



TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING,
PULCHOWK CAMPUS

THESIS NO: 069/MSCS/659
(FINAL REPORT)

PDTI-DBN: PREDICTING DRUG AND TARGET INTERACTION USING DEEP BELIEF
NETWORK

BY
OM PRAKASH MAHATO

A FINAL REPORT OF THESIS
SUBMITTED TO DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING,
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF M.Sc.
PROGRAM IN COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL
OCTOBER, 2016

PDTI-DBN: PREDICTING DRUG AND TARGET INTERACTION USING DEEP BELIEF
NETWORK

BY
OM PRAKASH MAHATO
ROLL NO: -069/MSCS/659

SUPERVISED BY
DR. AMAN SHAKYA

A FINAL REPORT OF THESIS
SUBMITTED TO DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING,
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF M.Sc.
PROGRAM IN COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL
OCTOBER, 2016

COPYRIGHT

The author has agreed that the library, Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering may make this report freely available for inspection. Moreover, the author has agreed that permission for extensive copying of this project report for scholarly purpose may be granted by the professors(s) who supervised the project work recorded herein or, in their absence, by the Head of the Department wherein the project report was done. It is understood that the recognition will be given to authors of this report and to the Department of Electronics and Computer Engineering, Pulchowk Campus, and Institute of Engineering in any use of the material in this project report. Copying or publication or the other use of this report for financial gain without approval of the Department of Electronics and Computer Engineering, Pulchowk Campus, Institute of Engineering and author's written permission is prohibited. Request for permission to copy or to make any other use of the material in this report in whole or in part should be addressed to:

Head

Department of Electronics and Computer Engineering

Pulchowk Campus, Institute of Engineering

Lalitpur, Kathmandu

Nepal

RECOMMENDATION

The undersigned certify that they have read and recommended to the Department of Electronics and Computer Engineering for acceptance, a thesis entitled “PDTI-DBN: PREDICTING DRUG AND TARGET INTERACTION USING DEEP BELIEF NETWORK”, submitted by Mr. OM PRAKSH MAHATO in partial fulfillment of the requirement for the award of the degree of “Master of Science in COMPUTER SYSTEM AND KNOWLEDGE ENGINEERING”.

DR. AMAN SHAKYA
SUPERVISOR

EXTERNAL EXAMINER

ABSTRACT

These days, virtual screening is applied in systematic drug design process which reduces cost and time for drug discovery. Virtual screening is soft computing technique being used for docking ligands from huge databases in selecting protein-receptor, targeting to correct drug. Due to non-linear nature of big-biological data, it is difficult to classify dockable and non-dockable ligands. Therefore, a machine learning method is used to train the classifier for separating intractable drug-target pair. However, the existing machine learning approaches have their several limitations on recent non-linear feature space of biological data. These days, deep learning approaches show advantages over many state-of-the art machine learning methods in complex applications. So, in this thesis, a new approach called **PDTI-DBN** framework was proposed to predict the interaction between drug and targets efficiently. The DBN (Deep Belief Network) is used to extract the high level features from 2D chemical substructure represented in fingerprint format. DBN, Stack of Restricted Boltzmann Machines is being trained by a greedy layer-wise unsupervised fashion and the result from this pre-training phase is used to initialize the parameters prior to Back-propagation (BP) used for fine tuning. The fine-tuning phase is composed by Multi Layer Perception (MLP) which shares all forward weights with RBMs. Similarly, logistic regression layer is staked as output layer. Then it is fine-tuned using BP of error derivative to build classification model that directly predict whether a drug interact with a target of interest. Based on evaluation on gold standard data, it is shown that this DBN model improves the throughput by five folds with around 99% accuracy for drug and target interaction prediction and its maximum F1-score obtained 73% with good AUC value from ROC curve.

Keywords: Predicting drug-target interaction, Virtual Screening, Deep Belief Network

ACKNOWLEDGEMENT

First of all, I would like to express my surrender and dependence onto Lord Krishna for supporting and inspiring from inside and outside to complete this thesis. I feel deep sense of gratitude to my supervisor Dr. Aman Shakya for his great help and advices during thesis period. I would like to thank to Dr. Sanjeeb Prasad Pandey, M.Sc. Program Coordinator, Department of Computer System and Knowledge Engineering, Institute of Engineering for the precious and inspiring guidance, suggestions, support and encouragement required throughout the thesis. I can't forget Professor Dr. Subarna Shakya and Professor Dr. Shashidhar Ram Joshi for encouraging me to complete my thesis as soon as possible. I would like to thank Dr. Diwakar Panta ,HOD Department of Electronics and Computer Engineering, Pulchowk Campus, for providing generous support and adequate physical facilities during the thesis work.

I am grateful to all those who have directly or indirectly contributed during the study. I would like to thank my friends for their encouragement, support and help during the research work, especially Mr. Md. Hasan Ansari From Nepal Telecom, Mrs Alka Jha, Mrs Sumitra Lama, Mr. Sajann Kushwaha , Mr. Manoj Kumar Singh form NEA for their valuable suggestions and co-operation in my research work. Finally, I don't have adequate word to express my indebtedness to my family for their love, support and encouragement.

Om Prakash Mahato

October, 2016

TABLE OF CONTENTS

COPYRIGHT.....	iii
RECOMMENDATION	iv
ABSTRACT.....	v
ACKNOWLEDGEMENT	vi
TABLE OF CONTENTS.....	vii
LIST OF ACRONYMS, SYMBOLS AND ABBREVIATIONS.....	ix
LIST OF FIGURES	x
LIST OF TABLE	xi
CHAPTER 1: INTRODUCTION.....	1
1.1 Background	1
1.2 Problem Statement	5
1.3 Objective of the study	5
1.4 Scope and Limitations.....	6
1.5 Organization of the Thesis	6
CHAPTER 2: LITERATURE REVIEW	7
CHAPTER 3: METHODOLOGY	10
3.1 System Methodology Overview.....	10
_3.1.1 Data Collection	10
_3.1.2 System Flow Diagram.....	16
_3.1.3 Deep Learning.....	19
_3.1.4 Experimental Design.....	33
_3.1.5 Evaluation Indices.....	33
_3.1.6 Predicting Drug-Target Interaction Example.....	35

3.2	System Design Overview	51
_3.2.1	Flowchart	51
_3.2.2	Dimension Reduction.....	53
_3.2.3	Learning	54
_3.2.4	Modules Developed	55
CHAPTER 4:	RESULT AND EVALUATION	56
4.1	Results	56
4.2	Evaluation.....	57
_4.2.1	Using threshold 0.5	58
_4.2.2	Using threshold 0.35, 0.45, 0.5 and 0.75	58
4.3	Result Summary and Comparison with existing prediction methods	59
_4.3.1	Result Summary.....	59
_4.3.2	Comparison with Existing Prediction Methods	62
CHAPTER 5:	CONCLUSION AND FUTURE WORK	63
5.1	Conclusion.....	63
5.2	Future Work	63
REFERENCES:	64
APPENDIX A	66
APPENDIX B	67

LIST OF ACRONYMS, SYMBOLS AND ABBREVIATIONS

ANN	Artificial Neural Network
BP	Back Propagation
DBN	Deep Belief Network
ML	Machine Learning
RBM	Restricted Boltzmann Machine
RF	Random Forest
SVM	Support Vector Machine

LIST OF FIGURES

Figure 1.1: Drug Discovery Pipeline	1
Figure 1.2: Crystal structure of protein in complex with drug	2
Figure 1.3: A typical virtual screening protocol	3
Figure 3.1: System Methodology Overview	10
Figure 3.2: Representation of a molecular substructure as fingerprint	13
Figure 3.3: Representation for substructure feature in fingerprint format	14
Figure 3.4: Screenshot of drug and substructure relationship fingerprint file	14
Figure 3.5: Screenshot of drug and target (protein) interaction file	15
Figure 3.6: System overview for Drug target interaction (PDTI-DBN) prediction Model.	16
Figure 3.7: Pre-training and Fine-tuning process Diagram	17
Figure 3.8: Restricted Boltzmann Machine (RBM).....	20
Figure 3.9: Contrastive Divergence (CD-k).....	26
Figure 3.10: (a) and (b) The graphical model representations of an RBM and a DBN.....	28
Figure 3.11: Pre-training and Fine-tuning of stack of RBMs	31
Figure 3.12: Confusion matrix for performance evaluation	33
Figure 3.13: Positive hidden sampling of RBM0	39
Figure 3.14: Negative visible sampling of RBM0	40
Figure 3.15: Negative hidden sampling of RBM0.....	41
Figure 3.16: Positive hidden sampling of RBM1	42
Figure 3.17: Negative visible sampling	43
Figure 3.18: Negative hidden sampling	44
Figure 3.19: Pre-training: Forward layer wise calculation	46
Figure 3.20: Fine-tuning: Layer wise calculation	47
Figure 3.21: Flowchart Development for PDTI-DBN framework	53
Figure 4.1: Bar chart graph of drug-target interaction prediction result.....	59
Figure 4.2: ROC curve of Five Experiment's average result.....	60
Figure 4.3: Time versus minibatch graph of 881 features and 286 features pre-training	61
Figure 4.4: Bar chart of prediction result in terms of Sn, Pre, Spc, Acc and F1	62

LIST OF TABLE

Table 3.1: Drug, its structure and interacting Targets [DrugBank [27] and UniProt[5]]	35
Table 3.2: Drug and Sub-Structure Fingerprint Sample	36
Table 3.3: Drug and Target interaction Sample	36
Table 3.4: Initialization of W_{ij} of RBM0	37
Table 3.5: Initialization of h_i of RBM0	37
Table 3.6: Initialization of b_j of RBM0	37
Table 3.7: Initialization of W_{ij} of RBM1	38
Table 3.8: Initialization of h_i of RBM1	38
Table 3.9: Initialization of W_{ij} of Logistic Layer	38
Table 3.10: Initialization of b_j of Logistic Layer	38
Table 3.11: Input feature of Nicotine Drug	39
Table 3.12: positive hidden mean of RBM0	39
Table 3.13: Positive hidden sample of RBM0	40
Table 3.14: Pegative visible mean output of RBM0	40
Table 3.15: Negative visible sample output of RBM0	40
Table 3.16: Negative hidden mean output of RBM0	41
Table 3.17: Negative hidden sample output of RBM0	41
Table 3.18: modified weight (W_{ij}) of RBM0	41
Table 3.19 modified h-bias (h_i) of RBM0	42
Table 3.20: modified v-bias (b_j) of RBM0	42
Table 3.21: Output of RBM0 as input to RBM1	42
Table 3.22: Positive hidden mean of RBM1	43
Table 3.23: Positive hidden sample of RBM1	43
Table 3.24: Negative visible mean output of RBM1	43
Table 3.25: Negative visible sample output of RBM1	43
Table 3.26: Negative hidden mean output	44
Table 3.27: Negative hidden sample output	44
Table 3.28: modified weight (W_{ij}) in Rbm1	44
Table 3.29: modified h-bias h_i in Rbm1	45
Table 3.30: Modified v-bias v_j in Rbm1	45

Table 3.31: Fine-tuning: input to RBM0	45
Table 3.32: Fine-tuning: output of RBM0	45
Table 3.33: Fine-tuning: output of RBM0 as input to RBM0.....	46
Table 3.34: Fine-tuning: output of RBM1	46
Table 3.35: Output given input in logistic layer before softmax function.....	46
Table 3.36: Output given input in logistic layer after softmax function.....	47
Table 3.37: Nicotine interacting with target t0 and t1 output	47
Table 3.38: calculated error in output layer	47
Table 3.39: Updated weight in logistic layer	48
Table 3.40: Updated bias in logistic layer	48
Table 3.41: Δ error calculation in RBM1.....	48
Table 3.42: Fine-tuning: Updated weight in RBM1 due to back propagation.....	48
Table 3.43: Fine-tuning: Updated bias in RBM1 due to back propagation	49
Table 3.44: Δ error calculation.. By error propagating from layer RBM1 to layer RBM0	49
Table 3.45: Fine-tuning: Updated weight in RBM0 due to back propagation.....	49
Table 3.46: Fine-tuning: Updated bias in RBM0 due to back propagation	49
Table 3.47: Testing Nicotine Drug sub-structure feature input	50
Table 3.48: Predicted output of Nicotine drug with targets after first iteration training	50
Table 3.49: Actual Nicotine Drug interaction with targets.....	50
Table 3.50: Predicted output for Nicotine drug with proteins	50
Table 3.51: Learning Parameter Dimension	54
Table 3.52: List of implemented modules	55
Table 4.1: hyper parameter for experiment1.....	56
Table 4.2 : Predicted Output table (row: drug and column UniProt Protein ID).....	56
Table 4.3: hyper-parameter for experiment 2 and 3.	56
Table 4.4: Experiment wise output of PDTI-DBN framework.....	57
Table 4.5: Prediction outputs of Experiment 1, 2, 3, 4 and 5.	57
Table 4.6: Sensitivity and Specificity of from predicted output with multipoint threshold	58
Table 4.7: Average of five experiment results from table	60

CHAPTER 1: INTRODUCTION

1.1 Background

In drug design and discovery, diverse computational chemistry approaches are used to calculate and predict drug binding to its target and the chemical properties for designing potential new drug. Protein-Ligands binding affinity is the principal component of many vital processes, such as cellular signaling, gene regulation and metabolism, which depend upon proteins binding to some substrate molecule. As shown in figure 1.1, For Drug discovery, there are three phases 1. HIT identification 2. HIT to LEAD formation and 3. LEAD optimization. HIT identification is first stage of drug discovery where computation algorithm can be applied called virtual screening for finding drug–target binding (as shown in figure 1.2) [1].

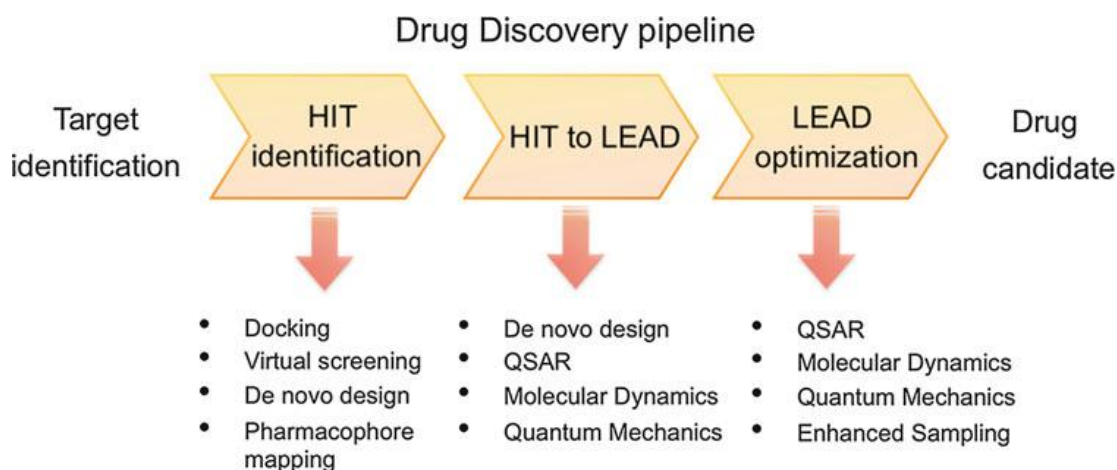


Figure 1: Drug Discovery Pipeline

Initially High throughput Screening (HTS) was being done for automation facility. However, due to the large number of ligands that need to be screened, HTS is not fast and cost effective enough as a lead identification method in the initial phases of drug discovery [2]. Drug development currently remains an expensive and time-consuming process with extremely low success rate: it typically takes 10–15 years and \$800 million–1 billion to bring a new drug to market [3]. In

recent decades, the rate of the number of new drugs approved by the US Food and Drug Administration versus the amount of money invested in pharmaceutical research and development has significantly declined [4].

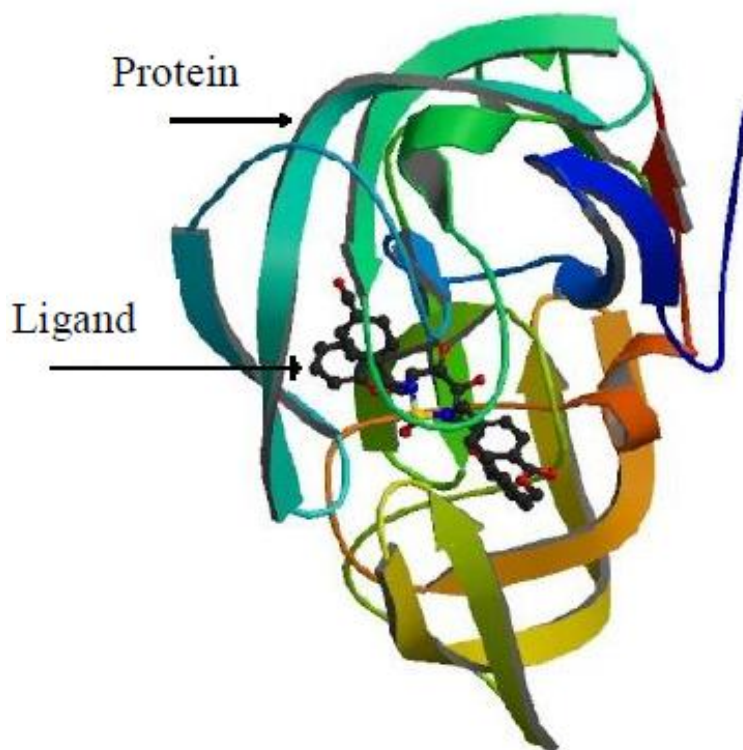


Figure 2: Crystal structure of protein in complex with drug

Therefore, as seen figure 1.3, a computational methods referred to as virtual screening (Computer-aided design, In Silico) are employed to complement HTS by narrowing down the number of ligands to be physically screened, which reduces the time and cost of laboratory experiment (In vitro) . In virtual screening, information such as structure and physicochemical properties of a ligand, protein, or both, are used to estimate binding affinity (or binding free energy), which represents the strength of association between the ligand (drug or molecule) and its receptor protein (target). The most popular approach to predicting binding affinity in virtual

screening is structure-based in which physicochemical interactions between a ligand and receptor are deduced from the 3D structures of both molecules [5, 6].

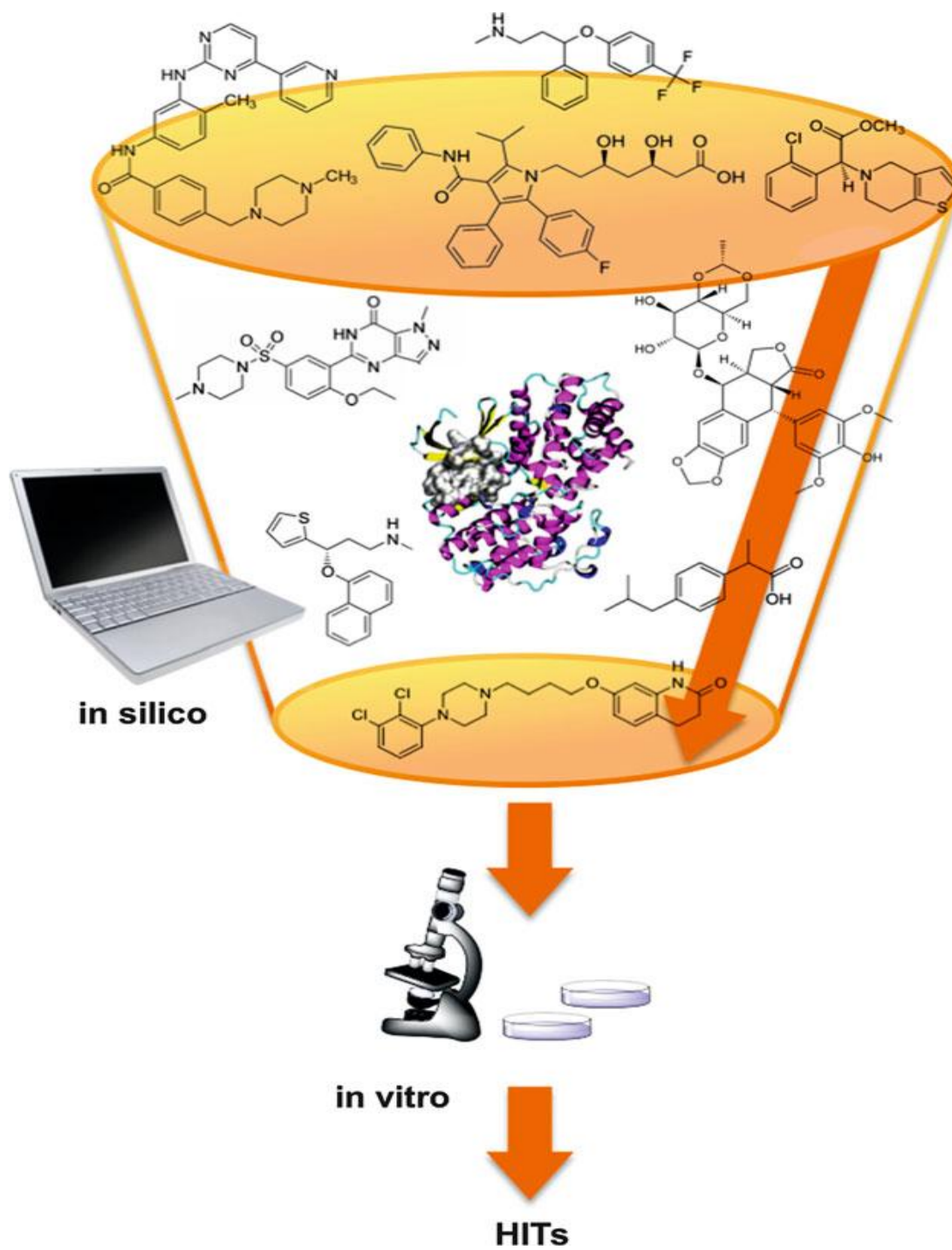


Figure 3: A typical virtual screening protocol

In network pharmacology [7], the assumption of "one drug for one target for one disease" (on which traditional drug discovery is based) is challenged and the relationships between drugs and targets become complicated. One drug may act on multiple targets while there are also proteins that are targeted by two or more compounds. Therefore, In Silico Prediction, identification of the interactions between chemical compounds and proteins plays a critical role in network pharmacology, drug discovery, drug target identification, elucidation of protein functions, and drug repositioning [7].

A strong support for the possibility of drug repositioning is the increasingly accepted concept of 'polypharmacology', i.e. individual drugs can interact with multiple targets rather than a single target. For Example, Drug Nicotine (DrugBank ID DB00184) interacts with 10 targets (UniProt ID, P17787, P30532, P30926, P32297, P43681, Q05901, Q15822, Q15825, Q9G226 and Q9UGM1) [8]. There is a challenge to reduce the side-effect as one drug interacts more than single targets. Therefore, a new approach for drug discovery is needed to minimize the side-effects as well as maximizing the protein-ligand affinity that is indentifying the possible all drug-target interactions.

In particular, machine-learning based methods, which have been successfully applied to various prediction problems in biology, have the potential to effectively learn the relationships among compounds and target proteins to predict new drug-target interactions from the viewpoint of chemo genomics. In this thesis, a framework of ligand based virtual screening using Deep Belief Networks (DBN) is proposed. Employing multiple hidden layers and training the layers using both supervised and unsupervised learning creates a deep learning (belief) network [9].

Deep learning networks are one of the latest and most powerful machine learning techniques for pattern recognition. DBN can be routinely applied to 1000 data sets of compound sub structure 2D fingerprint compound 881 dimensional descriptors without and with feature reduction fashion. For reducing the feature dimension, only the sub structure feature which are present in more than 1000 compounds are selected. By doing so, feature dimension is reduced to 286 features which is advantageous for time complexity. There are two phase trainings in DBN, one for pre-training phase which is in unsupervised learning based on only unlabeled 2D substructure fingerprint data. The pre-training phase extracts the low level sub structure features of drugs (compound) which initializes the weights for post supervised training. In pre-training, stack of RBM (Restricted Boltzmann machine) with greedy layer wise algorithm is applied in 3 to 4

hidden layers having 2000 neurons in each hidden layer. The second training phase is supervised training which is employed for fine tuning using known drug and target dataset of the same drugs which are being used in pre-training phase. Back propagation method is used for fine tuning purpose. Also, logistic regression is stacked as output layer. Finally, different set of drugs are being taken for cross validation the model.

1.2 Problem Statement

- It is difficult to predict the interaction between new drugs and targets due to very low known drug-target interactions available.
- It is challenge to optimize the drug design process that is there is no such a model which shows maximizing confidential in drug-target interaction with lowest side-effect provision.
- The exiting applied machine learning approaches have several limitations due non-linear nature of biological data. For example, recently famous machine learning approach like Support vector machine has shallow architecture and also it does not capture nonlinear relationship among the exiting features.

1.3 Objective of the study

Based on the above background and concept, the objective of this thesis to build a DTI-DBN framework which fulfills the following sub-objectives:

- To build PDTI-DBN framework using stack of RBM and a logistic layer at output layer.
- To evaluate the model by applying cross-validation technique with gold standard dataset with prediction of new drug-target interaction.

1.4 Scope and Limitations

This thesis work considers following scope and limitations.

Scope of Thesis:

- It is only feature based prediction applying Deep learning with Deep Belief Network (DBN) with fixed learning rate , drop out and momentum.
- Only drug compound 2D substructure finger print is used as pre- training dataset.
- Evaluation with several prediction evaluation approaches like ROC curve has been employed.

Limitations of Thesis:

- Adaptation learning rate has not been employed.
- Protein sequence feature domain profile has not been used in training purpose.

1.5 Organization of the Thesis

This report consists of the following five chapters

Chapter 1 is an introductory chapter which includes background of the study, literature review related to the research, problem statement, objectives of thesis and scope and limitations of the work.

Chapter 2 presents the general overview of the Deep Learning, its Concept and Components. It describes about Restricted Boltzmann Machine (RBM) and Contrastive Divergence for its training strategy. Also, The Deep Belief Network is described by constructing a stack of RBM.

Chapter 3 presents details of Methodology used for Drug and Target Interaction. It contains Data summary, system overview with a block diagram. Also, Experimental Setup and System specification are tabulated in this chapter. Finally, one complete example from training to prediction is illustrated for a pair of drug and target prediction.

Chapter 4 describes about the model evaluating guides and then it is discussed about the result obtained from this thesis on drug target interaction prediction.

Chapter 5 lists the references used for this thesis completion and the future enhanced of the thesis.

CHAPTER 2: LITERATURE REVIEW

A number of network-based approaches have been proposed for predicting unknown interactions between drugs and targets. A supervised learning framework was developed based on a bipartite graph, which integrates both chemical and genomic spaces by mapping them into a unified space. KRM (kernel regression-based method) was used to interaction prediction between drug and target [10]. Again, it was presented a new approach, called Bipartite Local Model (BLM), to predict unknown DTIs by combining independent drug-based and target-based prediction results using a supervised learning method called SVM (Support Vector Machine) [11]. Also, they combined KRM and SVM latter. However, for drug-candidate compounds or target-candidate proteins that currently has no sufficient known interactions available. BLM was extended as BLM-NII integrating neighbor-based interaction profile. It was shown that nuclear receptors interaction prediction worked well [12].

Also, bipartite graph had difficulty for predicting interaction between new drugs or new target for which there are no known interaction. It was proposed two matrix factorization methods using graph regularization [13]. Similarly, a first machine learning method was used starting from a DTI network to predict new ones with high accuracy. The calculation of the new interactions was done through the regularized least squares algorithm. The regularized least squares algorithm is trained using a kernel (GIP—Gaussian interaction profile) that summarizes the information in the network [14].

On the same data set (profiles) of similarity matrices (kernels) of drug and target proteins user were applied with supervised network inference algorithm added more heterogeneous biological data such as chemical structure, drug side effects, amino acid sequences and protein domains. It is web server model which user can use known interaction data or own interaction data[15]. However, Network-based approaches can only predict binary interaction drug and target pair. This is overcome by incorporating additional information about drug and target interaction. Restricted Boltzmann machine was used in their experiment for time in DTI for better performance in unsupervised learning fashion [16].

Also, it was proposed such a prediction model as state of art performance method which used probabilistic soft logic (PSL) on online data sets to overcome drug's unexpected therapeutic or adverse side effects. However, evaluating interactions locally (i.e. ignoring substructures or domains), false positive estimation gets increased [17]. Likewise, GIFT model was proposed which evaluates the drug–target interaction globally. EM (Expectation maximization) algorithm was used for this model [18].

For minimizing cost and risk, drug company used to design a “follow –on” drug for the targets having more drugs already discovered. It was proposed similarity-rank-based predictor (SRP) which takes evidence from “follow-on” drug observations. SRP is non-parametric as well as non statistical knowledge based model [19].

To overcome multi-target drugs and multi-drug targets nature, it was created a semi supervised-based learning framework called NormMullnf using collaborative filtering theory. Both labeled and unlabeled interaction information integrating with similarity information were used for principal component analysis in NormMullnf model [20]. Similarly, recommended system with collaborative filtering was proposed which uses both interaction matrix for drug–target and rating matrix of user-item. Also, evaluation was perform on four class of proteins effectively based on known interaction matrix [21]. However, Recommendation method (RM) does not take account important features in drug- target domain as it relies on Network-based inference (NBI). Again, it was presented new NBI method called DT-Hybrid which extended RM by adding drug and target similarity knowledge in tuned way [22].

Drug–target interaction depends on the factors which are based on the structural and physicochemical properties. Functional group or fragments are represented by the drug structure feature fingerprint. This fingerprint is used for training SVM for establishing drug-target interaction prediction model[23]. The model extended the structure-activity relationship methodology. To increase the performance, it was proposed a combination of support-vector machine and Random forest for solving DTI classification complexity. The model is based on feature of drug sub-structure fingerprint as well as physicochemical properties of the protein[24]. Similarly, it was presented Bi-gram PSSM model which is based on bi-gram features extracted

from the position Specific method. (PSSM) they show improved performance specifically for enzymes and ion-channels. They have used Support Vector Machine [SVM] for training purpose. However, they noted that the number of non-interacting drug–target pairs are usually extremely large in comparison with the number of interacting ones in existing drug–target interaction data so, performance reduced [25].

Again, Dockable and non-dockable ligands classification is improved [26]. They used three ML techniques (Support vector machine, Artificial Neural Networks and Random Forest) on a single problem domain (a Protease receptor of HIV). However, there are practical issues (i.e. slow on large problems, difficult to train, prone to over-fitting etc) occurring in methods like support vector method and random forest.

To overcome above issues, deep learning is being used increasable way in this decade. This improved the training algorithm, prevented over-fitting problems with advanced in computer hardware. It was proved that DNN was better predicted than RF on large diverse QSAR data sets. (still computational intensive) [27]. Similarly, it was presented DL-CPI (the abbreviation of Deep Learning for Compound-Protein Interactions prediction) prediction method is based on deep learning which shows over many state-of-the-machine learning method. DL-CPI employs deep neural network (DNN) to effectively learn the representations of compound-protein pairs. It overcomes limitations of other machine learning due to the non-linear and imbalanced nature of biological data [16].

CHAPTER 3: METHODOLOGY

3.1 System Methodology Overview

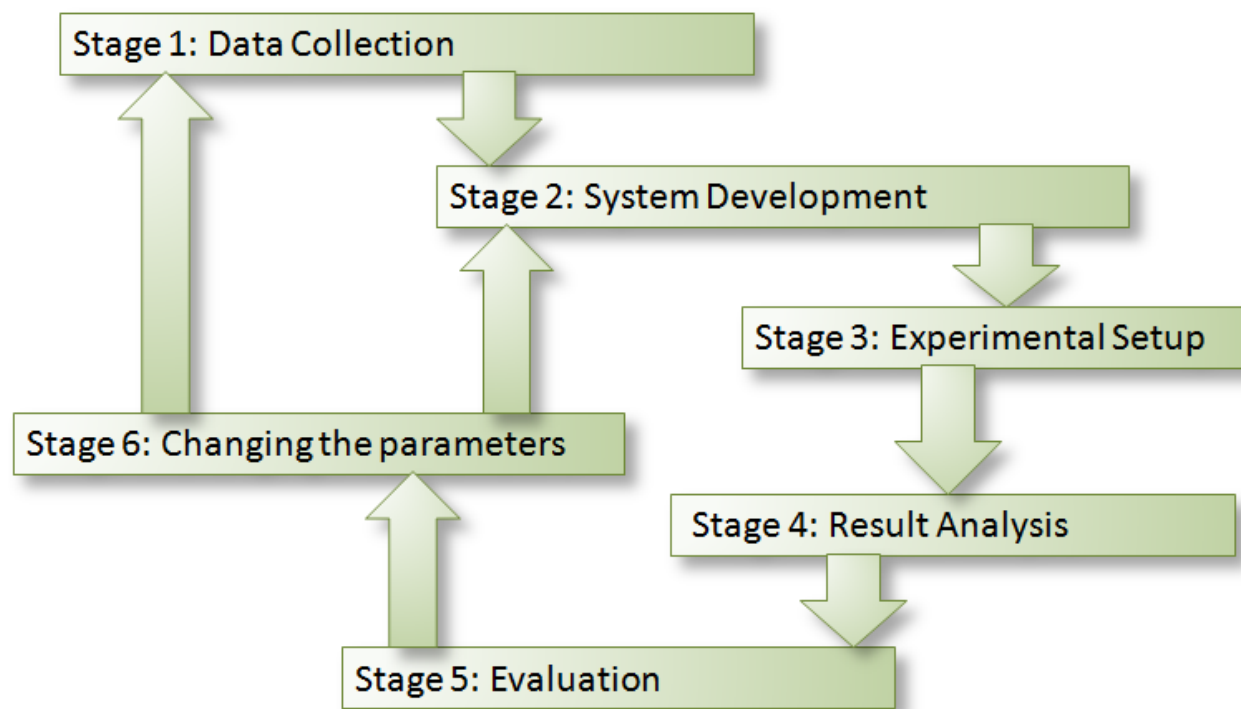


Figure 4: System Methodology Overview

As shown in figure 3.1, System Methodology goes are six stages: data collection, System Development, Experimental Setup, Result Analysis, Evaluation and Comparison and Changing Hyper-parameters. The details of each stage are explained in following sub-sections.

3.1.1 Data Collection

The PubChem System generates a binary substructure fingerprint for chemical structures. These fingerprints are used by PubChem for similarity neighboring and similarity searching.

A substructure is a fragment of a chemical structure. A fingerprint is an ordered list of binary (1/0) bits. Each bit represents a Boolean determination of, or test for, the presence of, for

example, an element count, a type of ring system, atom pairing, atom environment (nearest neighbors), etc., in a chemical structure. The native format of the PubChem Substructure.

Fingerprint property is binary data with a four byte integer prefix, where this integer prefix indicates the length of the bit list. For the ASN.1 and XML formatted data, this property is stored in a PC-InfoData container, as described by the PCSubstance ASN.1 definition or XML schema: <ftp://ftp.ncbi.nlm.nih.gov/pubchem/specifications/>

For each compound, it is used basic substructures as features, and constructed a fingerprint (a binary vector where "1" indicates the presence of a certain feature) of features to represent the compound. The fingerprints were obtained from the PubChem database [28], and each compound is represented as a 881-dimension binary vector. The feature vector composed of different seven sections which are described in below.

Section 1: Hierarchic Element Counts - These bits test for the presence or count of individual chemical atoms represented by their atomic symbol. (it runs from bit position 0 to 114)

Bit Position	Bit Substructure
0	≥ 4 H
1	1 ≥ 8 H
2	2 ≥ 16 H

Section 2: Rings in a canonic Extended Smallest Set of Smallest Rings (ESSSR) ring set - These bits test for the presence or count of the described chemical ring system. An ESSSR ring is any ring which does not share three consecutive atoms with any other ring in the chemical structure. For example, naphthalene has three ESSSR rings (two phenyl fragments and the 10-membered envelope), while biphenyl will yield a count of only two ESSSR rings.(Bit position 115 to 262)

Bit Position	Bit Substructure
115	≥ 1 any ring size 3
116	≥ 1 saturated or aromatic carbon-only ring size 3
117	≥ 1 saturated or aromatic nitrogen-containing ring size 3

Section 3: Simple atom pairs – These bits test for the presence of patterns of bonded atom pairs, regardless of bond order or count.(Bit position 263 to 326)

Bit Position	Bit Substructure
263	Li-H
64	Li-Li

Section 4: Simple atom nearest neighbors – These bits test for the presence of atom nearest neighbor patterns, regardless of bond order (denoted by "~") or count, but where bond aromaticity (denoted by ":") is significant.(Bit position 327 to 415)

Bit Position	Bit Substructure
327	C(~Br) (~C)
328	C(~Br) (~C) (~C)
329	C(~Br) (~H)

Section 5: Detailed atom neighborhoods – These bits test for the presence of detailed atom neighborhood patterns, regardless of count, but where bond orders are specific, bond aromaticity matches both single and double bonds, and where "-", "=", and "#" matches a single bond, double bond, and triple bond order, respectively.(Bit 416 to 459)

Bit Position	Bit Substructure
416	C=C
417	C#C

Section 6: Simple SMARTS patterns – These bits test for the presence of simple SMARTS patterns, regardless of count, but where bond orders are specific and bond aromaticity matches both single and double bonds.(Bit 460 to 712)

Bit Position	Bit Substructure
460	C-C-C#C
461	O-C-C=N

Section 7: Complex SMARTS patterns – These bits test for the presence of complex SMARTS patterns, regardless of count, but where bond orders and bond aromaticity are specific.(Bit 713 to 880)

Bit Position	Bit Substructure
713	Cc1ccc(C)cc1
714	Cc1ccc(O)cc1

Example: By means of above descriptors, each molecule can be described based on a set of fingerprints of structural keys, which is represented as a Boolean array. There is one-to-one correspondence between each pattern and bit in the finger print. For each pattern, if its corresponding substructure is present in the given molecule, the corresponding bit in the fingerprint is set to 1; conversely, it is set to 0 if the substructure is absent in the molecule. Two examples of substructure fingerprints generated with above descriptors are shown in Figure 3.2 and Figure 3.3.

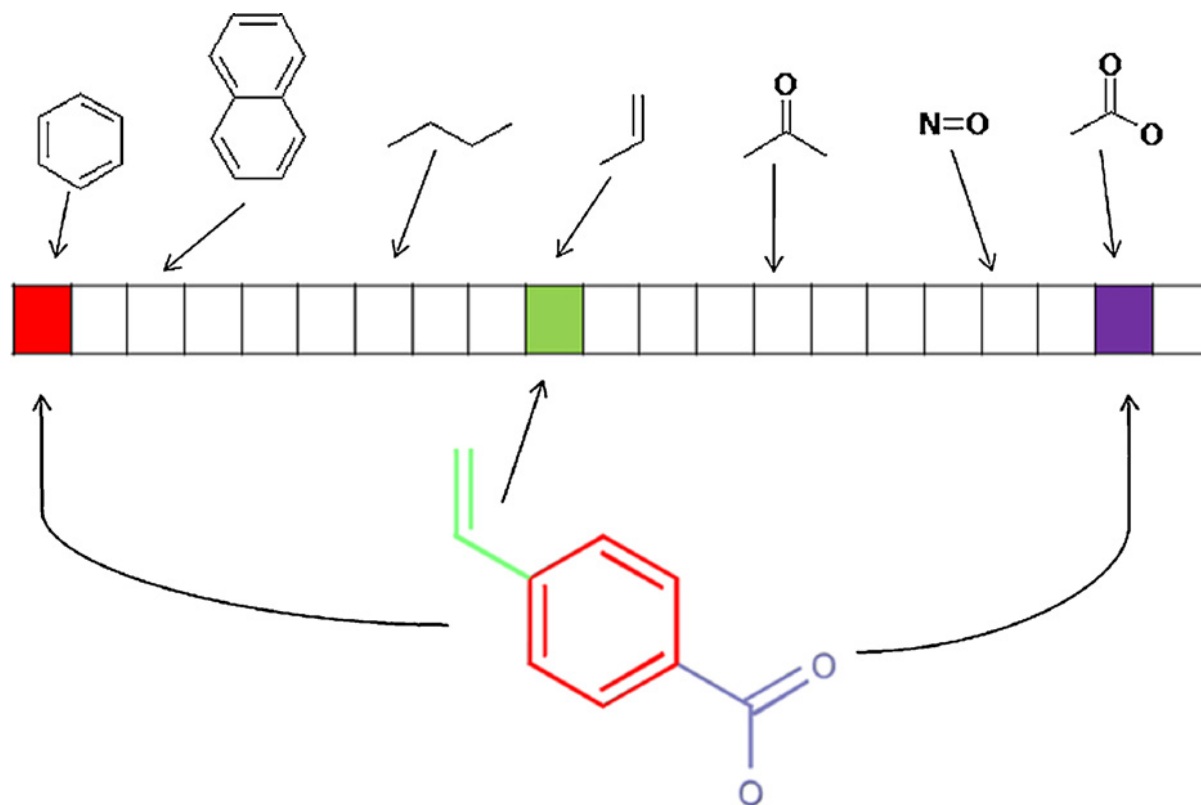


Figure 5: Representation of a molecular substructure as fingerprint

Summary of data sources:

- **Chemical data source:** Drug and PubChem chemical substructures relationships:

It includes 122,022 associations between 1007 drugs and 881 PubChem chemical substructures. The descriptions of the 881 chemical substructures are represented in a file as shown in figure 3.4.

- **Protein data source:** Drug and UniProt [8] target proteins relationships

It includes 3,152 associations between 1007 drugs and 775 target proteins. This has been generated from targets of DrugBank [29] and is represented in a matrix as shown in figure 3.5.

Above data can be downloaded from the URL:

<http://astro.temple.edu/~tua87106/drugreposition.html>

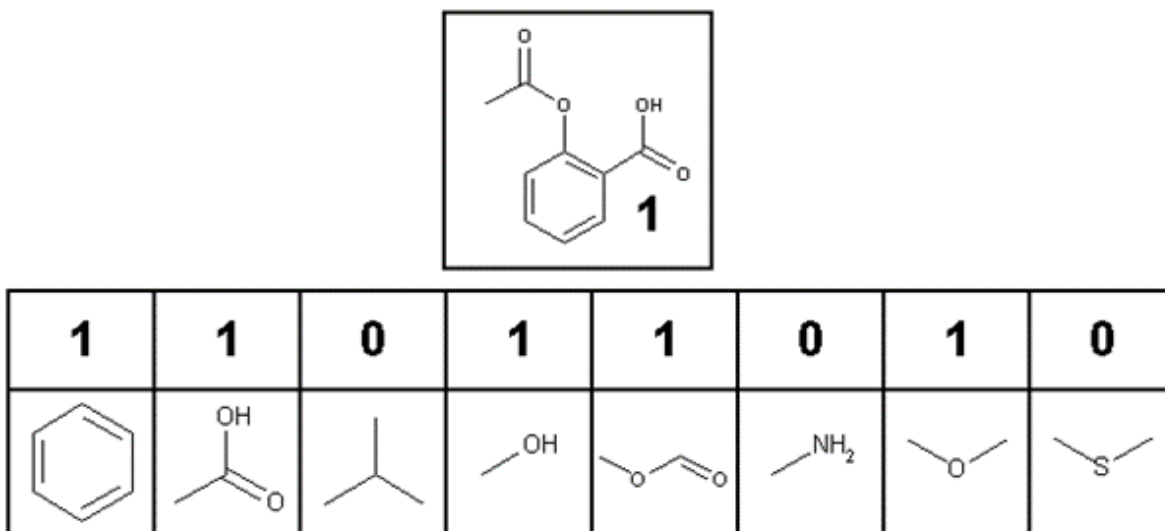


Figure 6: Representation for substructure feature in fingerprint format.

	A	B	C	D	E	F	G	H
1		SUB 0	SUB 1	SUB 2	SUB 3	SUB 4	SUB 5	SUB 6
2	Pravastatin	1	1	0	0	0	0	0
3	Fluvoxamine	0	0	0	0	0	0	0
4	Valsartan	0	0	0	0	0	0	0
5	Ramipril	1	0	0	0	0	0	0
6	Masoprocol	0	0	0	0	0	0	0
7	Flunisolide	1	0	0	0	0	0	0
8	Baclofen	0	0	0	0	0	0	0
9	Pentagastrin	1	0	0	0	0	0	0
10	Nicotine	0	0	0	0	0	0	0
11	Cevimeline	0	0	0	0	0	0	0
12	Lorazepam	0	0	0	0	0	0	0

Figure 7: Screenshot of drug and substructure relationship fingerprint file

Each row represents approved drugs by DrugBank and each column represents particular substructure presence or absence in terms of binary 1 or 0 respectively. There are total 1007 drugs and 881 substructures in the fingerprint file named as drug_structure.csv. As in above figure, all the rows having Pravastatin, Fluvoxamine, valsatan, Ramipril terms etc in first columns are drug names whereas all the columns except first having SUB0, SUB1, SUB2 etc are substructure features indices. As already described SUB0 is bit 0 substructure, SUB1 is bit 1

substructure and so on. The drug Pravastatin belongs to bit 0 and bit 1 position substructure features so that 1 is kept the corresponding cells where as 0 for other cells.

	A	B	C	D	E	F	G	H
1		A2A2V4	A8MPY1	A9UF02	B8DCL9	B8DD61	C1KC03	O00180
2	Pravastatin	0	0	0	0	0	0	0
3	Fluvoxamine	0	0	0	0	0	0	0
4	valsartan	0	0	0	0	0	0	0
5	Ramipril	0	0	0	0	0	0	0
6	Masoprocol	0	0	0	0	0	0	0
7	flunisolide	0	0	0	0	0	0	0
8	Baclofen	0	0	0	0	0	0	0
9	Pentagastrin	0	0	0	0	0	0	0
10	Nicotine	0	0	0	0	0	0	0
11	cevimeline	0	0	0	0	0	0	0
12	Lorazepam	0	1	0	0	0	0	0

Figure 8: Screenshot of drug and target (protein) interaction file

Here rows represent drug names whereas column represents proteins represented by UniProtKB ID. There are total 1007 drugs and 775 proteins in the file. Each cell in file represents the interaction of corresponding drug and protein. For example, Lorazepam drug is associated with A82V4 so 1 is present in the corresponding cell of Lorazepam drug and A2A2V4 protein, whereas others cells contain 0 that is no association between corresponding drugs and proteins. For paravastin is not linked with A2A2V4 protein so there is 0 in its interaction cell. And so on for others.

3.1.2 System Flow Diagram

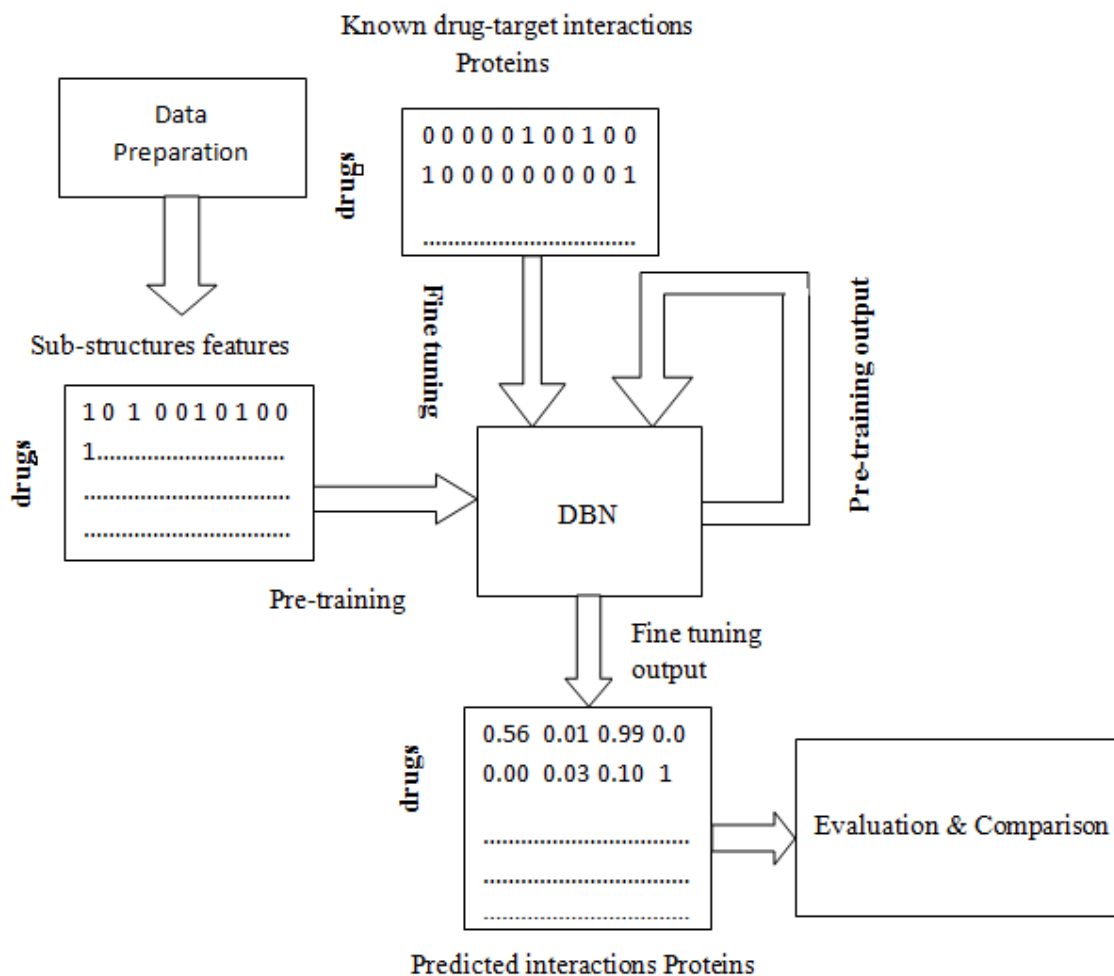


Figure 9: System overview for Drug target interaction (PDTI-DBN) prediction Model.

As shown in figure 3.6 , the system consists of pre-training , fine tuning and evaluation and comprasion sections. In pre-pretraining section, there is data input that it drug sub-structure finger print sequence as explained in dataset section in 3.1.1. Each drug sequence in binary code is fed the input layer of the deep belief Network(DBN) where first layer RBM takes it as visible units. There are 881 dimentional feature vecotor in each drug. So that the number of inputs of the First RBM as visible units are 881. DBN is stack of 3 hidden layer of RBM and one logistic layer at output layer. First RBM samples the each drug suing equation 3.2.4 and sampled data is agained back to to visible units for re-sampling at visiblle units using equation 3.2.5. After doing that, the paramers (weight between visible and hidden, visible uni bias and hidden unit

bias) are adjusted using equation 3.3.7, 3.3.10 and 3.3.11 accordingly. The output of the first RBM is representation of feature detected from the drug substructure fingerprint in probabilistic way. That the RBM is being trained by CD-1 algorithm as explained in section 3.1.3 ‘training RBM’ in chapter 3. First sampled output from the first

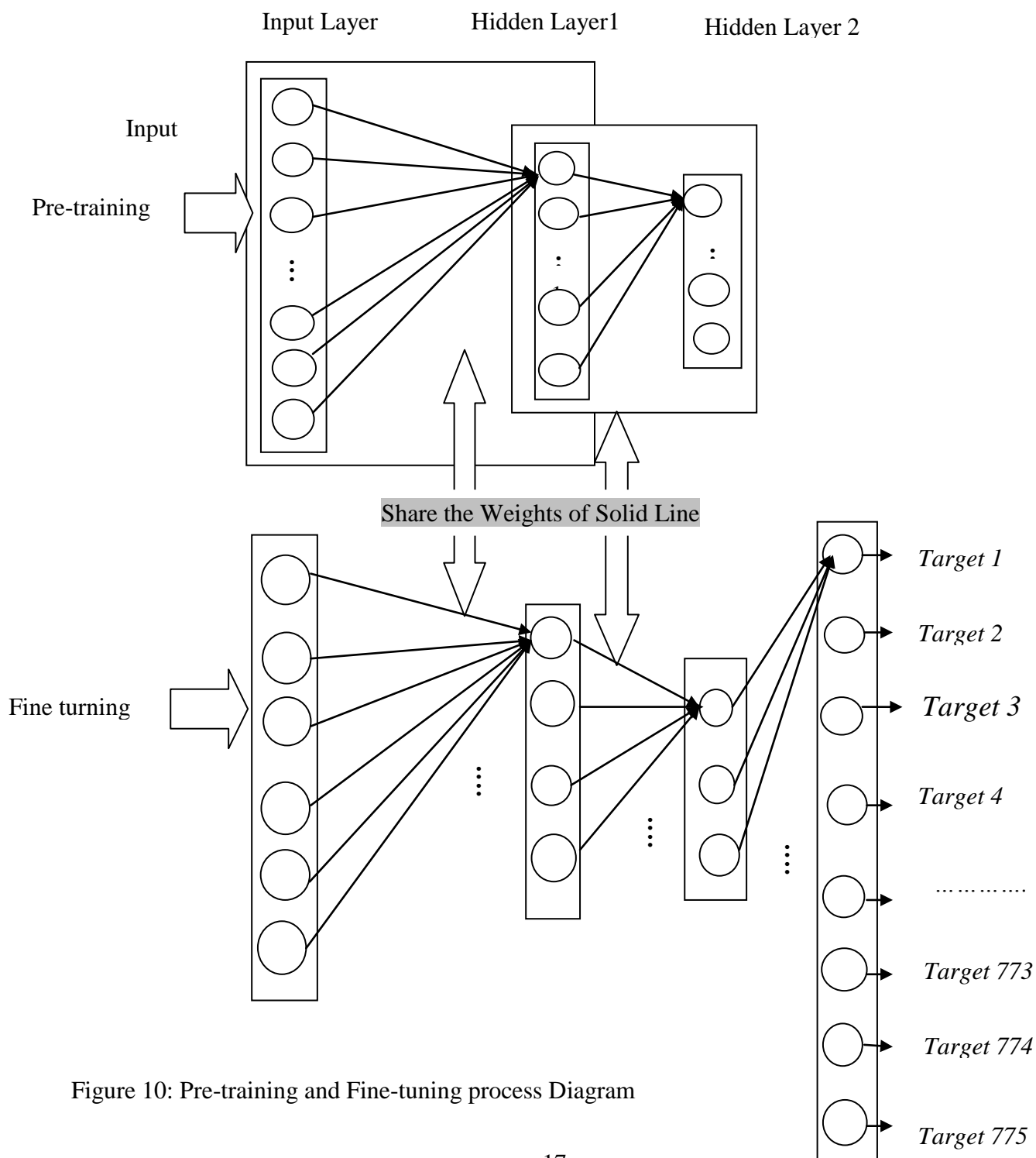


Figure 10: Pre-training and Fine-tuning process Diagram

Figure 3.7 shows: The pre-training initializes the parameter of the fine-tuning phase, pre-training consists of a stack of RBMs with layer-wise unsupervised training and fine-tuning consists of a stack of multi-hidden layer MLP with regression as output layer with BP training.

RBM is fed again another RBM as input. Second RBM again samples it for further abstract feature detection in hierarchical fashion. In this way, pre-training is being proceeded for all RBM in the system which is in unsupervised manner. After finishing the pre-training for number of drugs using a set of minibatch (a set of drugs), the system goes to fine-tuning section. In fine tuning section, the system trains the DBN in supervised way in which labeled data is used. The labeled data which is a simple two dimension matrix where row contains the drug and column contain the proteins. The cell corresponding the particular drug and particular protein (target) identifies for the drug and target interaction. '1' denotes interaction whereas '0' denotes no interaction. Based on this interaction output and corresponding drug feature fingerprint as input, the system is being trained using logistic regression at output layer and back propagation method in remaining hidden layers. The whole procedure for pre-training and fine tuning is illustrated in following figure 3.7.

As shown in Figure 3.7, the weights and bias parameter adjusted by pre-training are shared by the fine-tuning process. From diagram, it is clear that there are total 775 outputs which represent the targets (protein) to be interacted with drug which is fed at input. In fine tuning, the interacting target becomes '1' and missing target becomes '0' for corresponding drug input. For example, If Amoxicillin is taken as input then target Q8XJ01 has '1' and other has '0' at output layer. At the time of pre-training, only feature vector is feed to the input layer and sampling output is taken at each layer. The detailed procedure is explained step by step in section 3.3 for one complete example. As soon as pre-training finished, both input feature vector with corresponding output interaction target are fed to the system. The prediction is found at using feed forward calculation and then prediction output and actual output are compared for error calculation. The error obtained is back-propagated for parameter adjustment so that the accuracy is high.

For testing the system drug of corresponding feature vector is fed at input and prediction result is recorded. Finally, the prediction result is in the form of real number between 0 and 1. This real numbered prediction result is made binary by comparing with standard threshold 0.5 value. If the greater than 0.5 then output is considered as 1 otherwise 0. Further, result is compared with gold standard dataset and evaluation is done in following section in detail.

3.1.3 Deep Learning

Deep learning is one of the most cutting-edge machine learning techniques. Deep learning allows computational models that are composed of multiple processing layers to learn representations of data with multiple levels of abstraction. These methods have dramatically improved the state-of-the-art in speech recognition, visual object recognition, object detection and many other domains such as drug discovery and genomics. Deep models and their learning algorithms are inspired by the structure and information processing mechanism of human brain. Each deep model has a deep structure that consists of a number of non-linear hidden feature layers and hierarchical feature abstraction mechanism. Deep learning is part of a broader family of machine learning methods based on learning representations of data. For example, an observation (e.g., an image) can be represented in many ways such as a vector of intensity values per pixel, or in a more abstract way as a set of edges, regions of particular shape, etc. In other words, deep learning essential to build deep architectures for extracting multiple levels of distributed features of the input automatically. There are several ways of generating deep architectures, such as Convolution Neural Network (CNNs), Stacked Autoencoders (SAs), Recursive Neural Network (RNNs) and Deep Belief Network (DBNs). [30]

3.1.3.1 Restricted Boltzmann Machine

A Boltzmann Machine (BM) is a generative stochastic neural network that can learn a probability distribution over its inputs. A Restricted Boltzmann Machine (RBM) is further restricted to abandon visible-visible and hidden-hidden connections. A RBM is a two layer probabilistic bipartite undirected graph model as illustrated in Figure 3.8, which is constructed with a number of visible and hidden nodes (random variables). Each node has a bias and a connection weight with the nodes in the different layer.

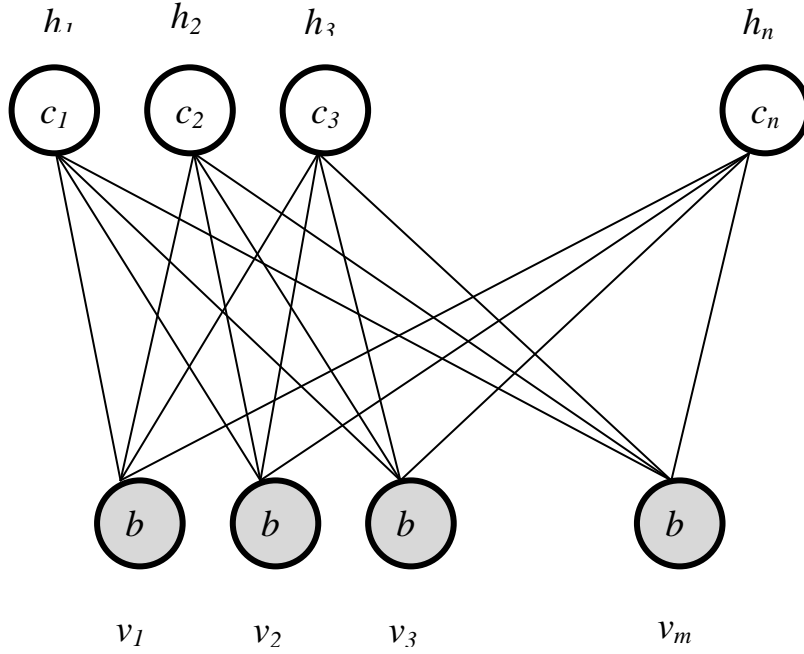


Figure 11: Restricted Boltzmann Machine (RBM)

As shown in figure, RBM consists of m visible units $\mathbf{v} = (v_1, v_2, \dots, v_m)$ representing the observable data, and n hidden units $\mathbf{h} = (h_1, h_2, \dots, h_n)$ to capture the dependencies between the observed variables. In binary RBMs, focus in this thesis, the random variables (\mathbf{v}, \mathbf{h}) takes values $(\mathbf{v}, \mathbf{h}) \in (0,1)^{m+n}$. An RBM is an energy-based probabilistic model, in which the Gibbs probability distribution is defined through an energy function. Its probability is defined as

$$P(\mathbf{v}, \mathbf{h}) = \frac{e^{-E(\mathbf{v}, \mathbf{h})}}{Z} \quad \text{equation (3.1)}$$

Where the energy function is given by

$$E(\mathbf{v}, \mathbf{h}) = \sum_i \sum_j w_{i,j} h_i v_j - \sum_j b_j v_j - \sum_i c_i h_i \quad \text{equation (3.2)}$$

And Z is partition function which is given by summing over all possible pair of visible and hidden vectors:

$$Z = \sum_{\mathbf{v}, \mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})} \quad \text{equation (3.3)}$$

In equation, 3.2, For all $i \in (1, \dots, n)$ and $j \in (1, \dots, m)$, $w_{i,j}$ is a real valued weight associated with the edge between the units v_j and h_i , and b_j and c_i are real valued bias terms associated with the j th visible and the i th hidden variable, respectively.

The graph of an RBM has connections only between the layer of hidden and the layer of visible variables, but not between two variables of the same layer. In terms of probability, this means that the hidden variables are independent given the state of the visible variables and vice versa:

$$\begin{aligned} P(\mathbf{h}|\mathbf{v}) &= \frac{P(\mathbf{v}, \mathbf{h})}{P(\mathbf{v})} \\ &= \frac{P(\mathbf{v}, \mathbf{h})}{\sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h})} \\ &= \prod_{i=1}^n P(h_i|\mathbf{v}) \end{aligned}$$

Where,

$$\begin{aligned} P(h_i|\mathbf{v}) &= \frac{e^{(h_i W_i v + b_i h_i)}}{1 + e^{(b_i + h_i W_i v)}} \\ P(h_i = 1|\mathbf{v}) &= \frac{e^{(W_i v + b_i)}}{1 + e^{(b_i + W_i v)}} \\ &= \frac{1}{1 + e^{(-b_i - W_i v)}} \end{aligned}$$

$$P(h_i = 1|\mathbf{v}) = \text{sigm} \left(\sum_{j=1}^m W_{i,j} v_j + c_i \right) \quad \text{equation (3.4)}$$

And similarly,

$$P(\mathbf{v}|\mathbf{h}) = \prod_{j=1}^m P(v_j|\mathbf{h})$$

$$P(v_j = 1|h) = \text{sigm} \left(\sum_{i=1}^n W_{i,j} h_i + b_j \right) \quad \text{equation (3.5)}$$

Thus, due to the absence of connections between hidden variables, the conditional distributions $p(h | v)$ and $p(v | h)$ factorize nicely. The conditional independence between the variables in the same layer makes Gibbs sampling especially easy: instead of sampling new values for all variables subsequently, the states of all variables in one layer can be sampled jointly. Thus, Gibbs sampling can be performed in just two steps: sampling a new state h for the hidden neurons based on $p(h | v)$ and sampling a state v for the visible layer based on $p(v | h)$. This is also referred to as **block Gibbs sampling** [31].

3.1.3.2 Training RBM

Unsupervised learning means learning an unknown distribution based on sample data. This includes finding new representations of data that foster learning, generalization, and communication. [24]. If it is assumed that the structure of the graphical model is known and that the energy function belongs to a known family of functions parameterized by $\Theta = (b_j, c_i, w_{ij})$ used in equation 3.2, unsupervised learning of a data distribution with an Markov Random Field (MRF) means adjusting the parameters Θ . It is considered the training data $S = (x_1, \dots, x_L)$. The data samples are assumed to be independent and identically distributed (i.i.d.). That is, they are drawn independently from some unknown distribution. The Gibbs distribution of an MRF describes the joint probability distribution of (V, H) and one is usually interested in the marginal distribution of V , which is given by

$$p(v) = \frac{1}{Z} \sum_h e^{-E(v,h)} \quad \text{equation (3.6)}$$

A standard way of estimating the parameters of a statistical model is maximum-likelihood estimation. Applied to MRFs, this corresponds to finding the MRF parameters that maximizes the maximum the probability of S under the MRF distribution, training corresponds to finding parameter Θ that maximizes the likelihood given training data.

Restricted Boltzmann machines are MRFs with hidden variables and RBM learning algorithms as based on gradient ascent on the log-likelihood. For a model of the for Equation 3.6 with parameter Θ , the log-likelihood given a single training example v is

$$\ln p(v|\Theta) = \ln \frac{1}{Z} e^{-E(v,h)}$$

$$= \ln \sum_h e^{-E(v,h)} - \ln \sum_{v,h} e^{-E(v,h)} \quad \text{equation (3.7)}$$

And for the gradient, we get

$$\begin{aligned} \frac{\partial(\ln p(v|\Theta))}{\partial\Theta} &= \frac{\partial(\ln \sum_h e^{-E(v,h)})}{\partial\Theta} - \frac{\partial(\ln \sum_{v,h} e^{-E(v,h)})}{\partial\Theta} \\ &= - \sum_h p(h|v) \frac{\partial E(v,h)}{\partial\Theta} + \sum_{v,h} p(v,h) \frac{\partial E(v,h)}{\partial\Theta} \end{aligned} \quad \text{equation (3.8)}$$

In above equation 3.8, it is used conditional probability which can be written as

$$\begin{aligned} p(h|v) &= \frac{p(v,h)}{p(v)} \\ &= \frac{\frac{1}{Z} e^{-E(v,h)}}{\frac{1}{Z} \sum_h e^{-E(v,h)}} \\ &= \frac{e^{-E(v,h)}}{\sum_h e^{-E(v,h)}} \end{aligned} \quad \text{equation (3.9)}$$

For RBMs the first term (positive phase) of Equation 3.8 (i.e. the expectation of the energy gradient under the conditional distribution of the hidden variables given a training example \mathbf{v}) can be computed efficiently because it factorizes nicely. For example, taking $\Theta = w_{i,j}$, we get:

$$\begin{aligned}
\frac{\partial E(v, h)}{\partial W_{i,j}} &= \frac{\partial}{\partial W_{i,j}} \left(- \sum_{i,j} W_{i,j} h_i v_j - \sum_j b_j v_j - \sum_i b_i h_i \right) \\
&= - \frac{\partial}{\partial W_{i,j}} \sum_{i,j} W_{i,j} h_i v_j \\
&= - \mathbf{h}_i \mathbf{v}_j
\end{aligned}$$

$$\nabla_w E(v, h) = -\mathbf{h}\mathbf{v}^T \quad \text{equation (3.10)}$$

And

$$\begin{aligned}
\sum_h p(h|v) \frac{\partial E(v, h)}{\partial w_{i,j}} &= - \sum_h p(h|v) \mathbf{h}_i \mathbf{v}_j \\
&= - \sum_h \prod_{k=1}^n p(h_k|v) \mathbf{h}_i \mathbf{v}_j \\
\mathbb{E}_{p(h|v)} \left[\frac{\partial E(v, h)}{\partial w_{i,j}} \right] &= - \sum_{h_i \in \{0,1\}} h_i v_j P(h_i|v) \\
&= -v_j P(H_i = 1|v)
\end{aligned}$$

$$\mathbb{E}_{p(h|v)} [\nabla_w E(v, h)] = -\text{sigm} \left(\sum_{j=1}^m W_{i,j} v_j + c_i \right) \cdot v_j \quad \text{equation (3.11)}$$

Using the factorization trick in equation 3.11, the derivative of the log-likelihood of a single pattern v with respect to the weight $w_{i,j}$ becomes

$$\frac{\partial (\ln p(v|\theta))}{\partial w_{i,j}} = - \sum_h p(h|v) \frac{\partial E(v, h)}{\partial w_{i,j}} + \sum_{v,h} p(v, h) \frac{\partial E(v, h)}{\partial w_{i,j}}$$

$$\begin{aligned}
&= \sum_h p(h|v) \mathbf{h}_i \mathbf{v}_j - \sum_v p(v) \sum_h p(h|v) \mathbf{h}_i \mathbf{v}_j \\
&= v_j P(H_i = 1|v) - \sum_v p(v) v_j P(H_i = 1|v) \quad \text{equation (3.12)}
\end{aligned}$$

For the mean the mean of this derivative over a training set $S = (v_1, \dots, v_L)$, often the following notations are used:

$$\begin{aligned}
\frac{1}{L} \sum_{v \in S} \frac{\partial(\ln p(v|\Theta))}{\partial w_{i,j}} &= \frac{1}{L} \sum_{v \in S} \left[\sum_h \mathbb{E}_{p(h|v)} \left[\frac{\partial E(v, h)}{\partial w_{i,j}} \right] + \sum_{v, h} \mathbb{E}_{p(h, v)} \left[\frac{\partial E(v, h)}{\partial w_{i,j}} \right] \right] \\
&= \frac{1}{L} \sum_{v \in S} \left[\sum_h \mathbb{E}_{p(h|v)} [\mathbf{h}_i \mathbf{v}_j] + \sum_{v, h} \mathbb{E}_{p(h, v)} [\mathbf{h}_i \mathbf{v}_j] \right] \\
&= \langle \mathbf{h}_i \mathbf{v}_j \rangle_{p(h|v)q(v)} - \langle \mathbf{h}_i \mathbf{v}_j \rangle_{p(h, v)} \quad \text{equation (3.13)}
\end{aligned}$$

With q denoting the empirical (data) distribution. This gives the often stated rule

$$\sum_{v \in S} \frac{\partial(\ln p(v|\Theta))}{\partial w_{i,j}} \propto \langle \mathbf{h}_i \mathbf{v}_j \rangle_{data} - \langle \mathbf{h}_i \mathbf{v}_j \rangle_{model} \quad \text{equation (3.14)}$$

Analogously to Equation 3.12, we get the derivative with respect to the bias parameter b_j of j^{th} visible variable,

$$\frac{\partial(\ln p(v|\Theta))}{\partial b_j} = v_j - \sum_v p(v) v_j \quad \text{equation (3.15)}$$

and with respect to the bias parameter c_i of the i^{th} hidden variable

$$\frac{\partial(\ln p(v|\Theta))}{\partial c_i} = P(H_i = 1|v) - \sum_v p(v) P(H_i = 1|v) \quad \text{equation (3.16)}$$

In above Equations 3.12, 3.15 and 3.16, there is the difference between two expectations: the expected values of the energy function under the model distribution and under the conditional distribution of hidden variables given the training example. For calculating the second term

(negative phase) of each mentioned equations is difficult. Directly calculating this sums, which run over all values of the variables, leads to a computational complexity which is in general exponential in number of variables of the MRF. To avoid this computational burden, the expectations can be approximated by samples drawn from the corresponding distributions based on Markov Chain Monte Carlo (MCMC) techniques. This method has a low converging speed, and it is difficult to define an adequate step size during training stage [32]. Also, Obtaining unbiased estimates of the log-likelihood gradient using MCMC methods typically requires many sampling steps. These resulted in a long training time for RBM to reach steady state. Hinton et al. suggested using the Contrastive Divergence (CD-k) theory, which shortens the calculation time required while maintaining the same level of accuracy. These days, CD has become a standard way to train RBMs.

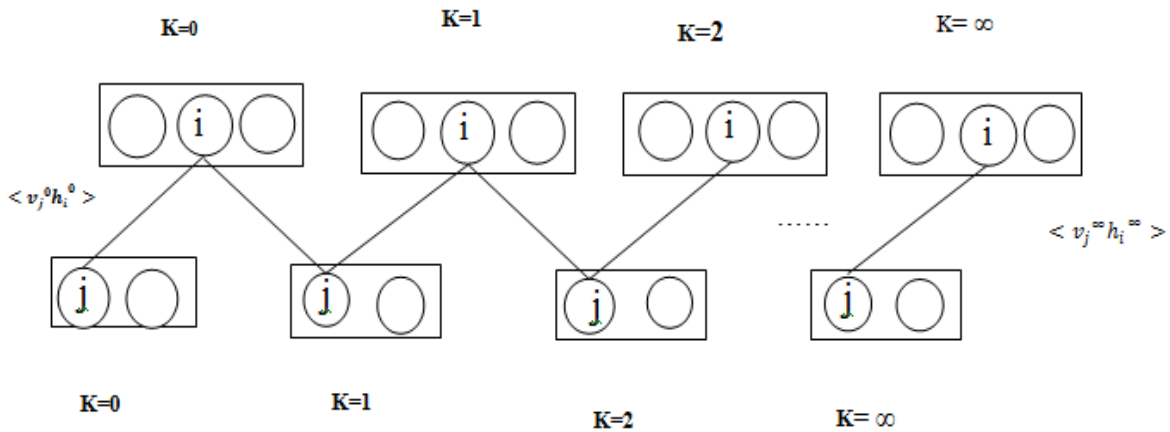


Figure 12: Contrastive Divergence (CD-k)

The idea of K-step contrastive divergence learning (CD-k) is quite simple instead of approximating the second term in the term in the likelihood gradient by a sample from the RBM-distribution (which would require running a Markov chain until the stationary distribution is reached), a Gibbs chain is run for only k steps .A each step in the Markov chain, visible units are samples given hidden units, hidden units are sampled given visible units as illustrated in Figure ...and in equations 3.17 and 3.18.

$$h^{(k+1)} \sim p(h^{(k+1)} | v^{(k)}) \quad \text{equation (3.17)}$$

$$v^{(k+1)} \sim p(v^{(k+1)}|h^{(k+1)}) \quad \text{equation (3.18)}$$

In the figure 3.9, when $k \rightarrow \infty$, samples $(v^{(k)}, h^{(k)})$ are guaranteed to be accurate sample of $p(v, h)$ but it is time-consuming. According to Hinton, The CD learning uses two tricks to speed up the sampling process. The first on it to initialize the Markov chain with a training example, and the second on is to obtain samples, after only k -steps of Gibbs sampling. A lot of experiments show that the performances of the approximations are still very good when $k=1$. It is also seen that CD learning provides an approximation of log-likelihood gradient that has been found to be a successful update rule for training probabilistic models. Variation justification can provide a theoretical proof to the convergence of the learning process. Conducting CD-1 learning by using Equations 3.4 and 3.5 namely $v=v^{(0)} \rightarrow h^{(0)} \rightarrow v^{(1)} \rightarrow h^{(1)}$. It is easy to get updating rules for parameter (w_{ij}, b_j, c_i) . The pseudo-code is demonstrated in Algorithm 1.

Algorithm 1: Updating rules for RBM

Input: $v^{(0)}$ is a training example form training distribution for RBM;

ϵ is the learning rate for updating the parameters.

W_{ij} is the visible-hidden connection weight matrix.

b_j is the bias vector for input (visible) units.

C_i is the bias vector for hidden units.

Output: The updated parameters in the RBM: W_{ij}, b_j and C_i

for all hidden units i **do**

 Compute $p(h_i^{(0)}=1|v^{(0)})$ using equation 3.4 (positive phase)

 Sample $h_i^{(0)} \in (0,1)$ from $p(h_i^{(0)}=1|v^{(0)})$.

end for

for all visible units j **do**

 Compute $p(v_j^{(1)}=1|h^{(0)})$ using equation 3.5

 Sample $v_j^{(1)} \in (0,1)$ from $p(v_j^{(1)}=1|h^{(0)})$.

end for

for all hidden units i **do**

Compute $p(h_i^{(1)}=1|v^{(1)})$ using equation 3.4 (negative phase)
 Sample $h_i^{(1)} \in (0,1)$ from $p(h_i^{(1)}=1|v^{(1)})$.

end for

Update:

$$W_{ij} = W_{ij} + \epsilon(v^{(0)} * p(h_i^{(0)}=1|v^{(0)}) - v^{(1)} * p(h_i^{(1)}=1|v^{(1)}))$$

$$b_j = b_j + \epsilon(v^{(0)} - v^{(1)})$$

$$c_i = c_i + \epsilon(p(h_i^{(0)}=1|v^{(0)}) - p(h_i^{(1)}=1|v^{(1)}))$$

return $W_{i,j}$, b_j and C_i

3.1.3.3 Deep Belief Network

The DBN is a direct acyclic graph except from the top two layers that form an undirected bipartite graph. The top two layers is what gives the DBN the ability to unroll into a deep autoencoder (DA) and perform reconstructions of the input data[30]. The DBN consists of a visible layer, output layer and a number of hidden layers as shown in figure 3.10 (b)

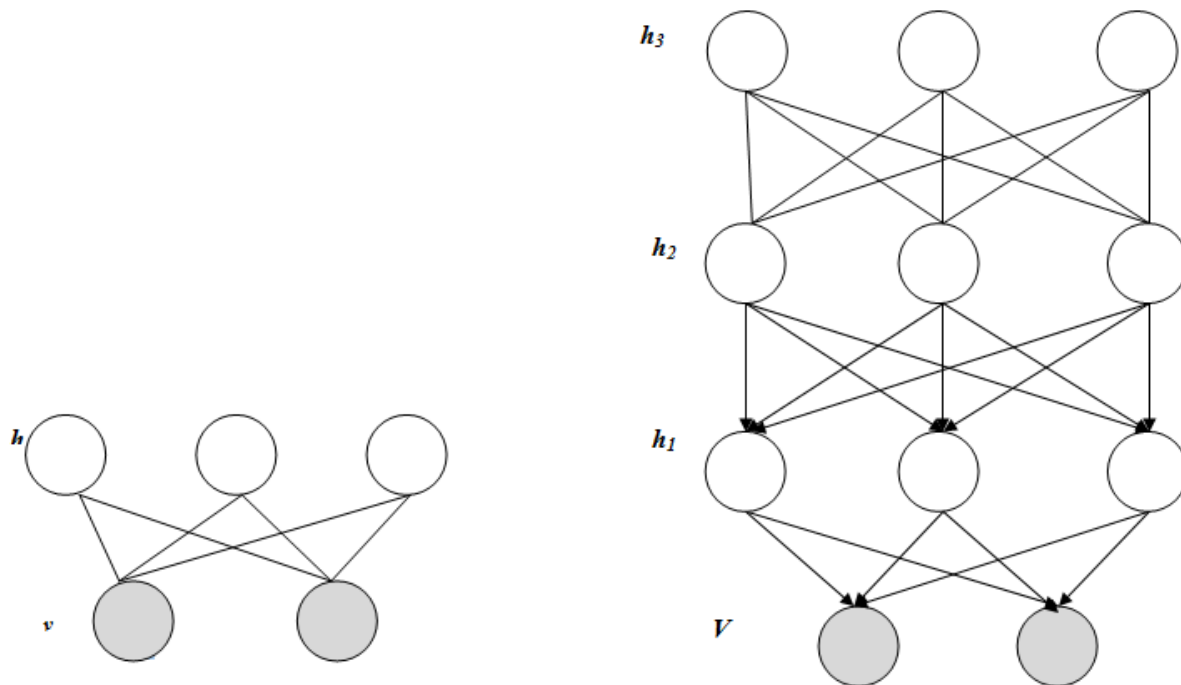


Figure 13: (a) and (b) The graphical model representations of an RBM and a DBN

Fig 3.10(a) shows undirected bipartite graphical model with one layer of stochastic visible units connected to one layer of stochastic hidden units. An RBM can be considered as a density model which describes the distribution of visible units. Given training samples of visible units, the model parameters of an RBM can be estimated under maximum likelihood criterion using contrastive divergence (CD) algorithm explained 3.1.3.1 and 3.1.3.2 sections above. Figure 3.10(b) shows the graphical representation of a DBN with three hidden layers. A DBN is a probabilistic generative model that contains many layers of hidden units. The top two layers form an undirected bipartite graph with the lower layers forming a directed sigmoid belief network as already defined. DBN can be constructed from stack of RBMs.

The training process of the DBN is defined by two steps: pre-training and fine-tuning. In pre-training the layers of the DBN are separated pair wise to form restricted Boltzmann machines (RBM). Each RBM is trained independently, such that the output of the lower RBM is provided as input to the next higher-level RBM and so forth. This way the layers of the DBN are trained as partly independent systems. The goal of the pre-training process is to achieve approximations of the model parameters (that is connection weight between visible and hidden unit, visible unit bias and hidden unit bias).

Hinton showed that RBMs can be stacked and trained in a greedy manner to form so-called Deep Belief Networks (DBN) [9]. DBNs are graphical models which learn to extract a deep hierarchical representation of the training data. They model the joint distribution between observed vector v and the ℓ hidden layers h^k as follows:

$$P(v, h^1, \dots, h^\ell) = \left(\prod_{k=0}^{\ell-2} p(h^k | h^{k+1}) \right) p(h^{\ell-1}, h^\ell) \quad \text{equation (3.19)}$$

Where, $v=h^{(0)}$, $p(h^{(k-1)}|h^{(k)})$ is a conditional distribution for the visible units conditioned on the hidden units of the RBM at level k which can be easily calculated by using equation 3.4 and 3.5, and $p(h^{\ell-1}, h^\ell)$ is the visible-hidden joint distribution in the top-level RBM. This is illustrated in the figure 3.10 (b).

Thus, in pre-training DBN works like feed forward network. All hidden layers are independent to each other so each RBM are trained one after another i.e. the output of the first RBM is given to second RBM as input. Again the output of the second RBM is given to input of the third RBM as input and so on. Finally, the parameter of each hidden layers are initialized by pre-training which overcomes the limitation of gradient based learning. After the initializing DBN parameters by pre-training, the second stage of DBN that is fine tuning approach is applied in supervised fashion. The fine tuning is done by using Back Propagation algorithm explained in Appendix: A. In the output of the first training result is compared with the actual output and then error is calculated and then error is propagated to back for re-updating the connection weight between visible layer and hidden layers in each RBM so that better accuracy is performed. This process is illustrated the figure 3.3 proposed by Hinton in 2006 [9].

Hinton showed that an image of high dimension data can be converted to low-dimension codes by training multilayer neural network with small central layer to reconstruct high-dimensional input vectors. So, for that a similar decoder network can be applied into recover the data from code as shown in figure 3.10 (b). Starting from with random weights in the two encoder and decoder networks, they can be trained together by minimizing the discrepancy. The required gradients are easily obtained by using chain rule to propagate error derivatives first through the decode network and then through the encoder network. The whole system is called “autoencoder” and is depicted in figure 3.11.

A detail training algorithm for DBN is illustrated in Algorithm -2 .

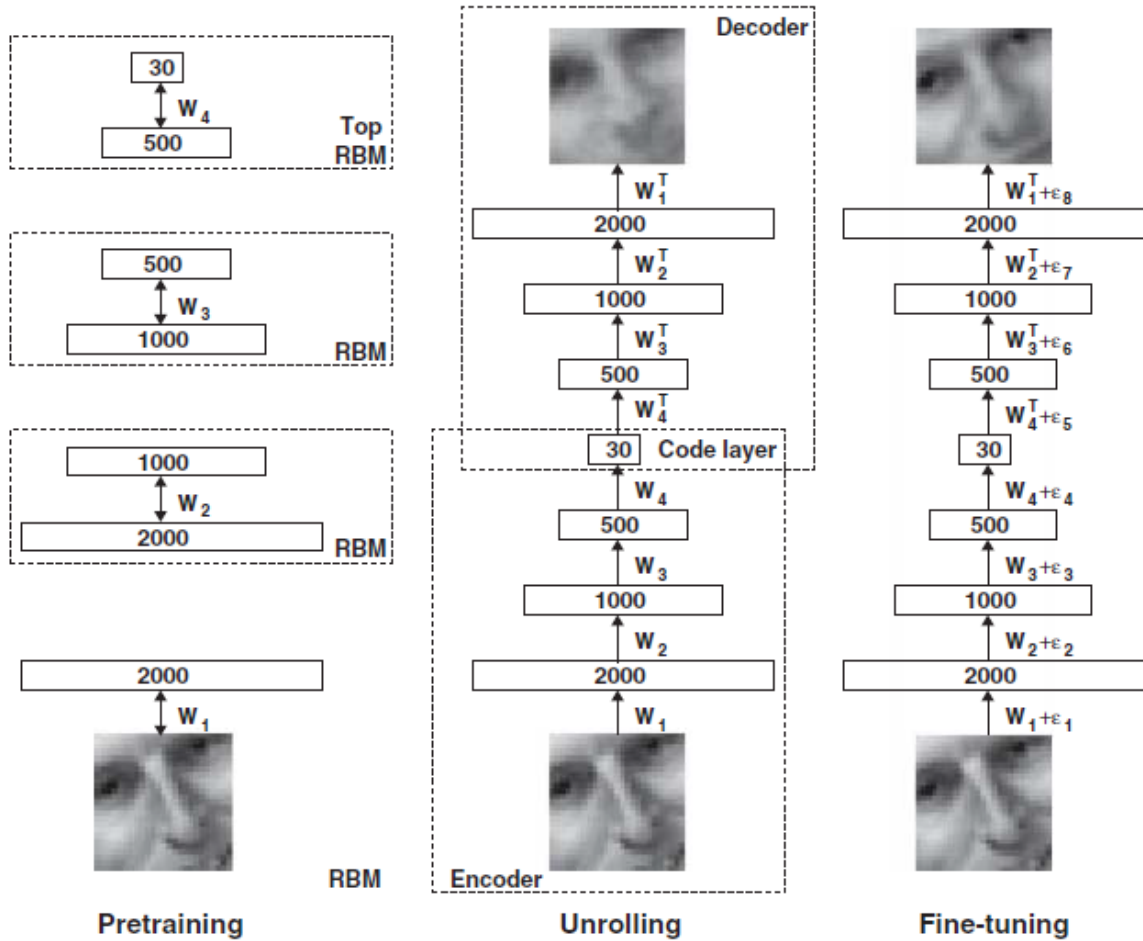


Figure 14: Pre-training and Fine-tuning of stack of RBMs

Pre-training consists of learning a stack of restricted Boltzmann machines (RBMs), each having only one layer of feature detectors. The learned feature activations of one RBM are used as the “data” for training the next RBM in the stack. After the pre-training, the RBMs are “unrolled” to create a deep autoencoder, which is then fine-tuned using back propagation of error derivatives. [6]

Algorithm 2: Training DBN

Pre- training :

Input : take input $v^{(0)}$ vector

for all hidden layers ℓ **do**

```

if first layer
    input=  $v^{(0)}$ 
else
    input= pre-input
    - train RBM using equations 3.4 and 3.5 and as mentions in
      Algorithm 1 .
    -find output sample from given input
    -update parameters using equations 3.12, 3.15 and 3.16
    assign:
        rep-input= output
end for
Fine- training:
    Find: error,
        for all output units  $i$  do
             $dy[i] = \text{actual output}[i] - \text{predicted output}[i]$ 
        end for
    for all hidden layers  $\ell$  to layer 1 do
        if  $\ell$  layer
            error =  $dy$  for all 'k' output units.
        else
            error= pre-layer- $dy$  (previous calculated) for all 'k' output units
        find error correction term 'del_y' from previous hidden layer
            for all pre-input  $j$  do
                 $del[j]=0$ 
                for all pre-output  $k$  do
                     $del[j]=del[j]+error[k]*\text{pre-layer-}w[k][j]$ 
                end for
                 $del\_y[j]= del[j]*\text{pre-layer-input}[j]*(1- \text{pre-layer-input}[j])$ 
            end for
        update the weight and bias for the current hidden layer for all units

```

```

new_weight= old_weight + learning_rate*del_y*current_input
new_bias = old_bias + learning_rate * del_y* current_input

```

end for

3.1.4 Experimental Design

From data set of 10007 drugs, randomly, training set of 997 drugs and testing set of 10 drugs is separated for every experiments. There were total 5 experiments performed. The hyper-parameters, training dataset and testing dataset are changed from experiment to experiment. In this thesis, experiment 1 to experiment 3 were done by changing hyper-parameters and experiment 4 and experiment 5 were done by changing the training dataset and testing dataset with hyper-parameters of experiment 3.

3.1.5 Evaluation Indices

In this study, the confusion matrix, also known as a contingency table or an error matrix [33] is used to represent the prediction performance of a predictor. Figure 3.12 illustrates an exemplary confusion matrix, where TP, FP, TN and FN are abbreviations for true positives, false positives, true negatives, and false negatives, respectively. For the convenience of subsequent descriptions, it is used to [TP FP; FN TN] to represent a confusion matrix.

		Condition (as determined by "Gold standard")		
		Condition Positive	Condition Negative	
Test Outcome	Test Outcome Positive	True Positive	False Positive (Type I error)	Positive predictive value = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Test Outcome Positive}}$
	Test Outcome Negative	False Negative (Type II error)	True Negative	Negative predictive value = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Test Outcome Negative}}$
		Sensitivity = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Condition Positive}}$	Specificity = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Condition Negative}}$	

Figure 15: Confusion matrix for performance evaluation

From a confusion matrix, five routinely used evaluation indices in this bio-informatics field, i.e. Specificity (Sp) , Precision (Pre) , Sensitivity (Sn) or Recall (Rec), Accuracy (Acc), F1-measure and Matthews correlation coefficient (MCC), can be computed as follows [27]:

$$Sn \text{ or } Rec = \frac{TP}{TP + FN} \quad \text{equation (3.20)}$$

$$Pre = \frac{TP}{TP + FP} \quad \text{equation (3.21)}$$

$$Sp = \frac{TN}{TN + FP} \quad \text{equation (3.22)}$$

$$Acc = \frac{TP + TN}{TP + TN + FP + FN} \quad \text{equation (3.23)}$$

$$F1 - Measure = \frac{2 * Pre * Rec}{Pre + Rec} \quad \text{equation (3.24)}$$

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP) * (TN + FP) * (TN + FP) * (TN + FN)}} \quad \text{equation (3.25)}$$

To evaluate drug-target interaction, the sensitivity, precision, specificity, accuracy and F1-score are calculated using equations 3.20. 3.21, 3.22, 3.23 and 3.24 respectively. As accuracy is not perfect evaluating term because it is based on single point threshold value. So, ROC curve is used for evaluating the DTI-DBN model for evaluating efficient way.

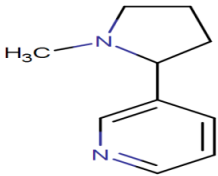
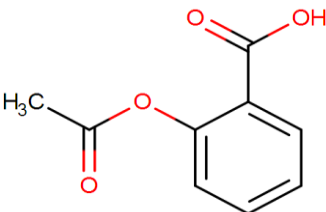
An ROC curve demonstrates several things:

- It shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).
- The closer the curve follows the left-hand border and then the top border of the ROC space, the more accurate the test.
- The closer the curve comes to the 45-degree diagonal of the ROC space, the less accurate the test.

3.1.6 Predicting Drug-Target Interaction Example

Let us consider one Nicotine drug form Table 3.1 which has chemical structure as shown in Drug Structure column. Its chemical sub-structure fingerprint is as shown in table 3.2. It is noted that the drug Nicotine has sub –structure SUB 9, SUB 10, SUB 11 but not others because there is presence of binary ‘1’ in SUB 9, SUB 10 and SUB 11 columns whereas ‘0’ in other columns. Also, from the third column of Table 3.1, it is clear that there Drug Nicotine (DB00184) is interacted with targets (proteins, Q15822, Q15825, Q9G226, Q9UGM1, P36544, P43681, Q05901). For simplicity the Interaction between ‘drug Nicotine and Q15822’ and ‘drug Nicotine and Q15822’ pairs are only taken from the dataset (drug and protein interaction table) as shown in Table 3.3. The presence of ‘0’ in Nicotine row with targets P07451, P07477, P07510, P07550 and P07900 indicates that there is no missing interaction between them.

Table 1: Drug, its structure and interacting Targets [DrugBank [27] and UniProt[5]]

Drug	Drug Structure	Target
Nicotine (DB00184)		UniProtKB - Q15822 UniProtKB - Q15825 UniProtKB - Q9G226 UniProtKB - Q9UGM1 UniProtKB - P36544 UniProtKB - P43681 UniProtKB - Q05901
Aspirin (DB00945)		UniProtKB - P23219 UniProtKB - P35354 UniProtKB - Q04828

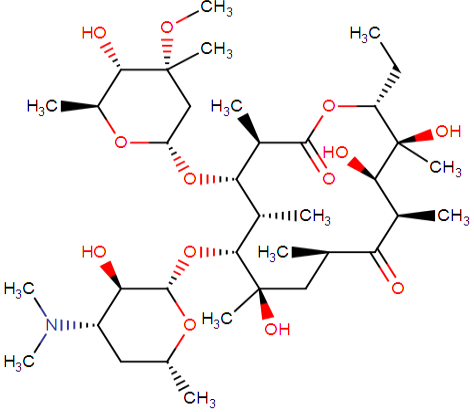
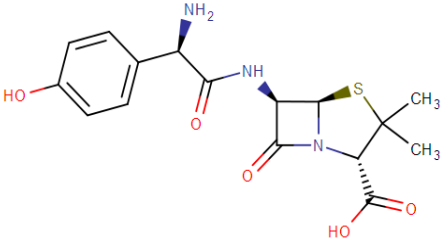
<p>Erythromycin DB00199</p>		<p>UniProtKB - P60725 UniProtKB - P61177</p>
<p>Amoxicillin DB01060</p>		<p>UniProtKB - Q8XJ01</p>

Table 2: Drug and Sub-Structure Fingerprint Sample

	SUB 0	SUB 8	SUB 9	SUB 10	SUB 11	SUB 12	SUB 13	SUB 16
Nicotine	0	0	1	1	1	0	0	0
Acetylsali	0	0	1	1	1	0	0	0
Erythromy	1	0	1	1	1	1	1	0
Amoxicilli	1	0	1	1	1	1	0	0

Table 3: Drug and Target interaction Sample

	Q15822	P07451	Q15825	P07477	P07510	P07550	P07900
Nicotine	1	0	1	0	0	0	0
Erythromycin	0	0	0	0	0	0	0

A step by step training example for Drug Nicotine having sub-structure feature (SUB9, SUB10, SUB11, SUB12, SUB13 and SUB16) is explained below. This each stepping result is traced from logs generated by system.

Step 1: Initializing number of Hidden layers, Number of neurons per Hidden layer, Weight W_{ij} , visible unit bias b_j , hidden unit bias h_i , number of inputs and Number of outputs in each hidden Layer (RBMSs) and Logistic Layer.

Initialization.....

Total number of Hidden Layers =2

Number of neurons in layer1=5

Number of neurons in layer2=4

RBM 0

no of inputs= 6

no of outputs= 5

Weight (W_{ij}) between visual and Hidden Units

Table 4: Initialization of W_{ij} of RBM0

	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
H[0]	-0.14158	0.1489352	-0.021041	0.06460518	0.0587995	0.0429279
H[1]	0.0249014	-0.037343	-0.103624	0.15775947	-0.1561477	0.110136
H[2]	0.0800287	0.1471575	0.1086465	0.00473589	0.03135864	-0.090575
H[3]	-0.06561	0.0650149	-0.023912	0.15515912	0.07514362	-0.054947
H[4]	0.1527631	-0.004892	-0.084809	0.06507426	-0.0935222	-0.129605

Hbias (h_i)

Table 5: Initialization of h_i of RBM0

h0	h1	h2	h3	h4
0	0	0	0	0

Vbias (b_j)

Table 6: Initialization of b_j of RBM0

b0	b1	b2	b3	b4	b5
0	0	0	0	0	0

RBM 1

no of inputs= 5

no of outputs= 4

Weight (w_{ij}) between visual and Hidden Units

Table 7: Initialization of W_{ij} of RBM1

	V[0]	V[1]	V[2]	V[3]	V[4]
H[0]	-0.1301792	0.1845587	0.0459248	0.086398	-0.1674031
H[1]	-0.0104486	-0.1203621	0.0246692	0.1414439	-0.1593327
H[2]	0.0673256	0.1677341	-0.0047118	0.0111143	0.0141208
H[3]	0.1461995	0.1312866	0.0920315	0.1107707	0.0651723

Hbias (h_i)

Table 8: Initialization of h_i of RBM1

h0	h1	h2	h3
0	0	0	0

Vbias (b_j)

b0	b1	b2	b3	b4
0	0	0	0	0

Logistic Layer.....

no of inputs= 4

no of outputs= 3

weight

Table 9: Initialization of W_{ij} of Logistic Layer

	V[0]	V[1]	V[2]
H[0]	0	0	0
H[1]	0	0	0
H[2]	0	0	0

Bias

Table 10: Initialization of b_j of Logistic Layer

b0	b1	b2
0	0	0

Step 2: Pre-training RBM0 with input Nicotine drug sub-structure feature

Pre-training.....

Total no of training data =1

Pr-training Learning rate =0.1

Total number of pre-training epochs =1

RBM 0

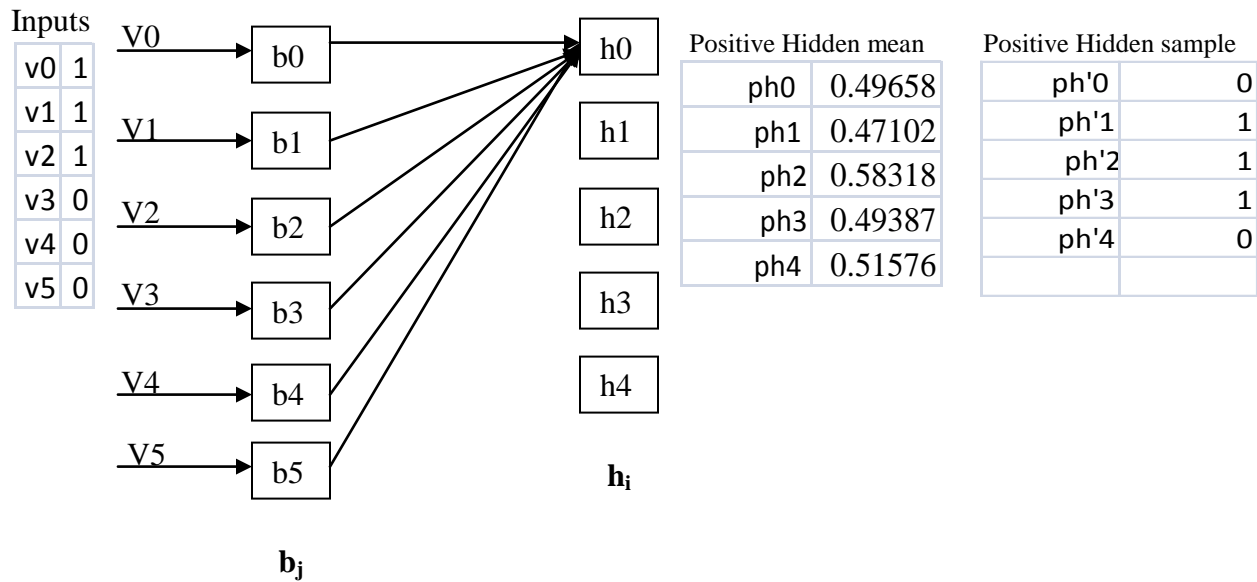


Figure 16: Positive hidden sampling of RBM0

Input (vj)

Table 11: Input feature of Nicotine Drug

v0	v1	v2	v3	v4	v5
1	1	1	0	0	0

positive hidden mean(p(Hi=1|v)) (using equation 3.4)

Table 12: positive hidden mean of RBM0

ph0	ph1	ph2	ph3	ph4
0.49658	0.47102	0.58318	0.49387	0.51576

Positive hidden sample (binomial output of hidden mean)

Table 13: Positive hidden sample of RBM0

ph'0	ph'1	ph'2	ph'3	ph'4
0	1	1	1	0

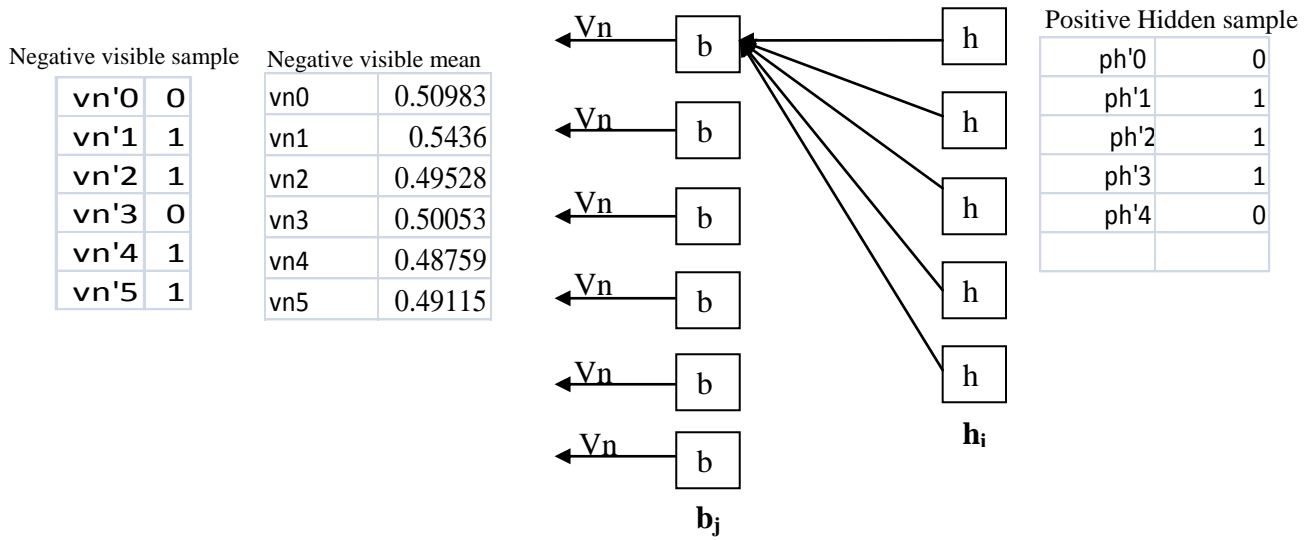


Figure 17: Negative visible sampling of RBM0

Negative visible mean using equation 3.5)

Table 14: Negative visible mean output of RBM0

vn0	vn1	vn2	vn3	vn4	vn5
0.50983	0.5436	0.49528	0.50053	0.48759	0.49115

negative visible sample ($p(v_j=1|h)$) (binomial output of negative visible mean)

Table 15: Negative visible sample output of RBM0

vn'0	vn'1	vn'2	vn'3	vn'4	vn'5
0	1	1	0	1	1

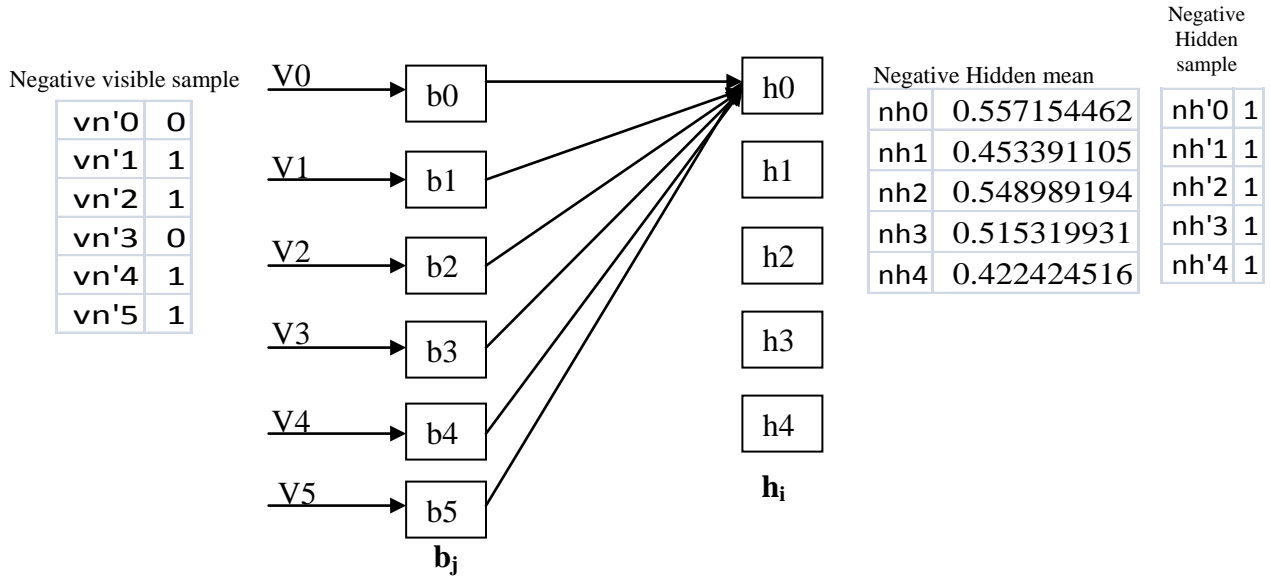


Figure 18: Negative hidden sampling of RBM0

Negative hidden mean using equation 3.4)

Table 16: Negative hidden mean output of RBM0

nh0	nh1	nh2	nh3	nh4
0.557154462	0.4533911	0.548989194	0.515319931	0.42242

Negative hidden sample (binomial output of negative hidden mean)

Table 17: Negative hidden sample output of RBM0

nh'0	nh'1	nh'2	nh'3	nh'4
1	1	1	1	1

Modified weight w_{ij} (using equation 3.12)

Table 18: modified weight (W_{ij}) of RBM0

	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
H[0]	-0.0919	0.14288	-0.0271	0.06461	0.00308	-0.0128
H[1]	0.072	-0.0356	-0.1019	-0.1578	-0.2015	0.0648
H[2]	0.13835	0.15058	0.11207	0.00474	-0.0235	-0.1455
H[3]	-0.0162	0.06287	-0.0261	0.15516	0.02361	-0.1065
H[4]	0.20434	0.00444	-0.0755	0.06507	-0.1358	-0.1718

Modified h-bias h_i (using equation 3.15).....

Table 19 modified h-bias (h_i) of RBM0

h0	h1	h2	h3	h4
-0.0557154	0.05466089	0.045101081	0.04846801	-0.042242452

Modified v-bias (b_j) (using equation 3.16).....

Table 20: modified v-bias (b_j) of RBM0

b0	b1	b2	b3	b4
0.1	0	0	0	-0.1

Step 3: Pre-train RBM 1 using output of RBM0 as input to RBM1 (same as step 2)

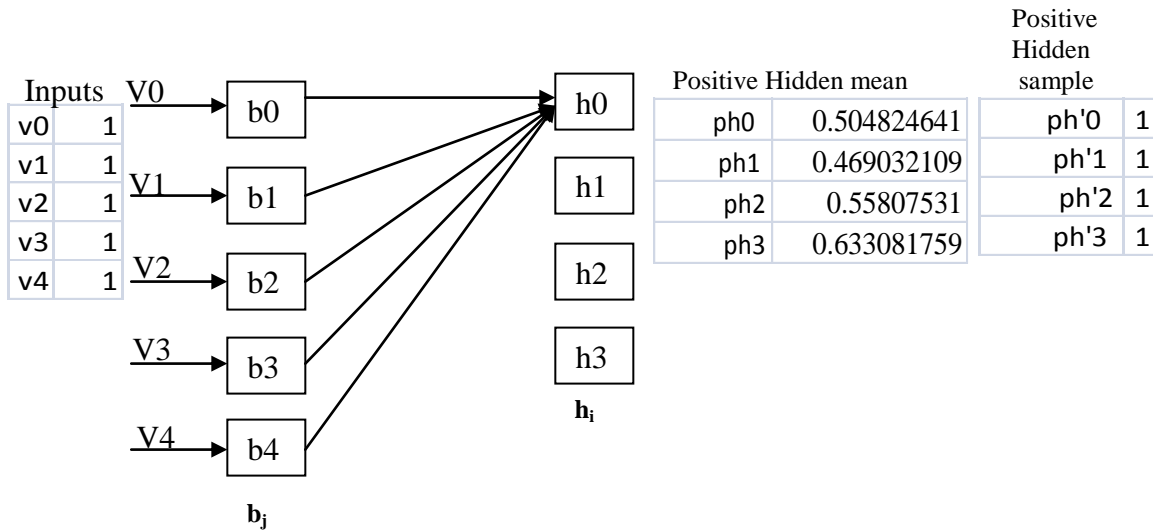


Figure 19: Positive hidden sampling of RBM1

Input

Table 21: Output of RBM0 as input to RBM1

v0	v1	v2	v3	v4
1	1	1	1	1

Positive hidden mean

Table 22: Positive hidden mean of RBM1

ph0	ph1	ph2	ph3
0.504824641	0.469032109	0.55807531	0.633081759

Positive hidden sample

Table 23: Positive hidden sample of RBM1

ph'0	ph'1	ph'2	ph'3	ph'4
1	1	1	1	1

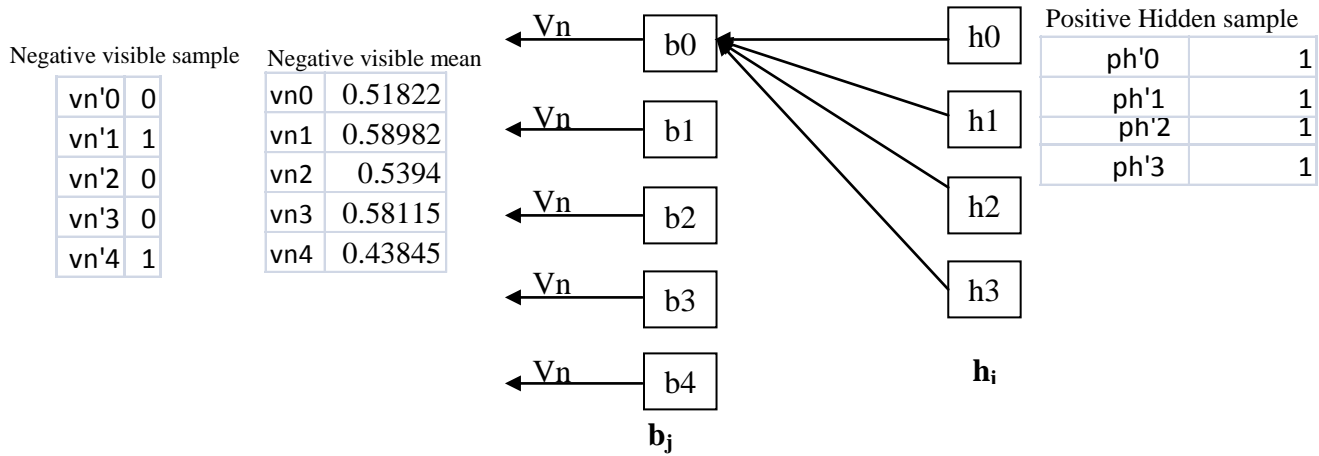


Figure 20: Negative visible sampling

Negative visible mean

Table 24: Negative visible mean output of RBM1

vn0	vn1	vn2	vn3	vn4
0.51821626	0.589819011	0.539396597	0.581150547	0.438453028

Negative visible sample

Table 25: Negative visible sample output of RBM1

vn'0	vn'1	vn'2	vn'3	vn'4	vn'5
0	1	0	0	1	

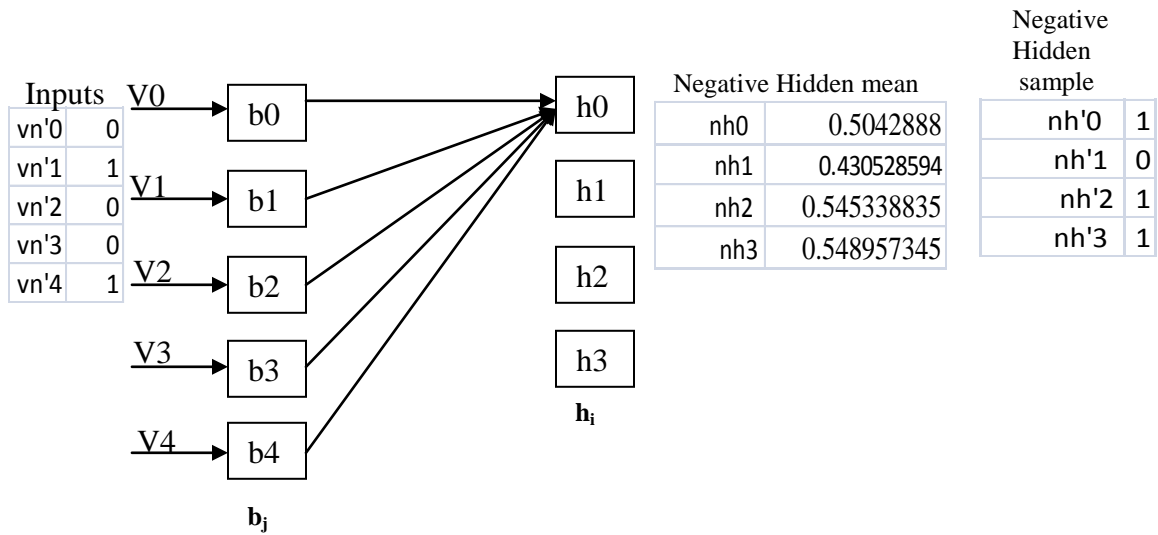


Figure 21: Negative hidden sampling

Negative hidden mean

Table 26: Negative hidden mean output

nh0	nh1	nh2	nh3
0.5042888	0.430528594	0.545338835	0.548957345

Negative hidden sample

Table 27: Negative hidden sample output

nh'0	nh'1	nh'2	nh'3
1	0	1	1

Modified weight (W_{ij})

Table 28: modified weight (W_{ij}) in RbM1

	V[0]	V[1]	V[2]	V[3]	V[4]
H[1]	-0.07969678	0.184612297	0.096407271	0.136880444	-0.1673495
H[2]	0.036454639	-0.116511758	0.071572438	0.188347063	-0.1554824
H[3]	0.123133156	0.169007717	0.051095762	0.044693245	0.01539445
H[4]	0.209507673	0.139698998	0.155339636	0.174078916	0.07358472

modified hbias.....

Table 29: modified h-bias h_i in Rbm1

h0	h1	h2	h3
0.04957112	0.056947141	0.045466117	0.045104265

modified vbias.....

Table 30: Modified v-bias v_j in Rbm1

b0	b1	b2	b3	b4
0.1	0	0.1	0.1	0

Step 4: fine tuning using feed forward calculation as given Algorithm 2

Fine-training.....

Total no of training data =1

Fine-training Learning rate =0.1

Total number of fine-training epochs =1

Forward layer wise calculation...(exactly like pre-training)

Layer 1

Input

Table 31: Fine-tuning: input to RBM0

v0	v1	v2	v3	v4	v5
1	1	1	0	0	0

Output (Using Equation 2.2.4)

Table 32: Fine-tuning: output of RBM0

ph'0	ph'1	ph'2	ph'3	ph'4
0	1	0	1	1

Layer 2 (Taking output of Layer1 as input to Layer 2)

Input

Table 33: Fine-tuning: output of RBM0 as input to RBM0

v0	v1	v2	v3	v4
0	1	0	1	1

Output (Using Equation 2.2.4)

Table 34: Fine-tuning: output of RBM1

ph'0	ph'1	ph'2	ph'3
0	1	1	0

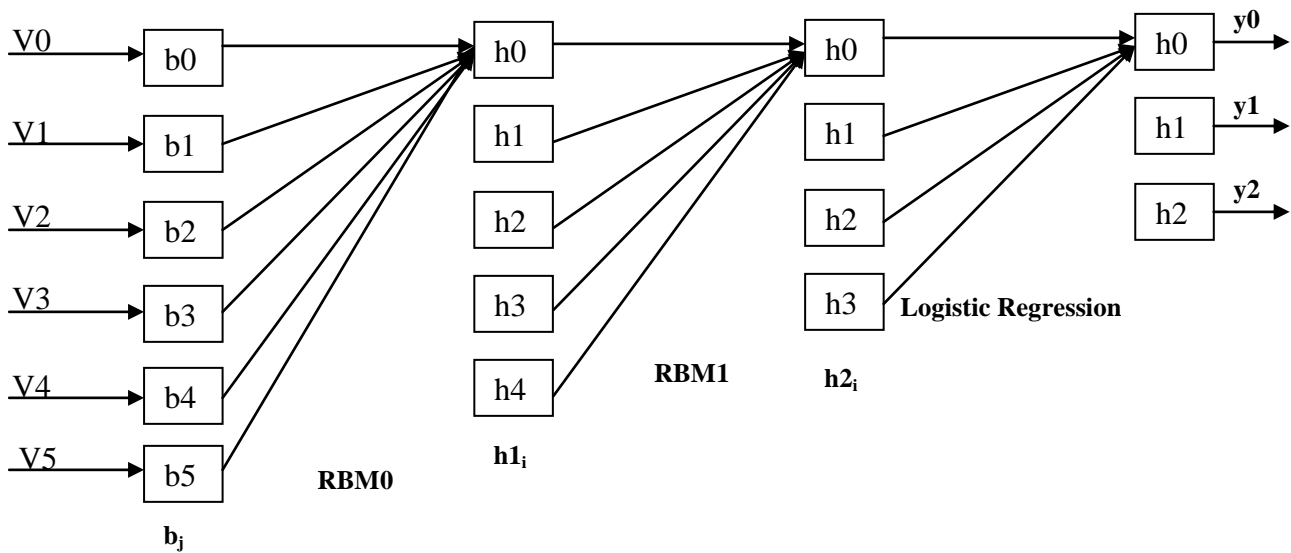


Figure 22: Pre-training: Forward layer wise calculation

Step 6: Logistic Layer training at output Layer...(detail algorithm is in APPENDX B)

Logistic Layer Training...

Y given x before softmax

Table 35: Output given input in logistic layer before softmax function

$y'0$	$y'1$	$y'2$
0	0	0

Y given x After softmax

Table 36: Output given input in logistic layer after softmax function

$y0$	$y1$	$y2$
0.333333333	0.333333333	0.333333333

Actual output

Table 37: Nicotine interacting with target $t0$ and $t1$ output

$t0$	$t1$	$t2$
1	0	1

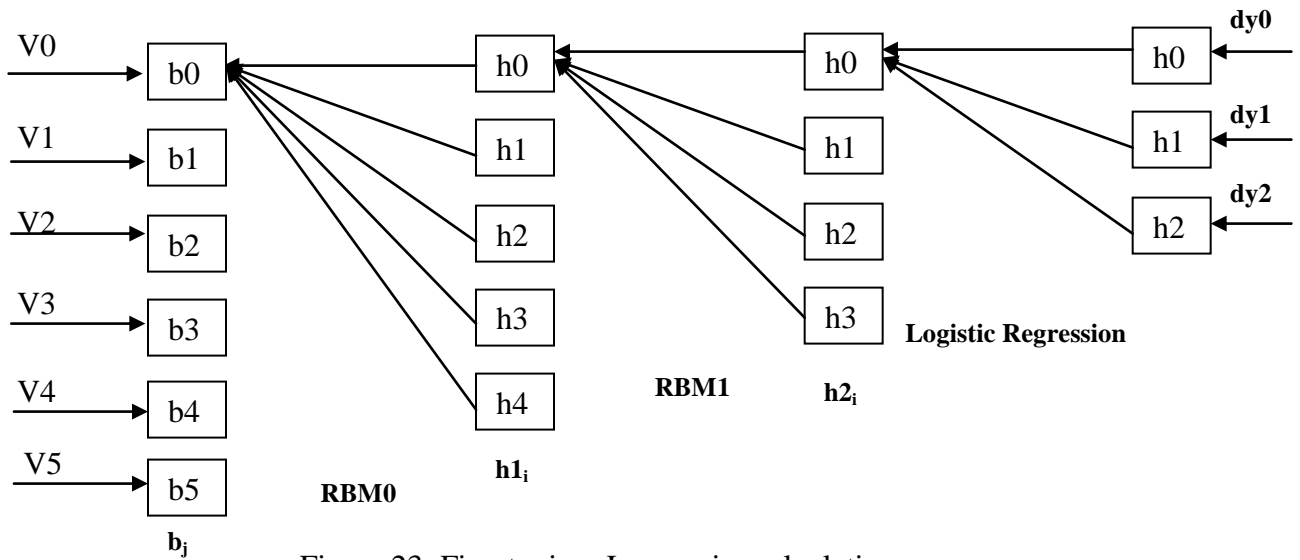


Figure 23: Fine-tuning: Layer wise calculation

Change in output

Table 38: calculated error in output layer

t0-y0	t1-y1	t2-y2
0.666666667	-0.333333333	0.666666667

Modified weight in Logistic Layer

Table 39: Updated weight in logistic layer

	V[0]	V[1]	V[2]	V[3]
H[0]	0	0.066666667	0.066666667	0
H[1]	0	-0.033333333	-0.033333333	0
H[2]	0	0.066666667	0.066666667	0

Modified bias in Logistic Layer

Table 40: Updated bias in logistic layer

b0	b1	b2
0.066666667	-0.033333333	0.066666667

Step 7: Update the weight in each hidden layer by back propagating the error calculated from Logistic Layer as given in algorithm 2.

Back Propagation...

In layer 2 (RBM1)

Del error calculation..

Table 41: Δ error calculation in RBM1

dy0	dy1	dy2	dy3
0	0	0	0

Modified Weight

Table 42: Fine-tuning: Updated weight in RBM1 due to back propagation

	V[0]	V[1]	V[2]	V[3]	V[4]
H[0]	-0.079697	0.184612297	0.0964073	0.1368804	-0.16735
H[1]	0.0364546	-0.1165118	0.0715724	0.1883471	-0.155482
H[2]	0.1231332	0.16900772	0.0510958	0.0446932	0.015394
	0.2095077	0.139699	0.1553396	0.1740789	0.073585

Bias

Table 43: Fine-tuning: Updated bias in RBM1 due to back propagation

b0	b1	b2	b3
0.04957112	0.05694714	0.045466117	0.045104265

In RBM 0

del calculation.. by error propagating from RBM1.

Table 44: Δerror calculation.. By error propagating from layer RBM1 to layer RBM0

dy0	dy1	dy2	dy3	dy4
0	0	0	0	0

Modified Weight

Table 45: Fine-tuning: Updated weight in RBM0 due to back propagation

	V[0]	V[1]	V[2]	V[3]	V[4]	V[5]
H[0]	-0.091923	0.142877633	-0.0271	0.064605	0.0031	-0.0127876
H[1]	0.072003	-0.03558026	-0.1019	-0.15776	-0.2015	0.06479689
H[2]	0.1383465	0.150576333	0.11207	0.004736	-0.0235	-0.1454741
H[3]	-0.016223	0.062870283	-0.0261	0.155159	0.0236	-0.1064794
H[4]	0.2043391	0.004441674	-0.0755	0.065074	-0.1358	-0.1718479

Bias

Table 46: Fine-tuning: Updated bias in RBM0 due to back propagation

b0	b1	b2	b3
-0.055715446	0.05466089	0.045101081	0.048468007
			-0.042242452

Step 8: checking the prediction result using a same drug used in training

Testing.....

The of testing data=1

Testing Drug sub-structure fingerprint input.....

Table 47: Testing Nicotine Drug sub-structure feature input

v0	v1	v2	v3	v4	v5
1	1	1	0	0	0

Predicted Drug-Target interaction.....

Table 48: Predicted output of Nicotine drug with targets after first iteration training

y0	y1	y3
0.35561256	0.288774879	0.221937198

Actual Drug-Target interaction.....

Table 49: Actual Nicotine Drug interaction with targets

t0	t1	t2
1	0	1

This is prediction error more as it this is only result after one epoch. So there is need of increasing number of epochs to update the parameters for good result.

After 1000 epochs in pre-training and 500 epochs in fine training we get

Predicted Drug-Target interaction.....

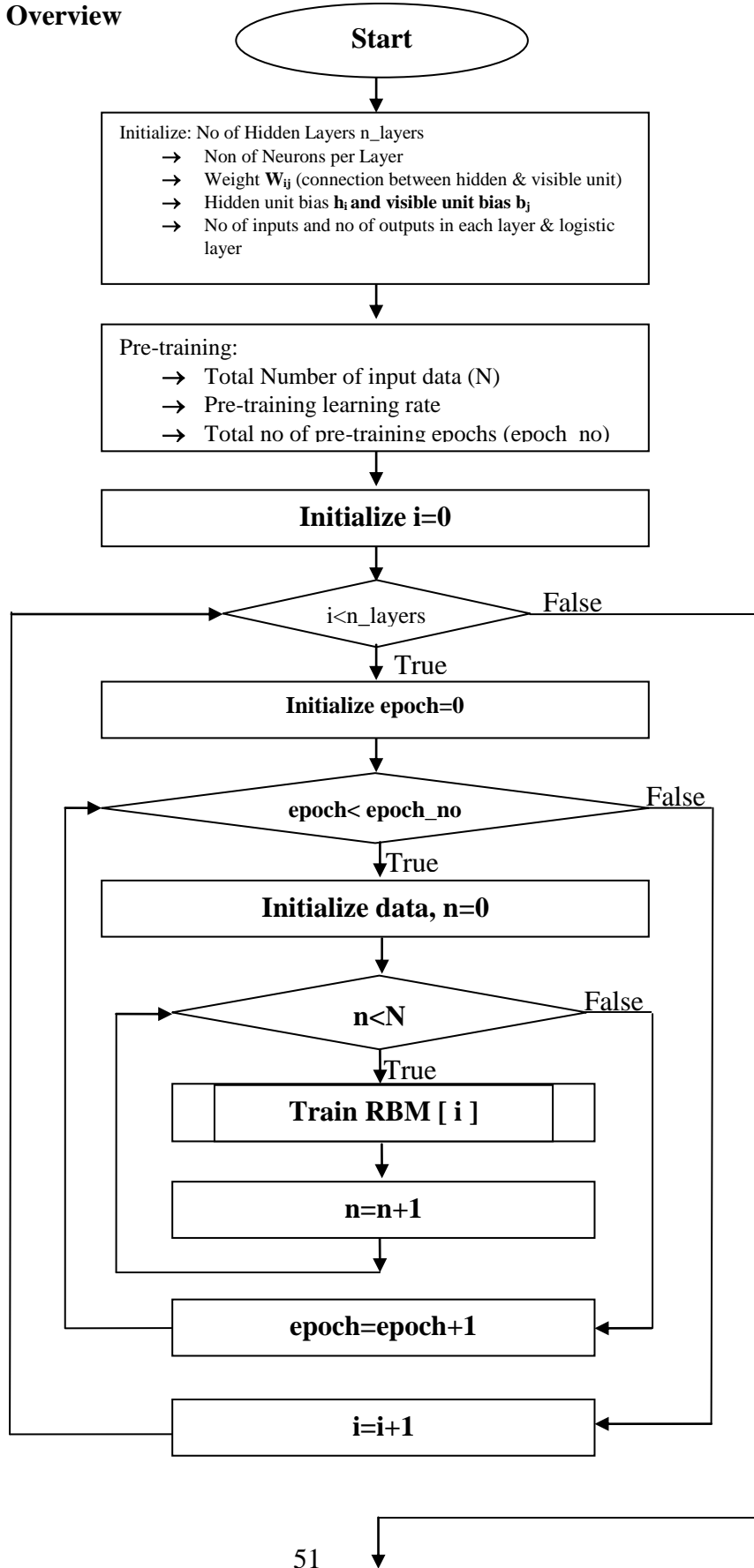
Table 50: Predicted output for Nicotine drug with proteins

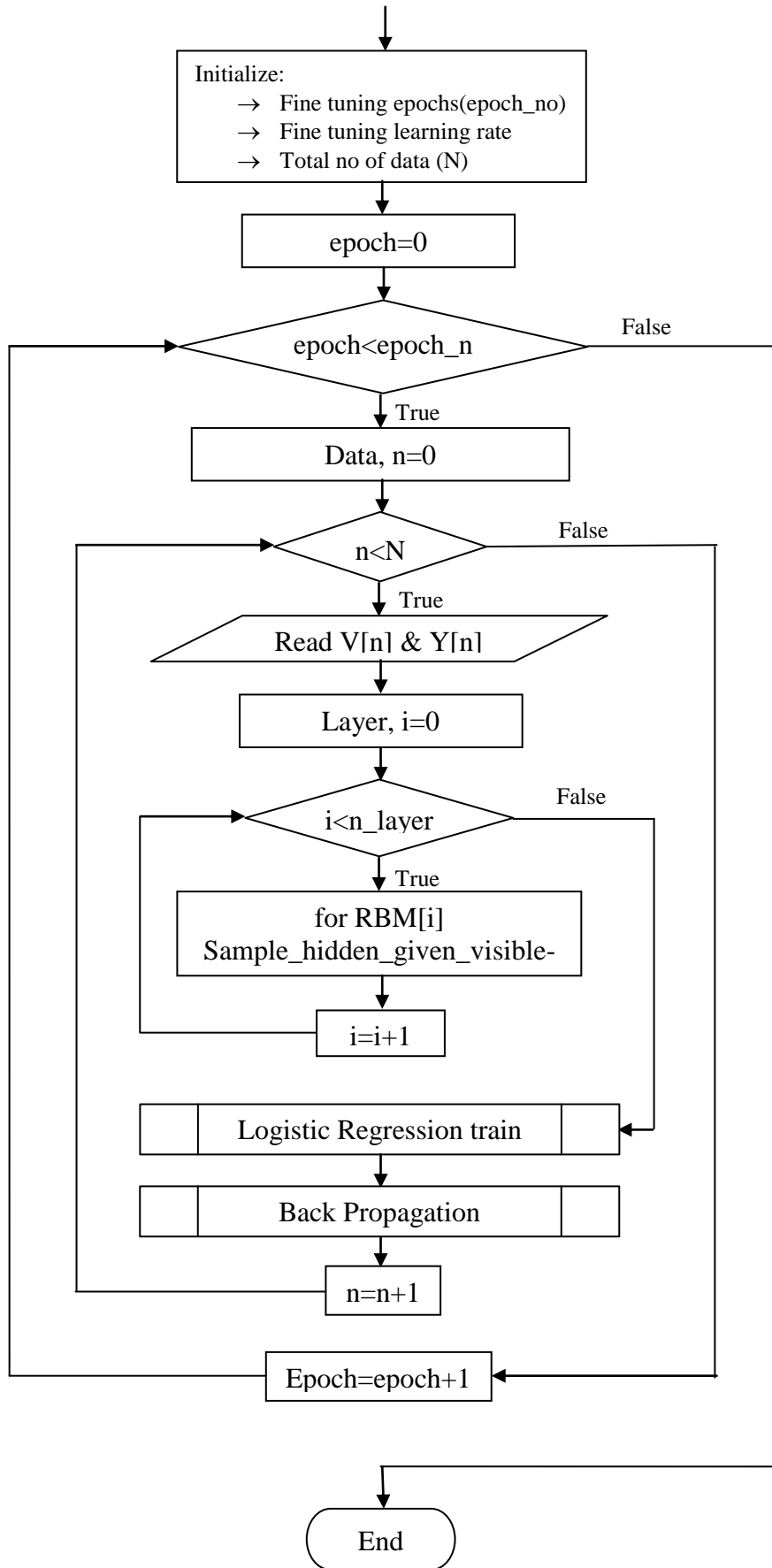
y0	y1	y2
0.5	7.31E-54	0.5

Step 9: converting the result in binaries form suing threshold of 0.5 we get 100% prediction in this case Nicotine drug.

3.2 System Design Overview

3.2.1 Flowchart





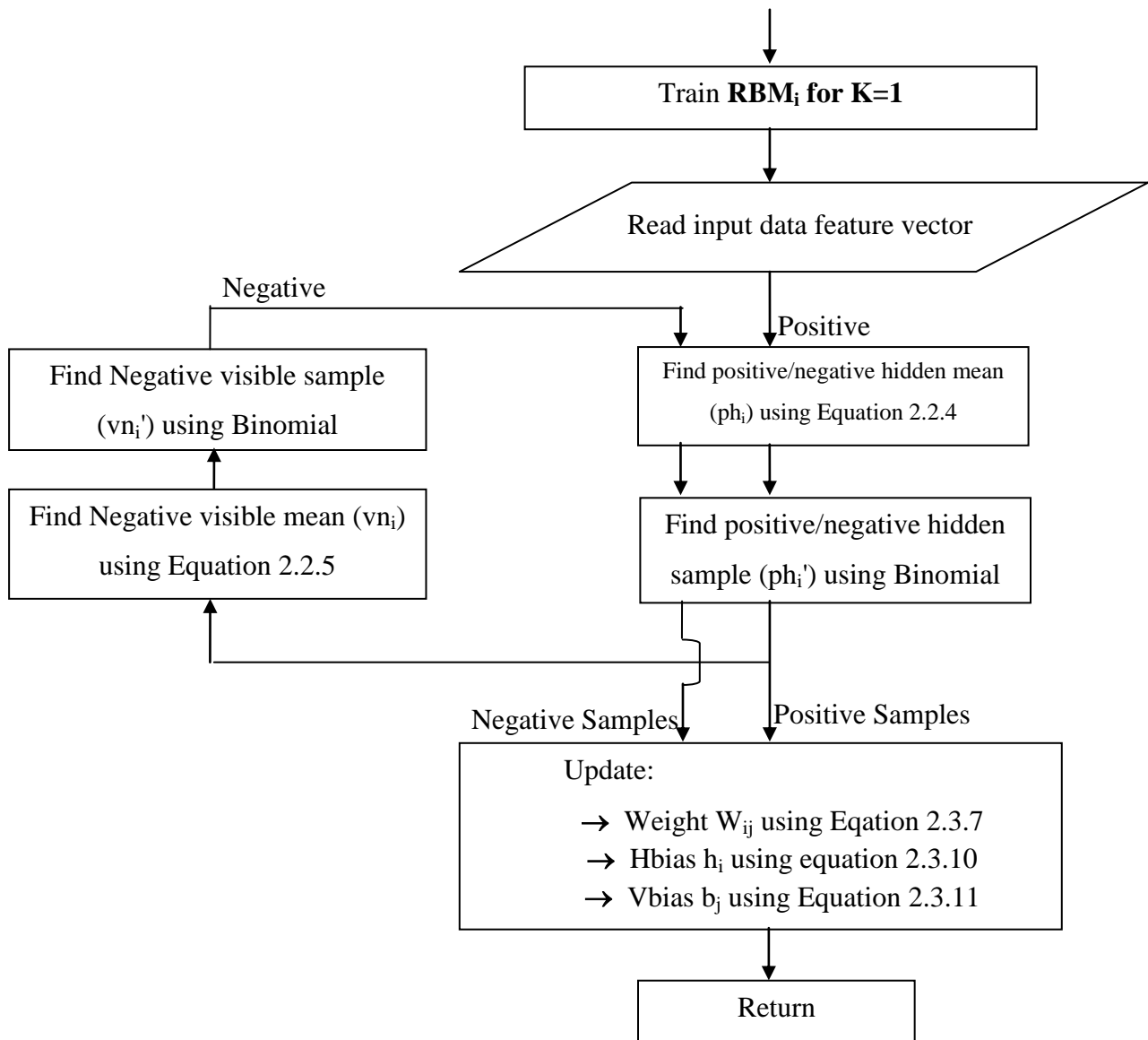


Figure 24: Flowchart Development for PDTI-DBN framework

3.2.2 Dimension Reduction

As training time taken for 881-dimensional feature training is very high, there is need of feature reduction. Although, DBN oneself is feature detector in feature reducing fashion, it was tried to check the performance by removing the features who is not present in more than 100 drugs. For better performance, it has been also planned to cascade PCA (principal component Analysis) for feature reduction purpose.

3.2.3 Learning

For learning features from drug-structure component, the drug-substructure feature is encoded in binary fingerprint. Using this fingerprint, DBN is used for feature learning using both supervised and unsupervised way. Initially, unsupervised is used which extracts the feature from drug as fingerprint and RBM of DBN is used to encode the fingerprint features.

The learning component using in DBN are listed below:

- I. The weight between visible layer and hidden layer, W_{ij} .
- II. The visible bias (offset) b_j
- III. Hidden layer bias (offset) h_i

The dimension of learning parameters depends on the input feature dimension, number of hidden layers and number of neurons hidden layers. For example: the following table illustrates the parameter dimension while using 881 dimension feature or 286 feature dimensions and 3 hidden layers of each 2000 neurons.

Table 51: Learning Parameter Dimension

	RBM 0	RBM 1	RBM 2
881 feature inputs	b_j of size 881 $(j=0,1,2,\dots,880)$ h_i of size 2000 $(i=0,1,2,\dots,1999)$ W_{ij} of 881×2000	b_j of size 2000 $(j=0,1,2,\dots,1999)$ h_i of size 2000 $(i=0,1,2,\dots,1999)$ W_{ij} of 2000×2000	b_j of size 2000 $(j=0,1,2,\dots,1999)$ h_i of size 2000 $(i=0,1,2,\dots,1999)$ W_{ij} of 2000×2000
286 feature inputs	b_j of size 286 $(j=0,1,2,\dots,285)$ h_i of size 2000 $(i=0,1,2,\dots,1999)$ W_{ij} of 881×2000	b_j of size 2000 $(j=0,1,2,\dots,1999)$ h_i of size 2000 $(i=0,1,2,\dots,1999)$ W_{ij} of 2000×2000	b_j of size 2000 $(j=0,1,2,\dots,1999)$ h_i of size 2000 $(i=0,1,2,\dots,1999)$ W_{ij} of 2000×2000

All above learning parameters are initially assigned by random value and then update by training features of each drug one by one in iterative way. After finishing the pre-training, fine tuning is done using pre-trained adjusted parameter and known interaction (drug-target interaction profile

dataset) in supervised way. Finally, the system goes in equilibrium stage, by updated parameters. This is the case of fully learned stage.

3.2.4 Modules Developed

The total eight different modules with unique algorithms have been implemented in this thesis which is as tabulated below:

Table 52: List of implemented modules

MODULE	Programming Language	Source
Input/output	Java	Own Algorithm+ Self Implementation
RBM	Java	Hinton Algorithm+Self Implementation
DBN	Java	Hinton Algorithm+Self Implementation
Logistic Layer	Java	Algorithm +self Implementation
Back Propagation	Java	Algorithm +self Implementation
Binarizing output	VBA	Own algorithm integrating with Excel
Evaluation Module	VBA	Own algorithm integrating with Excel
Visualization Module	Excel	Excel framework

CHAPTER 4: RESULT AND EVALUATION

4.1 Results

There are altogether 5 experiments performed using different hyper-parameters and using cross-fold techniques. The obtained results with corresponding parameters are listed with experiment wise below:

Experiment 1:

Table 53: hyper parameter for experiment1.

Patameters	Values
No of Hidden Layers	3
Total number of Neurongs per layer	{1000, 1000, 1000 }
Activation Function	Sigmoid
size of minibatch	50
Total number of input data	1000
pre-training epchos	1000
pre-training learning rate	0.1
fine-tuning epchos	500
fine-tuning learning rate	0.1

Table 54 : Predicted Output table (row: drug and column UniProt Protein ID)

	A2A2V4	A8MPY1	A9UF02	B8DCL9	B8DD61	O00305
Pravastatin	9.81E-04	9.81E-04	9.81E-04	9.81E-04	9.81E-04	9.81E-04
Fluvoxamine	1.90E-04	1.90E-04	1.90E-04	1.90E-04	1.90E-04	1.90E-04
Valsartan	5.63E-05	5.63E-05	5.63E-05	5.63E-05	5.63E-05	5.63E-05
Ramipril	0.00122	0.00122	0.00122	0.00122	0.00122	0.00122
Masoprocol	0.001285	0.001285	0.001285	0.001285	0.001285	0.001285
Flunisolide	0.001274	0.001274	0.001274	0.001274	0.001274	0.001274
Baclofen	0.001025	0.001025	0.001025	0.001025	0.001025	0.001025

Experiment 2 and 3: Changing number of hidden units per layer in experiment2 and changing number of sub-structure features from 881 to 286 in experiment 3.

Table 55: hyper-parameter for experiment 2 and 3.

Patameters	Exp 2 Values	Exp 3 Values
No of Hidden Layers	3	1
Total number of Neurongs per layer	{2000, 2000, 2000 }	{2000,2000,2000}
Activation Function	Sigmoid	Sigmoid
size of minibatch	50	50
Total number of input data	1000	1000
pre-training epchos	1000	1000
pre-training learning rate	0.1	0.1
fine-tuning ecphos	500	500
fine-tuning learning rate	0.1	0.1
no of input drug data features	881	286

The output of above three experiments using single threshold 0.5 for binary classification and using equation 3.20, 3.21, 3.22, 3.23 and 3.24 between actual interaction and predicted interaction, are tabulated in Table 4.4 and Table 4.5 respectively as below:

Table 56: Experiment wise output of PDTI-DBN framework

	Exp1	Exp2	Exp3	Exp 4	Exp5
True Positive (TP)	3	14	13	13	12
True Negative (TN)	7727	7726	7725	7720	7723
False positive (FP)	0	2	2	7	4
False Negative (FN)	20	8	10	10	11

Using the same hyper-parameters of experiment 3 (Exp3), additional experiment 4(Exp4) and experiment 5(Exp5) were performed by exchanging the training and testing data for 3 cross-fold experiments.

4.2 Evaluation

Evaluation has been performed using the evaluation technique from previous sections,

Table 57: Prediction outputs of Experiment 1, 2, 3, 4 and 5.

	Exp 1	Exp2	Exp 3	Exp4	Exp5

Sn	0.130435	0.636364	0.565217391	0.565217	0.521739
Pre	1	0.875	0.866666667	0.65	0.75
Spc	1	0.999741	0.991741	0.991094	0.991482
Acc	0.997419	0.99871	0.998451613	0.997806	0.998065
F1	0.230769	0.736842	0.684210526	0.604651	0.615385

4.2.1 Using threshold 0.5

Table 4.5 shows the output performed from experiments 1-5 i.e. (Exp1-Exp5) . From the table , prediction can evaluated by indices sensitivity (Sn) , Precision (Pre), Specificity (Spc), Accuracy (Acc) , F1-Score(F1). As the single point threshold is not efficient for binary classification for predicting the model, there is need of taking variable threshold values for binary classification on the prediction output. Taking threshold 0.35, 0.45 and 0.5, result and evaluations are done in next section 4.2.2.

4.2.2 Using threshold 0.35, 0.45, 0.5 and 0.75

Taking threshold 0.35, 0.45, 0.5 and 0.75, we get the following results in each experiment 2 to experiment 5. Experiment 1 result has not been included as it was worse due to less number of neurons i.e. non-adjustment of hyper-parameters.

Table 58: Sensitivity and Specificity of from predicted output with multipoint threshold

Experiments	Threshold	Sensitivity (Sn)	Specificity(Spc)	1-Secificity
Experiment 2	0.35	0.5612	0.99922	0.00078
	0.45	0.6120	0.992	0.008
	0.5	0.63636	0.99741	0.009
	0.75	0.7321	0.98	0.02
Experiment 3	0.35	0.0434	0.99904	0.00096
	0.45	0.0524	0.99353	0.00647
	0.5	0.5621	0.991741	0.008249
	0.75	0.6122	0.97292	0.02708

Experiment 4	0.35	0.0434	0.99904	0.0096
	0.45	0.0434	0.99353	0.00647
	0.5	0.565217	0.991094	0.008906
	0.75	0.60991	0.9812	0.0188
Experiment 5	0.35	0.0312	0.9992	0.0008
	0.45	0.1241	0.9991	0.0009
	0.5	0.5621	0.991482	0.00029
	0.75	0.62121	0.98013	0.01987

4.3 Result Summary and Comparison with existing prediction methods

4.3.1 Result Summary

In Figure 4.1, it is clearly seen that the True positive (TP), False positive (FP), False Negative (FN) and TN negative result of all experiments. From figure, it is clearly seen that number of True positive of Experiment 2 is better than others. It is due to presence of 881 features in experiment 2 .

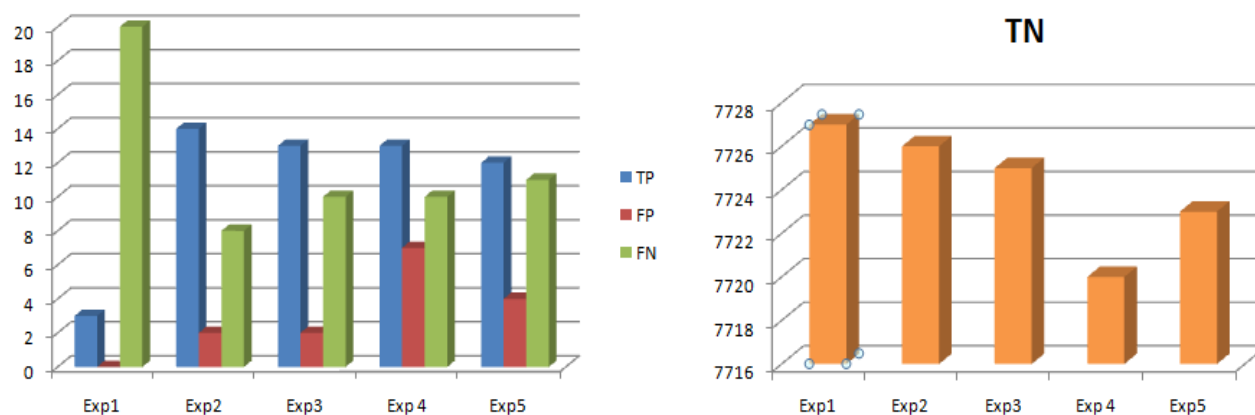


Figure 25: Bar chart graph of drug-target interaction prediction result

For drawing ROC curve of PDTI-DBN framework, the average sensitivity and specificity of five experiments with multi threshold are calculated as given in table 4.7 and its corresponding ROC curve is as shown in figure 4.2.

Table 59: Average of five experiment results from table

	Sensitivity	Specificity	1-Specificity
0.35	0.1698	0.999125	0.000875
0.45	0.207975	0.99454	0.00546
0.5	0.58144425	0.99293175	0.00706825
0.75	0.643855	0.9785625	0.0214375

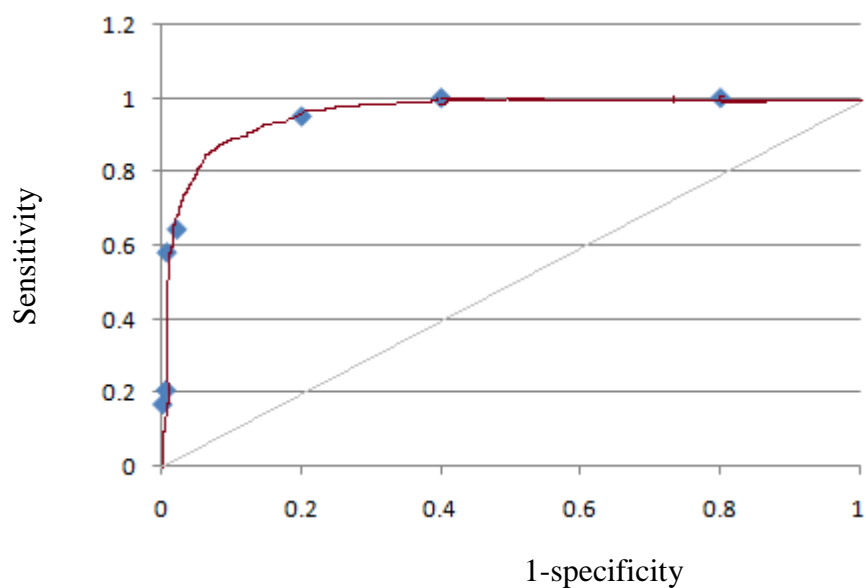


Figure 26: ROC curve of Five Experiment's average result

Form above ROC curve, the AUC can be easily calculated which numerically better value than existing drug-target interaction model.

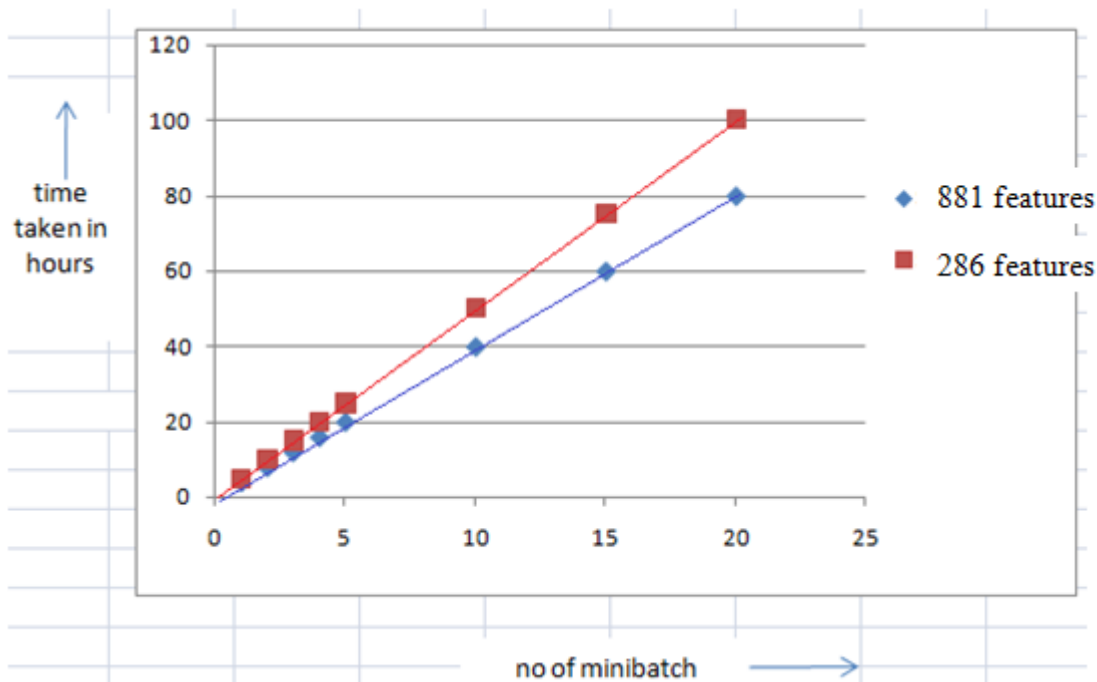


Figure 27: Time versus minibatch graph of 881 features and 286 features pre-training

From the figure 4.3 , it is shown that time taken by minibatch of 50 drugs is more when 881 drug features are present in compare to 286 feature presence. For one mini-batch training in 881 – dimensional feature space, it takes 5 hours where as for reduced features to 286 mini-batches training takes only 4 hours. It saves the time for training but result is worse than 881 dimensional feature.

From figure 4.4, by comparing the results obtained from experiments, it is clear that experiment 2 has better performance than other experiments. One more important point is that the PDTI-DBN framework predicts well even the less important feature are removed from feature space.

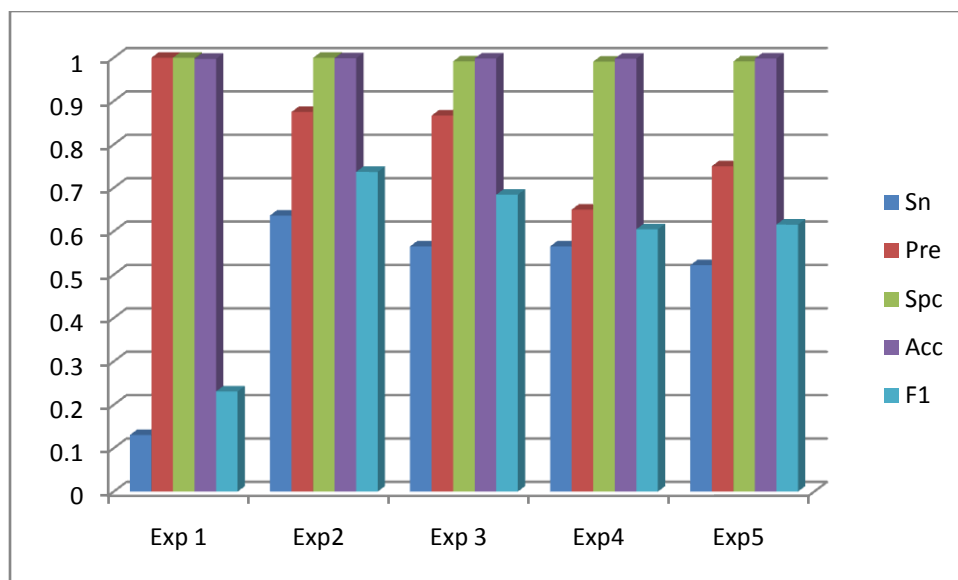


Figure 28: Bar chart of prediction result in terms of Sn, Pre, Spc, Acc and F1

4.3.2 Comparison with Existing Prediction Methods

As shown in above result, it can be said that the accuracy of PDTI-DBN is more than 99% which is comparatively between than existing approaches for drug target interaction. All the existing systems even in different dataset and using different other machine learning approaches have maximum 98% accuracy. Similarly, as sheen ROC curves, there is curve above the 45 degree angle line which shows DTI-DBN predicts well.

One more this is analyzed that DTI-DBN takes 7days to train 1000 drug features, there is difficult to get adjusted hyper-parameter frequently to get better performance. To reduce the training time period, it is seen that there is need of some parallel computation technique for better hyper-parameter adjusting.

CHAPTER 5: CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this thesis, Drug-target interaction prediction was performed efficiently using deep belief network. The drug sub-structure finger print was used as pre-training data set for the DBN and with known corresponding drug-target interaction profile was used for fine tuning of DBN using back-propagation method. Also logistic layer was stacked at output layer for better update of Parameters initialized by pre-training. From the five experiments with different hyper-parameters and different random training and testing data set of 1007 drugs, it is concluded as:

- PDTI-DBN framework predicts the drug-target interaction efficiently with good AUC value and 99% efficiency for gold standard data.
- The maximum F1-score of this framework was obtained as 73% in experiment 2 with 881 dimensional features and it get little bit declined when the feature is reduced to 286 dimensional features.

5.2 Future Work

- As deep learning has high computation complexity, there is need of parallel programming approach to reduce the training time. DTI-DBN framework developed using sequential programming is to be converted into parallel using CUDA programming which is executed on GPU for better performance on different hyper-parameters..
- PDTI-DBN framework will be extended with adapting Deep Belief Network with hyper-parameters.

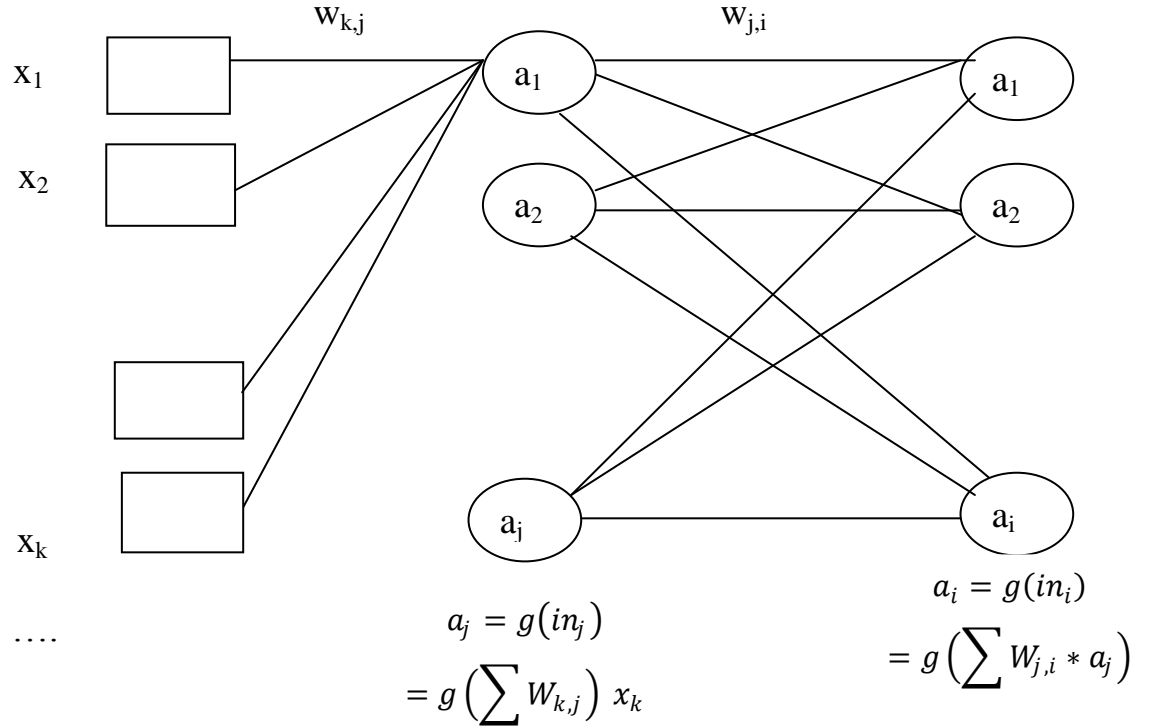
REFERENCES:

- [1] G. P. a. M. D. Vivo, "Computational Chemistry for Drug Discovery," *Springer Science+Business Media Dordrecht, Encyclopedia of Nanotechnology*, pp. 1–15, 2015.
- [2] P. D. Lyne, "structure-based virtual screening:an overview," *Elsevier Science Ltd, research focus*, vol. 7, pp. 1048–1055, 2002.
- [3] J. A. DiMasi, "new drug development in the United States from 1963 to 1999," *Clinical Pharmacology and Therapeutics*, vol. 69, pp. 286–296, 2001.
- [4] B. B. a. R. Zimmel, "prospects for productivity," *Nature Reviews: Drug Discovery*, vol. 3, pp. 451–457, 2004.
- [5] G. Nidhi Singh, David M. Ferguson, and Christopher R. McCurdy, "a combined ligand-based and targetbased drug design approach for g-protein coupled receptors: a pplication to salvinorin a, a selective kappaopioid receptor agonist," *J Comput Aided Mol Des*, vol. 7, pp. 471–493, 2006.
- [6] J. M. B. Tami J. Marrone, and J. Andrew Mc-Cammon, "structure-based drug design:computational advances," *Annu. Rev. Pharmacol. Toxicol*, pp. 71–90, 1997.
- [7] A. L. Hopkins, "network pharmacology: the next paradigm in drug discovery," *Nature Chemical Biology*, vol. 4, pp. 682–690, 2008.
- [8] Consortium, "The Universal Protein Resource ,The UniProt," 2008.
- [9] G. E. H. a. R. R. Salakhutbinov, "Reducing the Dimensionality of Data With Neural Network," *Science* vol. 313, pp. 504-507, July 28 2006.
- [10] M. A. Yoshihiro Yamanishi, Alex Gutteridge , Wataru Honda, "Prediction of drug–target interaction networks from the the local information and neighbors," *Bioinformatics*, vol. 24, pp. i232-i240, 2008.
- [11] K. B. a. Y. Yamanishi, "Supervised prediction of drug–target interactions using bipartite local models," vol. 25, pp. 2397–2403, 2009.
- [12] C.-K. K. Jian-Ping Mei, Peng Yang, Xiao-Li Li, Jie Zheng, "Drug-Target Interaction Prediction by Learning From Local Information and Neighbors," *Bioinformatics Advance Access Published*, pp. 1–8, November 17 2012.
- [13] P. Z. Ali Ezzat, Mein Wu, Xiao-Li Li, and Chee-keong Kwoh, "Drug-Target Interaction Prediction with Graph Regularized Matrix Factorization," pp. 1–11, January 2016.
- [14] S. B. N. Twan van Laarhoven, Elena Marchiori, "Gaussian interaction profile kernels for predicting drug–target," *Bioinformatics*, vol. 27, pp. 3036–3043, 2011.
- [15] Y. Y. Masaaki Kotera, Yuki Moriya, RYusuke Sawada, Minoru Kanehisa and Susumu Goto, "DITIES: drug-target interaction network inference engine based on supervised analysis," *Nucleic Acids Research*, pp. 1-7, 2014.
- [16] M. S. Kai Tian, Shuigeng Zhou, Jihong Gauan, "Boosting Comound-Pretein Interaction Prediction by Deep Learning," *Bioinfofrmatics and Biomedicine (BIBM)*, pp. 29-34, 2015.
- [17] B. H. Shbeir Fakhraei, Louiqa Raschid and Lise Getoor, "Network-Based Drug-Target Interaction Prediction with Probablistic Soft Logic," *Transactions on Computational Biology and Bioinformatics*, vol. 11, pp. 775-787, 2014.
- [18] T. C. Songpeng Zu, Shao Li, "Global Optimization-based Inference of Chemogenomic Features from Drug-Target Interacitons," 2015.
- [19] S.-M. Y. Jian-Yu Shi, "SRP: A Concise Non-parametric Similarity-Rank-based Model for Predicting Drug-Target Interactions," *International Conference on Bioinformatics and Biomedicine (BIBM)*, pp. 1636-1641, 2015.

- [20] B. L. Lihong Peng, Wen Zhau, Zejun Li., "Predicting Drug-Target Interactions with Multi-information Fusion," *Biomedical and Health Informatics*, pp. 1–12, 2016.
- [21] A. Koochi, "Prediction of drug-target interactions using popular Collaborative Filtering methods," *GENSIPS*, pp. 58-61, 2013.
- [22] A. P. Salvatore Alaimo, Rosalba Giugno and Alfredo Ferro, "Drug-Target prediction through domain- tuned network-based inference," pp. 1-5, June 15 2013.
- [23] S. L. Dong-Sheng Cao, Qing-Song Xu, Hong-Mei Lu, Jian-Hua Huang, Qian-Nan Hu, Yi-Zeng Liang, "Large-scale prediction of drug-target interactions using protein sequences and drug topological structures," *Analytica Chimica Acta, Elsevier ScienceDirect*, vol. 752, pp. 1 - 10, 2012.
- [24] H. L. O. Edgar D. Coelho, "Ensemble-Based Methodology for the Prediction of Drug-Target Interactions," *International Symposium on Computer-Based Medical Systems*, pp. 36-41, 2016.
- [25] S. K. Zaynab Mousavian, Kaveh Kavousi, Ali Masoudi-Nejad, "Drug-target interaction prediction from PSSM based evolutionary information," *Journal of pharmacological and Toxicological Methods*, vol. 78, pp. 42-51, 2015.
- [26] R. P. Upul Senanayake, and Roshan Ragel, "Machine Learning based Search Space Optimization for Drug Discovery," pp. 68-75, 2013.
- [27] R. P. S. Junshui Ma, Andy LiaW, George E. Dahl and Vladimir Svetnik, "Deep Neural Nets as a Method for Quantitative Structure-Activity Relationships," *Chemical Information and Modeling*, pp. A-L, December 17 2014.
- [28] P. A. T. Sunghwan Kim, Evan E. Bolton, "PubChem Substance and Compound databases," *Nucleic Acids Research*, vol. 44, pp. D1202–D1213, September 2015.
- [29] C. K. David S. Wishart, An Chi Guo, Dean Cheng, Savita Shrivastava., "DrugBank: a knowledgebase for drugs, drug actions and drug targets.," *Nucleic Acids Research*, vol. 36, pp. D901–D906, 2008.
- [30] Y. Bengio, "Learning deep architecture of AI," *Foudation and Trends in Machine Learning*, pp. 1-127, 2009.
- [31] A. n. I. a. C. Igel, "Training Restricted Boltzmann Machines: An Introdcution*," *Pattern Recognition*, vol. 47, pp. 25-39, 2014.
- [32] B. W. L. Xue Sen Lin, Xin Yi Yang, "Engine Components Fault Diagnosis Using an Improved Method of Deep belief networks," pp. 454-459, 2016.
- [33] B. R. Sofia Visa, Anca Ralescu and Esther van der Knaap, "Confusion Matrix-based Feature Selection," vol. 37.

APPENDIX A

Back Propagation Algorithm



In the above, there is input layer, one hidden layer and one output layer. x_1, x_2, \dots, x_k are input of k units. Hidden layer has j units and there are i outputs. $W_{k,j}$ is weight between input and hidden layer and $w_{j,i}$ is the weight between hidden layer and output layer. Using forward calculation output y calculated at first and then error is obtained and expressed as :

$$E_{error} = \frac{1}{2}(y - a_i)^2$$

$$\frac{\partial E}{\partial W_{k,j}} = - \sum_i (y_i - a_i) \frac{\partial g(in_i)}{\partial W_{k,j}}$$

$$\frac{\partial E_r}{\partial W_{k,j}} = - \sum_i (y_i - a_i) \frac{\partial a_i}{\partial W_{k,j}}$$

Where, $(y_i - a_i) * g'(in_j) = \Delta_i$ (Let).

Now, above equation reduced to

$$\begin{aligned}
&= - \sum_i \Delta_i W_{j,i} g'(in_k) \frac{\partial(in_k)}{\partial W_{k,j}} \\
&= - \sum_i \Delta_i W_{j,i} g'(in_k) \frac{\partial}{\partial W_{k,j}} \left(\sum_{k'} W_{k,j} * a_{k'} \right) \\
&= - \sum_i \Delta_i W_{j,i} g'(in_k) a_k \\
&= (-a_k) \Delta_j
\end{aligned}$$

$$\therefore W_{k,j} \leftarrow W_{k,j} + \alpha a_k \Delta_j \quad \text{Equation(A.1)}$$

Where,

$$\Delta_j = \Delta_i W_{j,i} g'(in_k) \text{ and } \alpha \text{ is learning rate.}$$

APPENDIX B

Protein ID with Complete name

Target	Organism	Name
UniProtKB - Q15822	ACHA2_HUMAN	Neuronal acetylcholine receptor subunit alpha-2
UniProtKB - Q15825	ACHA6_HUMAN	Neuronal acetylcholine receptor subunit alpha-6
UniProtKB - Q9G226	Q9G226_RANSY	ATP synthase subunit A
UniProtKB - Q9UGM1	ACHA9_HUMAN	Neuronal acetylcholine receptor subunit alpha-9
UniProtKB - P36544	ACHA7_HUMAN	Neuronal acetylcholine receptor subunit alpha-7
UniProtKB - P43681	ACHA4_HUMAN	Neuronal acetylcholine receptor subunit alpha-4
UniProtKB - Q05901	ACHB3_HUMAN	Neuronal acetylcholine receptor subunit alpha-3
UniProtKB - P23219	PGH1_HUMAN	Prostaglandin G/H synthase 1
UniProtKB - P35354	PGH2_HUMAN	Prostaglandin G/H synthase 2

UniProtKB - Q04828	AK1C1_HUMAN	Aldo-keto reductase family 1 member C1
UniProtKB - P60725	RL4_ECO57	50S ribosomal protein L4
UniProtKB - P61177	RL22_ECO57	50S ribosomal protein L22
UniProtKB - Q8XJ01	PBPA_CLOPE	Penicillin-binding protein 1A

Source: <http://www.uniprot.org/uniprot/>

Drug	Description
Nicotine	Nicotine is highly toxic alkaloid. It is the prototypical agonist at nicotinic cholinergic receptors where it dramatically stimulates neurons and ultimately blocks synaptic transmission. Nicotine is also important medically because of its presence in tobacco smoke. [PubChem]
Aspirin	The prototypical analgesic used in the treatment of mild to moderate pain. It has anti-inflammatory and antipyretic properties and acts as an inhibitor of cyclooxygenase which results in the inhibition of the biosynthesis of prostaglandins. Acetylsalicylic acid also inhibits platelet aggregation and is used in the prevention of arterial and venous thrombosis. (From Martindale, The Extra Pharmacopoeia
Erythromycin	Erythromycin is a macrolide antibiotic produced by <i>Streptomyces erythreus</i> . It inhibits bacterial protein synthesis by binding to bacterial 50S ribosomal subunits; binding inhibits peptidyl transferase activity and interferes with translocation of amino acids during translation and assembly of proteins. Erythromycin may be bacteriostatic or bactericidal depending on the organism and drug concentration
Amoxicillin	A broad-spectrum semisynthetic antibiotic similar to ampicillin except that its resistance to gastric acid permits higher serum levels with oral administration. Amoxicillin is commonly prescribed with clavulanic acid (a beta lactamase inhibitor) as it is susceptible to beta-lactamase degradation. [PubChem]

Source: <http://www.drugbank.ca/drugs>