

A Machine Learning Approach To Protein Fold Recognition

Submitted in partial fulfillment of the requirements

of the degree of

Bachelor of Engineering

by

Ashwin Fernandes 21

Kavya Kotian 32

Chiraag Limaye 36

Karan Manghi 39

Supervisor:

Mr. Uday Nayak



UNIVERSITY OF MUMBAI

A Machine Learning Approach To Protein Fold Recognition

Submitted in partial fulfillment of the requirements

of the degree of

Bachelor of Engineering

by

Ashwin Fernandes	21
Kavya Kotian	32
Chiraag Limaye	36
Karan Manghi	39

Supervisor:

Mr. Uday Nayak



Department of Information Technology

Don Bosco Institute of Technology
Vidyavihar Station Road, Mumbai - 400070
2016-2017

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

CERTIFICATE

This is to certify that the project entitled “**Machine Learning Approach To Protein Fold Recognition**” is a bonafide work of

Ashwin Fernandes 21

Kavya Kotian 32

Chiraag Limaye 36

Karan Manghi 39

submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **Undergraduate** in **Bachelor of Information Technology**

Date:24/04/2017

(Prof. Uday Nayak)
Supervisor

(Prof. Janhavi B.)
HOD, IT Department

(Dr. Prasanna Nambiar)
Principal

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Project Report Approval for B.E.

This project report entitled “A Machine Learning Approach To Protein Fold Recognition” by Ashwin Fernandes, Kavya Kotian, Chiraag Limaye, Karan Manghi is approved for the degree of Bachelor of Engineering in Information Technology

(Examiner’s Name and Signature)

1. _____

2. _____

(Supervisor: Uday Nayak)

1. _____

Date: 24/04/2017

Place: Department Of Information Technology, DBIT

DON BOSCO INSTITUTE OF TECHNOLOGY

Vidyavihar Station Road, Mumbai - 400070

Department of Information Technology

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea / data / fact / source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

(_____))
(**Ashwin Fernandes 21**)

(_____))
(**Kavya Kotian 32**)

(_____))
(**Chiraag Limaye 36**)

(_____))
(**Karan Manghi 39**)

Date: 24/04/2017

Abstract

Proteins are key functional units in living organisms and are involved in many biological processes in the cell. Protein folding is the physical process by which a protein chain acquires its native 3-dimensional structure. The phenomenon of protein folding is very important in Biology as a protein's three dimensional structure largely determines its function. The process for identifying these structurally similar proteins is called fold recognition and forms the basis of the fold recognition problem.

Several experimental methods including X-ray crystallography, NMR spectroscopy, and electron microscopy have been used to determine protein structure. However, due to the significant cost and time for using those methods, the number of proteins with known structure(s) is significantly smaller than the number of known protein sequences (i.e. by a factor of approximately 200) and the sequence-structure gap is still increasing.

This project aims at training neural networks to predict if a given query-template protein pair belongs to the same structural fold. Thus trying to achieve accurate comparison of a protein pair.

Keywords: Neural Networks, Protein Fold, Protein Folding, Protein Fold Recognition

Contents

1	Introduction	2
1.1	Problem Statement	2
1.2	Scope of the Project	2
1.3	Current Scenario	2
1.4	Need for the Proposed System	5
2	Review of Literature	6
2.1	Summary of the investigation in the published papers	6
2.2	Algorithm	8
3	Analysis and Design	11
3.1	Methodology / Procedure adopted	11
3.2	Analysis	12
3.2.1	Software Requirements	12
3.3	System Architecture / Design	12
3.3.1	Modules and their description	12
4	Implementation	15
4.1	Implementation Plan <i>for Sem – 8</i>	15
4.2	Coding Standard	16
4.3	Dataset	18
5	Results and Discussion	22
6	Conclusion & Future Work	24
	Appendix	27
	Query protein	27
	Template Protein	27

Fold Recognition Problem	27
Sequence Identity	27
Sequence-sequence alignment	27
Profile-profile alignment	27
MUSCLE	27
CLUSTALW	27
T-Coffee	27
HHSearch	27
HMMer	27
BLAST	27
PSI-BLAST	27
IMPALA	27

References	29
-------------------	-----------

Acknowledgement	30
------------------------	-----------

List of Figures

1.1	BLAST GUI	3
1.2	FOLDpro GUI	4
2.1	Neural Network	10
3.1	Block diagram of the Neural Network Implementation	13
4.1	Implementation Timeline for Sem-8	15
4.2	Gantt Chart for Sem-8	16

List of Tables

5.1	Top Twenty input Features	22
-----	-------------------------------------	----

Chapter 1

Introduction

1.1 Problem Statement

To train neural networks to predict if a protein pair belongs to the same structural fold.

1.2 Scope of the Project

The dataset used for training the neural networks is constructed such that protein pairs have lower than 40 percentage sequence identity. A protein from the dataset is paired with all the other proteins and pairwise similarity is calculated. Pairwise similarity features are obtained through five types of sequence alignment and/or protein structure prediction tools (i.e., sequence-sequence alignment, sequence-family information, sequence-profile alignment, profile-profile alignment and structural information) and selected based on their use in prior works.[7]

These pairwise similarities serve as inputs to the neural network. 84 such features are considered.[7] The neural network is trained and tested with a dataset file of size 3.8MB[7].

1.3 Current Scenario

- **BLAST**

The BLAST programs are widely used tools for searching protein and DNA databases for sequence similarities. For protein comparisons, a variety of definitional, algorithmic and statistical refinements described here permits the execution time of the BLAST programs to be decreased substantially while enhancing their sensitivity to weak similarities. [4]

The image shows the NCBI BLAST (Basic Local Alignment Search Tool) web interface. The top section is titled 'Enter Query Sequence' and includes a large text input field for 'Enter accession number(s), gi(s), or FASTA sequence(s)', a 'Clear' button, and a 'Query subrange' section with 'From' and 'To' input fields. Below this is an 'Or, upload file' section with a 'Choose File' button and 'No file chosen' text. There is also a 'Job Title' input field and a checkbox for 'Align two or more sequences'. The second section is 'Choose Search Set', which includes a 'Database' dropdown menu set to 'Non-redundant protein sequences (nr)', an 'Organism' input field with a suggestion icon, an 'Exclude' checkbox, and an 'Entrez Query' input field. The third section is 'Program Selection', which shows a list of algorithms: 'Quick BLASTP (Accelerated protein-protein BLAST)', 'blastp (protein-protein BLAST)' (which is selected), 'PSI-BLAST (Position-Specific Iterated BLAST)', 'PHI-BLAST (Pattern Hit Initiated BLAST)', and 'DELTA-BLAST (Domain Enhanced Lookup Time Accelerated BLAST)'.

Figure 1.1: BLAST GUI

In bioinformatics, BLAST for Basic Local Alignment Search Tool is an algorithm for comparing primary biological sequence information, such as the amino-acid sequences of proteins or the nucleotides of DNA sequences. A BLAST search enables a researcher to compare a query sequence with a library or database of sequences, and identify library sequences that resemble the query sequence above a certain threshold.[5]


- **FOLDpro**

A two-stage machine learning, information retrieval, approach to fold recognition. First, it uses alignment methods to derive pairwise similarity features for query-template protein pairs. Second, it apply support vector machines to these features to predict the structural relevance (i.e. in the same

fold or not) of the query-template pairs. For each query, the continuous relevance scores are used to rank the templates. The FOLDpro approach is modular, scalable, and effective. Using predictions of the top-ranked template, the sensitivity is about 85%, 56%, and 27% at the family, super-family, and fold levels respectively. Using the 5 top-ranked templates, the sensitivity increases to 90%, 70%, and 48%[3].

FOLDpro is a web server to predict protein 3D structure using a machine learning fold recognition approach. It makes predictions in three steps.

- Step 1: Use a machine learning information retrieval approach to rank template proteins for the query protein, integrating a variety of similarity features.[3]



FOLDpro: Protein Fold Recognition and Template-Based 3D Structure Prediction

Email address(where the prediction will be sent):

Target Name(required):

Protein sequence(one plain sequence, no headers):

Figure 1.2: FOLDpro GUI

- Step 2: Generate profile-profile alignments between the query protein and the top ranked template proteins. Multiple templates are used to improve both the alignment and the structure modeling if necessary.[3]
- Step 3: Based on the query-template alignments and 3D structures of the templates, Modeller (Sali and Blundell, 1993) is used to generate structure models for the query protein. Five models are generated for the query. The models ranked higher are generated based on the

templates ranked higher. So they are presumably, but not always, better than the models ranked lower (e.g., fold1.pdb is likely better than fold2.pdb. fold2.pdb is likely better than fold3.pdb).[3]

1.4 Need for the Proposed System

Given scores of the 84 scores extracted[7] for any two proteins our system can tell if the two contain similar folds. The existing software FOLDpro takes into account 54 input features.[3] On conducting literature survey it was observed that to increase accuracy 30 more features contribute to the result.[3] Thus our system considers 84 features cited by the latest discovery.

Several experimental methods including X-ray crystallography, NMR spectroscopy, and electron microscopy have been used to determine protein structure. However, due to the significant cost and time for using those methods, the number of proteins with known structure(s) is significantly smaller than the number of known protein sequences (i.e. by a factor of approximately 200) and the sequence-structure gap is still increasing[7]. Therefore, the construction of computational approaches and tools to predict a protein's three dimensional structure from its sequence is an important problem not only for :

- Understanding the relationship between protein structure function[6].
- Advancement in protein-based biotechnologies and drug discovery[8].

An important task in protein structure prediction is to identify proteins that have similar tertiary structures (from among those that have already been determined experimentally). By identifying such proteins, their structures can be used as a template to model the unknown structure of another protein[7].

Chapter 2

Review of Literature

2.1 Summary of the investigation in the published papers

Recent Progress in Machine Learning-Based Methods for Protein Fold Recognition:

Understanding how proteins adopt their 3D structure remains one of the greatest challenges in science. Determination of the fold category of a protein is crucial as it reveals the 3D structure of proteins. Classification of a protein of unknown structure under a fold category is called fold recognition, which is a fundamental step in the determination of the tertiary structure of a protein.[14]

- Systematically reviewed recent progress in machine learning-based protein fold recognition methods.
- Compared with the traditional experimental methods, machine learning-based methods offer better advantages in terms of robustness and reliable performance.

Machine learning-based methods categorized into two classes according to the learning algorithms used in prediction models:

1. (1) Single classifier-based methods
2. (2) Ensemble classifier-based methods.

- Single classifier-based methods involve SVM classifier and it has been highly efficient in several fields of bioinformatics such as in protein remote homology detection, protein structural classification, and DNA-binding protein prediction.
- Ensemble classifier-based methods use one specific feature descriptor to encode query proteins with feature representations; the feature representations are trained with each single basic classifier to create n single classifier models. They explore the functional domain information and sequential evolution information.

- The different methods are compared on a public benchmark dataset.
- A public dataset, DD (Ding and Dubchak), is used as a benchmark dataset for performance comparison of the existing methods.
- The DD dataset is comprised of a training dataset and a testing dataset, both of which cover 27 protein fold classes in the SCOP database.
- The training dataset contains 311 protein sequences with 40% residue identity, while the testing dataset contains 383 protein sequences with 35% residue identity.
- 20 methods were evaluated and compared on the DD dataset.

In general, machine learning-based methods can be successfully applied in protein fold recognition. In the future, machine learning methods will be extensively applied in other similar but unexplored fields, such as disease-causing amino acid change prediction, protein-protein binding site or interaction prediction, and DNA-protein binding site or interaction prediction.

Protein family classification with Neural networks[11]:

- Protein families are defined to group together proteins that share similar structure.
- It focusses on training vector representations for protein sequences and investigate various neural network models for predicting a protein's family.
- Dataset used: Universal Protein Resource (UniProt).
- Used Global Vector for amino acid sequence representation.
- Support Vector Machine was used to classify the proteins. It took 7 hours to train the model.

Next-generation sequencing technologies generate large amounts of biological sequence information in the form of DNA/RNA sequences. From DNA sequences we also know the amino acid sequences of proteins, which are the fundamental molecules that perform most biological functions. The functionality of a protein is thus encoded in the amino acid sequence and understanding the sequence-function relationship is a major challenge in bioinformatics. Investigating protein functional often involves structural studies (crystallography)

or biochemical studies, which require time consuming efforts. Protein families are defined to group together proteins that share similar function, and the aim of our project is to predict protein family from raw sequence. We focus on training informative vector representations for protein sequences and investigate various neural network models for the task of predicting a protein's family.

The Protein Data Bank[13]:

The Protein Data Bank (PDB; <http://www.rcsb.org/pdb/>) is the single world-wide archive of structural data of biological macromolecules. This paper describes the goals of the PDB, the systems in place for data deposition and access, how to obtain further information, and near-term plans for the future development of the resource. These are exciting and challenging times to be responsible for the collection, curation and distribution of macromolecular structure data. Since the RCSB assumed responsibility for data deposition in February 1999, the number of depositions has averaged approximately 50 per week. However, with the advent of a number of structure genomics initiatives worldwide this number is likely to increase. We estimate that the PDB, which at this writing contains approximately 10 500 structures, could triple or quadruple in size over the next 5 years. This presents a challenge to timely distribution while maintaining high quality. The PDB's approach of using modern data management practices should permit us to scale to accommodate a large data influx.

2.2 Algorithm

The Algorithm used in this project :Recurrent Neural Networks Our system consists of 3 layers. Input layer, hidden layer and output layer. The input layer consists of 84 nodes as we have 84 comaprison results to be considered. Explanation of backpropagation with respect to our system:

1. Sigmoidal function is implemented by this algorithm. A binary sigmoidal function has binary '0' and '1' but class labels used are 1 and 2, hence it is deduced that this algorithm is using a sigmoidal function. The use of the sigmoidal function will be clearer as we explain further. The learning rate is a variable value which can be changed manually. In our system, we have kept it as 0.3. The use of the learning rate will also be clearly understood .
2. Initially random weights are assigned to each synapse(connection between

nodes of different layers) in the system. Based on these weights and the values for each of the 84 nodes, an output for each of the nodes is calculated by multiplying the weights and the node values.

3. The input to the next node is calculated by applying the sigmoidal function to the output of the node to which it is connected. This is called forward propagation. The reason why we use the sigmoidal function is that the derivative of the sigmoidal function can be calculated by simple multiplication and hence no need to use calculus libraries.
4. If $f(x)$ is the sigmoidal function then its derivative is defined as $f(x) * [1 - f(x)]$. This is how simple it is. Moving on, by doing this, we can find out the input to the output layer and hence its activation function. The activation function is nothing but the sigmoidal function.
5. Now based on the sigmoidal function of the output layer, we calculate the error in the final output. This is done with the help of stochastic gradient descent. It is a mathematical concept, where we try to achieve a global minima.
6. Global minima in terms of error is achieved in our case because we want to minimise the error. The number or amount of steps on the graph which we need to take in order to achieve global minima are governed by the learning rate.
7. If we take a very high learning rate then we might miss out on reaching the global minima because then we would be varying the weights by a large amount. Also if we take a very small learning rate then it would take a lot of time to achieve the global minima. Hence we selected the learning rate in such a way that it is optimum. Based on this error calculated, we calculate the change in weights for all the synapses. This change in weights is then added to the original weights and the new weights are found. This is called back propagation of error.

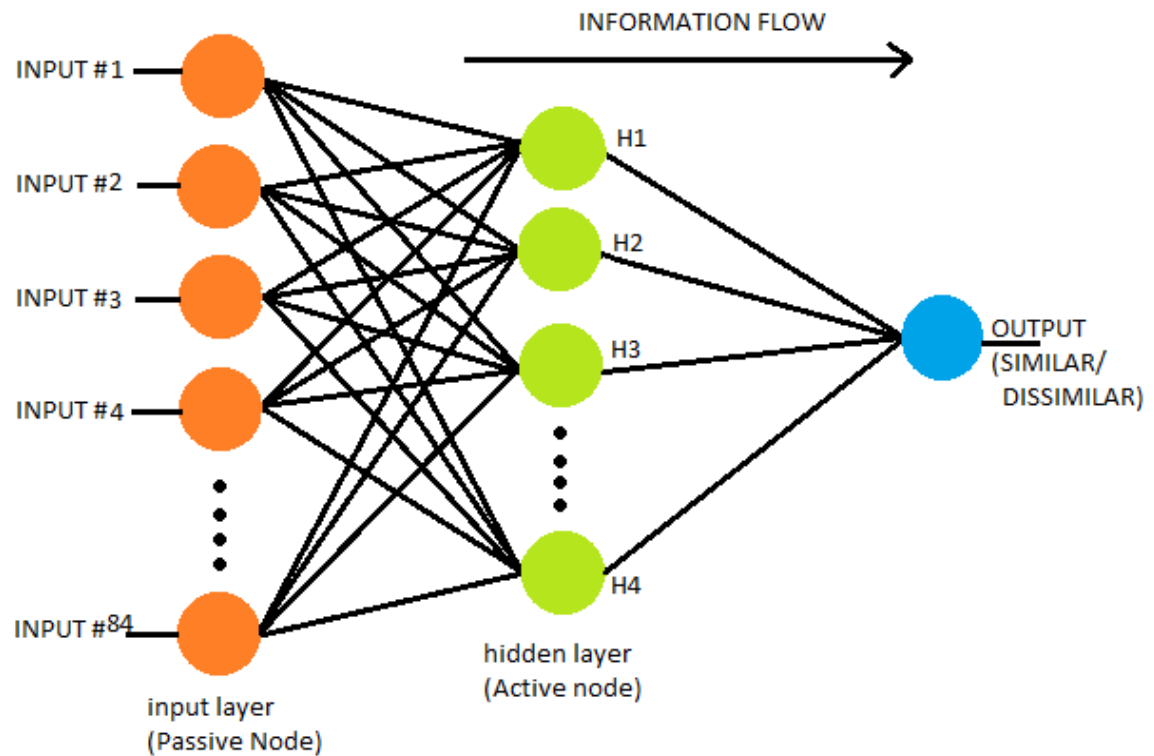


Figure 2.1: Neural Network

8. This goes on till the number of epochs is equal to the number of epochs that we have defined. Finally when all the epochs are done, the final weights are found. Based on these weights, the system can predict an unseen pair to be similar or dissimilar.

Chapter 3

Analysis and Design

3.1 Methodology / Procedure adopted

The methodology/model adopted would be **Iterative Model**.

The incremental build model is a method of software development where the product is designed, implemented and tested incrementally (a little more is added each time) until the product is finished. It involves both development and maintenance. The product is defined as finished when it satisfies all of its requirements. This model combines the elements of the waterfall model with the iterative philosophy of prototyping. The product is decomposed into a number of components, each of which is designed and built separately (termed as builds). Each component is delivered to the client when it is complete. This allows partial utilization of the product and avoids a long development time. It also avoids a large initial capital outlay and subsequent long waiting period. This model of development also helps ease the traumatic effect of introducing a completely new system all at once. There are, however, several problems with this model. The incremental model applies the waterfall model incrementally.[9]

Tasks involved :

These tasks are common to all the models[9]

- 1. Communication: helps to understand the objective.
- 2. Planning: required as many people (software teams) work on the same project but different function at same time.
- 3. Modeling: involves business modeling, data modeling, and process modeling.
- 4. Construction: this involves the reuse software components and automatic code.
- 5. Deployment: integration of all the increments.

Weekly meetings were conducted twice, on Wednesdays and Thursdays during project hours. Progress of the project was monitored and measured by our

Project guide Prof. Uday Nayak every week.

3.2 Analysis

This project being an inter-disciplinary project required extensive research referring to multiple research papers to help us understand the concept thoroughly. The requirements gathering and the feasibility study phase was quite long as discovery of a new concepts would lead to reformatting of the requirements and the execution plan all over again.

3.2.1 Software Requirements

- Anaconda 4.2.0
- Python 3.5.2
- Spyder 3.0.0
- Intel core i3 with 4 gb ram

3.3 System Architecture / Design

The existing software FOLDpro takes into account 54 input features.[2] While our system considers 84 features cited by the latest discovery.

3.3.1 Modules and their description

Input module

- 84 feature comparison results of each protein pair constitutes 1 row in the dataset. The last number on each row is a 1 or a 2. 1 signifies that the two proteins are similar and 2 signifies that they are not similar. We have taken 1,2 and not 1,0 as the class labels as we are using these class labels directly in the formula of the "normalise_dataset" function.
- If we take a 0 then there occurs a "divide-by-zero-error". So instead of converting the class labels from 0,1 to a set which does not cause an error, we are directly using 1,2 as the class labels. It doesn't reduce the size of the program by a large factor. But it does reduce and as we know that every

little drops forms an ocean. It is an innovative method that we have used here.

- We have also used the concepts of k-folds where the training test set is alternatively selected from the dataset provided to it for training.

Neural Network implementation module :

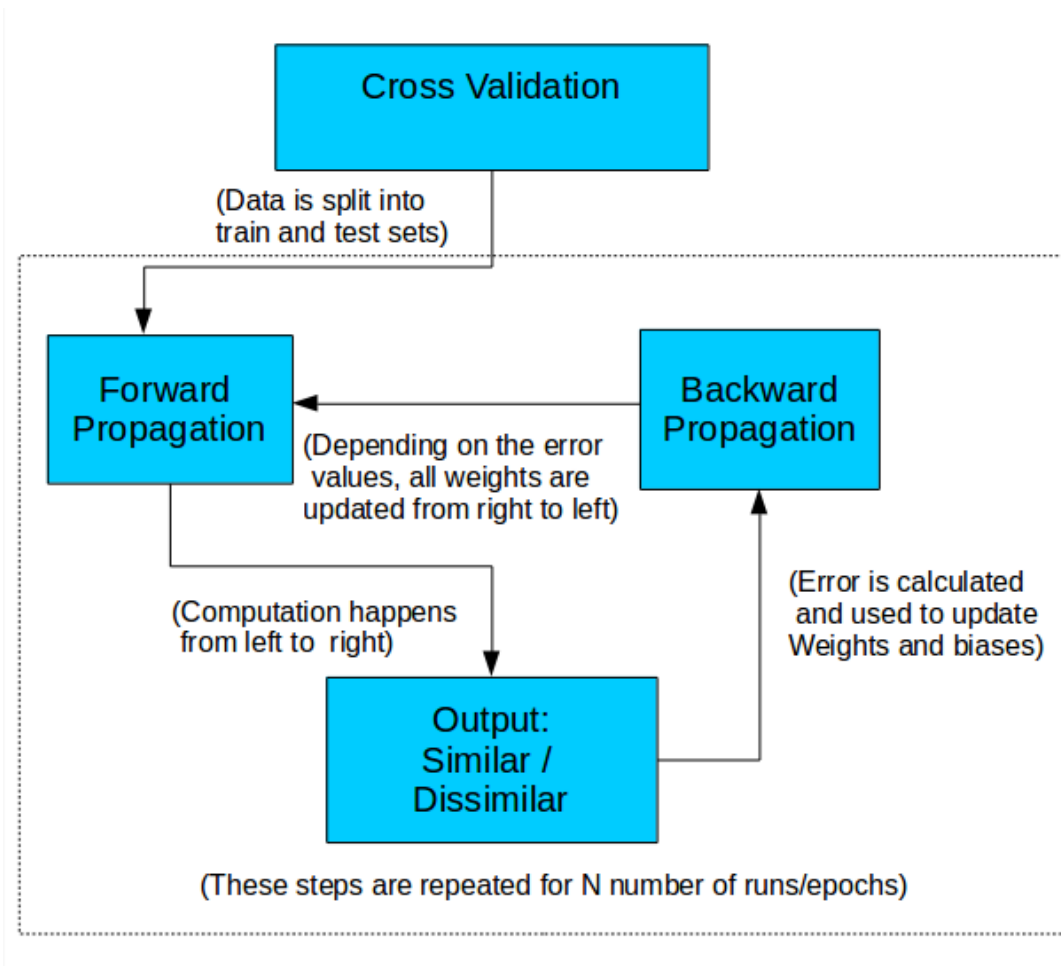


Figure 3.1: Block diagram of the Neural Network Implementation

- Cross Validation is performed which means the dataset is divided into training and testing sets. For instance there are 10 comparison scores in the dataset, the first 9 scores will be used as training set and the 10th score will be used as the testing set. For the next iteration, the first 8 scores and the 10th score will be used as the training set and the 9th score will be used as the testing set and so on. Thus the entire dataset is trained as well as tested.
- The neural network is fed scores and the forward propagation computes

whether the proteins are similar or not by giving the output as 1 or 2.

- Depending on the output obtained, error is calculated. If the actual output is equal to the desired output, there is no error. But if it does not match with the desired output then there is an error.
- After calculating the error, we activate the back propagation function where the error value is used to update all weights in the neural network. After updating the weights, forward propagation function is activated and it again computes the result.
- These steps are repeated for N iterations which is specified in the program.

Chapter 4

Implementation

4.1 Implementation Plan for Sem – 8

Implementation Plan for the Sem - 8 to include the following:

- Our Implementation Plan involved splitting into groups of 2 at a time, one group would focus on the dataset extraction at a time while the other group would focus on building the neural network.
- After 2-3 weeks would swap work hence everyone is responsible for contribution to all the modules mentioned.
- Work breakdown mainly involved
 - Literature survey
 - Feasibility study
 - Requirements gathering
 - Dataset Extraction
 - Coding the Neural Network

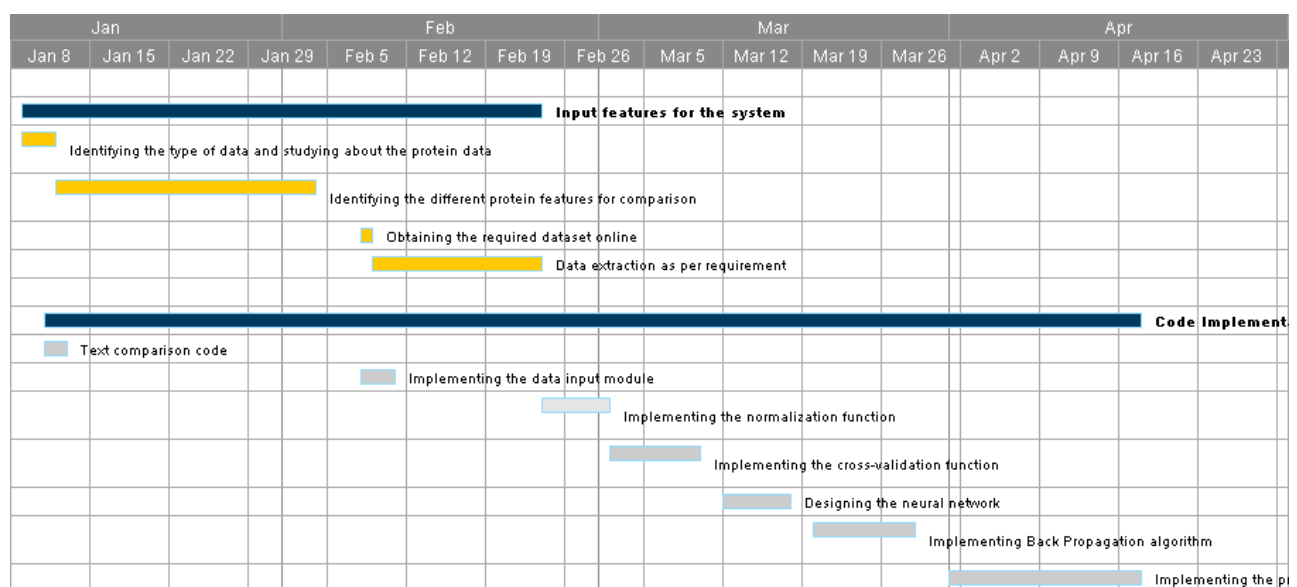


Figure 4.1: Implementation Timeline for Sem-8

	Task Name	Start Date	End Date	Dur
1				
2	Input features for the system	01/09/17	02/23/17	34d
3	Identifying the type of data and studying about the protein data	01/09/17	01/11/17	3d
4	Identifying the different protein features for comparison	01/12/17	02/03/17	17d
5	Obtaining the required dataset online	02/08/17	02/08/17	1d
6	Data extraction as per requirement	02/09/17	02/23/17	11d
7				
8	Code Implementation	01/11/17	04/17/17	69d
9	Text comparison code	01/11/17	01/12/17	2d
10	Implementing the data input module	02/08/17	02/10/17	3d
11	Implementing the normalization function	02/24/17	03/01/17	4d
12	Implementing the cross-validation function	03/02/17	03/09/17	6d
13	Designing the neural network	03/12/17	03/17/17	6d
14	Implementing Back Propagation algorithm	03/20/17	03/28/17	7d
15	Implementing the prediction function	04/01/17	04/17/17	12d

Figure 4.2: Gantt Chart for Sem-8

4.2 Coding Standard

Functions used can be described as follows:

- `load_csv`:
It is used to take the dataset as an input and then read the dataset row by row
- `dataset_minmax`:
Find the minimum and maximum value for each column inorder to normalise the data. It is used in the next function called "normalise_dataset"
- `normalise_dataset`:
It is used to achieve standardization by converting all values in the range of 0 and 1
- `cross_validation_split`:
It splits the dataset into "k" folds where k is defined at the beginning of the program

- **accuracy_metric:**

This function calculates the accuracy of the program by comparing the predicted and actual values

- **evaluate_algorithm:**

It invokes the "cross_validation_split" function and with the help of that it calculates the accuracy of the program by invoking the "accuracy_metric" function

- **activate:**

It calculates the net output of each neuron by multiplying the weight and the input of each neuron

- **transfer:**

It is the neuron activation function which is further used when we calculate the derivative of the function. Such a function is selected on purpose as its derivative is $f(x)*(1-f(x))$ if $f(x)$ is the transfer function. This function is called Sigmoidal Function

- **forward_propagate:**

Here, we use the "transfer" and "activate" function to calculate the inputs for the next layer from the previous layer

- **transfer_derivative:**

Used to calculate the derivative of each output to be further used for stochastic gradient descent

- **backward_propagate_error:**

The error found between the predicted value and the actual value is propagated to the previous layers taking into account the learning rate and the gradient descent

- **update_weights:**

The change on weights, delta, is calculated and then it is added to the original weights for each synapse

- **train_network:**

Here, the number of iterations(epochs) are taken into account and it simply invokes the "backward_propagate_error", "forward_propagate" and "update_weights" functions to achieve the prediction and learning

- `initialize_network`:
Here, random initial weights are assigned to every synapse
- `back_propagation`:
This function is the heart of the program and governs and invokes and combines the values of all functions
- `predict`:
This function is used when we provide an unseen dataset. It will use all the learning that the system has accumulated and will use that to to classify the proteins as similar or dissimilar

4.3 Dataset

The protein dataset consists of proteins with less than 40% similarity which is extracted from the SCOP dataset as per prior works.[10] Pairs of proteins from this database are formed by combinations of one protein with every other protein. These pairs are then used to produce the input to the neural network. Pariwise input features in categories of sequences, protein families, sequence-sequence alignment, sequence-profile alignment, and profile-profile alignment.[7] These input values are scores generated from number of external alignment tools including MUSCLE, CLUSTALW, T-Coffee, HHSearch, HMMer, BLAST, PSI-BLAST, IMPALA, PALIGN, PRC, and Compass. [7]

The input feature name list [7] :

- 1 query length
- 2 target length
- 3 cosine of composition of monomer of query and template sequences
- 4 correlation of composition of monomer of query and template sequences
- 5 gaussian function of composition of monomer of query and template sequences
- 6 cosine of composition of dimer of query and template sequences
- 7 correlation of composition of dimer of query and template sequences
- 8 gaussian function of composition of dimer of query and template sequences
- 9 cosine of composition of monomer of query and template families
- 10 correlation of composition of monomer of query and template families
- 11 gaussian function of composition of monomer of query and template families

- 12 cosine of composition of dimer of query and template families
- 13 correlation of composition of dimer of query and template families
- 14 gaussian function of composition of dimer of query and template families
- 15 Palign sequence alignment score1
- 16 Palign sequence alignment score2
- 17 Clustalw sequence alignment score
- 18 Clustalw profile-profile alignment score
- 19 Lobster (coach) profile-profile alignment score
- 20 Psiblast profile-sequence alignment score
- 21 Psiblast alignment evalule
- 22 Psiblast normalized alignment length
- 23 Psiblast identity rate of alignment
- 24 Psiblast positive rate of alignment
- 25 Hmmer(pfam) alignment score
- 26 Hmmer(pfam) alignment evalule
- 27 Hmmer(search) alignment score
- 28 Hmmer(search) alignment evalule
- 29 Impala sequence-profile alignment score
- 30 Impala alignment evalule
- 31 Impala normalized alignment length
- 32 Impala alignment identity rate
- 33 Impala alignment positive rate
- 34 Rpsblast sequence-profile alignment score
- 35 Rpsblast alignment evalule
- 36 Rpsblast alignment normalized length
- 37 Rpsblast alignment indentity rate
- 38 Rpsblast alignment positive rate
- 39 secondary structure match ratio
- 40 relative solvent accessibility match ratio
- 41 average contact probability (cmappro, 8 Angstrom)
- 42 cosine of residue contact num (cmappro, 8 Angstrom)
- 43 correlation of residue contact num (cmappro, 8 Angstrom)
- 44 cosine of residue contact order (cmappro, 8 Angstrom)
- 45 corr_of_residue_contact_ord (cmappro, 8 Angstrom)
- 46 average contact probability (bmappro, 8 Angstrom)
- 47 cosine of residue contact num (bmappro, 8 Angstrom)

- 48 correlation of residue contact num (bmappro, 8 Angstrom)
- 49 cosine of residue contact order (bmappro, 8 Angstrom)
- 50 corr_of_residue_contact_ord (bmappro, 8 Angstrom)
- 51 average contact probability (cmappro, 12 Angstrom)
- 52 cosine of residue contact num (cmappro, 12 Angstrom)
- 53 correlation of residue contact num (cmappro, 12 Angstrom)
- 54 cosine of residue contact order (cmappro, 12 Angstrom)
- 55 corr_of_residue_contact_ord (cmappro, 12 Angstrom)
- 56 average contact probability (bmappro, 12 Angstrom)
- 57 cosine of residue contact num (bmappro, 12 Angstrom)
- 58 correlation of residue contact num (bmappro, 12 Angstrom)
- 59 cosine of residue contact order (bmappro, 12 Angstrom)
- 60 corr_of_residue_contact_ord (bmappro, 12 Angstrom)
- 61 beta-residue in beta-sheet pairing probability
- 62 percentage of helix in query
- 63 percentage of beta-strand in query
- 64 percentage of coil in query
- 65 percentage of exposed residue in query
- 66 percentage of buried residue in query
- 67 percentage of helix in template
- 68 percentage of beta-strand in template
- 69 percentage of coil in template
- 70 percentage of exposed residue in template
- 71 percentage of buried residue in template
- 72 consine of SS and SA composition
- 73 correlation of SS and SA composition
- 74 gaussian function of SS and SSA composition
- 75 dot product of SS and SA composition
- 76 prc-hmm coemis score (on hmmer profile)
- 77 prc-hmm simple score (on hmmer profile)
- 78 prc-hmm reverse score (on hmmer profile)
- 79 prc-hmm coemis score (on chk profile)
- 80 prc-hmm simple score (on chk profile)
- 81 prc-hmm reverse score (on chk profile)
- 82 HHsearch profile-profile alignment score
- 83 Compass alignment score

84 Compass evalule

A typical dataset for a pair of proteins is given as :

#1aca-d1aca 1abra-d1abra

2 0.510748 2.510000 0.698011 0.012915 0.809200 0.270975 0.042790 0.876369
 0.868627 0.452407 0.876726 0.600178 0.263828 0.935275 0.439352 0.339785
 0.093023 1.755814 -2.197674 0.169767 -0.430783 0.139535 0.330000 0.330000
 -2.615116 0.000000 -0.873256 -0.198451 0.136047 1.504077 0.232558 0.100000
 0.350000 0.118605 2.639057 0.069767 0.000000 0.500000 0.220930 0.581395
 0.022947 0.526593 0.191639 0.537929 0.301708 0.013903 0.553371 0.262824
 0.503738 0.262899 0.282084 0.763291 0.427663 0.761204 0.578318 0.241353
 0.732094 0.309420 0.710735 0.470353 0.000000 0.627907 0.000000 0.372093
 0.546512 0.453488 0.374502 0.223108 0.402390 0.446215 0.553785 0.932924
 0.682792 0.692490 0.879876 0.150000 0.122093 -0.009302 0.154651 0.094186
 -0.031395 0.077907 0.395349 0.084429

(Where Protein 1: 1aca-d1aca Protein 2: 1abra-d1abra The first value '2' indicates that the two proteins do not have similar folds. Values second onwards are the scores of 84 features listed above.)

Chapter 5

Results and Discussion

If a system with lesser inputs is desired we could consider the 20 input scores listed below as they contribute to the top information gain from among the 84

Input Feature	Information Gain
HHSearch score	0.0375
COMPASS evalue	0.0370
PRC reverse score on chk profile	0.0354
PRC reverse score on HMM profile	0.0341
HMMer pfam evalue	0.0287
Dot product of SS and RSA vectors	0.0266
HMMer search evalue	0.0264
SS match ratio	0.0263
Correlation of SS and RSA vectors	0.0263
PRC simple score on HMM profile	0.0248
Cosine of SS and RSA vectors	0.0246
Gaussian kernel on SS and RSA vectors	0.0237
COMPASS score	0.0235
PRC coemis score on HMM profile	0.022
PSI-BLAST evalue	0.0205
IMPALA evalue	0.0181
RPS-BLAST evalue	0.0180
SA match ratio	0.0154
Cosine of residue contact num (8A)	0.0150
HMMer search score	0.0142

Table 5.1: Top Twenty input Features

features we have considered .

The resulting system is such that the neural network is trained and tested with the help of a dataset of size 3.8MB .

Given scores of the 84 scores extracted[7] for any two proteins our system can

tell if the two contain similar folds. The existing software FOLDpro takes into account 54 input features.[3] On conducting literature survey it was observed that to increase accuracy 30 more features contribute to the result.[3] Thus our system considers 84 features cited by the latest discovery.

Chapter 6

Conclusion & Future Work

In this work, we have constructed a neural network such that given a protein pair it tells us whether it belongs to the same fold

The accuracy that the system varies with the size of the dataset. With a smaller dataset of the order of Kbs, it can give an accuracy of 100%. But as the dataset size increases, goes in the orders of Mbs, it gives an average accuracy of around 75%. Also, the accuracy depends on the learning rate. Very high or very low learning rate may reduce the accuracy. Hence for every dataset(size) we take, we need to tweak the learning rate so as to achieve a high accuracy. By training and getting an experience of a number of datasets having different sizes, we could get used to a constant or a small range of learning rates. For the current testings that we did, we found a learning rate of 0.3 to be optimum. Moreover, the accuracy also depends on the number of folds and number of epochs taken. More the number of epochs and folds, greater will be the accuracy but the time taken to give the output will increase. Please note that the "accuracy" of the system doesn't take into account the time factor. It just considers the correctness of the predictions. Hence in order to keep a balance between the time taken and correctness, we need to take optimum number of epochs and folds.

As the project is an interdisciplinary project the literature survey took quite a while. Also bioinformatics offers an abundance of data on their various websites like PDB, UniProt, NCBI etc. Streamlining the data which we would be working with so as to provide a notable contribution to the fold recognition was a tedious procedure. Data Preprocessing module was undertaken and thus working on feature generation scripts for obtaining scores from the output files was a task that was taking a long time. After researching a bit more we could extract datasets from a repository collated by Taeho Jo, Jie Hou, Jesse Eickholt Jianlin Cheng[7]

This project considers a basic Recurrent Neural Network and tells if the given two proteins have similar folds.

Further scope can be increased by adding a Data preprocessing module. This would involve extracting input features by giving the protein pairs present in

their FASTA format to various sequence alignment tools and applying feature generation scripts to extract the scores from the output files.

Also the neural network module can be replaced with modules using more complex algorithms like HMM algorithm, Restricted Boltzman machines, Convolutional Neural Networks etc to offer comparisons on which handles the Protein Fold Recognition problem better.

Appendix - I

Query protein - A protein sequence on which the analysis is performed.

Template Protein - A protein sequence which serves as a model for comparing other proteins.

Fold Recognition Problem - The Protein Folding Problem is the obstacle that scientists confront when they try to predict 3D structure of proteins based on their amino acid sequence.

Sequence Identity - Sequence identity is the amount of characters which match exactly between two different sequences. Hereby, gaps are not counted and the measurement is relational to the shorter of the two sequences.

Sequence-sequence alignment - It is a way of arranging the sequences of proteins to identify regions of similarity that may be a consequence of functional, structural, or evolutionary relationships between the sequences.

Profile-profile alignment - It is a multiple sequence comparison method for finding and aligning distantly related sequences which gives better results than sequence sequence alignment and sequence profile alignment.

MUSCLE - Multiple Sequence Comparison by Log-Expectation (MUSCLE) is computer software for multiple sequence alignment of protein and nucleotide sequences.

CLUSTALW - Clustal W is a general purpose multiple sequence alignment program for DNA or proteins. It produces biologically meaningful multiple sequence alignments of divergent sequences.

T-Coffee - T-Coffee (Tree-based Consistency Objective Function For alignment Evaluation) is a multiple sequence alignment software using a progressive approach.

HHSearch - HHsearch is an open-source software program for protein sequence searching that is part of the free HH-suite software package.

HMMer - HMMER is used for searching sequence databases for sequence homologs, and for making sequence alignments. It implements methods using probabilistic models called profile hidden Markov models (profile HMMs).

BLAST - Basic Local Alignment Search Tool finds regions of similarity between biological sequences. The program compares nucleotide or protein sequences to sequence databases and calculates the statistical significance.

PSI-BLAST - Position-Specific Iterative BLAST (PSI-BLAST) (blastpgp) This program is used to find distant relatives of a protein. First, a list of all closely related proteins is created. These proteins are combined into a general "profile" sequence, which summarises significant features present in these sequences.

IMPALA - It is designed for the complementary procedure of comparing a single query sequence with a database of PSI-BLAST- generated PSSMs (position-specific scoring matrix).

References

- [1] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, Philip E. Bourne; *The Protein Data Bank. Nucleic Acids Res* 2000; 28 (1): 235-242. doi: 10.1093/nar/28.1.235
- [2] Leyi Wei and Quan Zou; *Recent Progress in Machine Learning-Based Methods for Protein Fold Recognition*; School of Computer Science and Technology, Tianjin University, Tianjin 300354
- [3] Jianlin Cheng, Pierre Baldi; *A machine learning information retrieval approach to protein fold recognition*; *Bioinformatics* 2006; 22(12):1456-1463. doi:10.1093/bioinformatics/btl102
- [4] Stephen F. Altschul, Thomas L. Madden, Alejandro A. Schäffer, Jinghui Zhang, Zheng Zhang, Webb Miller, David J. Lipman; *BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res* 1997; 25 (17): 3389-3402. doi: 10.1093/nar/25.17.3389
- [5] BLAST <http://en.wikipedia.org/wiki/BLAST> , last modified on 12 April 2017, at 02:26
- [6] Jo, T. et al. *Homology Modeling of an Algal Membrane Protein, Heterosigma Akashiwo Na⁺ -ATPase. Membrane* 35(2), 80–85 (2010).
- [7] Taeho Jo, Jie Hou, Jesse Eickholt Jianlin Cheng; *Improving Protein Fold Recognition by Deep Learning Networks*; SCIENTIFIC REPORT 5, ARTICLE NUMBER: 17573 (2015)
- [8] Baker, D. Centenary Award and Sir Frederick Gowland Hopkins Memorial Lecture. *Biochemical Society Transactions* 42(2),225–229 (2014).
- [9] Pressman, Roger (2010). *Software Engineering: A Practitioner's Approach. Boston: McGraw Hill.* pp.41–42. ISBN9780073375977.
- [10] Murzin, A. G., Brenner, S. E., Hubbard, T. Chothia, C. *SCOP: a structural classification of proteins database for the investigation of sequences and structures. Journal of molecular biology* 247, 536–540 (1995).

-
- [11] Matt Spencer, Jesse Eickholt, and Jianlin Cheng; *A Deep Learning Network Approach to ab initio*;IEEE/ACM TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, VOL.12, NO. 1, JANUARY/FEBRUARY 2015
- [12] Timothy K. Lee,Tuan Nguyen;*Protein Family Classification with Neural Networks*, STANFORD UNIVERSITY
- [13] Helen M. Berman, John Westbrook, Zukang Feng, Gary Gilliland, T. N. Bhat, Helge Weissig, Ilya N. Shindyalov, Philip E. Bourne; *The Protein Data Bank. Nucleic Acids Res* 2000; 28 (1): 235-242. doi: 10.1093/nar/28.1.235
- [14] Leyi Wei and Quan Zou;*Recent Progress in Machine Learning-Based Methods for Protein Fold Recognition*;School of Computer Science and Technology, Tianjin University, Tianjin 300354

Acknowledgements

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and Organizations. We would like to extend my sincere thanks to all of them. We are highly indebted to prof. Uday Nayak for their guidance and constant supervision as well for providing necessary information regarding the project and also for his support for completing the Project.

We would like to express my gratitude towards our project Coordinator Prof. Tayyabali Sayyad, Head of The Department IT, Prof. Janhavi Baikerikar and Principal of DBIT, Dr. Prasanna Nambiar for their kind co-operation and encouragement which helped us in completion of this project. Our thanks and appreciation also go to our parents and the entire college staff who willingly helped us out with their abilities.

(_____)
(**Ashwin Fernandes 21**)

(_____)
(**Kavya Kotian 32**)

(_____)
(**Chiraag Limaye 36**)

(_____)
(**Karan Manghi 39**)

Date: 24/04/2017