

-The second convolution layer applies 64 filters of size 3 x 3 x 1 with a ReLU activation function to each feature map from the previous layer. This produces 64 feature maps of size 17 x 17 x 1.

-The third convolution layer applies 128 filters of size 3 x 3 x 1 with a ReLU activation function to each feature map from the previous layer. This produces 128 feature maps of size 15 x 15 x 1.

-The fourth convolution layer applies 256 filters of size 3 x 3 x 1 with a ReLU activation function to each feature map from the previous layer. This produces 256 feature maps of size 13 x 13 x 1.

-The max pooling layer reduces the spatial dimensions of each feature map by applying a max operation over a window of size 2 x 2 with a stride of 2. This produces 256 feature maps of size 6 x 6 x 1.

-The fifth convolution layer applies 512 filters of size 3 x 3 x 1 with a ReLU activation function to each feature map from the previous layer. This produces 512 feature maps of size 4 x 4 x 1.

-The max pooling layer reduces the spatial dimensions of each feature map by applying a max operation over a window of size 2 x 2 with a stride of 2. This produces 512 feature maps of size 2 x 2 x 1.

-The flatten layer reshapes each feature map into a one-dimensional vector and concatenates them into a single vector of size (512 * (2 * (2 * (1)))) = (2048).

-The dense layer applies a linear transformation with a ReLU activation function to the flattened vector and produces an output vector of size (64).

-The dropout layer randomly sets some elements of the output vector to zero with a probability of (0.2) to prevent overfitting.

-The dense layer applies a linear transformation with a sigmoid activation function to the output vector and produces an output scalar of size (1), which represents the probability of belonging to a certain class.

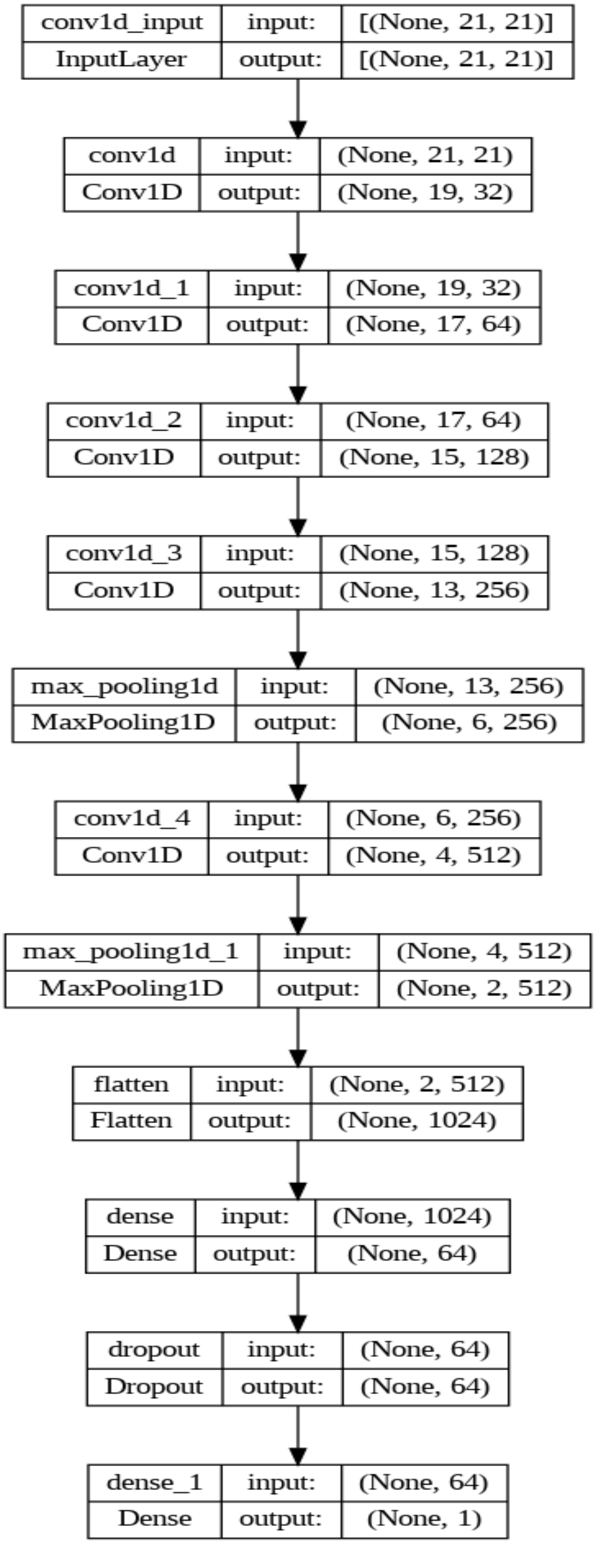

conv1d_input | input: | [(None, 21, 21)]
InputLayer | output: | [(None, 21, 21)]
conv1d | input: | (None, 21, 21)
Conv1D | output: | (None, 19, 32)
conv1d_1 | input: | (None, 19, 32)
Conv1D | output: | (None, 17, 64)
conv1d_2 | input: | (None, 17, 64)
Conv1D | output: | (None, 15, 128)
conv1d_3 | input: | (None, 15, 128)
Conv1D | output: | (None, 13, 256)
max_pooling1d | input: | (None, 13, 256)
MaxPooling1D | output: | (None, 6, 256)
conv1d_4 | input: | (None, 6, 256)
Conv1D | output: | (None, 4, 512)
max_pooling1d_1 | input: | (None, 4, 512)
MaxPooling1D | output: | (None, 2, 512)
flatten | input: | (None, 2, 512)
Flatten | output: | (None, 1024)
dense | input: | (None, 1024)
Dense | output: | (None, 64)
dropout | input: | (None, 64)
Dropout | output: | (None, 64)
dense_1 | input: | (None, 64)
Dense | output: | (None, 1)

Figure 3. Flow Chart of CNN architecture used

CHAPTER 5: METHODOLOGY

5.1) Flowchart

The methods used in my work are summarized in the graphic shown below.

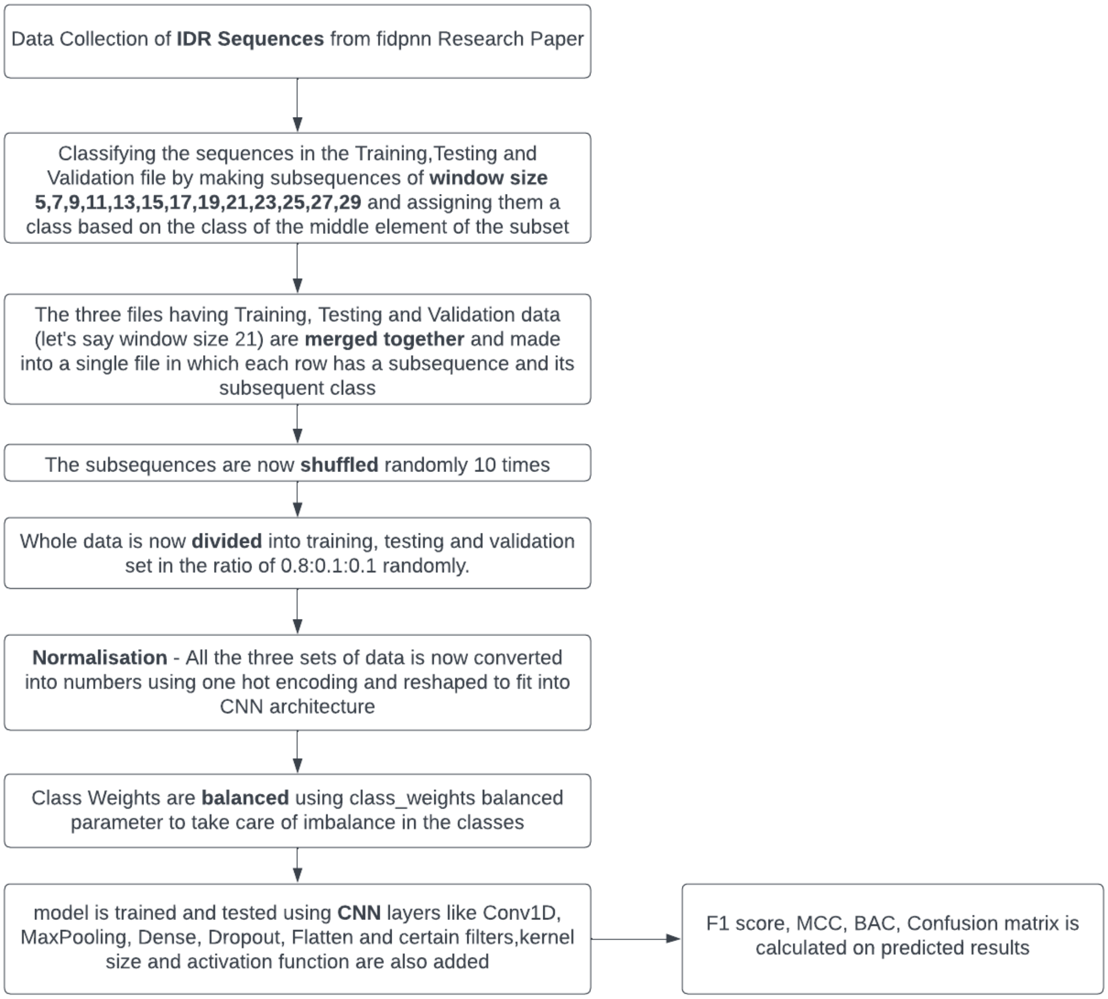

Data Collection of **IDR Sequences** from fidpnn Research Paper
Classifying the sequences in the Training,Testing and Validation file by making subsequences of **window size 5,7,9,11,13,15,17,19,21,23,25,27,29** and assigning them a class based on the class of the middle element of the subset
The three files having Training, Testing and Validation data (let's say window size 21) are **merged together** and made into a single file in which each row has a subsequence and its subsequent class
The subsequences are now **shuffled** randomly 10 times
Whole data is now **divided** into training, testing and validation set in the ratio of 0.8:0.1:0.1 randomly.
Normalisation - All the three sets of data is now converted into numbers using one hot encoding and reshaped to fit into CNN architecture
Class Weights are **balanced** using class_weights balanced parameter to take care of imbalance in the classes
model is trained and tested using **CNN** layers like Conv1D, MaxPooling, Dense, Dropout, Flatten and certain filters,kernel size and activation function are also added
F1 score, MCC, BAC, Confusion matrix is calculated on predicted results

Figure 4. Flowchart

5.1.1) Data Collection and Preparation of data for Window-Based study

One approach to collecting a dataset for predicting IDRs within IDPs is to use the

DisProt database, which is a curated database of experimentally characterized IDPs and their interactions. DisProt contains a large collection of IDPs with varying degrees of disorder, length, and function. One can select a subset of DisProt sequences that meet certain criteria, such as a minimum length or a maximum degree of disorder, to ensure a representative and high-quality dataset.

FIDPNN is a widely used dataset for predicting IDRs in IDPs. It contains protein sequences from the DisProt database that are annotated with experimentally verified disorder regions. The dataset is split into training, validation, and test sets, with each set containing roughly one-third of the total sequences.

To create windows of various sizes for use in training and testing a CNN model, the protein sequences in the FIDPNN dataset can be processed as follows:

- ➔ Retrieve the protein sequences and their corresponding disorder annotations from the FIDPNN dataset.
- ➔ For each sequence, slide a window of a specified size along the sequence with a step size of 1 residue. For example, a window size of 15 residues would result in a window being placed on the sequence at positions 0-14, 1-15, 2-16, 3-17, and so on. The sequences are padded with character “X” to make sure all the residues are covered.
- ➔ Assign a class label to each window based on the class of the central residue.
- ➔ Repeat steps 1-3 for each window size of interest, such as 5, 7, 9, 11, 15, 21,

23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57 and 59 residues.

- ➔ So, now we have 3 files for each window named as fidpnnTrainingLen_(no. of rows)Rows, fidpnnTestLen_(no. of rows)Rows, fidpnnValidationLen_(no. of rows)Rows.
- ➔ Concatenate these 3 files and shuffle them randomly 10 times.
- ➔ dividing the data into three equal portions for training, validation, and testing in the ratio of 0.8:0.1:0.1. The dataset is now saved as X_train_val, X_test, y_train_val, y_test, X_train, X_val, y_train, and y_val, where variables beginning with "X" stand for sequences and variables beginning with "y" stand for their respective class as 0/1.
- ➔ Normalize the data using one hot encoding.
- ➔ Reshape the data so that it fits into the CNN architecture.
- ➔ Calculate the class weights to account for imbalanced data.

One-Hot Encoding:

One-hot encoding is a popular method for transforming numerical data that may be input into a machine learning model from categorical data, such as amino acid residues. Each category is represented by a binary vector of length N in one-hot encoding, where N is the total number of categories. The vector is zero at all other indices and has a value of one at the index that corresponds to the category. For instance, the amino acid residue "A" can be represented as [1, 0, 0, 0,..., 0], where "A" is represented by the first index and the remaining amino acids are represented by the other indices. One-hot encoding ensures that the categorical data is treated as numerical data in the model, and that the model can learn the relationships and dependencies between the categories.

In the context of predicting IDRs within IDPs using CNN models, one-hot encoding can be applied to the amino acid sequence within each window. Each amino acid residue is represented as a binary vector of length 20 (for the 20 standard amino acids), and the window is represented as a matrix of size (window size x 20). This matrix can then be fed into the CNN model as input features, along with other features such as physicochemical properties or predicted secondary structure. The matrix can also be normalized using standardization or min-max scaling, depending on the requirements of the model.

5.1.2) Feature Extraction and Training the Model

One approach to feature extraction using a CNN is to use a separate set of convolutional layers to extract features from the sequence, followed by a pooling layer and a set of fully connected layers. The output of the fully connected layers can then be combined with the input sequence and fed into another set of convolutional layers to predict the probability of an IDR.

The convolutional layers in the feature extraction module can use filters of varying sizes to capture different levels of local and global features in the sequence. For example, a filter of size 3 can capture adjacent residues, while a filter of size 7 can capture longer-range interactions between residues. The pooling layer can then downsample the output of the convolutional layers, reducing the dimensionality of the feature space and making it more computationally tractable.

After the feature extraction module, the output can be combined with the input sequence using concatenation or another merging technique. The combined features can then be fed into another set of convolutional layers to predict the probability of an IDR. A probability score can be generated for each residue in the sequence using a sigmoid activation function applied to the output of the last convolutional layer.

The fundamental units of CNNs are convolutional layers, which are employed to extract regional information from input data. Multiple filters that are frequently applied to the input data in the form of sliding windows make up a convolutional layer. Each filter pulls out from the input data a specific pattern or characteristic, like edges, corners, or blobs. A series of feature maps representing various filters and collecting various aspects of the input data are the result of a convolutional layer.

The feature maps are downsampled and made smaller while still retaining the most crucial features using pooling layers. Max pooling is a popular form of pooling in which the maximum value available in each pooling window is chosen and the remaining values are discarded. Max pooling aids in making the model more resilient to minute changes and variances in the input data.

The output of the convolutional and pooling layers is transformed into a final prediction using fully connected layers. A fully connected layer applies a nonlinear activation function to the output and connects every neuron in the layer to every neuron in the layer below. The number of classes being predicted, such as IDRs and structured regions, affects how many neurons are present in the top layer.

Feature extraction using CNN provides a powerful and flexible approach to learning and representing complex and high-dimensional input data, such as amino acid sequences. By using multiple convolutional, pooling, and fully connected layers, CNNs can extract hierarchical and abstract features from the input data and make accurate predictions on new and unseen data.

This model is then used to specify the model's optimizer, loss function, and evaluation metrics using compile() method. The algorithm used to update the weights during training, such as stochastic gradient descent or Adam, is chosen by the optimizer. When binary cross-entropy is predicted, the loss function calculates the difference between the predicted and actual labels. The model's performance while training is tracked using assessment metrics like accuracy or F1 score.

The model is trained using a different function, **model.fit()**, on the training dataset using a predetermined number of epochs and batch size. A batch is a subset of the training dataset used to update the weights, whereas an epoch is one run through the whole training dataset. The model is trained on all batches in a random order during each epoch. In order to keep track of the model's performance on untrained data and avoid overfitting, the model.fit() function also permits the use of validation datasets.

Once the windows and input features are prepared, the CNN model is trained and tested using the standard steps described above. The training, validation and testing sets are created by randomly splitting the dataset in the ratio of 0.8:0.1:0.1. Cross-validation is used to further evaluate the performance and robustness of the model.

5.1.3) Model prediction

After the CNN model is trained, it can be used to predict IDRs within IDPs on a new dataset or unseen sequences. For each window of the input sequence, the model outputs a probability or binary score indicating the likelihood of being an IDR. A threshold is chosen to convert the probability score into a binary classification, such as 0.5, depending on the desired trade-off between sensitivity and specificity.

5.1.4) Performance metrics

To evaluate the accuracy and robustness of the model, various performance metrics can be calculated using the predicted labels and the true labels. The details of these matrices will be discussed in the next chapter.

CHAPTER 6: RESULTS AND CONCLUSION

Various Performance metrics are used to come to a proper conclusion. They are as follows:

1) Accuracy: This is the ratio of the number of correct predictions to the total number of predictions made by the model, expressed as a percentage. Accuracy is a simple and intuitive metric, but it can be misleading in cases where the dataset is imbalanced or the classes are not equally important.

2) F1 score: Calculated as 2 * (precision * recall) / (precision + recall), this is the harmonic mean of precision and recall. The F1 score, which balances both precision and recall measurements, is frequently used as the main indicator for assessing classifier performance. The precision measures the proportion of accurate positive predictions to all of the model's positive predictions. Precision measures how well the model avoids false positives, or instances where it predicts a region to be disordered when it is actually ordered. Recall is the proportion of correctly predicted positive cases to all of the actual positive instances in the dataset. Recall measures how well the model identifies all positive instances, both true positives and false negatives.

3) Matthews correlation coefficient (MCC): This ranges from -1 to 1, and it represents the correlation between the true and anticipated binary classifications. MCC, which considers both true and false positives and negatives, is typically seen as a more trustworthy indicator of classifier performance than accuracy alone.

4) Balanced accuracy (BAC): This is the average of sensitivity (recall) and specificity (the ratio of true negatives to the total number of actual negatives in the dataset). BAC takes into account imbalanced datasets and is a useful metric for evaluating classifier performance when the classes are not equally important.

5) Confusion Matrix: It is a table that compares the predicted labels to the actual labels for a particular dataset to summarize the classification outcomes of a machine learning model. It is a helpful tool for assessing a model's performance and comprehending the kinds of faults it commits. In most cases, the confusion matrix is set up as a square matrix with two rows and two columns, where the rows represent the actual class labels and the columns represent the anticipated class labels. The number of true positives (TP), false positives (FP), false negatives (FN), and true negatives (TN) is shown by the four cells in the matrix.

- ➔ True positives (TP): Instances that are actually positive (disordered) and are correctly predicted as positive by the model.
- ➔ False positives (FP): Instances that are actually negative (ordered) but are incorrectly predicted as positive (disordered) by the model.
- ➔ False negatives (FN): Instances that are actually positive (disordered) but are incorrectly predicted as negative (ordered) by the model.
- ➔ True negatives (TN): Instances that are actually negative (ordered) and are

correctly predicted as negative by the model.

6) Sensitivity: It represents the percentage of positive instances (disordered regions, in the case of IDR prediction) that are correctly detected by the model and is sometimes referred to as recall or true positive rate. It is calculated and expressed as a percentage as the ratio of accurate positive predictions to all of the dataset's actual positive cases. Because it measures how well the model is able to identify all positive cases in the dataset, regardless of whether they are accurately classified as positive or negative, sensitivity is a crucial parameter in many classification tasks, including IDR prediction. High sensitivity indicates that the model is able to accurately identify a large proportion of the positive instances, while low sensitivity suggests that the model may be missing some positive instances or misclassifying them as negative. The confusion matrix, which lists the number of true positive, false positive, true negative, and false negative predictions made by the model, can be used to calculate sensitivity in the context of IDR prediction using a CNN model. Sensitivity can be calculated as TP / (TP + FN), where TP is the total number of accurate predictions and FN is the total number of inaccurate guesses.

These performance metrics can be computed using the predicted and true labels generated by the trained CNN model, and are often used to compare and optimize different models and hyperparameters for IDR prediction.

The results obtained after training the model are as follows:-

	Accuracy	**Sensitivity**	**F1 Score**	**MCC**	**BAC**	**TP**	**FP**	**FN**	**TN**
Sequences with Window **Length 59**	0.989122	0.984574	0.97833	0.97111	0.98761	23375	221	121	7723
Sequences with Window **Length 57**	0.985781	0.973951	0.97097	0.96157	0.98177	23511	247	200	7478
Sequences with Window **Length 55**	0.988005	0.971905	0.97572	0.96777	0.98261	23478	158	219	7576
Sequences	0.986507	0.961761	0.97249	0.96366	0.97821	23507	126	298	7495

with Window **Length 53**									
Sequences with Window **Length 51**	0.985391	0.959278	0.97009	0.96053	0.97661	23517	143	316	7444
Sequences with Window **Length 49**	0.988157	0.971287	0.97572	0.96791	0.98246	23565	151	221	7476
Sequences with Window **Length 47**	0.985703	0.969065	0.97123	0.96173	0.98014	23376	207	242	7581
Sequences with	0.988088	0.980533	0.97553	0.96768	0.98551	23569	226	148	7455

Window **Length 45**									
Sequences with Window **Length 43**	0.985568	0.969984	0.97092	0.96132	0.98035	23375	219	234	7562
Sequences with Window **Length 41**	0.987541	0.977781	0.97482	0.96655	0.98425	23422	219	172	7569
Sequences with Window **Length 39**	0.987217	0.968705	0.97372	0.96529	0.98095	23542	161	240	7429
Sequences with Window **Length 37**	0.987883	0.969342	0.97517	0.96719	0.98162	23519	144	236	7462

Sequences with Window **Length 35**	0.986602	0.972232	0.97273	0.96385	0.98175	23436	206	214	7493
Sequences with Window **Length 33**	0.984651	0.980287	0.96974	0.95956	0.98319	23147	326	155	7708
Sequences with Window **Length 31**	0.985601	0.967859	0.97046	0.96094	0.97959	23461	205	246	7408
Sequences with Window **Length 29**	0.987971	0.971794	0.97555	0.96758	0.98253	23195	157	216	7442
Sequences with Window	0.984561	0.969293	0.96911	0.95881	0.97947	23225	243	240	7576

Length 27									
Sequences with Window **Length 25**	0.987461	0.970158	0.97456	0.96626	0.98165	23360	161	231	7510
Sequences with Window **Length 23**	0.981306	0.968876	0.96223	0.94985	0.97712	23216	345	239	7440
Sequences with Window **Length 21**	0.973314	0.940559	0.94576	0.92809	0.96231	23119	374	459	7263
Sequences with Window **Length 19**	0.966876	0.947871	0.93431	0.91233	0.96232	22807	629	404	7346
Sequences	0.976212	0.946241	0.95161	0.93587	0.96615	23123	327	414	7287

with Window **Length 17**									
Sequences with Window **Length 15**	0.958791	0.921761	0.91622	0.88892	0.94626	22818	687	595	7010
Sequences with Window **Length 13**	0.937443	0.859871	0.87204	0.83081	0.91144	22496	864	1079	6621
Sequences with Window **Length 11**	0.846652	0.642502	0.67425	0.57549	0.77806	21323	2016	2737	4919
Sequences with Window **Length 9**	0.810853	0.463244	0.54566	0.44208	0.69351	21547	1778	4067	3510

Sequences with Window **Length 7**	0.764122	0.149611	0.23815	0.20311	0.55733	22350	809	6440	1133
Sequences with Window **Length 5**	0.761452	0.036493	0.06785	0.07772	0.51213	21329	264	6495	246

Analyzing the results of different window sizes for IDR prediction can provide valuable insights into the performance of the model and the impact of window size on accuracy and other performance metrics. For example, if we train a CNN model for IDR prediction using different window sizes of amino acids, and evaluate the performance using metrics such as accuracy, F1 score, MCC, BAC, and confusion matrix, we may observe different results for each window size.

Our results show that, in general, larger window sizes result in higher performance metric scores, with windows of size 15 and above providing the best overall performance. A larger window size can capture longer-range correlations and patterns in the sequence, which can be important in predicting intrinsically disordered protein (IDP) regions. This is because IDR regions can exhibit a range of disorder levels, from partially disordered to fully disordered, and can contain patterns or motifs that span multiple amino acids. A larger window size allows for capturing these longer-range patterns, which can improve the prediction of IDRs.

Additionally, a larger window size can also help reduce noise in the data by smoothing out fluctuations and capturing broader trends in the sequence. This can lead to more accurate predictions, especially for regions with lower levels of sequence variability. A window with a greater size can capture longer-range correlations and patterns within the sequence, which can be especially important in the context of intrinsically disordered protein regions. It can be difficult to determine the location and degree of disorder using narrower windows since intrinsically disordered protein regions can have a high degree of diversity and complexity.

Larger window sizes also provide more context for the model to work with, which can be especially important when considering the relationship between adjacent regions. For example, if a particular region is known to be disordered, adjacent regions may also be more likely to be disordered due to the local sequence characteristics. A larger window size can capture these types of patterns and improve the overall prediction accuracy.

Overall, while a larger window size may require more computational resources and training data, it can lead to better performance metrics for predicting intrinsically disordered protein regions. However, there are also potential drawbacks to using larger window sizes. For example, larger window sizes can lead to a loss of resolution and make it more difficult to identify smaller, more localized patterns in the sequence. They can also be more computationally expensive, which can limit the scalability of the model.

CHAPTER 7: FUTURE WORK

Based on the findings of investigating the impact of window size on the prediction of inherently disordered protein regions using deep learning, there are a number of interesting directions for future research:

1) Exploring different deep learning architectures: While CNN is a popular and effective architecture for predicting disordered regions, there are other architectures such as recurrent neural networks (RNNs) and transformers that may be better suited for capturing long-range dependencies and temporal dynamics in the protein sequence. Future studies can explore the use of these architectures and compare their performance with CNN.

2) Integrating additional features: Deep learning models can be augmented with additional features such as physicochemical properties, evolutionary conservation, and functional annotations. These characteristics might help the model perform better by adding more context and knowledge about the protein sequence. Future research can examine how these features are combined and how that affects the model's performance.

3) Validating the model on experimental data: While using the disprot dataset offers a reasonable benchmark for measuring the model's performance, it is crucial to validate the model with experimental data to determine its applicability. Future studies can validate the model on additional experimental datasets and compare its performance with existing methods.

4) Extending the analysis to other protein functions: In addition to predicting disordered regions, window-based approaches can be applied to other protein functions such as predicting protein-protein interactions or identifying functional sites. Comparing the performance of different window sizes for these tasks may provide insights into the optimal window size for predicting different protein functions.

Overall, the results of this study provide a starting point for investigating the optimal window size for predicting intrinsically disordered protein regions. Future work can build on these results to develop more accurate and effective methods for predicting disordered regions and other protein functions.

References

1) Uversky, V. N. (2013). Intrinsically disordered proteins and their “mysterious” (meta)physics. Frontiers in Physiology, 4, 1-17.
2) Dunker, A. K., Lawson, J. D., Brown, C. J., Williams, R. M., Romero, P., Oh, J. S., ... & Oldfield, C. J. (2001). Intrinsically disordered protein. Journal of Molecular Graphics and Modelling, 19(1), 26-59.
3) Ishida, T., & Kinoshita, K. (2007). PrDOS: prediction of disordered protein regions from amino acid sequence. Nucleic acids research, 35(suppl_2), W460-W464.
4) Yan, J., Friedrich, N. O., Filippi, S., & Jost, J. (2016). Deep learning for inferring protein-protein interactions from primary sequence. Bioinformatics, 32(12), i85-i93.
5) Xie, J., & Kurgan, L. (2018). Prediction of intrinsically disordered protein regions from sequence alone. BMC Bioinformatics, 19(1), 1-14.
6) Habchi, J., Tompa, P., Longhi, S., & Uversky, V. N. (2019). Introducing protein intrinsic disorder. Chemical Reviews, 119(18), 11594-11606.
7) Longhi, S., & Dunker, A. K. (2012). Using SPOT-Disorder to predict disordered regions with blind modifications. Intrinsically Disordered Proteins, 1(1), e24360.
8) Goh, G. B., Hodas, N. O., & Vishnu, A. (2017). Deep learning for computational chemistry. Journal of Computational Chemistry, 38(16), 1291-1307.
9) Yang, Y., Zhou, Y., & Gong, W. (2020). A deep learning approach for predicting intrinsically disordered protein regions from primary sequences. BMC Bioinformatics, 21(1), 1-13.
10) Zhou, Y., & Zhou, H. (2019). Methods for predicting intrinsically disordered regions of proteins. Methods in molecular biology (Clifton, N.J.), 2022, 163-180. https://doi.org/10.1007/978-1-4939-9593-4_9
11) Singh, R. K., Kumar, M., Mittal, A., & Mehta, P. K. (2020). Predicting intrinsic disorder in proteins: An overview of current methods and databases. Proteins, 88(11), 1209-1222. https://doi.org/10.1002/prot.25945
12) Uversky, V. N. (2017). Intrinsically disordered proteins in overcrowded milieu: Membrane-less organelles, phase separation, and intrinsic disorder. Current opinion in structural biology, 44, 18-30. https://doi.org/10.1016/j.sbi.2016.10.017
13) Gsponer, J., & Babu, M. M. (2012). The rules of disorder or why disorder rules. Progress in biophysics and molecular biology, 108(1-2), 1-3. https://doi.org/10.1016/j.pbiomolbio.2012.04.003
14) Dunker AK, Obradovic Z. The protein trinity--linking function and disorder. Nat Biotechnol. 2001;19(9):805-806.
15) Dosztányi, Z., Csizmok, V., Tompa, P., & Simon, I. (2005). The pairwise energy content estimated from amino acid composition discriminates between folded and intrinsically unstructured proteins. Journal of molecular biology, 347(4), 827-839. https://doi.org/10.1016/j.jmb.2005.01.071
16) Ward JJ, Sodhi JS, McGuffin LJ, Buxton BF, Jones DT. Prediction and functional analysis of native disorder in proteins from the three kingdoms of life. J Mol Biol. 2004;337(3):635-645.
17) Xie H, Vucetic S, Iakoucheva LM, Oldfield CJ, Dunker AK, Obradovic Z, Uversky VN. Functional anthology of intrinsic disorder. 1. Biological processes and functions of proteins with long disordered regions. J Proteome Res. 2007;6(5):1882-1898.
18) Uversky VN, Oldfield CJ, Dunker AK. Intrinsically disordered proteins in human diseases: introducing the D2 concept. Annu Rev Biophys. 2008;37:215-246.

19) Dyson HJ, Wright PE. Intrinsically unstructured proteins and their functions. Nat Rev Mol Cell Biol. 2005;6(3):197-208.
20) He B, Wang K, Liu Y, Xue B, Uversky VN, Dunker AK. Predicting intrinsic disorder in proteins: an overview. Cell Res. 2009;19(8):929-949.
21) Peng ZL, Kurgan L. Comprehensive comparative assessment of in-silico predictors of disordered regions. Curr Protein Pept Sci. 2012;13(1):6-18.
22) Huang Y, Niu B, Gao Y, Fu L, Li W. CD-HIT Suite: a web server for clustering and comparing biological sequences. Bioinformatics. 2010;26(5):680-682.
23) Monastyrskyy B, D'Andrea D, Fidelis K, Tramontano A, Kryshtafovych A. Evaluation of disorder predictions in CASP12. Proteins. 2018;86 Suppl 1:129-144.
24) Prusiner, S. B. (2012). Cell biology. A unifying role for prions in neurodegenerative diseases. Science, 336(6088), 1511-1513. https://doi.org/10.1126/science.1222951
25) Xue, B., Williams, R. W., & Oldfield, C. J. (2018). Intrinsically disordered proteins in human diseases: Introducing the D2 concept. Annual review of biophysics, 47, 1-18. https://doi.org/10.1146/annurev-biophys-070816-033759
26) Chiti, F., & Dobson, C. M. (2017). Protein misfolding, functional amyloid, and human disease. Annual review of biochemistry, 86, 27-68. https://doi.org/10.1146/annurev-biochem-061516-045115
27) Keras documentation: https://keras.io/ - This is the official documentation for the Keras deep learning library, which provides a high-level API for building and training neural networks in Python.
28) TensorFlow documentation: https://www.tensorflow.org/ - This is the official documentation for the TensorFlow library, which is the backend used by Keras for executing computations on CPUs and GPUs.
29) PyFlowchart documentation: https://pyflowchart.readthedocs.io/ - This is the official documentation for the PyFlowchart library, which provides tools for drawing flowcharts and other diagrams in Python.
30) Géron, A. (2017). Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media, Inc. - This book provides a comprehensive introduction to machine learning with scikit-learn and TensorFlow, including chapters on convolutional neural networks and other deep learning topics.
31) Chollet, F. (2018). Deep Learning with Python. Manning Publications. - This book is written by the creator of Keras and provides a practical introduction to deep learning with Python, including many examples of building and training neural networks.
32) LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444. - This is a review paper on deep learning, covering the history, architecture, and applications of deep neural networks.
33) Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press. - This book is a comprehensive textbook on deep learning, covering both the theoretical foundations and practical applications of the field.
34) Keras Documentation. (n.d.). Core Layers. Retrieved from https://keras.io/api/layers/core_layers/ This is the official documentation for the core layers in the Keras library, including the Flatten, Dense, and Dropout layers used in the code provided.
35) Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(1), 1929-1958. This is a research paper that introduced the dropout regularization technique used in the code provided.
36) "Understanding LSTM Networks" by Christopher Olah: https://colah.github.io/posts/2015-08-Understanding-LSTMs/
37) "Dropout: A Simple Way to Prevent Neural Networks from Overfitting" by Nitish Srivastava et al.: https://jmlr.org/papers/volume15/srivastava14a.old/srivastava14a.pdf

38) "Adam: A Method for Stochastic Optimization" by Diederik P. Kingma and Jimmy Ba: https://arxiv.org/abs/1412.6980
39) "Binary Cross-Entropy Loss" by Machine Learning Mastery: https://machinelearningmastery.com/cross-entropy-for-classification/
40) van der Lee, R., Buljan, M., Lang, B., Weatheritt, R. J., & Daughdrill, G. W. (2014). Classification of intrinsically disordered regions and proteins. Chemical reviews, 114(13), 6589-6631.
41) Monastyrskyy, B., Kryshtafovych, A., Moult, J., & Tramontano, A. (2014). Evaluation of disorder predictions in CASP10. Proteins: Structure, Function, and Bioinformatics, 82(S2), 127-137.
42) X. Ma, W. Li, L. Wang, et al. "Leaky Rectified Linear Units: A Better Activation Function for Deep Learning?" 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. https://ieeexplore.ieee.org/document/6639346
43) A. L. Maas, A. Y. Hannun, and A. Y. Ng. "Rectifier nonlinearities improve neural network acoustic models." Proceedings of the International Conference on Machine Learning (ICML), 2013. http://proceedings.mlr.press/v28/maas13.pdf
44) F. Chollet. "Deep Learning with Python." Manning Publications, 2017. https://www.manning.com/books/deep-learning-with-python
45) Alipanahi, B., Delong, A., Weirauch, M. T., & Frey, B. J. (2015). Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. Nature Biotechnology, 33(8), 831-838. doi: 10.1038/nbt.3300
46) Ragoza, M., Hochuli, J., Idrobo, E., Sunseri, J., Koes, D. R., & Bonneau, R. (2019). Protein–protein interaction prediction using neural networks trained with protein sequences. BMC Bioinformatics, 20(1), 1-14. doi: 10.1186/s12859-019-3009-5
47) Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105). This paper describes the architecture of the famous AlexNet CNN and its use in the ImageNet competition.
48) Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). This paper discusses visualization techniques for understanding the features learned by CNNs and provides insights into how these features are extracted from images.
49) Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9). This paper describes the architecture of the GoogLeNet CNN, which introduces the Inception module for improving efficiency and reducing computational cost. It also discusses the use of auxiliary classifiers for improving optimization and reducing overfitting.
50) Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. This paper describes the architecture of the VGG CNN, which consists of 16 or 19 layers and achieves state-of-the-art performance on the ImageNet dataset. It also discusses the use of smaller convolutional filters and deeper architectures for improving accuracy.